DEDICATION TO GENE H. GOLUB ON THE OCCASION OF HIS 15TH BIRTHDAY

SIAM has short abbreviations for the names of its journals—SIREV, SINUM, SISSC, etc. For over a year now, this particular issue of SIMAX has been known as SIGENE, because it is dedicated to Gene Golub on the occasion of his 15th birthday. Yes, that's right, 15th. Gene gets to legally celebrate his birthday only every fourth year. But you probably already knew all that. Because you probably already know Gene.

I can't think of anybody else who knows more people, and who is really liked by more people, than Gene Golub. That's because he has devoted his life to this business—to numerical analysis and scientific computing. In fact, most of us have actually been to his home at one time or another. We're his family. And, for many of us, he is part of our extended family.

Gene's professional family is richly international. Over the last 30 years, thousands of people have visited Gene, studied with Gene, and worked with Gene and his group at Stanford. Many of these have been young people from outside the United States: grad students from Latin America and Europe and post docs and visiting scholars from all over the world. And then Gene has returned the visits. For someone who says he hates to travel, he sure does a lot of it.

Through these many friends, Gene has deeply affected our profession. He has helped to make it warm, friendly, personal, and fun.

Thanks, Gene.

Gene earned three degrees in Mathematics from the University of Illinois in the fifties. He spent a year on a fellowship at Cambridge (where his office mate was Velvel Kahan), a year each at Lawrence Radiation Lab and Space Technology Labs, and then, in 1962, he came to Stanford. He's been there ever since, although it seems like Stanford allows him to take a sabbatical of some kind almost every other quarter. He has honorary degrees from Oxford, Linköping, Grenoble, Waterloo, Dundee, and Illinois. He was elected to the Swedish Royal Academy of Engineering Sciences in 1986 and to the U.S. National Academy of Engineering in 1990.

His California license plate appropriately says "Dr. SVD." He has published almost 100 papers. The first, with Richard Varga on Chebyshev iterative methods, was published in Numerische Mathematik in 1961. The most recent is a joint paper with Peter Arbenz, soon to appear in SIMAX. But the September 23, 1991 issue of his CV already lists four more papers "accepted for publication."

His book with Charlie Van Loan, *Matrix Computations*, is now in its second edition and is the definitive reference and text in the field.

He has edited or coedited half a dozen conference proceedings. He has been on the editorial board of fifteen different journals. So far, twenty students have done their Ph.D. dissertations under Gene's supervision. More are on the way. An interdepartmental degree program in Scientific Computing and Computational Mathematics is now in its third year at Stanford and Gene is the director.

Many of his professional activities have involved SIAM. He's been on the Council and the Board of Trustees. He was President of SIAM from 1985 to 1987. And, he is the Founding Editor of not one, but two, SIAM journals: SISSC and SIMAX.

Gene was recently named the Fletcher Jones Professor of Computer Science at Stanford (an honor he shares with Don Knuth). But this is not his first endowed chair. For years, the Numerical Analysis Group at Stanford had offices in an old, intimate house on the edge of campus known as Serra House. Once during that time, Gene was offered an endowed chair at another prestigious university. The crew in Serra House responded by giving Gene a new piece of furniture for his office—the Serra House Chair.

He stayed at Stanford.

Happy 15th Birthday, Gene, from all of us. Now is the time to start planning the celebration of your 16th. It's OK to have the party at your place, isn't it?

Cleve Moler

EDITOR'S NOTE

The SIAM staff is indebted to everyone who made the publication of this issue possible, but especially to Dan Boley, Jack Dongarra, Alan George, Paul van Dooren, and Bob Plemmons, our industrious and discreet SIGENE editors. We also thank John de Pillis for supplying the portrait of Gene and Cleve Moler for providing an overview of Gene's first 15 years.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 1–9, January 1992

CONSTRAINED MATRIX SYLVESTER EQUATIONS*

JEWEL B. BARLOW†, MOGHEN M. MONAHEMI†, AND DIANNE P. O'LEARY‡

Dedicated to Gene Golub on the occasion of his 60th birthday, with gratitude for his tradition of fruitful research in linear algebra, inspired by applications.

Abstract. The problem of finding matrices L and T satisfying $TA - FT = LC$ and $TB = 0$ is considered. Existence conditions for the solution are established and an algorithm for computing the solution is derived. Conditions under which the matrix $[C^T, T^T]$ is full rank are also discussed. The problem arises in control theory in the design of reduced-order observers that achieve loop transfer recovery.

Key words. Sylvester operator, matrix Liapunov equation, loop transfer recovery

AMS(MOS) subject classifications. 93B40, 65F30, 15A06

1. Introduction.

In this paper we consider the following problem: Let n, m, and p be given integers. Given $A \in \mathcal{R}^{n\times n}$, $B \in \mathcal{R}^{n\times p}$, $C \in \mathcal{R}^{m\times n}$, and $F \in \mathcal{R}^{(n-m)\times(n-m)}$, find $L \in \mathcal{R}^{(n-m)\times m}$ and $T \in \mathcal{R}^{(n-m)\times n}$ such that

$$TA - FT = LC, \tag{1}$$

$$TB = 0, \tag{2}$$

$$\text{and} \quad \begin{bmatrix} T \\ C \end{bmatrix} \quad \text{is full rank.} \tag{3}$$

Sylvester [10] considered the homogeneous version of (1) in an 1884 paper. For this reason, (1) is often called a matrix Sylvester equation. Liapunov considered (1) with $A^T = F$ and $LC = I$ in an 1892 monograph [6].

The constrained Sylvester problem (1), (2), and (3) arises in control theory, in the design of reduced-order observers that achieve precise loop transfer recovery [11], [7]. Here, the state model of the system is

$$\dot{x} = Ax + Bu,$$
$$y = Cx,$$

and the observer $z \in \mathcal{R}^{(n-m)\times 1}$ satisfies

$$\dot{z} = Fz + (TB)u + Ly.$$

Tsui [11] has shown that the constrained Sylvester problem is the relevant one to consider in the design of L and T.

* Received by the editors February 11, 1991; accepted for publication (in revised form) June 18, 1991.

† Aerospace Engineering Department, University of Maryland, College Park, Maryland 20742.

‡ Computer Science Department and Institute for Advanced Computer Studies, University of Maryland, College Park, Maryland 20742 (oleary@cs.umd.edu). The work of this author was supported by Air Force Office of Scientific Research grant AFOSR-87-0158.

In §2 we discuss existence and uniqueness of solutions to matrix Sylvester equations. The section following that concerns existence and uniqueness of solutions to the constrained Sylvester problem (1) and (2). The computational algorithm developed in that section is summarized in §4. In §5 we consider conditions under which that algorithm produces a solution to the full problem (1), (2), and (3).

2. Existence of solutions to matrix Liapunov equations. It is well known that a matrix Liapunov equation

$$T\mathcal{A} - \mathcal{B}T = \mathcal{C}$$

has a unique solution T for every choice of $\mathcal{C}$ if and only if $\mathcal{A} \in \mathcal{R}^{n\times n}$ and $\mathcal{B} \in \mathcal{R}^{m\times m}$ have no common eigenvalues. In this section we briefly review this result and related results for the case of common eigenvalues. Our purpose is merely to establish enough notation to discuss conditions under which the full rank condition (3) is violated. Therefore, to simplify the discussion, we consider only the case in which $\mathcal{A}$ and $\mathcal{B}$ each have a complete set of eigenvectors. The general case is studied using the Jordan canonical forms of these matrices (see, for example, [4, Chap. 8]) but leads to the same conclusions.

Let the eigendecomposition of $\mathcal{A}$ be $\mathcal{A} = U_{\mathcal{A}} D_{\mathcal{A}} U_{\mathcal{A}}^{-1}$, where $D_{\mathcal{A}}$ is diagonal with elements α_j, $j = 1, \cdots, n$. Similarly, let $\mathcal{B} = U_{\mathcal{B}} D_{\mathcal{B}} U_{\mathcal{B}}^{-1}$, where $D_{\mathcal{B}}$ is diagonal with elements β_i, $i = 1, \cdots, m$.

Then $T\mathcal{A} - \mathcal{B}T = \mathcal{C}$ is equivalent to

$$U_{\mathcal{B}}^{-1} T \mathcal{A} U_{\mathcal{A}} - U_{\mathcal{B}}^{-1} \mathcal{B} T U_{\mathcal{A}} = U_{\mathcal{B}}^{-1} \mathcal{C} U_{\mathcal{A}},$$

or, with definitions $\hat{\mathcal{C}} = U_{\mathcal{B}}^{-1} \mathcal{C} U_{\mathcal{A}}$ and $\hat{T} = U_{\mathcal{B}}^{-1} T U_{\mathcal{A}}$,

$$\hat{T} D_{\mathcal{A}} - D_{\mathcal{B}} \hat{T} = \hat{\mathcal{C}}.$$

Writing this equation componentwise, we obtain

$$(\alpha_j - \beta_i)\hat{t}_{ij} = \hat{c}_{ij}, \qquad i = 1, \cdots, m, \qquad j = 1, \cdots, n. \tag{4}$$

This leads to the standard result that there is a unique solution $\hat{T}$ (and therefore a unique T) for every choice of $\mathcal{C}$ if and only if $\alpha_j - \beta_i \neq 0$ for all values of i and j.

If any $\alpha_J - \beta_I = 0$, then there is no solution to (4) if $\hat{c}_{IJ} \neq 0$, and an infinite number of solutions if $\hat{c}_{IJ} = 0$, since $\hat{t}_{IJ}$ is then arbitrary. Since $T = U_{\mathcal{B}} \hat{T} U_{\mathcal{A}}^{-1}$, each arbitrary component of $\hat{T}$ contributes a term $\hat{t}_{IJ} u_I v_J^T$ to T, where u_I is the Ith column of $U_{\mathcal{B}}$ and v_J^T is the Jth row of $U_{\mathcal{A}}^{-1}$.

In the problem of interest, the matrix $\mathcal{C}$ is LC, where L is to be determined, so it is sometimes possible to produce a solution to the Liapunov equation even if there are common eigenvalues.

Later we will be interested in conditions that ensure that a matrix related to $\hat{T}$ and $\hat{\mathcal{C}}$ be full column rank. Suppose there exists a nonzero vector u such that $\mathcal{A}u = \lambda u$ and $\mathcal{C}u = 0$. (This is equivalent to saying that the control system is *not observable.*) If λ is a simple eigenvalue of $\mathcal{A}$, then a column of $\hat{\mathcal{C}}$ and the corresponding column of $\hat{T}$ will be zero. If λ is a multiple eigenvalue, then all columns of $\hat{\mathcal{C}}$ may be nonzero, but there will be linear dependence among columns of $\hat{\mathcal{C}}$ corresponding to eigenvectors of that eigenvalue. In this case, the corresponding columns of $\hat{T}$ will have the same linear dependence, since the values α_j in (4) are all equal.

3. Development of an algorithm for the constrained problem. We examine the question of existence and uniqueness of the solution to (1) and (2), and we develop an algorithm for determining the solution if it exists.

Note that there are $(n-m)p$ equations in (2) and $(n-m)n$ equations in (1). There are $(n-m)n$ unknowns in T and $(n-m)m$ unknowns in L, so there are more equations than unknowns if $m < p$.

We may assume that B has full column rank; if not, throwing away the redundant columns does not change the problem. Therefore, the number of rows n in B must be greater than the number of columns p; otherwise, the only solution to $TB = 0$ is $T = 0$. (This is an explanation of the fact that loop transfer recovery cannot be accomplished if the circuit is broken at an "output point.")

Therefore, we may assume that $n > p$ *and* $\operatorname{rank}(B) = p$.

We can eliminate the constraint $TB = 0$ by using the QR factorization of B to define an unconstrained matrix Z. To do this, factor B as

$$B = W \begin{bmatrix} S \\ 0 \end{bmatrix}, \tag{5}$$

where $S \in \mathcal{R}^{p\times p}$ is full rank and $W \in \mathcal{R}^{n\times n}$ is an orthogonal matrix: $W^TW = I$. If we partition W into its first p columns W_1 and its remaining $n-p$ columns W_2, we have

$$TB = T[W_1, W_2] \begin{bmatrix} S \\ 0 \end{bmatrix} = TW_1S.$$

Since the columns of W_2 form a basis for the orthogonal complement of the subspace spanned by the columns of W_1, and since $TW_1S = 0$ if and only if $TW_1 = 0$, we know that

$$T = ZW_2^T \tag{6}$$

for some matrix $Z \in \mathcal{R}^{(n-m)\times(n-p)}$.

Substituting this in the Sylvester equation (1) and multiplying on the right by the nonsingular matrix W, we obtain

$$ZW_2^TA[W_1, W_2] - FZW_2^T[W_1, W_2] = LC[W_1, W_2].$$

This yields the two relations

$$ZA_2 - FZ = LC_2, \tag{7}$$

$$ZA_1 = LC_1, \tag{8}$$

where $A_1 = W_2^TAW_1 \in \mathcal{R}^{(n-p)\times p}$, $A_2 = W_2^TAW_2 \in \mathcal{R}^{(n-p)\times(n-p)}$, $C_1 = CW_1 \in \mathcal{R}^{m\times p}$, and $C_2 = CW_2 \in \mathcal{R}^{m\times(n-p)}$.

We now consider two cases, based on the relation between p, the number of controls, and m, the number of observed variables. *We assume in both cases that* C_1 *is full rank.*

3.1. Case I: $p > m$. As noted above, in this situation, there are more equations than unknowns, and in general, no solution exists.

The RQ factorization of the $m \times p$ matrix C_1 is

$$C_1 = [\hat{R}, 0] \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix},$$

where $Q_1 \in \mathcal{R}^{m \times p}$, $Q_2 \in \mathcal{R}^{(p-m) \times p}$, $\hat{R} \in \mathcal{R}^{m \times m}$, and $\hat{R}$ has rank m.

Now (8) gives us the relation

$$ZA_1Q^T = LC_1Q^T = [L\hat{R}, 0],$$

or, letting $A_1Q^T = G = [G_1, G_2]$,

$$ZG_1 = L\hat{R}, \qquad ZG_2 = 0.$$

Using this formula in (7), we obtain

$$ZA_2 - FZ = ZG_1\hat{R}^{-1}C_2,$$

or

$$ZG_2 = 0, \tag{9}$$

$$Z(A_2 - G_1\hat{R}^{-1}C_2) - FZ = 0. \tag{10}$$

We now have a problem in exactly the same form as the original equations (2) and (1), except that the Sylvester equation (10) is homogeneous. Further reduction proceeds exactly as above in order to find a change of variables that produces an unconstrained Sylvester equation. However, unless $A_2 - G_1\hat{R}^{-1}C_2$ and F have at least one common eigenvalue, the only solution to (10) is $Z = 0$.

We will not consider this case further.

3.2. Case II: $p \leq m$. Consider a QR factorization of the $m \times p$ matrix C_1 as

$$C_1 = [Q_1, Q_2] \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix},$$

where $Q_1 \in \mathcal{R}^{m \times p}$, $Q_2 \in \mathcal{R}^{m \times (m-p)}$, $\hat{R} \in \mathcal{R}^{p \times p}$, and $\hat{R}$ has rank p.

Now let

$$\hat{L} = LQ = [\hat{L}_1, \hat{L}_2],$$

where $\hat{L}_1 \in \mathcal{R}^{(n-m) \times p}$ and $\hat{L}_2 \in \mathcal{R}^{(n-m) \times (m-p)}$. From (8), we have that

$$ZA_1 = LC_1 = \hat{L} \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix},$$

so

$$\hat{L}_1 = ZA_1\hat{R}^{-1}.$$

Using this formula in (7), and letting

$$Q^TC_2 = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix},$$

we obtain

$$ZA_2 - FZ = [\hat{L}_1, \hat{L}_2]Q^T C_2 = [ZA_1\hat{R}^{-1}, \hat{L}_2] \begin{bmatrix} E_1 \\ E_2 \end{bmatrix},$$

or

$$Z(A_2 - A_1\hat{R}^{-1}E_1) - FZ = \hat{L}_2 E_2. \tag{11}$$

We have succeeded in reducing the original constrained Sylvester problem (2) and (1) to an unconstrained one through the change of variables $T = ZW_2^T$. The $(n-m)(m-p)$ entries of $\hat{L}_2$ are free parameters. For each choice of $\hat{L}_2$, (11) has a unique solution, as long as the matrices $\hat{A} \equiv A_2 - A_1\hat{R}^{-1}E_1$ and F have no common eigenvalues. Section 2 discusses the existence of the solution in the case of common eigenvalues.

Note. If C_1 fails to have full rank, then we have

$$C_1 = [Q_1, Q_2] \begin{bmatrix} \hat{R} & \hat{P} \\ 0 & 0 \end{bmatrix},$$

where $Q_1 \in \mathcal{R}^{m\times r}$, $Q_2 \in \mathcal{R}^{m\times(m-r)}$, $\hat{R} \in \mathcal{R}^{r\times r}$, $\hat{P} \in \mathcal{R}^{r\times(p-r)}$, and $\hat{R}$ has rank r. A derivation following the steps above leads to the Sylvester equation (11) but with the side constraint

$$Z(A_{12} - A_{11}\hat{R}^{-1}\hat{P}) = 0,$$

where A_{11} denotes the first r columns of A_1 and A_{12} denotes the remaining columns. Thus we reduced the problem to a smaller constrained Sylvester equation of the same form as the original, and the process needs to be repeated. □

4. The resulting algorithm. The following algorithm computes a solution to the constrained Sylvester problem (1) and (2).

$$TA - FT = LC, \tag{12}$$

$$TB = 0, \tag{13}$$

under the assumptions that $n > m > p$, $\operatorname{rank}(CB) = p$, and (redundantly) $\operatorname{rank}(B) = p$.

Step 1. Factor B into its QR factors

$$B = W \begin{bmatrix} S \\ 0 \end{bmatrix},$$

where $S \in \mathcal{R}^{p\times p}$ is full rank and $W \in \mathcal{R}^{n\times n}$ is an orthogonal matrix: $W^TW = I$. Partition W into its first p columns W_1 and its remaining $n-p$ columns W_2.

Step 2. Set $A_1 = W_2^T A W_1 \in \mathcal{R}^{(n-p)\times p}$, $A_2 = W_2^T A W_2 \in \mathcal{R}^{(n-p)\times(n-p)}$, $C_1 = CW_1 \in \mathcal{R}^{m\times p}$, and $C_2 = CW_2 \in \mathcal{R}^{m\times(n-p)}$.

Step 3. Perform a QR factorization of the $m \times p$ matrix C_1 as

$$C_1 = [Q_1, Q_2] \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix},$$

where $Q_1 \in \mathcal{R}^{m\times p}$, $Q_2 \in \mathcal{R}^{m\times(m-p)}$, $\hat{R} \in \mathcal{R}^{p\times p}$, and $\hat{R}$ has rank p.

Step 4. Let

$$Q^T C_2 = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix},$$

where $E_1 \in \mathcal{R}^{p\times(n-p)}$ and $E_2 \in \mathcal{R}^{(m-p)\times(n-p)}$.

Step 5. Solve the Sylvester matrix equation $Z(A_2 - A_1\hat{R}^{-1}E_1) - FZ = \hat{L}_2E_2$, where the entries of $\hat{L}_2$ are chosen randomly.

Step 6. Set $\hat{L}_1 = ZA_1\hat{R}^{-1}$ and $L = [\hat{L}_1, \hat{L}_2]Q^T$.

Step 7. Set $T = ZW_2^T$.

The software tasks needed to implement the algorithm are matrix multiplication, the QR factorization [3], and an algorithm for solving unconstrained Sylvester problems [1], [5]. The highest-order terms in the operation counts are cubic in n, m, and p, with constants depending on the specific choice of software. There is substantial potential parallelism in the computation, since there are well-known parallel algorithms for each of these basic tasks; see, for example, [9], [8], and the references therein.

For examples of applications of this algorithm to loop transfer recovery, see [7].

5. Necessary conditions and sufficient conditions for solutions to the full problem. In this section we develop some conditions that are necessary in order to obtain a solution T to the problem (1), (2), and (3). For ease of reference, we define

$$\mathcal{T} = \begin{bmatrix} T \\ C \end{bmatrix},$$

where $T \in \mathcal{R}^{(n-m)\times n}$ and $C \in \mathcal{R}^{m\times n}$.

Recall that we already assumed, without loss of generality, that $n > p$ and $\text{rank}(B) = p$. We will consider the case $p < m$, since the other case has a solution only under accidental conditions. We also restrict ourselves to the case in which F has no eigenvalues in common with the matrix $\hat{A} \equiv A_2 - A_1\hat{R}^{-1}E_1$ of (11). Under these circumstances, (1) and (2) always have a solution, and the only issue is the rank of $\mathcal{T}$.

Recall that W and Q are $n \times n$ orthogonal matrices. We note that $\mathcal{T}$ is full rank if and only if the matrices

$$\mathcal{T}W = \begin{bmatrix} T \\ C \end{bmatrix} W = \begin{bmatrix} 0 & Z \\ C_1 & C_2 \end{bmatrix}$$

and

$$Q^T\mathcal{T}W = \begin{bmatrix} 0 & Z \\ Q_1^TC_1 & E_1 \\ Q_2^TC_1 & E_2 \end{bmatrix} = \begin{bmatrix} 0 & Z \\ \hat{R} & E_1 \\ 0 & E_2 \end{bmatrix},$$

are full rank, and it is sometimes easier to work with these.

NECESSARY AND SUFFICIENT CONDITION 1. *For $\mathcal{T}$ to be full rank, it is necessary and sufficient that $Q_1^TC_1$ (or, equivalently, C_1) and $[Z^T, E_2^T]$ be full rank.*

Our goal is to express such conditions more obviously in terms of the data matrices A, B, C, and F.

NECESSARY CONDITION 1. *The matrix C must have full rank m.*

Proof. If $\alpha^T C = 0$ for some nonzero α, then $[0^T, \alpha^T]\mathcal{T} = 0$ and $\mathcal{T}$ is not full rank. □

NECESSARY CONDITION 2. *The system A, B, C must be regular* [2, p. 661], *i.e., the matrix CB must have full rank p.*

Note. This condition is also necessary and sufficient for the existence of a full rank triangular factor $\hat{R}$ for C_1. □

Proof. Recall from (5) that the first p columns of W span the range of B. For $\mathcal{T}W$ to be full rank, it is necessary that C_1 have full column rank p. Now, $C_1 = CW_1 = CBS^{-1}$, so it is necessary that CB be full rank. □

NECESSARY CONDITION 3. *The system must be observable, i.e., the only vector y satisfying $Ay = \mu y$ and $Cy = 0$ must be the vector $y = 0$.*

Note. If we add the assumption that A and F have no common eigenvalues, then this result is easy to prove. Suppose there is a nonzero y satisfying $Ay = \mu y$ and $Cy = 0$. Then, since $TA - FT = LC$,

$$TAy - FTy = (\mu I - F)Ty = LCy = 0,$$

so $Ty = 0$. This fact, along with $Cy = 0$, implies that $\mathcal{T}$ is rank deficient. If we avoid this extra assumption on the eigenvalues of A and F, then the outline of the proof is similar, but it must be done in terms of the reduced Sylvester equation (11). □

Proof. Suppose there is a nonzero y satisfying $Ay = \mu y$ and $Cy = 0$, and let

$$W^T y = \begin{bmatrix} y_1 \\ y_2 \end{bmatrix}$$

so that $y = W_1 y_1 + W_2 y_2$. Then

$$Ay = A(W_1 y_1 + W_2 y_2) = \mu(W_1 y_1 + W_2 y_2).$$

Multiplying by W_2^T and using the definitions following (8), we obtain

$$W_2^T A W_1 y_1 + W_2^T A W_2 y_2 = A_1 y_1 + A_2 y_2 = \mu y_2.$$

Now, y_2 is an eigenvector of $\hat{A}$, since

$$\begin{aligned} \hat{A}y_2 &= (A_2 - A_1 \hat{R}^{-1} Q_1^T C W_2) y_2 \\ &= \mu y_2 - A_1(y_1 + \hat{R}^{-1} Q_1^T C W_2 y_2) \\ &= \mu y_2 - A_1(y_1 + \hat{R}^{-1} Q_1^T C(y - W_1 y_1)) \\ &= \mu y_2 - A_1(y_1 + \hat{R}^{-1} Q_1^T C y - \hat{R}^{-1} Q_1^T C W_1 y_1) \\ &= \mu y_2, \end{aligned} \tag{14}$$

since $Cy = 0$ and $CW_1 = Q_1 \hat{R}$, so $Q_1^T C W_1 = \hat{R}$. Further,

$$E_2 y_2 = Q_2^T C W_2 y_2 = Q_2^T C(y - W_1 y_1) = Q_2^T C W_1 y_1 = 0,$$

since $Q_2^T C W_1 = 0$.

Consider the reduced Sylvester equation (11) $Z\hat{A} - FZ = \hat{L}_2 E_2$, and multiply by y_2:

$$Z\hat{A}y_2 - FZy_2 = \mu Z y_2 - FZy_2 = (\mu I - F)Zy_2 = \hat{L}_2 E_2 y_2 = 0,$$

so, since μ is not an eigenvalue of F, we must have $Zy_2 = 0$.

We have a vector y_2 that satisfies $E_2 y_2 = 0$ and $Zy_2 = 0$, so $Q^T TW$ is rank deficient. Thus, observability is necessary for a full rank $\mathcal{T}$. □

These necessary conditions are not sufficient, as shown by the following example.

Example. Let

$$A = \begin{bmatrix} -3 & 0 & -3 \\ 0 & 1 & 1 \\ -1 & 0 & -2 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

It is easy to see that C, B, and CB are all full rank. The eigenvectors of A are the columns of the matrix

$$\begin{bmatrix} 0 & 1.0000 & 1.0000 \\ 1 & -0.0819 & 0.4523 \\ 0 & 0.4343 & -0.7676 \end{bmatrix},$$

so the system is observable. We calculate

$$E_2 = \begin{bmatrix} 1 & 0 \end{bmatrix}, \qquad \hat{A} = \begin{bmatrix} 1 & 0 \\ 0 & -3 \end{bmatrix},$$

so in using the decompositions in §2 to solve the reduced Sylvester equation of Step 5 we obtain

$$\hat{C} = \hat{L}_2 \begin{bmatrix} 1 & 0 \end{bmatrix}.$$

Since this matrix has a zero column regardless of the choice of $\hat{L}_2$, the solution matrix will as well, and we will have a rank deficient $\mathcal{T}$. □

NECESSARY CONDITION 4. *The reduced system must be observable, i.e., the only vector y satisfying $\hat{A}y = \mu y$ and $E_2 y = 0$ must be the vector $y = 0$.*

Proof. The proof of this result, motivated by the example above, follows from the discussion at the end of §2. □

If Necessary Conditions 1–4 are satisfied, then we conjecture that there exists a choice of the matrices $\hat{L}_2$ and F so that the algorithm yields a solution to the constrained Sylvester problem (1), (2) satisfying the full rank condition (3). The first two necessary conditions guarantee the existence of full rank triangular factors for B in Step 1 and C_1 in Step 3. Some freedom in the choice of F is needed so that its eigenvalues are distinct from those of $\hat{A}$, guaranteeing the existence of a solution of the reduced Sylvester equation in Step 5. The remaining freedom in F, the other two necessary conditions, and freedom in the choice of $\hat{L}_2$ in Step 5 can be used in satisfying the condition that $[Z^T, E_2^T]$ be full rank. In practice, of course, the eigenvalues of any given matrix F will virtually always be distinct from those of A, and thus the algorithm will successfully compute a solution to the constrained problem, although it may not be possible to satisfy the full rank condition.

See [7] for numerical computations using this algorithm in control design of a flexible arm, helicopter flight, and aircraft flight dynamics.

Acknowledgments. We are grateful to the referees for their helpful comments.

REFERENCES

[1] R. H. BARTELS AND G. W. STEWART, *Algorithm 432: Solution of the matrix equation $AX + XB = C$*, Comm. ACM, 15 (1972), pp. 820–826.

[2] J. J. D'Azzo and C. H. Houpi, *Linear Control System Analysis and Design: Conventional and Modern*, McGraw-Hill, New York, 1988.

[3] J. J. Dongarra, C. B. Moler, J. R. Bunch, and G. W. Stewart, *LINPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[4] F. R. Gantmacher, *The Theory of Matrices*, Chelsea, New York, 1977.

[5] G. H. Golub, S. Nash, and C. Van Loan, *A Hessenberg–Schur method for the problem* $AX + XB = C$, IEEE Trans. Automat. Control, AC-24 (1979), pp. 909–913.

[6] A. Liapounoff, *Problème Général de la Stabilité du Mouvement*, in Annals of Mathematical Studies 17, Princeton University Press, Princeton, NJ, 1947.

[7] M. M. Monahemi, J. B. Barlow, and D. P. O'Leary, *The design of reduced-order Luenberger observers with precise* LTR, Tech. Report TR-2600, UMIACS TR-91-17, University of Maryland, Computer Science Department, Baltimore, MD, January 1991.

[8] D. P. O'Leary and G. W. Stewart, *Data-flow algorithms for parallel matrix computations*, Comm. ACM, 28 (1985), pp. 840–853.

[9] D. P. O'Leary and P. Whitman, *Parallel* QR *factorization by Householder and modified Gram–Schmidt algorithms*, Parallel Comput., 16 (1990), pp. 99–112.

[10] J. J. Sylvester, *Sur l'équation en matrices* $px = xq$, in Comptes Rendus de l'Académie des Sciences, 1884, pp. 67–71. Reprinted in Collected Mathematical Papers of James Joseph Sylvester, Cambridge, 1912, IV, pp. 176-180.

[11] C.-C. Tsui, *A new approach to robust observer design*, Internat. J. Control, 47 (1988), pp. 745–751.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 10–19, January 1992

002

THE COMPONENTWISE DISTANCE TO THE NEAREST SINGULAR MATRIX*

JAMES DEMMEL†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. The singular value decomposition of a square matrix A answers two questions. First, it measures the distance from A to the nearest singular matrix, measuring distance with the two-norm. It also computes the condition number, or the sensitivity of A^{-1} to perturbations in A, where sensitivity is also measured with the two-norm. As is well known, these two quantities, the minimum distance to singularity and the condition number, are essentially reciprocals. Using the algorithm of Golub and Kahan [*SIAM J. Numer. Anal.*, Ser. B, 2 (1965), pp. 205–224] and its descendants, these quantities may be computed in $O(n^3)$ operations. More recent sensitivity analysis extends this analysis to perturbations of different maximum sizes in each entry of A. One may again ask about distance to singularity, condition numbers, and complexity in this new context. It is shown that there can be no simple relationship between distance to singularity and condition number, because the condition number can be computed in polynomial time and the distance to singularity is NP-complete. Nonetheless, there are some useful inequalities relating the two, especially in the case of componentwise relative perturbations.

Key words. distance to singularity, conditioning, singular value decomposition

AMS(MOS) subject classifications. 65F35, 15A12

1. Introduction.

The singular value decomposition (SVD) of a real n by n matrix A is written $A = U\Sigma V^T$, where U is n by n and orthogonal, V is n by n and orthogonal, $\Sigma = \mathrm{diag}(\sigma_1, \cdots, \sigma_n)$ is n by n and diagonal, and $\sigma_1 \geq \cdots \geq \sigma_n \geq 0$. It is well known that σ_n is the distance from A to the nearest singular matrix, where we measure distance either in the two-norm $\|\cdot\|_2$ or the Frobenius (Euclidean) norm $\|\cdot\|_F$. Furthermore, the condition number of A with respect to inversion is $\kappa(A) \equiv \|A\|_2 \cdot \|A^{-1}\|_2 = \sigma_1/\sigma_n$. This means that a small perturbation δA of A causes a small perturbation of A^{-1} bounded by

$$\frac{\|(A+\delta A)^{-1} - A^{-1}\|_2}{\|A^{-1}\|_2} \leq \kappa(A) \cdot \frac{\|\delta A\|_2}{\|A\|_2} + O(\|\delta A\|_2^2).$$

Note that if we normalize A to have unit norm $\|A\|_2 = 1$, then the condition number and smallest singular value are reciprocals. The first successful algorithm for computing the SVD was described by Golub and Kahan in [13] and further developed in [5]–[7], [14], [10]; it can compute the SVD stably in $O(n^3)$ floating point operations.

More recent perturbation theory and error analysis have tried to extend this analysis by considering structured linear systems, where δA is no longer bounded by a "round ball" of the form $\|\delta A\|_2 \leq \omega$ or $\|\delta A\|_F \leq \omega$. Instead, we bound δA in terms of a fixed nonnegative matrix E and the smallest scalar ω satisfying

$$|\delta A_{ij}| \leq \omega E_{ij}$$

for all i and j. We will write this set of inequalities in the shorthand $|\delta A| \leq \omega E$, where $|\delta A|$ is the matrix of absolute entries of δA and the inequality is interpreted

* Received by the editors February 11, 1991; accepted for publication (in revised form) May 8, 1991. This research was supported by National Science Foundation grants DCR-8552474 and ASC-8715728. This author is also a Presidential Young Investigator.

† Mathematics Department and Computer Science Division, University of California, Berkeley, California 94720 (demmel@cs.berkeley.edu).

componentwise. Another way to describe ω is

$$\omega = \max_{ij} \frac{|\delta A_{ij}|}{E_{ij}}$$

where $0/0$ is interpreted as 0. Thus we can think of ω as a scaled infinity-norm of δA, interpreting δA as a vector of dimension n^2.

Given this componentwise way to measure δA, we can ask four related questions, motivated by analogous properties of the SVD:

(1) How big can $\|(A+\delta A)^{-1} - A^{-1}\|/\|A^{-1}\|$ be as a multiple $\kappa(A,E)\cdot\omega$ of ω for small ω? $\kappa(A,E)$ is called the *condition number of A with respect to E.*

(2) What is the smallest ω for which there is a δA satisfying $|\delta A| \leq \omega E$ and $A+\delta A$ is singular? Denote this minimum value of ω by $\underline{\omega}(A,E)$. $\underline{\omega}(A,E)$ is the distance from A to the nearest singular matrix in the metric defined by E.

(3) What is the complexity of computing $\kappa(A,E)$ and $\underline{\omega}(A,E)$?

(4) How are $\kappa(A,E)$ and $\underline{\omega}(A,E)$ related? In particular, are they reciprocals?

The first question was answered by Bauer in 1966 and Skeel in 1979–80 [4], [21], [22]; we survey this material in §2 of the paper. In particular, we show that $\kappa(A,E)$ can be reliably computed in $O(n^3)$ time (or even less if fast matrix multiplication/inversion techniques due to Strassen, Pan, Winograd, and others are used [17]).

The second question was answered by Rohn in 1989 [18]; we discuss this result in §3. The answer is a simple expression involving 4^n eigenproblems of dimension n.

A more recent result of Rohn and Poljak answers question (3); they show that computing $\underline{\omega}(A,E)$ is NP-complete, even in the apparently simple case where $E_{ij} = 1$. We discuss this result in §4, and show that their result implies that any numerical algorithm that computes a sufficiently accurate approximation to $\underline{\omega}(A,E)$ must be able to solve a particular NP-complete problem exactly. Thus, unless P=NP, which is widely believed to be false, computing $\underline{\omega}(A,E)$ will take an exponential amount of time.

We now have a simple answer to question (4) above. Since $\kappa(A,E)$ can be computed in polynomial time (in fact $O(n^3)$), and computing $\underline{\omega}(A,E)$ is NP-complete, there cannot be any simple (i.e., polynomial) way to compute one from the other. In particular, they cannot be reciprocals.

Nonetheless, there are some simple inequalities relating $\kappa(A,E)$ and $\underline{\omega}(A,E)$, especially for the particular value $E = |A|$. This corresponds to relative perturbations in each entry of A, and is the value of E of most interest in practical computations [4], [21], [22], [1], [2], [15]. In §5 we show that the maximum value of $(\kappa(AD, |AD|))^{-1}$ over all nonsingular diagonal D bounds $\underline{\omega}(A,|A|)$ from below and cannot be much smaller, possibly unless A is close to a certain set of codimension 4. We prove they are close in a variety of special cases, and conjecture that they are always close.

2. Bauer and Skeel's perturbation theory. We consider the system of linear equations $Ax = b$ and the perturbed system $(A+\delta A)(x+\delta x) = b$. Subtracting these two equations and rearranging yields

$$\delta x = A^{-1} \cdot \delta A \cdot (x + \delta x).$$

Applying the triangle inequality yields

$$|\delta x| \leq |A^{-1}| \cdot |\delta A| \cdot |x + \delta x| \leq |A^{-1}| \cdot \omega E \cdot |x + \delta x|$$

and taking norms on both sides and dividing by $\| \, |x + \delta x| \, \|$ yields

$$\frac{\| \, |\delta x| \, \|}{\| \, |x + \delta x| \, \|} \leq \omega \| \, |A^{-1}| \cdot E \, \|.$$

If $\| \cdot \|$ denotes the infinity norm, we can show that this inequality is attainable. More generally, it is only attainable to within a constant depending on the choice of vector norm. This justifies the definition

$$\kappa(A, E) \equiv \| \, |A^{-1}| \cdot E \, \|.$$

Later in this paper we will use several different norms $\| \cdot \|$.

3. Rohn's formula for the distance to a singular matrix. To state Rohn's result [18], [19], we need some notation. S_1 and S_2 will denote *signature matrices*, i.e., any diagonal matrices with diagonal entries ± 1. Note that there are 2^n signature matrices of dimension n. $\rho_0(X)$ denotes the *real spectral radius* of X:

$$\rho_0(X) = \max\{|\lambda| : \lambda \text{ is a real eigenvalue of } X \}$$

or 0 if X has no real eigenvalues. Now we may state Rohn's theorem.

THEOREM 3.1 (see [18]).

$$\underline{\omega}(A, E) = \frac{1}{\max_{S_1, S_2} \rho_0(S_1 A^{-1} S_2 E)}$$

where the maximum is taken over all pairs of signature matrices.

In particular, consider the case $E_{ij} = 1$. Then $E = uu^T$ where u is the column vector of all ones, and

$$\rho_0(S_1 A^{-1} S_2 E) = \rho_0(S_1 A^{-1} S_2 uu^T) = \rho_0(u^T S_1 A^{-1} S_2 u) = |u^T S_1 A^{-1} S_2 u|,$$

so

$$\underline{\omega}(A, E) = \frac{1}{\max_{S_1, S_2} |u^T S_1 A^{-1} S_2 u|} = \frac{1}{\max_{|x|=|y|=u} |x^T A^{-1} y|}$$

where the maximum is over all pairs of vectors, each of which has entries ± 1. Note the close resemblance of this to the formula for the smallest singular value; the only difference is that we take the maximum over all pairs of unit vectors $\|x\|_2 = \|y\|_2 = 1$.

4. Rohn and Poljak's NP-completeness theorem. In [20] it is shown that computing $\max_{|x|=|y|=u} x^T B y$ is NP-complete. In other words, unless P = NP, there is no significantly cheaper algorithm than simply computing all possible $x^T B y$. Their reduction is to the max-cut problem, which we outline here.

Let $G = (N, E)$ be a weighted graph with nodes N, edges E and edge weights $w : E \to \mathbf{R}$. Let $S \subset N$ and define the *cut* δS as the set of all edges with one endpoint in S and one not in S. Let $w(\delta S)$ be the sum of the weights of all edges in δS. The *max-cut* problem is to find S maximizing $w(\delta S)$. Given any positive integer m, it is NP-complete to decide if there is an S such that $w(\delta S)$ is at least m; this is true even if each edge has weight 1, so that $w(\delta S)$ is just the cardinality of δS [12].

Rohn and Poljak reduce max-cut to computing $\max_{|x|=|y|=u} x^T B y$ as follows. Given $G = (N, E)$ and $w \equiv 1$, $n =$ the cardinality of N, and $e =$ the cardinality of E, represent N by the integers from 1 to n. Define the matrix B by

$$B_{ij} = \begin{cases} 0 & \text{if } (i,j) \notin E, \quad i \neq j, \\ -1 & \text{if } (i,j) \in E, \quad i \neq j, \\ r & \text{if } i = j. \end{cases}$$

Here r is any integer exceeding $2e$; this guarantees that the matrix B is diagonally dominant and in fact well conditioned if we want. Then Rohn and Poljak show that if MC is the value of the max-cut, then

$$\max_{|x|=|y|=u} x^T B y = 2 \cdot MC + r \cdot n - e.$$

They do this as follows: since B is diagonally dominant, $x = y$ when the maximum of $x^T B y$ is attained. Thus

$$\begin{aligned} y^T B y &= \sum_{ij} b_{ij} y_i y_j = -\frac{1}{2} \sum_{ij} b_{ij} (y_i - y_j)^2 + \sum_{ij} b_{ij} \\ &= 2 \cdot card(\delta S) + r \cdot n - e, \end{aligned}$$

where $card(\delta S)$ is the cardinality of the cut set defined by $S = \{i : y_i = 1\}$. Maximizing on both sides yields the result.

This reduces the NP-complete problem max-cut to $\max_{|x|=|y|=u} x^T B y$. A similar reduction works in the opposite direction.

Now we show that any sufficiently accurate numerical algorithm for computing $\underline{\omega}(A, uu^T) = 1/\max_{|x|=|y|=u} x^t A^{-1} y$ would have to do an exponential amount of work, unless P=NP. From the previous discussion, we see that by taking $r = 4e \leq 4n^2$ in B, for example, we can guarantee that the usual condition number of B is less than 3. Letting $A = B^{-1}$, we see that each entry of $A^{-1} = B$ can be computed quite accurately, say, with an error $\sum_{ij} |\delta B_{ij}|$ bounded by $1/6$. (This requires accuracy increasing with n, but suffices to prove the point.) Suppose that the algorithm for $\max_{|x|=|y|=u} x^t B y$ is stable in the sense that it computes the correct answer to within $\pm 1/6$ for a slightly perturbed matrix $B + \delta B'$ with $\sum_{ij} |\delta B'_{ij}| < 1/6$. Then since

$$|\max_{|x|=|y|=u} x^t (B + \delta B') y - \max_{|x|=|y|=u} x^t B y| \leq \sum_{ij} |\delta B'_{ij}| < 1/6,$$

we finally get that the error in the computed value of $\max_{|x|=|y|=u} x^T A^{-1} y$ is less than $1/6 + 1/6 + 1/6 = 1/2$. Since the true value is an integer, we can round to the nearest integer to get it exactly. Thus we can solve an NP-complete problem exactly, as claimed.

Note that this also proves the more general result that there is no simple (polynomial) function yielding $\underline{\omega}(A, E)$ given any data computable in polynomial time from the entries of A and E.

5. Inequalities relating $\kappa(A, E)$ and $\underline{\omega}(A, E)$. Even though there can be no simple formula relating $\kappa(A, E)$ and $\underline{\omega}(A, E)$, we may still seek to relate them by inequalities. The following proposition gives a simple one-sided inequality.

PROPOSITION 5.1. *Let $\kappa(A, E) = \| \, |A^{-1}| \cdot E \, \|$, where $\| \cdot \|$ is any operator norm. Let $\rho(\cdot)$ denote the spectral radius. Then*

$$\underline{\omega}(A, E) \geq \frac{1}{\rho(|A^{-1}| \cdot E)} \geq \frac{1}{\kappa(A, E)}.$$

If A^{-1} has a checkerboard sign pattern, the first inequality is an equality.

Proof. The Perron–Frobenius theorem implies that

$$\rho_0(S_1 A^{-1} S_2 E) \leq \rho(|A^{-1}| \cdot E) \leq \| \, |A^{-1}| \cdot E \, \|.$$

Now apply Rohn's theorem. The first inequality is attainable if A^{-1} has a checkerboard sign pattern. □

We give another more elementary proof of the second inequality in the theorem in the special case of norms satisfying the additional properties

$$\|X\| \leq \| \, |X| \, \| \quad \text{and} \quad 0 \leq X \leq Y \Rightarrow \|X\| \leq \|Y\|.$$

Choose any $\omega < (\kappa(A,E))^{-1}$. Then if $|\delta A| \leq \omega E$, we have

$$\|A^{-1}\delta A\| \leq \| \, |A^{-1}| \cdot |\delta A| \, \| \leq \omega \| \, |A^{-1}|E \, \| < 1,$$

implying that $A + \delta A = A(I + A^{-1}\delta A)$ is invertible. Thus $(\kappa(A,E))^{-1} \leq \underline{\omega}(A,E)$.

Since we can find matrices whose spectral radius and norm differ arbitrarily, we cannot expect to bound $\underline{\omega}(A,E)$ from above by any function of $\kappa(A,E)$. For example, let

$$A = \begin{bmatrix} \epsilon & 1 \\ 0 & 1 \end{bmatrix}$$

and $E = |A|$. In this case $\rho(|A^{-1}| \cdot E) = 1$ and $\| \, |A^{-1}| \cdot E \, \| \approx 2/\epsilon$.

This last example makes it clear that to bound $\underline{\omega}(A,E)$ from above by some function of $\kappa(A,E)$, we need to have E depend on A in some way. We begin by noting that if D_1 and D_2 are arbitrary nonsingular diagonal matrices, then

$$\underline{\omega}(A,E) = \underline{\omega}(D_1AD_2, D_1ED_2)$$

since $|\delta A| \leq \omega E$ if and only if $|D_1\delta AD_2| \leq \omega |D_1ED_2|$ and $A+\delta A$ is singular if and only if $D_1AD_2 + D_1\delta AD_2$ is singular. Similarly, $\rho(|A^{-1}| \cdot E) = \rho(|(D_1AD_2)^{-1}| \cdot |D_1ED_2|)$. On the other hand, although $\kappa(A,E)$ is independent of row scaling by D_1, since

$$\kappa(A,E) = \| \, |A^{-1}| \cdot E \, \| = \| \, |(D_1A)^{-1}| \cdot (D_1E) \, \| = \kappa(D_1A, D_1E),$$

it is not independent of column scaling by D_2, as the last example shows.

By letting the column scaling D_2 vary, we can prove the following theorem.

THEOREM 5.2. *Let $\| \cdot \|$ denote any p-norm. Then*

$$\underline{\omega}(A,E) \geq \frac{1}{\rho(|A^{-1}|E)} = \frac{1}{\min_{D_2} \kappa(AD_2, ED_2)}.$$

If $|A^{-1}| \cdot E$ is irreducible, the minimizing D_2 can be given as follows. Let x and y^T be the right and left Perron vectors of $|A^{-1}| \cdot E$, respectively, and let q satisfy $1/p + 1/q = 1$. Then $D_{2,ii} = y_i^{-1/p} \cdot x_i^{1/q}$.

Proof. The proof is a variation on the proof of Lemma I in [3]. There Bauer shows that for $B > 0$ and $C > 0$,

$$\min_{D_1, D_2} (\|D_1BD_2\| \cdot \|D_2^{-1}CD_1^{-1}\|) = \rho(BC). \tag{1}$$

Let $x_1 > 0$ and $y_1^T > 0$ be the right and left Perron vectors, respectively, of BC, and $x_2 > 0$ and $y_2^T > 0$ the corresponding Perron vectors of CB. Then Bauer shows that the minimizing D_1 and D_2 are given by $D_{1,ii} = y_{1i}^{1/p} x_{1i}^{-1/q}$ and $D_{2,ii} = y_{2i}^{-1/p} x_{2i}^{1/q}$.

We note that Bauer's argument depends only on the positivity of the components of x_i and y_i, which according to the Perron–Frobenius theorem is guaranteed just by

the irreducibility of BC and CB. In particular, we can take $C = I$, as long as B is irreducible.

Now consider the minimization problem (1) subject to the side constraint $D_1 = D_2^{-1}$. Along with the choice $C = I$ this changes our minimization problem into $\min_{D_2} \|D_2^{-1} B D_2\|$. This means that the minimum can be no smaller than $\rho(BC) = \rho(B)$. But the choice $C = I$ means $x_1 = x_2$ and $y_1 = y_2$, so from the formulas for the unconstrained minimizing D_1 and D_2, we see that $D_1 = D_2^{-1}$ and so the constrained and unconstrained problems attain the same minimum.

If $B = |A^{-1}| \cdot E$ is irreducible, this proves the theorem. If $B = |A^{-1}| \cdot E$ is reducible, we can reduce to the irreducible case as follows. Assume without loss of generality that B has been permuted to block upper triangular form with blocks B_{ij} such that the diagonal blocks B_{ii} are irreducible (the diagonal blocks correspond to the strongly connected components of the graph of B). It is possible to choose D_2' so that $D_2'^{-1} B D_2'$ has the same diagonal blocks as B but the off-diagonal blocks are as small as we like. Now we can choose D_2'' to minimize the norm of each diagonal block of $(D_2' D_2'')^{-1} B (D_2' D_2'')$ individually. This makes the overall norm as close to $\rho(B)$ as desired. Note that there is not necessarily any D_2 that attains the minimum in the case where two diagonal blocks of B have spectral radius $\rho(B)$. □

This proof works for a wider class of norms than the p-norms; see [3] for details.

We do not know how weak the inequality in Theorem 5.2 can be. We can get an upper bound for $\underline{\omega}(A, E)$ based on perturbing one component of A at a time.

PROPOSITION 5.3.

$$\frac{1}{\max_{ij} |A_{ij}^{-1}| \cdot E_{ji}} \geq \underline{\omega}(A, E).$$

Let A be n by n. Then the upper bound is always within a constant of the trace of $|A^{-1}| \cdot E$:

$$\frac{n^2}{\operatorname{tr}(|A^{-1}| \cdot E)} \geq \frac{1}{\max_{ij} |A_{ij}^{-1}| \cdot E_{ji}} \geq \frac{1}{\operatorname{tr}(|A^{-1}| \cdot E)}.$$

Proof. The perturbation $A_{ij} + \delta A_{ij}$ of A_{ij} that make $A + \delta A$ singular is precisely $(A_{ji}^{-1})^{-1}$; this follows from block Gaussian elimination as in [8]. This corresponds to $\omega = 1/(|A_{ji}^{-1}| \cdot E_{ij})$ in $|\delta A| \leq \omega E$. Minimizing ω over all i and j produces the upper bound. The relationship with $\operatorname{tr}(|A^{-1}|E)$ is elementary. □

This upper bound can be arbitrarily weak, as can be seen from the example

$$A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad E = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Here both off-diagonal entries of A must be perturbed simultaneously to make it singular.

To make further progress, we specialize to the case $E = |A|$. This corresponds to making relative perturbations in each entry of A, and is the choice of E of most interest in numerical applications [4], [21], [22], [1], [2], [15]. We will present a number of special cases where the inequality in Theorem 5.2 is nearly sharp, i.e., where $\underline{\omega}(A, |A|)$ is bounded above by a constant multiple of $1/\rho(|A^{-1}| \cdot |A|)$. In fact, we will show that it is only possible for $\underline{\omega}(A, |A|)$ to greatly exceed $1/\rho(|A^{-1}| \cdot |A|)$ in case A is close to a set of codimension 4, a very thin set indeed. Based on this, we conjecture that $\underline{\omega}(A, |A|)$ is always bounded above by a constant multiple of $1/\rho(|A^{-1}| \cdot |A|)$.

THEOREM 5.4. *The inequality*

$$\frac{\gamma}{\rho(|A^{-1}| \cdot |A|)} \geq \underline{\omega}(A, |A|) \geq \frac{1}{\rho(|A^{-1}| \cdot |A|)} \tag{2}$$

holds in the following cases:

(1) *For 2 by 2 matrices with* $\gamma = 4$;
(2) *For triangular matrices with* $\gamma = 1$, *in which case* $\underline{\omega}(A, |A|) = 1/\rho(|A^{-1}| \cdot |A|) = 1$;
(3) *For n by n symmetric positive definite matrices with* $\gamma = n^2$;
(4) *For n by n matrices with* $1 \geq |A_{ij}| \geq \eta > 0$ *with* $\gamma = n^2/\eta$.

Remark. Theorem 5.4 (3) also appeared in [9].

Proof.

(1) If

$$A = \begin{bmatrix} a & b \\ c & d \end{bmatrix},$$

then a straightforward computation shows that the upper bound in Proposition 5.3 is given by $\max(|ad|, |bc|)/|ad-bc|$, and the lower bound by $1/\rho(|A^{-1}| \cdot |A|) = (\sqrt{ad} + \sqrt{bc})^2/|ad - bc|$. It is easy to verify that the upper bound is never more than four times the lower bound.

(2) To make an upper triangular matrix singular without perturbing below the diagonal, a diagonal entry must be made equal to 0. This means $\underline{\omega}(A, |A|) = 1$. Clearly, $|A^{-1}| \cdot |A|$ is also triangular with unit diagonal, so all its eigenvalues are 1.

(3) Write $A = DXD$ where D is diagonal with $D_{ii} = A_{ii}^{1/2}$ and $X_{ii} = 1$. Then $\rho(|A^{-1}| \cdot |A|) = \rho(|X^{-1}| \cdot |X|)$. Since the entries of X are bounded by 1 and the largest entry of X^{-1} is on its diagonal, we have $\max_{ij} |X_{ij}^{-1}| \cdot |X_{ji}| = \max_k X_{kk}^{-1}$. Also,

$$\rho(|X^{-1}| \cdot |X|) \leq \| \, |X^{-1}| \cdot |X| \, \|_\infty \leq \|X^{-1}\|_\infty \|X\|_\infty \leq n^2 \cdot \max_k X_{kk}^{-1}$$

yielding the result.

(Note that $\max_{kk} X_{kk}^{-1}$ is approximately the usual 2-norm condition number of X, and so is approximately the condition number for the eigenproblem for A, where we measure errors in eigenvalues by their relative error [11]. Thus we see that $\underline{\omega}(A, |A|)$ is approximately the reciprocal of the condition number for the eigenproblem for A.)

(4) It is easy to verify that $\rho(|A^{-1}| \cdot |A|) \leq \|A^{-1}\|_\infty \|A\|_\infty \leq \|A^{-1}\|_\infty n$ and $\max_{ij} |A_{ij}^{-1}| \cdot |A_{ji}| \geq \eta \max_{ij} |A_{ij}^{-1}| \geq \eta \|A^{-1}\|_\infty / n$. Combining these two inequalities yields the result. □

To proceed, let us without loss of generality restrict ourselves to the compact set of matrices $\mathcal{S}$ where $\|A\|_F = 1$. Let $\mathcal{N}$ be the subset of $\mathcal{S}$ consisting of nonsingular matrices; $\mathcal{N}$ is an open dense subset of $\mathcal{S}$. The existence of a constant γ so that (2) is true for all matrices is clearly implied by

$$f(A) \equiv \rho(|A^{-1}| \cdot |A|)/\mathrm{tr}(|A^{-1}| \cdot |A|) \tag{3}$$

being a bounded function on $\mathcal{N}$, because if it is bounded by μ, say, then we can take $\gamma = n^2\mu$ by Proposition 5.3. Since $|A^{-1}| \cdot |A| \geq I$, the denominator of $f(A)$ is bounded

below by n, and the only way $f(z)$ could be unbounded is for the numerator to be unbounded. Since A has Frobenius norm 1, this means A^{-1} must be unbounded. Thus, we can restrict our search for a bound $f(z) \leq \mu$ to a small neighborhood of the singular matrices.

The next theorem shows that in the generic case, that is, when we are close to a matrix of rank $n-1$ (in addition to a kind of irreducibility criterion), we expect $f(A)$ to be very close to 1. Thus if our upper and lower bounds on $\underline{\omega}(A, |A|)$ are ever very far apart, they must be near a set of codimension at least $\min(n, 4)$.

THEOREM 5.5. *Let A have rank $n-1$, $n \geq 3$, with right and left unit null vectors v and u^t: $Av = 0$ and $u^t A = 0$. Suppose $|u^t| \cdot |A| \cdot |v| \neq 0$. Then as $\delta A \to 0$ we have $f(A + \delta A) \to 1$. Thus $f(B)$ can only become unbounded (if it does at all) when B is close to a variety of codimension* $\min(n, 4)$.

Proof. Let $A + \delta A = U \Sigma V^t = \sum_{i=1}^{n} \sigma_i u_i v_i^t$ be the singular value decomposition of A, where $\|\delta A\|_2$ is small. Then $\sigma_n \leq \|\delta A\|_2$, the other σ_i are much larger, the last column u_n of U is close to u, and the last column v_n of V is close to v. Also, $A^{-1} = \sum_{i=1}^{n} \sigma_i^{-1} v_i u_i^t = \sigma_n^{-1} v u^t + O(\sigma_{n-1}^{-1})$. Thus

$$|A^{-1}| \cdot |A| = \sigma_n^{-1} |v| \cdot |u^t| \cdot |A| + O(\sigma_{n-1}^{-1}).$$

Now consider the term $\sigma_n^{-1} |v| \cdot |u^t| \cdot |A|$, which has rank 1. Since it has rank 1 and is nonnegative, its trace and spectral radius are identical: $\sigma_n^{-1} |u^t| \cdot |A| \cdot |v|$. By assumption, $|u^t| \cdot |A| \cdot |v|$ is nonzero, so for tiny δA, $|A^{-1}| \cdot |A|$ is a small perturbation of a nonnegative rank 1 matrix, and so $f(A + \delta A)$ is close to 1.

This argument breaks down either when A is nearly rank $n-2$, or $|u^t| \cdot |A| \cdot |v| = 0$. It is easy to verify that the set of matrices of rank at most $n-2$ has codimension 4. If $|u^t| \cdot |A| \cdot |v| = 0$, then either a row or column of A is zero (a set of codimension n), or at least four entries of A are zero (corresponding to the nonzero entries of u and v). □

Finally, we consider 3 by 3 matrices. If a nonzero 3 by 3 matrix is close to singular, then either it is close to rank 2 but far from rank 1, or its inverse has this property. This is enough to prove that $f(A)$ in (3) is bounded for all nonsingular 3 by 3 A.

THEOREM 5.6. *Let A be 3 by 3. Then there is a constant γ such that*

$$\frac{\gamma}{\rho(|A^{-1}| \cdot |A|)} \geq \underline{\omega}(A, |A|) \geq \frac{1}{\rho(|A^{-1}| \cdot |A|)}.$$

Proof. We wish to show that $f(A)$ is bounded for all nonsingular 3 by 3 A. We begin by restricting the set of nonsingular matrices for which we need to prove the result. In particular, we show that we can consider without loss of generality the bounded set $\mathcal{C}$ of matrices where each entry is in the range $[-1, 1]$ and each row, column, and 2 by 2 minor contains at least one ± 1.

If a matrix A is nonsingular, all its rows and all its columns are nonzero. By dividing each row of such a matrix by its largest entry, and each column of the resulting matrix by its largest entry, we get a new matrix where each row and each column contains at least one ± 1. We can further guarantee that no 2 by 2 minor of A fails to contain a ± 1. Suppose without loss of generality that the upper left 2 by 2 submatrix has largest absolute entry $a < 1$. Then entries (3,1), (3,2), (1,3), and (2,3) must all be ± 1. Now multiply rows 1 and 2 by $1/a$ and column 3 by a; this leaves the ± 1's in column 3 unchanged and introduces a ± 1 in the upper left 2 by 2 submatrix. All these transformations consist of diagonal scalings leaving $f(A)$

unchanged. Therefore, every nonsingular A has a representative in $\mathcal{C}$ with the same value of f, so it suffices to prove that f is bounded on $\mathcal{C}$.

Now let $\mathcal{K}$ be the closure of $\mathcal{C}$. $\mathcal{K}$ is a compact set with $\mathcal{K} - \mathcal{C}$ consisting of singular matrices. Next, we show that for every rank 2 matrix A in $\mathcal{K}$ we must have $|u^t| \cdot |A| \cdot |v| \neq 0$, in the notation of Theorem 4. We argue by contradiction. First, $z^t \equiv |u^t| \cdot |A|$ cannot be zero unless one or more rows of A is zero, which is impossible. Now $z^t|v|$ can be zero if and only each $z_i v_i = 0$. Clearly not every z_i can be nonzero, because then v would be 0. If two z_i were nonzero, only one v_i could be zero, implying that A had a zero column, a contradiction. So z must have exactly one nonzero component, and v at most two nonzero components. Since v cannot have exactly one nonzero component (which would imply A had a zero column), it must have exactly two nonzero components. The same reasoning shows that u has exactly two nonzero components. But if $|u^t| \cdot |A| \cdot |v| = 0$ and each of u and v has two nonzero components, A must have a 2 by 2 minor exactly 0, a contradiction. So $|u^T| \cdot |A| \cdot |v|$ cannot be zero.

By Theorem 5.5 this means around every rank 2, $A \in \mathcal{K}$, we can put a small ball such that for any $A' \in \mathcal{C}$ and in the ball, $f(A')$ is close to 1.

Now consider a rank 1 matrix $A \in \mathcal{K}$. We claim that all the entries of A are ± 1: Write $A = xy^t$, and suppose $A_{11} = x_1 y_1 = 1$. Then since each entry of A is bounded by 1 in absolute value, we must have $|x_i| \leq |x_1|$ and $|y_i| \leq |y_1|$ for $i = 1, 2$. So the only way to have ± 1's in the other rows and columns is to have all $|x_i|$ identical and all $|y_i|$ identical. Now by Theorem 5.4(4), $f(A')$ must be bounded for all $A' \in \mathcal{C}$ in a small ball surrounding A.

We have now constructed an open cover of the rank 2 and rank 1 matrices in $\mathcal{K}$ with the property that f is bounded on $\mathcal{C}$ intersected with their union. Around each $A \in \mathcal{C}$ we can also find an open set within which f is bounded, since f is continuous and finite on $\mathcal{C}$. All these sets taken together form an open cover of the compact set $\mathcal{K}$, so by the Heine–Borel theorem there is a finite subcover. Taking the maximum of f on this finite collection of sets yields a finite bound for f on all of $\mathcal{C}$. □

The evidence presented here leads us to make the following conjecture, which has also been made by Higham [16].

CONJECTURE. *There is a constant γ, possibly depending on dimension, such that*

$$\frac{\gamma}{\rho(|A^{-1}| \cdot |A|)} \geq \underline{\omega}(A, |A|) \geq \frac{1}{\rho(|A^{-1}| \cdot |A|)}$$

for all nonsingular A. Since

$$\rho(|A^{-1}| \cdot |A|) = \min_D \kappa(AD, |AD|)$$

where the minimum is taken over all nonsingular diagonal D, this would mean that the componentwise relative distance to the nearest singular matrix is approximately the reciprocal of the smallest condition number attainable by column scaling.

Note added in proof. Nick Highham and the author have found a 4 by 4 rank 2 matrix in a neighborhood of which the upper bound in Proposition 5.3 can be arbitrarily weak.

REFERENCES

[1] M. ARIOLI, J. DEMMEL, AND I. S. DUFF, *Solving sparse linear systems with sparse backward error*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 165–190.

[2] M. Arioli, I. S. Duff, and P. P. M. de Rijk, *On the augmented system approach to sparse least-squares problems*, Numer. Math., 55 (1989), pp. 667–684.
[3] F. L. Bauer, *Optimally scaled matrices*, Numer. Math., 5 (1963), pp. 73–87.
[4] ———, *Genauigkeitsfragen bei der Lösung linearer Gleichungssysteme*, Z. Angew. Math. Mech., 46 (1966), pp. 409–421.
[5] J. Bunch, J. Dongarra, C. Moler, and G. W. Stewart, LINPACK *User's Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.
[6] P. A. Businger and G. Golub, *Algorithm 358: Singular value decomposition of a complex matrix*, Comm. ACM, 12 (1969), pp. 564–565.
[7] P. Deift, J. Demmel, L.-C. Li, and C. Tomei, *The bidiagonal singular values decomposition and Hamiltonian mechanics*, Technical Report 458, Computer Science Department, Courant Institute, New York, July 1989. (LAPACK Working Note #11.) SIAM J. Numer. Anal., 28 (1991), pp. 1461–1514.
[8] J. Demmel, *The smallest perturbation of a submatrix which lowers the rank and constrained total least squares problems*, SIAM J. Numer. Anal., 24 (1987), pp. 199–206.
[9] ———, *On floating point errors in Cholesky*, Technical Report, Computer Science Department, University of Tennessee, Knoxville, TN, 1989. (LAPACK Working Note #14.)
[10] J. Demmel and W. Kahan, *Accurate singular values of bidiagonal matrices*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 873–912.
[11] J. Demmel and K. Veselić, *Jacobi's method is more accurate than* QR, Technical Report 468, Computer Science Department, Courant Institute, New York, October 1989. (LAPACK Working Note #15.) SIAM J. Matrix Anal. Appl., 13 (1992), to appear.
[12] M. Garey and D. Johnson, *Computers and Intractability*, W. H. Freeman, San Francisco, 1979.
[13] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, SIAM J. Numer. Anal. Ser. B, 2 (1965), pp. 205–224.
[14] G. Golub and C. Reinsch, *Singular value decomposition and least squares solutions*, Numer. Math., 14 (1970), pp. 403–420.
[15] D. J. Higham and N. J. Higham, *Backward error and condition of structured linear systems*, SIAM J. Matrix Anal. Appl., this issue.
[16] N. J. Higham, 1990, Private communication.
[17] V. Pan, *How can we speed up matrix multiplication?*, SIAM Rev., 26 (1984), pp. 393–416.
[18] J. Rohn, *Systems of linear interval equations*, Linear Algebra Appl., 126 (1989), pp. 39–78.
[19] ———, *Nonsingularity under data rounding*, Linear Algebra and its Applications, 139 (1990), pp. 171–174.
[20] J. Rohn and S. Poljak, *Radius of nonsingularity*, Math. Control. Systems. Sig., to appear.
[21] R. D. Skeel, *Scaling for numerical stability in Gaussian elimination*, J. Assoc. Comput. Mach., 26 (1979), pp. 494–526.
[22] ———, *Iterative refinement implies numerical stability for Gaussian elimination*, Math. Comput., 35 (1980), pp. 817–832.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 20–35, January 1992

003

ON THE SIGNIFICANCE OF NONGENERIC TOTAL LEAST SQUARES PROBLEMS*

SABINE VAN HUFFEL†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Total least squares (TLS) is one method of solving overdetermined sets of linear equations $AX \approx B$ that is appropriate when there are errors in both the observation matrix B and the data matrix A. Golub was the first to introduce this method into the field of numerical analysis and to develop an algorithm based on the singular value decomposition. However, as pointed out by Golub and Van Loan, some TLS problems fail to have a solution altogether. Van Huffel and Vandewalle described the properties of these so-called nongeneric problems and proposed an extension of the generic TLS problem, called nongeneric TLS, in order to make these problems solvable. They proved that the solution of these nongeneric TLS problems is still optimal with respect to the TLS criteria for any number of observation vectors in B if additional constraints are imposed on the TLS solution space. These constraints are further scrutinized in this paper and compared with other approaches in linear regression.

Key words. total least squares, singular value decomposition, errors-in-variables, latent root regression, collinearity

AMS(MOS) subject classifications. 15A18, 65F20

1. Introduction.

Solving sets of linear equations is a basic issue in many applications. If these sets $Ax \approx b$ are overdetermined, then generally only an approximate solution can be computed, for instance by using a least squares (LS) or total LS (TLS) approach. The basic principle of TLS is that the noisy data $[A; b]$, while not satisfying a linear relation, are modified with minimal effort, as measured by the Frobenius norm, in a "nearby" matrix $[\widehat{A}; \widehat{b}]$ which is rank-deficient so that the set $\widehat{A}x = \widehat{b}$ is compatible. This matrix $[\widehat{A}; \widehat{b}]$ is a rank one modification of the data matrix $[A; b]$. The term "total least squares" (TLS) was coined in [10] although its solution, using the singular value decomposition (SVD), was already introduced in [9] and [8]. More generally, the TLS problem is defined as follows.

DEFINITION 1 (Ordinary TLS problem). Given an overdetermined set of m linear equations $AX \approx B, B \in \mathcal{R}^{m \times d}$, in $n \times d$ unknowns X. The total least squares (TLS) problem seeks to

$$\underset{[\widehat{A}; \widehat{B}] \in \mathcal{R}^{m \times (n+d)}}{\text{minimize}} \quad \| [A; B] - [\widehat{A}; \widehat{B}] \|_F \tag{1}$$

$$\text{subject to} \quad R(\widehat{B}) \subseteq R(\widehat{A}). \tag{2}$$

Once a minimizing $[\widehat{A}; \widehat{B}]$ is found, then any X satisfying

$$\widehat{A}X = \widehat{B} \tag{3}$$

is called a TLS *solution* and $[\Delta\widehat{A}; \Delta\widehat{B}] = [A; B] - [\widehat{A}; \widehat{B}]$, the corresponding TLS correction.

* Received by the editors December 17, 1990; accepted for publication (in revised form) June 4, 1991. This work was supported by the Belgian National Fund for Scientific Research (N.F.W.O.).

† ESAT Laboratory, Department of Electrical Engineering, Katholieke Universiteit Leuven, Kardinaal Mercierlaan 94, B-3001 Heverlee, Belgium (vanhuffe@esat.kuleuven.ac.be).

Whenever the TLS solution is not unique, the minimum norm solution, denoted by $\hat{X}$, is singled out.

Unless stated otherwise, we assume that the set of equations $AX \approx B$ is *overdetermined*, i.e., $m > n$. For the *one-dimensional* problem, i.e., $d = 1$, the matrices are replaced by their corresponding vector notations, e.g., the vectors b and x are used instead of the matrices B and X.

$R(S)$ and $R_r(S)$ denote, respectively, the range and row space of a matrix S.

Denote the SVD of the $m \times n$ matrix A, $m > n$, by:

$$A = U'\Sigma' V'^T \tag{4}$$

with

$$\begin{aligned} &U' = [U'_1; U'_2], U'_1 = [u'_1, \cdots, u'_n], U'_2 = [u'_{n+1}, \cdots, u'_m], u'_i \in \mathcal{R}^m, U'^T U' = I_m, \\ &V' = [v'_1, \cdots, v'_n], \qquad v'_i \in \mathcal{R}^n, V'^T V' = I_n, \\ &\Sigma' = \operatorname{diag}(\sigma'_1, \cdots, \sigma'_n) \in \mathcal{R}^{m \times n}, \qquad \sigma'_1 \geq \cdots \geq \sigma'_n \geq 0 \end{aligned}$$

and denote the SVD of the $m \times (n + d)$ matrix $[A; B]$, $m > n$, by:

$$[A; B] = U\Sigma V^T \tag{5}$$

with

$$\begin{aligned} &U = [U_1; U_2], U_1 = [u_1, \cdots, u_n], U_2 = [u_{n+1}, \cdots, u_m], u_i \in \mathcal{R}^m, U^T U = I_m, \\ &V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \\ n & d \end{bmatrix} \begin{matrix} n \\ d \end{matrix} = [v_1, \cdots, v_n, v_{n+1}, \cdots, v_{n+d}], \qquad v_i \in \mathcal{R}^{n+d}, V^T V = I_{n+d}, \\ &\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_{n+t}) \in \mathcal{R}^{m \times (n+d)}, \quad t = \min\{m - n, d\}, \quad \sigma_1 \geq \cdots \geq \sigma_{n+t} \geq 0. \end{aligned}$$

For convenience of notation, we define $\sigma_i = 0$ if $m < i \leq n + d$.

Using this notation, we call the problem *generic* if for $\sigma_p > \sigma_{p+1} = \cdots = \sigma_{n+1}$ with $p \leq n$, the submatrix

$$V_\gamma = \begin{bmatrix} v_{n+1,p+1} & \cdots & v_{n+1,n+d} \\ \vdots & \ddots & \vdots \\ v_{n+d,p+1} & \cdots & v_{n+d,n+d} \end{bmatrix} \quad \text{with } v_{j,i} \text{ the } j\text{th component of } v_i \tag{6}$$

of V in (5) has full row rank d. If $\sigma_n > \sigma_{n+1}$ (i.e., $p = n$), this means that V_{22} in (5) is nonsingular (or $v_{n+1,n+1} \neq 0$ if $d = 1$). The TLS solution of generic problems, called generic TLS solution, can be computed with the algorithm of Golub and Van Loan [10] and is given by

$$\hat{X} = -Z\Gamma^{-1} \tag{7}$$

where Z, Γ are obtained by postmultiplying $[v_{p+1}, \cdots, v_{n+d}]$ with an orthogonal matrix Q such that

$$[v_{p+1}, \cdots, v_{n+d}]Q = \begin{bmatrix} Y & Z \\ 0 & \Gamma \\ n - p & d \end{bmatrix} \begin{matrix} n \\ d \end{matrix}. \tag{8}$$

For one-dimensional TLS problems, the generic TLS solution (7) reduces to a simple scaling of the last column vector $[z^T;\gamma]^T$ in (8):

$$\widehat{x} = -z/\gamma. \tag{9}$$

Whenever V_γ is rank-deficient (or $\gamma = 0$, if $d = 1$) the problem is called *nongeneric*. In this case, $\sigma'_n \leq \sigma_{n+1}$, as proven in [20], [19], and the TLS problem (1)–(2) fails to have a finite solution altogether. For example,

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 2 \\ 2 \end{bmatrix} \quad \text{and} \quad [\Delta A; \Delta b] = \begin{bmatrix} 0 & 0 & 0 \\ 0 & \varepsilon & 0 \\ 0 & \varepsilon & 0 \end{bmatrix},$$

then for all $\varepsilon > 0$, $b - \Delta\widehat{b} \in R(A - \Delta\widehat{A})$. Thus there is no smallest value of $\|[\Delta\widehat{A}; \Delta\widehat{b}]\|_F$ for which $b - \Delta\widehat{b} \in R(A - \Delta\widehat{A})$.

The following remark is important here. In this paper, we assume that the data have been properly scaled in advance. This assumption implies that the rows and columns of $[A; B]$ have been scaled in such a way that the errors in its entries are independent zero-mean variables with equal variance. The failure to approach these conditions may result in spurious indications of rank deficiency of the matrix V_γ in (6), a phenomenon that results in the detection of "artificial" nongeneric TLS problems that are not meaningful on the basis of what the source problem proclaims about the significance of each entry in $[A; B]$.

In [19], we described the properties of these nongeneric TLS problems and generalized the TLS computations, as given by Golub and Van Loan, in order to solve these problems. It was proven that the proposed generalization still satisfies the TLS criteria (1)–(2) for any number of right-hand side vectors b_i provided additional constraints are imposed on the TLS solution space. The significance of these problems and their constraints, as well as some additional properties, are investigated in more detail in this paper. Section 2 describes the one-dimensional nongeneric TLS problem ($d = 1$) while the multidimensional nongeneric TLS problem ($d > 1$) is considered in §3. Finally, §4 gives the conclusions.

2. The nongeneric one-dimensional TLS problem.

2.1. Problem description. Let (5) be the SVD of $[A; b]$. If $\sigma_p > \sigma_{p+1} = \cdots = \sigma_{n+1}$, $p \leq n$; then the TLS problem (1)–(2) fails to have a solution if all $v_{n+1,j} = 0$, $j = p+1, \cdots, n+1$. The properties of these so-called nongeneric TLS problems are described by [19, Thm. 3.1]. For clarity of exposition, we repeat this theorem here.

THEOREM 2.1 (Properties of the nongeneric one-dimensional TLS problem). *Let* (4) (*respectively,* (5)) *be the* SVD *of* A (*respectively,* $[A; b]$). *Let* b' *be the orthogonal projection of* b *onto* $R(A)$ *and* $[\widehat{A}; \widehat{b}] = [A; b](I - [z^T; \gamma]^T[z^T; \gamma])$ *the rank* n *approximation of* $[A; b]$, *computed from* (8)–(9). *If* $V'(\sigma_j)$ (*respectively,* $U'(\sigma_j)$) *is the right* (*respectively, left*) *singular subspace of* A *associated with* σ_j, *then the following relations can be proven:*

$$(10)\quad (a)\ v_{n+1,j} = 0 \iff v_j = \begin{bmatrix} v' \\ 0 \end{bmatrix} \quad \textit{with } v' \in V'(\sigma_j),$$

$$
\begin{array}{lllll}
(b) & v_{n+1,j} = 0 & \Longrightarrow & \sigma_j = \sigma'_k & \text{with } k = j-1 \text{ or } k = j \text{ and } 1 \le k \le n, \\
(c) & v_{n+1,j} = 0 & \Longrightarrow & b \perp u' & \text{with } u' \in U'(\sigma_j), \\
(d) & v_{n+1,j} = 0 & \Longleftrightarrow & u_j = u' & \text{with } u' \in U'(\sigma_j), \\
(e) & v_{n+1,j} = 0 & \Longrightarrow & b' \perp u' & \text{with } u' \in U'(\sigma_j), \\
(f) & v_{n+1,j} = 0 & \Longrightarrow & \widehat{b} \perp u' & \text{with } u' \in U'(\sigma_j).
\end{array}
$$

If σ_j is an isolated singular value, the converse of relation (c) *and* (e) *also holds.*

In particular, assume that $\sigma_n > \sigma_{n+1}$ and $v_{n+1,n+1} = 0$. Then, this theorem yields:

$$
\sigma_{n+1} = \sigma'_n,\ u_{n+1} = \pm u'_n,\ v_{n+1} = \begin{bmatrix} z \\ \gamma \end{bmatrix} = \begin{bmatrix} \pm v'_n \\ 0 \end{bmatrix} \tag{11}
$$

and b, b' as well as $\widehat{b}$ are orthogonal to u'_n.

It is easy to see that the generic TLS approximation $[\widehat{A}; \widehat{b}] = \sum_{i=1}^{n} \sigma_i u_i v_i^T$ and corresponding TLS correction matrix $[\Delta\widehat{A}; \Delta\widehat{b}] = \sigma_{n+1} u_{n+1} v_{n+1}^T$ minimizes $\|[\Delta\widehat{A}; \Delta\widehat{b}]\|_F$ but does *not* satisfy the constraint $\widehat{b} \in R(\widehat{A})$ and therefore, this $[\widehat{A}; \widehat{b}]$ does not solve the TLS problem (1)–(2). Indeed, using (4), (5), and (11), we obtain:

$$
\begin{aligned}
[\widehat{A}; \widehat{b}] &= [A; b] - [\Delta\widehat{A}; \Delta\widehat{b}] = [A; b] - \sigma_{n+1} u_{n+1} v_{n+1}^T \\
&= \left[\sum_{i=1}^{n} \sigma'_i u'_i v_i'^T ; b \right] - \sigma'_n u'_n [v_n'^T ; 0] = \left[\sum_{i=1}^{n-1} \sigma'_i u'_i v_i'^T ; b \right].
\end{aligned}
$$

Observe that this approximation makes $\widehat{A}$ rank-deficient! Indeed, A is reduced to a matrix $\widehat{A}$ of rank $n-1$ while b is not changed at all. Since $\operatorname{rank}(\widehat{A}) = n-1 < \operatorname{rank}([\widehat{A}; \widehat{b}]) = n$, it follows that $\widehat{b} \notin R(\widehat{A})$. Moreover, (11) also yields :

$$
\begin{aligned}
&[\widehat{A}; \widehat{b}] v_{n+1} = 0 \Rightarrow \widehat{A} v'_n = 0, \\
&[A; b] v_{n+1} = \sigma_{n+1} u_{n+1} \Rightarrow A v'_n + 0 \cdot b = A v'_n = \sigma'_n u'_n \approx 0 \quad \text{if } \sigma'_n \text{ is small.}
\end{aligned} \tag{12}
$$

This means that the solution v'_n of the set $\widehat{A}x = 0$ describes an approximate linear relation among the columns of A instead of estimating the desired linear relation between A and b.

In these situations, described by Theorem 2.1, the ordinary TLS problem (1)–(2) as such has no solution. Indeed, in order to satisfy $\widehat{b} \in R(\widehat{A})$, we have to make one direction $v \in R_r([A; b])$ orthogonal to $R_r([\widehat{A}; \widehat{b}])$, i.e., $[\widehat{A}; \widehat{b}] v = 0$, and such that $\widehat{A}x = \widehat{b}$ is compatible. If $v = \varepsilon v_n + \sqrt{1-\varepsilon^2}\, v_{n+1}$, then for every $\varepsilon > 0$, the correction matrix

$$
\begin{aligned}
[\Delta\widehat{A}; \Delta\widehat{b}] &= [A; b](\varepsilon v_n + \sqrt{1-\varepsilon^2}\, v_{n+1})(\varepsilon v_n + \sqrt{1-\varepsilon^2}\, v_{n+1})^T \\
&= (\varepsilon\sigma_n u_n + \sqrt{1-\varepsilon^2}\, \sigma_{n+1} u_{n+1})(\varepsilon v_n + \sqrt{1-\varepsilon^2}\, v_{n+1})^T
\end{aligned} \tag{13}
$$

satisfies $b - \Delta\widehat{b} \in R(A - \Delta\widehat{A})$. This implies that the data A, b are projected in the lower-dimensional subspace orthogonal to $\varepsilon\sigma_n u_n + \sqrt{1-\varepsilon^2}\, \sigma_{n+1} u_{n+1}$. The TLS corrections are given by :

$$
\|[\Delta\widehat{A}; \Delta\widehat{b}]\|_F = \sqrt{\varepsilon^2\sigma_n^2 + (1-\varepsilon^2)\sigma_{n+1}^2}. \tag{14}
$$

However, in order to satisfy (1) simultaneously, ε must be as small as possible. This "smallest" ε cannot be determined and therefore, there is *no "smallest"* $\|[\Delta\widehat{A}; \Delta\widehat{b}]\|_F$ for which $\widehat{b} \in R(\widehat{A})$. Moreover this approach makes the problem *very ill conditioned.* Indeed, using (13), the solution $\widehat{x}$ of $(A - \Delta\widehat{A})x = b - \Delta\widehat{b}$ is given by:

$$[\widehat{x}^T; -1]^T = -(\varepsilon v_n + \sqrt{1-\varepsilon^2}\, v_{n+1})/(\varepsilon v_{n+1,n} + \sqrt{1-\varepsilon^2}\, v_{n+1,n+1}) \tag{15}$$

$$= -\frac{v_n}{v_{n+1,n}} - \frac{\sqrt{1-\varepsilon^2}\, v_{n+1}}{\varepsilon v_{n+1,n}} \qquad \text{if } v_{n+1,n+1} = 0. \tag{16}$$

If ε is small, then the coefficients of $\widehat{x}$ are inflated and have *large* variances due to the influence of the second term in (16).

The following question now arises: how can we solve these nongeneric TLS problems in a meaningful way? From the earlier equations, it is clear that these problems only occur whenever A is nearly rank-deficient ($\sigma'_n \approx 0$) or when the set of equations is highly conflicting ($\sigma'_n \approx \sigma_{n+1}$ large). The latter situation is easily detected by inspecting the size of the smallest σ_i for which $v_{n+1,i} \neq 0$. If this singular value is large, the user can simply conclude that the data are not appropriate for linear modelling using TLS and as such reject the problem. In the first situation ($\sigma'_n \approx 0$), two options are possible. Either the user can remove the dependency between the columns of A by removing appropriate columns in A such that the remaining submatrix A_p of dimension $m \times p$, $p < n$, has full rank and then apply TLS to the reduced problem $A_p x \approx b$ in order to obtain an estimate of the remaining parameters. The position of the nonzero entries in v_{n+1} determines which columns of A, i.e., which variables in the model, are dependent. How to pick these columns is a problem of *subset selection.* Procedures for discarding or selecting columns in A are available in a variety of settings, e.g. see [13], [14], [18], [11, §12.2]. Another way is solving a nongeneric TLS problem defined below, i.e., additional constraints are imposed on the TLS problem (1)–(2) in order to make the problem solvable and stabilize the solution.

DEFINITION 2 (Nongeneric one-dimensional TLS problem). Given an overdetermined set of m linear equations $Ax \approx b$ in n unknowns x. Let (5) be the SVD of $[A; b]$. The nongeneric one-dimensional TLS problem seeks to

$$\underset{[\widehat{A}; \widehat{b}] \in \mathcal{R}^{m \times (n+1)}}{\text{minimize}} \quad \| [A; b] - [\widehat{A}; \widehat{b}] \|_F \tag{17}$$

$$\text{subject to } \widehat{b} \in R(\widehat{A}) \tag{18}$$

$$\text{and } [\Delta\widehat{A}; \Delta\widehat{b}] v_j = 0 \qquad \forall j : \max_{v_{n+1,p} \neq 0} \{p\} < j \leq n+1. \tag{19}$$

Once a minimizing $[\widehat{A}; \widehat{b}]$ is found, then any x satisfying

$$\widehat{A}x = \widehat{b} \tag{20}$$

is called a *nongeneric* TLS *solution* and $[\Delta\widehat{A}; \Delta\widehat{b}] = [A; b] - [\widehat{A}; \widehat{b}]$ the corresponding nongeneric TLS correction.

2.2. Solution and properties. The nongeneric TLS problem is solved as follows.

THEOREM 2.2 (Nongeneric one-dimensional TLS solution). *Let* (5) *be the* SVD *of* $[A;b]$ *and assume* $v_{n+1,j} = 0$ *for* $j = p+1, \cdots, n+1$, $p \leq n$. *If* $\sigma_{p-1} > \sigma_p$ *and* $v_{n+1,p} \neq 0$, *then*

$$[\widehat{A};\widehat{b}] = U\widehat{\Sigma}V^T \quad \text{with } \widehat{\Sigma} = \operatorname{diag}(\sigma_1, \cdots, \sigma_{p-1}, 0, \sigma_{p+1}, \cdots, \sigma_{n+1}) \tag{21}$$

solves the nongeneric TLS *problem* (17)–(19). *The corresponding nongeneric* TLS *correction matrix is*

$$[\Delta\widehat{A};\Delta\widehat{b}] = [A;b] - [\widehat{A};\widehat{b}] = \sigma_p u_p v_p^T \tag{22}$$

and

$$\widehat{x} = -\frac{1}{v_{n+1,p}}[v_{1,p}, \cdots, v_{n,p}]^T \tag{23}$$

exists and is the unique solution to $\widehat{A}x = \widehat{b}$.

Proof. The nongeneric TLS approach searches in the $(n+1)$-dimensional row space of $[A;b]$ for an approximation $[\widehat{A};\widehat{b}]$ of rank n with minimal deviation $[\Delta\widehat{A};\Delta\widehat{b}]$ such that (18) is satisfied and $[\Delta\widehat{A};\Delta\widehat{b}]v_j = 0$, $\quad j = p+1, \cdots, n+1$. Since $[A;b] = \sum_{i=1}^{p} \sigma_i u_i v_i^T + \sum_{i=p+1}^{n+1} \sigma_i u_i v_i^T$ and the rank n TLS approximation $[\widehat{A};\widehat{b}] = [A;b] - [\Delta\widehat{A};\Delta\widehat{b}]$, these last constraints imply that :

$$[\widehat{A};\widehat{b}] = C + \sum_{i=p+1}^{n+1} \sigma_i u_i v_i^T \tag{24}$$

where C is a rank $(p-1)$ approximation of $[A;b]_p = \sum_{i=1}^{p} \sigma_i u_i v_i^T$.

Hence, the nongeneric TLS approach must look in the p-dimensional space $R([A;b]_p)$ for an approximation C of rank $p-1$ with minimal approximation effort $\|[\Delta\widehat{A};\Delta\widehat{b}]\|_F$ such that (18) is satisfied. Using the Eckart–Young–Mirsky theorem [11, Thm. 2.5.2], [20, Thm. 2.3] and the properties of the SVD, this minimizing $[\Delta\widehat{A};\Delta\widehat{b}]$ is given by (22). The condition $\sigma_{p-1} > \sigma_p$ ensures that (22) is the unique minimizer. The nongeneric TLS approximation $[\widehat{A};\widehat{b}]$ of rank n is then given by (21). Since v_p belongs to the null space of $[\widehat{A};\widehat{b}]$, i.e., $[\widehat{A};\widehat{b}]v_p = 0$, and since $v_{n+1,p} \neq 0$, (18) is satisfied, i.e., $\widehat{b} \in R(\widehat{A})$. The solution $\widehat{x}$ of the nongeneric TLS problem is then obtained by scaling v_p until its last component is -1, which proves (23). The uniqueness of $\widehat{x}$ follows from the condition that $\sigma_{p-1} > \sigma_p$. □

Theorem 2.2 is a correction of [17, Thm. 1-4]. The latter reference does not include condition (19) and as a result, the proof there is not correct. This condition is crucial in the statement of Theorem 2.2 because, without it, the nongeneric TLS problem is *not* solvable and as a result, the nongeneric TLS solution does not exist. For example, if $p = n$ in Theorem 2.2, the nongeneric TLS solution (23) is given by :

$$[\widehat{x}^T; -1]^T = -v_n / v_{n+1,n} \tag{25}$$

with corresponding nongeneric correction matrix $[\Delta\widehat{A};\Delta\widehat{b}] = \sigma_n u_n v_n^T$. Observe that a smaller $\|[\Delta\widehat{A};\Delta\widehat{b}]\|_F$ can be found if we omit the constraint $[\Delta\widehat{A};\Delta\widehat{b}]v_{n+1} = 0$, namely, every $[\Delta\widehat{A};\Delta\widehat{b}]$ of the form (13) with $0 < \varepsilon < 1$. However, since no "smallest" ε can be determined such that (17)–(18) are satisfied simultaneously, the nongeneric

TLS problem is not solvable without condition (19). Moreover, as said before, the corrections (13) make the problem very ill conditioned and inflate the coefficients of the corresponding solution. As shown in §3, these solutions are not even meaningful since they induce false correlations between A and b.

There is no restriction in imposing the condition $\sigma_{p-1} > \sigma_p$. Indeed, if $\sigma_{p-1} = \sigma_p$, then the nongeneric TLS solution still exists but is possibly not unique.

The nongeneric TLS solution (23) also equals the minimum norm TLS solution computed in the restricted row space

$$R_r([\widehat{A}; \widehat{b}]) = R_r([A; b]) \setminus R([v_p, \cdots, v_{n+1}]).$$

This result follows directly from (8) using the assumptions $v_{n+1,j} = 0$ for $j = p+1, \cdots, n+1$.

Since $v_{n+1,j} = 0$ for $j = p+1, \cdots, n+1$, vector v_p completely characterizes the linear relationship between A and b. Indeed,

$$[A; b]v_j = Av_j' = \sigma_j u_j, \qquad j = p+1, \cdots, n+1,$$
$$[A; b]v_p = \sigma_p u_p.$$

Observe that σ_p measures the degree of *incompatibility* of the TLS problem $Ax \approx b$ and thus indicates how closely b is linearly related to A. A small σ_p implies that $[A; b]v_p \approx 0$ and hence, there exists a strong linear relationship between A and b given by v_p. As σ_p is increasing, b becomes less dependent from the columns of A and hence the validity of the imposed linear model $Ax \approx b$ must be questioned.

Note also that nongeneric TLS problems do not occur if we are only interested in estimating a linear relationship between the columns of $[A; b]$, no matter which column is used as right-hand side. If, then, $v_{n+1,n+1} = 0$, we simply replace b by any column a_i of A provided $v_{i,n+1} \neq 0$.

In practice, $v_{n+1,n+1}$ is rarely equal to zero but close-to-nongeneric TLS problems in which $v_{n+1,n+1} \approx 0$ are not uncommon. As shown in Theorem 2.1, this occurs when σ_n' approaches σ_{n+1}. In those cases, the generic TLS solution can still be computed but is unstable and becomes very sensitive to data errors when $\sigma_n' - \sigma_{n+1}$ is very close to zero [10]. Identifying the problem as nongeneric stabilizes the solution and makes it rather insensitive to data errors, as shown below.

Example 1. Consider the TLS problem $A_0 x \approx b_0$,

$$A_0 = \begin{bmatrix} 1 & 0 \\ 0 & 10^{-4} \\ 0 & 0 \end{bmatrix} \quad \text{and} \quad b_0 = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix},$$

illustrated in Fig. 1 ($n = 2$). Since the set of equations is highly incompatible, the nongeneric TLS problem is likely to occur. Taking the SVD (5) of $[A_0; b_0]$, we observe that $v_{n+1,n+1} = 0$. Hence, $b_0 \perp u_n'$ and $v_{n+1} = v_n'$ by Theorem 2.1 (see Fig. 1) and the nongeneric TLS solution $\widehat{x}_0$ of the unperturbed problem $A_0 x \approx b_0$ must be computed. Using Theorem 2.2, we obtain $\widehat{x}_0 = [2/(\sqrt{5}-1),\ 0]^T$ from v_2.

Perturb A_0 and b_0 with

$$\Delta A = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 10^{-8} \end{bmatrix} \quad \text{and} \quad \Delta b = \begin{bmatrix} 0 \\ 10^{-8} \\ 0 \end{bmatrix}.$$

Taking the SVD (5) of $[A; b] = [A_0 + \Delta A; b_0 + \Delta b]$, we now obtain $v_{n+1,n+1} \approx 10^{-8}$ and $\sigma_n' - \sigma_{n+1} = \mathcal{O}(10^{-8})$. Computing the generic TLS solution (9) from v_{n+1} ($p = n$)

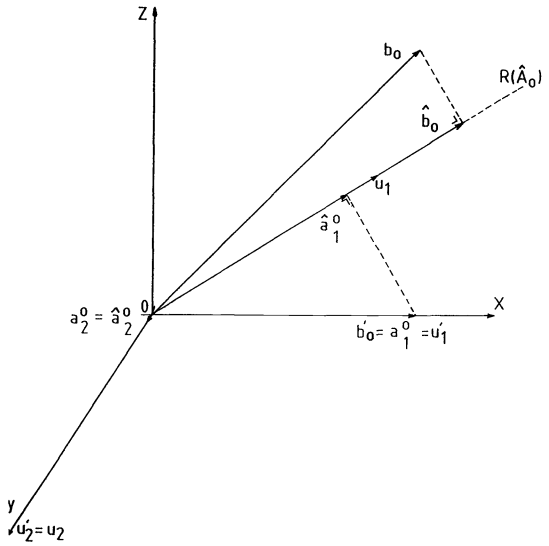


FIG. 1. *Geometric illustration of Example* 1. *X, Y, and Z are the coordinate axes.* A_0 *has two columns* a_1^0 *and* a_2^0, *and* b_0 *is the observation vector. The nongeneric* TLS *approximation is given by* $\widehat{a}_1^0, \widehat{a}_2^0, \widehat{b}_0$. *The* LS *approximation* b_0' *is the orthogonal projection of* b_0 *onto* $R(A_0)$. u_1', u_2' (*respectively,* u_1, u_2) *are the left singular vectors of* A_0 (*respectively,* $[A_0; b_0]$).

produces a very sensitive, unstable solution, given by $[1, 10^8 + 10^4]^T$. However, since the perturbation level is 10^{-8}, we may conclude that $b \perp u_n'$. Hence the nongeneric TLS solution must be computed. Using Theorem 2.2, we obtain $\widehat{x}_0 + \Delta\widehat{x} = [2/(\sqrt{5} - 1), 2 \cdot 10^{-8}/(\sqrt{5} - 3)]^T$ from v_n $(n = 2)$. Observe that $\|\Delta\widehat{x}\|_2 = 10^{-8}$ is very small and the perturbed TLS solution stable.

Using Theorem 2.1, the following useful characterization of the nongeneric TLS solution (23) can be proven.

THEOREM 2.3 (Closed-form expressions of the nongeneric TLS solution). *Let* (4) (*respectively,* (5)) *be the* SVD *of* A (*respectively,* $[A; b]$) *and assume that* $v_{n+1,j} = 0$ *for* $j = p + 1, \cdots, n + 1$, $p \leq n$. *If* $\sigma_{p-1} > \sigma_p$ *and* $v_{n+1,p} \neq 0$, *the nongeneric* TLS *solution equals:*

$$\widehat{x} = (A^T A - \sigma_p^2 I_n)^{-1} A^T b \tag{26}$$

$$= \sum_{i=1}^{p-1} (\sigma_j'^2 - \sigma_p^2)^{-1} \sigma_j' v_j' u_j'^T b. \tag{27}$$

Proof. Since $\begin{bmatrix} \widehat{x} \\ -1 \end{bmatrix}$ is proportional to v_p, it is an eigenvector of $[A; b]^T[A; b]$ corresponding to the eigenvalue σ_p^2. Hence, it satisfies the eigenvector equations:

$$[A; b]^T[A; b] \begin{bmatrix} \widehat{x} \\ -1 \end{bmatrix} = \begin{bmatrix} A^T A & A^T b \\ b^T A & b^T b \end{bmatrix} \begin{bmatrix} \widehat{x} \\ -1 \end{bmatrix} = \sigma_p^2 \begin{bmatrix} \widehat{x} \\ -1 \end{bmatrix} \tag{28}$$

or

$$(A^T A - \sigma_p^2 I_n)\widehat{x} = A^T b, \tag{29}$$

$$b^T A\widehat{x} + \sigma_p^2 = b^T b. \tag{30}$$

We first prove that $A^T A - \sigma_p^2 I$ is nonsingular. Hereto, $\sigma'_{p-1} > \sigma_p > \sigma'_p$ must hold.

Since the solution is nongeneric, it follows that $\sigma_p > \sigma_{p+1}$. Indeed, if by the contrary $\sigma_p \leq \sigma_{p+1}$, we necessarily have $\sigma_p = \sigma_{p+1}$ since $\sigma_i \geq \sigma_{i+1}$, for all i. In this case we would compute the minimum norm TLS solution $\begin{bmatrix} \widehat{x} \\ -1 \end{bmatrix}$ in $R([v_p, \cdots, v_{n+1}])$. Since $v_{n+1,p} \neq 0$ we can compute a generic TLS solution, a contradiction since $\widehat{x}$ is nongeneric. Thus $\sigma_p > \sigma_{p+1}$. Since we also assumed $\sigma_{p-1} > \sigma_p, \sigma_p$ is not a repeated singular value and hence v_p is also unique, up to a scaling factor.

Now, the interlacing property for singular values [11, Cor. 8.3.3] implies that:

$$\sigma_1 \geq \sigma'_1 \geq \cdots \geq \sigma'_{p-1} \geq \sigma_p \geq \sigma'_p \geq \sigma_{p+1} \geq \cdots \geq \sigma'_n \geq \sigma_{n+1}. \tag{31}$$

Since $v_{n+1,j} = 0$ for $j = p+1, \cdots, n+1$, it follows from Theorem 2.1(b) that $\sigma_j = \sigma'_{j-1}$. Hence, we have $\sigma_{p+1} = \sigma'_p$. Since $\sigma_p > \sigma_{p+1}$ is also assumed, we must have $\sigma_p > \sigma'_p$.

We next prove that $\sigma'_{p-1} > \sigma_p$ by contradiction. Assume that $\sigma'_{p-1} \leq \sigma_p$; then, from (31), we necessarily have that $\sigma'_{p-1} = \sigma_p$.

Since σ'_{p-1} is a singular value of A with corresponding singular vector v'_{p-1}, we obtain from the corresponding eigenvector equations:

$$A^T A v'_{p-1} = \sigma'^2_{p-1} v'_{p-1} = \sigma_p^2 v'_{p-1}$$

or

$$[A; b]^T [A; b] \begin{bmatrix} v'_{p-1} \\ 0 \end{bmatrix} = \sigma_p^2 \begin{bmatrix} v'_{p-1} \\ 0 \end{bmatrix}. \tag{32}$$

Equation (32) implies that

$$\begin{bmatrix} v'_{p-1} \\ 0 \end{bmatrix}$$

is also a right singular vector of $[A; b]$ corresponding to σ_p, a contradiction since the right singular vector v_p, corresponding to σ_p, is unique up to a scaling factor and has a nonzero last component: $v_{n+1,p} \neq 0$. Thus $\sigma'_{p-1} > \sigma_p$.

Since $\sigma'_{p-1} > \sigma_p > \sigma'_p$ is now proven, $(A^T A - \sigma_p^2 I_n)$ is nonsingular. Hence, (26) then follows immediately from (29). Now take the SVD of (26):

$$\widehat{x} = \sum_{j=1}^{n} (\sigma'^2_j - \sigma_p^2)^{-1} \sigma'_j v'_j u'^T_j b. \tag{33}$$

Since $v_{n+1,j} = 0$ for $j = p+1, \cdots, n+1$, we have from Theorem 2.1(c) that $b \perp u'_{j-1}$, i.e., $u'^T_i b = 0$ for $i = p, \cdots, n$. Substituting this into (33) proves (27). □

Observe that the expressions of the generic and nongeneric TLS solutions coincide for $p = n + 1$, i.e., when the additional constraints (19) vanish.

The following remark is important here. Comparing the expressions of the (non)generic TLS solution with the closed-form expression of the LS solution:

$$x' = (A^T A)^{-1} A^T b, \tag{34}$$

we observe that TLS applies a *de*regularizing procedure, a kind of *reverse* ridge regression. From a numerical point of view this looks bad since subtracting $\sigma_p^2 I$ makes

the problem more ill conditioned. However, we should realize here that this subtraction is essentially motivated by the statistics of the problem, as shown below. It is well known in regression analysis that the ordinary LS solution x' given by (34) is generally an *inconsistent* estimate of the true parameters x_0 in an errors-in-variables model given by:

$$A_0 x_0 = b_0, \quad A = A_0 + \Delta A \quad \text{and} \quad b = b_0 + \Delta b. \tag{35}$$

Only A, b are known. A_0, b_0 are unknown constants and the rows of the error matrix $[\Delta A; \Delta b]$ are assumed to be independently and identically distributed with common zero mean and associated error covariance matrix given by $\sigma_\nu^2 I$. It is easily proven [6], [16] that under quite general conditions x' does not converge to x_0 but ("plim" means probability limit):

$$\operatorname*{plim}_{m\to\infty} x' - x_0 = -\sigma_\nu^2 \left(\frac{1}{m} A_0^T A_0 + \sigma_\nu^2 I_n\right)^{-1} x_0 = -\sigma_\nu^2 \left(\operatorname*{plim}_{m\to\infty} \frac{1}{m} A^T A\right)^{-1} x_0. \tag{36}$$

This implies that the LS solution is asymptotically an *underbiased* estimate of the true parameters x_0 in (35). The asymptotic bias can be removed by subtracting the error covariance matrix $\mathcal{E}([\Delta A; \Delta b]^T [\Delta A; \Delta b]) = m\sigma_\nu^2 I_n$ from $A^T A$ in (34) ($\mathcal{E}$ denotes the expected value operator) and a consistent estimator can be derived [16], [15], [5]:

$$(A^T A - m\sigma_\nu^2 I_n)^{-1} A^T b. \tag{37}$$

This is clearly a modification of the ordinary LS estimate. It corrects for the extra sources of error in an errors-in-variables model and can be considered as a "method-of-moments" estimator in the following sense:

$$\mathcal{E}\left(\frac{1}{m} A^T A - \sigma_\nu^2 I\right) = \frac{1}{m} A_0^T A_0, \tag{38}$$

$$\mathcal{E}\left(\frac{1}{m} A^T b\right) = \frac{1}{m} A_0^T A_0 x_0. \tag{39}$$

Let (4) and (5) be the SVD of A and $[A; b]$, respectively. Assume first that $\sigma'_n > \sigma_{n+1}$, i.e., the TLS problem is generic. Under the assumptions of model (35), we can prove that $\lim_{m\to\infty} \frac{1}{m}\sigma_{n+1}^2 = \sigma_\nu^2$ with probability one [7], [16], which implies that (37) and (26) are asymptotically equivalent for $p = n + 1$. Hence, subtracting $\sigma_{n+1}^2 I$ makes the TLS solution (26) to a consistent estimate of the true parameters x_0 in (35). See, e.g., [7], [6], [16], [5], and [2] for a description of the consistency conditions of the TLS solution in errors-in-variables models of the form (35) or transfer function models arising in system identification.

If the TLS problem is nongeneric, then $\sigma_p^2 > \sigma_{p+1}^2 = \cdots = \sigma_{n+1}^2$ holds. From expression (26) it follows that the nongeneric TLS solution will be an overbiased estimate of the true parameters x_0 in (35). The smaller the difference $\sigma_p^2 - \sigma_{n+1}^2$, the smaller the bias, the more compatible the set of equations $Ax \approx b$ and the better the nongeneric TLS solution estimates the true parameters x_0. Moreover, if the assumptions of model (35) hold, the nongeneric TLS problem can be made generic by adding more equations provided $\lim_{m\to\infty} \frac{1}{m} A_0^T A_0$ exists and is positive definite [7, Lemma 3.3].

2.3. Additional constraints. Since $R_r([\Delta\widehat{A}; \Delta\widehat{b}]) = [\widehat{x}^T; -1]^T$, the additional constraints $[\Delta\widehat{A}; \Delta\widehat{b}]v_j = 0$ are equivalent with the requirement

$$\begin{bmatrix} \widehat{x} \\ -1 \end{bmatrix} \perp v_j \quad \forall j \quad \text{with } v_{n+1,j} = 0.$$

The introduction of these *additional constraints* in the nongeneric TLS approach can be motivated as follows. The singular vectors v_j with negligible or zero last component and associated with a small singular value, are called *nonpredictive multicollinearities* in linear regression since they reveal multicollinearities (i.e., approximate linear dependencies between the columns of A) in A that are of no (or negligible) value in predicting the response b. In fact, if $v_{n+1,n+1} = 0$, then using Theorem 2.1 yields (11) and (12). Since $\sigma'_n = \sigma_{n+1}$ is the smallest singular value, (12) means that A is nearly rank-deficient or else the set of equations is highly incompatible. Since also $b \perp u'_n = u_{n+1}$, there is *no correlation* between A and b in the direction of $u'_n = u_{n+1}$. Hence, it does not make sense here to apply a correction of the form (13) since this correction "creates" a correlation between $\widehat{A} = A - \Delta\widehat{A}$ and $\widehat{b} = b - \Delta\widehat{b}$ in the direction u_{n+1}. Applying such corrections would induce a wrong interpretation about the exact relation between A and b. If $b \perp u_{n+1}$, then also keep the TLS approximation $\widehat{b} \perp u_{n+1}$ as done in our nongeneric TLS approach. Therefore, the strategy of nongeneric TLS is to eliminate those directions in A which are *not* at all correlated with the observation vector b. Indeed, solving a set of equations is computing the relation between A and b. If there is no correlation between b and a left singular vector u'_j of A, i.e., $b \perp u'_j$, it is better to eliminate that direction rather than forcing a solution in that direction.

Latent root regression [21] follows a similar approach. It minimizes the residual sum of squares $\|Ax-b\|_2^2$ subject to the *same* constraints, i.e., $\left[\begin{smallmatrix} x \\ -1 \end{smallmatrix}\right] \perp$ all nonpredictive v_j. The latent root (LR) regression estimator was developed by Hawkins [12] and Webster, Gunst, and Mason [21] as an alternative for the principal component (PC) estimator when estimating the true parameters x_0 in a linear regression model, defined by:

$$\begin{aligned} Ax_0 = b_0, \qquad b = b_0 + \Delta b, \qquad A, b \text{ known}, \\ \mathcal{E}(\Delta b) = 0 \quad \text{and} \quad \mathcal{E}(\Delta b \Delta b^T) = \sigma_\nu^2 I. \end{aligned} \tag{40}$$

If (4) is the SVD of A, then the PC estimator, more commonly known as the minimum norm LS solution or truncated SVD solution [3] in numerical analysis, is given by:

$$x_{PC} = \sum_{j=1}^{p} \frac{u_j'^T b}{\sigma'_j} v'_j \tag{41}$$

where $v'_{p+1}, \cdots, v'_n$ are multicollinearities between the columns of A.

Let $[A; b]$ be the $m \times (n+1)$ matrix whose first n columns contain the standardized regressor variables and whose last column contains the standardized values of the response variable. Denote the singular values and corresponding right singular vectors of $[A; b]$ as in (5) and let $v_j^T = [v_j^{1T}; v_{n+1,j}]$. Then the LR estimator is given by:

$$x_{LR} = \sum_{j=1}^{p} f_j v_j^1 \tag{42}$$

where

$$f_j = -\frac{v_{n+1,j}\sigma_j^{-2}}{\sum_{q=1}^{p} v_{n+1,q}^2 \sigma_q^{-2}}$$

and $v_{p+1}, \cdots, v_{n+1}$ are the nonpredictive multicollinearities.

Observe that the LR estimator also makes

$$\begin{bmatrix} x_{LR} \\ -1 \end{bmatrix} \perp \text{all nonpredictive multicollinearities,}$$

corresponding to the $n-p+1$ deleted terms in (42). To accomplish this, the coefficients f_j in (42) are determined by first setting $f_j = 0$ if v_j identifies a nonpredictive multicollinearity and then selecting the remaining values to minimize the residual sum of squares. If $p = n + 1$, i.e., there are no nonpredictive v_j, then the latent root regression estimate equals the LS solution. This is a way of *stabilizing* the solution in linear regression when the data in A are nearly multicollinear ($\sigma_n' \approx 0$). It is well known that then the ordinary LS solution tends to be inflated and predicted values may be unreasonable.

The nongeneric TLS solution is another way of handling this multicollinearity problem and stabilizing the TLS solution. Nongeneric TLS minimizes the applied corrections $\|[A - \widehat{A}; b - \widehat{b}]\|_F$ subject to $\widehat{A}x = \widehat{b}$ and the *same* constraints as used in latent root regression. If there are *no* nonpredictive v_j, then the nongeneric TLS solution equals the generic TLS solution, as proven in Theorem 2.3. Moreover, Theorem 2.1 proves the equivalence between these nonpredictive v_j with $v_{n+1,j} = 0$ and corresponding multicollinearities v_i' of A. This implies that LR, PC, and nongeneric TLS eliminate the *same* multicollinearities. The relation between PC, LR, and (non)generic TLS depends primarily on the size of the smallest singular value σ_p, corresponding to a v_p with $v_{n+1,p} \neq 0$. If σ_p is small, close connections exist between nongeneric TLS and these biased regression estimators. This is evident by comparing the expressions (41) and (26) of the PC estimator and the nongeneric TLS estimator, respectively. The comparison between LR and TLS estimation is based on (23). Assuming $n - p + 1$ nonpredictive multicollinearities, we observe from (42) and (23) the strong, though different, relation between the two methods. As σ_p decreases, the contribution of v_p in (42) will dominate and thus x_{LR} approaches $\widehat{x}$. For those problems, the nongeneric TLS estimator looks promising but the merits of the nongeneric TLS estimator can be questioned when the sets of equations are more incompatible (i.e., σ_p increases). This occurs when the data are not appropriate for linear modelling. In this case the user must reject the problem as irrelevant from a linear modelling point of view. Large σ_p also occur in the presence of outliers in the data, i.e., large errors in the measurements. These outliers make the (non)generic TLS solution unstable and considerably deteriorate its accuracy [1]. The same holds for the other regression estimators, although the loss in accuracy is less dramatic. In these cases *robust* procedures which are rather insensitive to outliers should be considered, e.g., by downweighting measurement samples that give rise to high residuals; see, e.g., [1], [22].

3. The nongeneric multidimensional TLS problem. The properties given in Theorem 2.1 can be generalized to multidimensional nongeneric TLS problems $AX \approx B_{m\times d}$ ($d > 1$); see [19, Thm. 3.4]. Since conditions concerning the multiplicity of σ_{n+1} are imposed, this theorem does not describe the properties of *all*

nongeneric TLS problems. Let us first assume that $\sigma_n > \sigma_{n+1}$ and V_{22} is singular so that the generic multidimensional TLS solution $\widehat{X} = -V_{12}V_{22}^{-1}$ does *not exist.* Analogously to the one-dimensional problem, we could construct a multidimensional analogue $[\Delta\widehat{A}; \Delta\widehat{B}]$ of (13) such that $R(\widehat{B}) \subseteq R(\widehat{A})$. Since $\|[\Delta\widehat{A}; \Delta\widehat{B}]\|_F$ depends on ε, no "smallest" $\|[\Delta\widehat{A}; \Delta\widehat{B}]\|_F$ can be computed and hence the TLS problem (1)–(2) has no solution. In order to solve these problems and compute a solution with reduced sensitivity, the nongeneric one-dimensional problem (17)–(19) is generalized as follows.

DEFINITION 3 (Nongeneric multidimensional TLS problem). Given an overdetermined set of m linear equations $AX \approx B$, $B \in \mathcal{R}^{m\times d}$, in $n \times d$ unknowns X. Let (5) be the SVD of $[A; B]$. The nongeneric multidimensional TLS problem seeks to

$$\underset{[\widehat{A};\widehat{B}] \in \mathcal{R}^{m\times(n+d)}}{\text{minimize}} \quad \| [A; B] - [\widehat{A}; \widehat{B}] \|_F \tag{43}$$

$$\text{subject to } R(\widehat{B}) \subseteq R(\widehat{A}) \tag{44}$$

$$\text{and} \quad [\Delta\widehat{A}; \Delta\widehat{B}] \begin{bmatrix} w \\ 0 \end{bmatrix} \begin{matrix} n \\ d \end{matrix} = 0 \quad \forall \begin{bmatrix} w \\ 0 \end{bmatrix} \in R\left(\begin{bmatrix} V_1 \\ V_2 \end{bmatrix}\right) \tag{45}$$

where

$$\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \begin{matrix} n \\ d \end{matrix} = [v_{q+1}, \cdots, v_{n+d}]$$

and q maximal such that V_2 is nonsingular. Once a minimizing $[\widehat{A}; \widehat{B}]$ is found, then any X satisfying

$$\widehat{A}X = \widehat{B}$$

is called a *nongeneric* TLS *solution* and $[\Delta\widehat{A}; \Delta\widehat{B}] = [A; B] - [\widehat{A}; \widehat{B}]$, the corresponding nongeneric TLS correction.

In [19, Thm. 3.3] the nongeneric multidimensional TLS problem (43)–(45) is solved and the theorem shows how to compute the nongeneric TLS solution $\widehat{X}$. A proof is included in [20, Thm. 3.15]. This nongeneric TLS solution also equals the *minimum norm* TLS solution computed in the restricted row space

$$R_r([\widehat{\widehat{A}}; \widehat{\widehat{B}}])$$

given by $R_r([A; B]) \setminus R([v_{q+1}, \cdots, v_{n+d}])$ [20].

Observe that in generic problems the additional constraints (45) can be neglected. Consider the SVD (5) of $[A; B]$. If $\sigma_n > \sigma_{n+1}$, the TLS problem (1)–(3) is solvable provided V_{22} is nonsingular. Hence, $q = n$ in (45). Since V_{22} is nonsingular, no $\begin{bmatrix} w \\ 0 \end{bmatrix} \begin{matrix} n \\ d \end{matrix}$, $w \neq 0$, exists such that

$$\begin{bmatrix} V_{12} \\ V_{22} \end{bmatrix} y = \begin{bmatrix} w \\ 0 \end{bmatrix}$$

for some y and hence, the constraints (45) vanish. In case of nonuniqueness (assume that $\sigma_p > \sigma_{p+1} = \cdots = \sigma_{n+1}$ with $p \leq n$), the minimum norm generic TLS solution (7) is singled out with Γ nonsingular. Since all $\begin{bmatrix} w \\ 0 \end{bmatrix} \in R([v_{p+1}, \cdots, v_{n+d}])$ belong to

$$R\left(\begin{bmatrix} Y \\ 0 \end{bmatrix}\right) \text{ and } R\left(\begin{bmatrix} Y \\ 0 \end{bmatrix}\right) \perp R\left(\begin{bmatrix} Z \\ \Gamma \end{bmatrix}\right) = R\left(\begin{bmatrix} \widehat{X} \\ -I \end{bmatrix}\right)$$

because of the orthonormality of $[v_{p+1}, \cdots, v_{n+d}]Q$, the constraints (45) are automatically satisfied and need not be imposed.

It is worthwhile to discuss here in more detail the significance of the additional constraints (45). As mentioned in the previous section, the vectors $\left[\begin{smallmatrix} w \\ 0 \end{smallmatrix}\right]$ identify nonpredictive multicollinearities, i.e., approximate linear dependencies between the columns of A which do not reveal anything about the relation between A and B and which are of no (or little) value in predicting B. This is because:

(46)
$$\left([A;B]\begin{bmatrix} w \\ 0 \end{bmatrix}\right)^T \left([A;B]\begin{bmatrix} w \\ 0 \end{bmatrix}\right) = \alpha^2 \iff w^T A^T A w = \alpha^2, \qquad \sigma_{n+d} \leq \alpha \leq \sigma_{q+1}.$$

If σ_{q+1} is small, then $\alpha \approx 0$ and (46) implies:

$$Aw \approx 0 \qquad \forall \begin{bmatrix} w \\ 0 \end{bmatrix} \begin{matrix} n \\ d \end{matrix} \in R([v_{q+1}, \cdots, v_{n+d}]), \tag{47}$$

i.e., there exist $n - q$ independent approximate linear dependencies between the columns of A or, A is close to a rank q matrix. The closer the linear relations in (47) are to zero, the stronger are the multicollinearities and the more damaging are their effects on the generic TLS solution.

In order to stabilize the coefficients of the generic TLS solution, nongeneric TLS imposes the constraints (45). Since

$$R_r([\Delta\widehat{A}; \Delta\widehat{B}]) = R\left(\begin{bmatrix} \widehat{X} \\ -I_d \end{bmatrix}\right),$$

these imply that the nongeneric TLS solution $\widehat{X}$ must satisfy :

$$w^T \widehat{X} = 0 \quad \forall \begin{bmatrix} w \\ 0 \end{bmatrix} \in R([v_{q+1}, \cdots, v_{n+d}]), \tag{48}$$

i.e., all nonpredictive multicollinearities $\left[\begin{smallmatrix} w \\ 0 \end{smallmatrix}\right]$ are eliminated from the solution or estimator. The same is true for the ordinary LS solution. Small singular values σ'_i of A and their corresponding right singular vectors identify multicollinearities whose effects on the LS solution are more damaging as σ'_i goes to zero. In order to stabilize the LS solution, these small singular values are set to zero, i.e., A is reduced to a rank r matrix and the minimum norm solution $X' = \sum_{i=1}^{r} \sigma_i'^{-1} v'_i u_i'^T B$, called truncated SVD [3] in numerical analysis and principal component estimator in linear regression [4], is computed in the reduced r-dimensional subspace of $R(A)$. In a sense, we could thus say that the *nongeneric* TLS solution stabilizes the coefficients of the generic TLS solution in *errors-in-variables* models (35) just as the truncated SVD solution (or PC *estimator*) and the LR regression estimator (when $d = 1$) stabilize the coefficients of the LS solution in *regression* models (40) when *multicollinearities* are present in the data A, B. A more thorough statistical analysis is needed in order to reveal the merits of the nongeneric TLS estimator in multicollinearity problems.

Equation (46) shows that the nongeneric TLS estimator will be more accurate as σ_{q+1} goes to zero. Large σ_{q+1} imply highly incompatible TLS problems and hence the accuracy of the nongeneric TLS estimator, as well as the relevance of the problem to be solved, may be questioned.

If the conditions of [19, Thm. 3.4] are satisfied, then the vectors w in the additional constraints (45), identifying nonpredictive multicollinearities, are given by right singular vectors v' of A. If not, the nongeneric TLS solution still exists but in these cases, w can be any vector within the row space of A. These vectors still identify nonpredictive multicollinearities if the sum of squared projections of the rows of A onto w, i.e., $\alpha^2 = w^T A^T A w$, is small. This occurs when σ_{q+1} is not much larger than σ_{n+d}, which implies that w is most likely oriented along the lowest right singular vectors $v'_n, \cdots$ of A. However, large values of α will occur if at least one subset $Ax_i \approx b_i$ of $AX \approx B$ is highly incompatible. It affects the accuracy of the other solutions when solving the d-dimensional TLS problem. In this case, it is expected to obtain more accurate TLS solutions $\hat{x}_i, i = 1, \cdots, d$, by solving d separate one-dimensional TLS problems $Ax_i \approx b_i, i = 1, \cdots, d$.

4. Conclusions. Whenever multicollinearities, i.e., approximate dependencies, between the columns of A are as strong or even stronger than the linear relation between A and B, TLS problems $AX \approx B$ become very ill conditioned and even lack a solution satisfying the TLS criteria. This happens when the set of equations $AX \approx B$ is highly incompatible or when the data matrix A is (nearly) rank-deficient. In order to make these problems solvable and well conditioned, an extension of the generic TLS problem, called nongeneric TLS, has been proposed. Nongeneric TLS problems satisfy the same TLS criteria, as formulated in the generic TLS problem, but, in addition, impose constraints on the solution space. These additional constraints, which eliminate all multicollinearities between the columns of A from the TLS solution space, are scrutinized and compared with LS, LR, and PC regression. It is shown that LR and PC estimators eliminate the same multicollinearities from the LS estimator as does the nongeneric TLS estimator from the generic TLS estimator, thereby greatly reducing the estimator variances. In this sense we could thus say that nongeneric TLS stabilizes the coefficients of the generic TLS solution in errors-in-variables models, just as PC and LR stabilize the coefficients of the LS estimate in linear regression models, when multicollinearities are present in the data.

Finally, it should be noted that nongeneric problems do not occur if we are interested in estimating linear relationships between the columns of $[A; B]$ no matter which columns are used as right-hand side.

REFERENCES

[1] L. P. AMMANN AND J. W. VAN NESS, *Standard and robust orthogonal regression*, Comm. Statist. B—Simulation Comput., 18 (1989), pp. 145–162.

[2] M. AOKI AND P. C. YUE, *On a priori estimates of some identification methods*, IEEE Trans. Automat. Control., AC-15 (1970), pp. 541–548.

[3] T. F. CHAN AND P. C. HANSEN, *Computing truncated* SVD *least squares solutions by rank revealing* QR *factorizations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 519–530.

[4] N. R. DRAPER AND H. SMITH, *Applied Regression Analysis*, Second Edition, John Wiley, New York, 1981.

[5] W. A. FULLER, *Measurement Error Models*, John Wiley, New York, 1987.

[6] P. P. GALLO, *Properties of estimators in errors-in-variables models*, Ph.D. thesis, Institute of Statistics Mimeoseries # 1511, University of North Carolina, Chapel Hill, NC, October 1982.

[7] L. J. GLESER, *Estimation in a multivariate "errors in variables" regression model: Large sample results*, Ann. Statist., 9 (1981), pp. 24–44.

[8] G. H. GOLUB, *Some modified matrix eigenvalue problems*, SIAM Rev., 15 (1973), pp. 318–344.

[9] G. H. GOLUB AND C. REINSCH, *Singular value decomposition and least squares solutions*, Numer. Math., 14 (1970), pp. 403–420.

[10] G. H. GOLUB AND C. F. VAN LOAN, *An analysis of the total least squares problem*, SIAM J. Numer. Anal., 17 (1980), pp. 883–893.

[11] ———, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[12] D. M. HAWKINS, *On the investigation of alternative regressions by principal component analysis*, Appl. Statist., 22 (1973), pp. 275–286.

[13] R. R. HOCKING, *The analysis and selection of variables in linear regression*, Biometrics, 32 (1976), pp. 1–49.

[14] ———, *Developments in linear regression methodology* 1959–1982, Technometrics, 25 (1983), pp. 219–230.

[15] P. N. JAMES, P. SOUTER, AND D. C. DIXON, *Suboptimal estimation of the parameters of discrete systems in the presence of correlated noise*, Electron. Lett., 8 (1972), pp. 411–412.

[16] H. SCHNEEWEISS, *Consistent estimation of a regression with errors in the variables*, Metrika, 23 (1976), pp. 101–115.

[17] S. VAN HUFFEL, *Analysis of the total least squares problem and its use in parameter estimation*, Ph.D. thesis, Department of Electrical Engineering, K.U. Leuven, Leuven, Belgium, June 1987.

[18] S. VAN HUFFEL AND J. VANDEWALLE, *Subset selection using the total least squares approach in collinearity problems with errors in the variables*, Linear Algebra Appl., 88/89 (1987), pp. 695–714.

[19] ———, *Analysis and solution of the nongeneric total least squares problem*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 360–372.

[20] ———, *The Total Least Squares Problem: Computational Aspects and Analysis*, Frontiers in Applied Mathematics, Vol. 9, Society for Industrial and Applied Mathematics, Philadelphia, 1991.

[21] J. T. WEBSTER, R. F. GUNST, AND R. L. MASON, *Latent root regression analysis*, Technometrics, 16 (1974), pp. 513–522.

[22] R. H. ZAMAR, *Robust estimation in the errors-in-variables model*, Biometrika, 76 (1989), pp. 149–160.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 36–40, January 1992

004

SIGN-NONSINGULAR MATRIX PAIRS*

RICHARD A. BRUALDI† AND KEITH L. CHAVEY‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. This paper generalizes the notion of sign-nonsingular matrices to sign-nonsingular matrix pairs and proves an extremal theorem.

Key words. sign-nonsingular matrix, LU-factorization, indicator polynomial

AMS(MOS) subject classifications. 15A15, 15A09, 15A23

1. Introduction.

Let $S = [s_{ij}]$ $(1 \leq i, j \leq n)$ be a $(0, 1, -1)$-matrix of order n. Then S is a *sign-nonsingular matrix* (SNS-matrix) provided that each real matrix with the same sign pattern as S is nonsingular. There has been considerable recent interest in constructing and characterizing SNS-matrices [1], [4]. There has also been interest in strong forms of sign-nonsingularity [2]. In this paper we give a new generalization of SNS-matrices and investigate some of their basic properties.

Let $S = [s_{ij}]$ be a $(0, 1, -1)$-matrix of order n and let $C = [c_{ij}]$ be a real matrix of order n. The pair (S, C) is called a *matrix pair of order n*. Throughout, $X = [x_{ij}]$ denotes a matrix of order n whose entries are algebraically independent indeterminates over the real field. Let $S \circ X$ denote the Hadamard product (entrywise product) of S and X. We say that the pair (S, C) is a *sign-nonsingular matrix pair of order n*, abbreviated SNS-*matrix pair of order n*, provided that the matrix

$$A = S \circ X + C$$

is nonsingular for all positive real values of the x_{ij}. If $C = O$ then the pair (S, O) is a SNS-matrix pair if and only if S is a SNS-matrix. If $S = O$ then the pair (O, C) is a SNS-matrix pair if and only if C is nonsingular. Thus SNS-matrix pairs include both nonsingular matrices and sign-nonsingular matrices as special cases.

The pairs (S, C) with

$$S = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad C = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

and

$$S = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \qquad C = \begin{bmatrix} 0 & 0 & 1 \\ 0 & 2 & 0 \\ 3 & 0 & 0 \end{bmatrix}$$

are examples of SNS-matrix pairs.

Let (S, C) be a SNS-matrix pair of order n. In this note we prove that the number of nonzero entries of S is at most $(n^2 + 3n - 2)/2$. This result extends a theorem of Gibson [3] for SNS-matrices to SNS-matrix pairs. We also strengthen a theorem of

* Received by the editors May 13, 1991; accepted for publication (in revised form) July 15, 1991.

† Department of Mathematics, University of Wisconsin, Madison, WI 53706 (brualdi@math.wisc.edu or na.brualdi@na-net.ornl.gov). This research was partially supported by National Science Foundation grant DMS-8901445 and National Security Agency grant MDA904-89-H-2060.

‡ Department of Mathematics, University of Wisconsin, River Falls, WI 54022 (Keith.L.Chavey@uwrf.edu). This research was partially supported by National Security Agency grant MDA904-89-H-2060 and an Office of Education Fellowship administered by the Department of Mathematics of the University of Wisconsin at Madison.

Murota [5] and apply it to obtain a more general bound on the number of nonzero entries of S in terms of a parameter which is naturally associated with a SNS-matrix pair.

2. Main results. Let (S, C) be a matrix pair of order n. The determinant

$$\det(S \circ X + C)$$

is a polynomial in the indeterminates of X of degree at most n over the real field. We call this polynomial the *indicator polynomial* of the matrix pair (S, C) because of the following proposition.

THEOREM 2.1. *The matrix pair (S, C) is a* SNS*-matrix pair if and only if all the nonzero coefficients in its indicator polynomial have the same sign and there is at least one nonzero coefficient.*

Proof. Assume that (S, C) is a SNS-matrix pair. Clearly the indicator polynomial has a nonzero coefficient. Consider a monomial

$$b_{i_1,\cdots,i_k;j_1,\cdots,j_k} x_{i_1 j_1} \cdots x_{i_k j_k} \tag{1}$$

occurring in the indicator polynomial with a nonzero coefficient. By taking the x_{ij} that occur in (1) large and all others small, we see that any monomial that occurs in the indicator polynomial with a nonzero coefficient can be made to dominate all others. Hence all the nonzero coefficients have the same sign. The converse is immediate. □

For SNS-matrix pairs (S, C) with $C = O$ the indicator polynomial is a homogeneous polynomial of degree n. In this case Theorem 2.1 is a standard fact about SNS-matrices.

The following is a theorem of Murota [5].

THEOREM 2.2. *Let S and C be real matrices of order n, and let $A = S \circ X + C$. Assume that* $\det A$ *is a nonzero scalar. Then there exist permutation matrices P and Q such that PAQ has an LU-factorization*

$$PAQ = LU, \tag{2}$$

where L is a lower triangular real matrix, with 0*s above and nonzeros on its main diagonal, and where U is an upper triangular matrix whose entries are polynomials of degree at most* 1 *in the entries of X, with* 0*s below and* 1*s on its main diagonal.*

The next theorem is a strengthening of Theorem 2.2.

THEOREM 2.3. *Let S and C be real matrices of order n, and let $A = S \circ X + C$. Assume that* $\det A$ *is a nonzero scalar. Then there exist permutation matrices P and Q such that PSQ is a strictly upper triangular matrix. The matrix S has at most $n(n-1)/2$ nonzero entries, and if equality holds then for the same permutation matrices P and Q, PAQ is an upper triangular matrix.*

Proof. First suppose that $\det(A)$ is a nonzero scalar, and let P, Q, L, U be as in Theorem 2.2. We claim that for each $k = 2, \cdots, n$, there are entries $z_1, \cdots, z_{k-1}$ of X such that the entry in row i of column k of U is a linear polynomial in $z_1, \cdots, z_i$ $(i = 1, \cdots, k-1)$. This follows by induction using the fact that each entry of A contains at most one of the indeterminates of X. Since the first row of PAQ equals a nonzero scalar multiple of the first row of U, each entry in the first row of U contains at most one of the indeterminates of X. The second row of PAQ equals a nonzero scalar multiple of the second row of U plus a scalar multiple of the first row of U. Hence each entry of U in its second row contains at most one indeterminate not in the

entry of U immediately above it. Continuing like this we verify the claim. If z_i occurs in the (i,k) position of U, then z_i occurs in the (i,k) position of PAQ $(1 \le i \le k-1)$, and hence cannot occur on or below the main diagonal of PAQ. Therefore PSQ is a strictly upper triangular matrix and hence S has at most $n(n-1)/2$ nonzero entries.

Now assume that S has exactly $n(n-1)/2$ nonzero entries. Suppose that there is a nonzero entry of L below its main diagonal, say, in row i. Then the ith main diagonal entry of $PAQ = LU$ is not a scalar, contradicting the fact that PSQ is strictly upper triangular. Therefore L is a diagonal matrix and PAQ is upper triangular. □

In the proof of our main theorem we shall use the following theorem of Gibson [3]. If B is a real matrix, then $|B|$ denotes the matrix obtained from B by replacing each entry with its absolute value.

THEOREM 2.4. *The maximum number of nonzero entries in a* SNS*-matrix S of order n equals*

$$\frac{n^2+3n-2}{2}$$

with equality if and only if there exist permutation matrices such that $P|S|Q = T_n$ where

$$T_n = \begin{bmatrix} 1 & 1 & \cdots & 1 & 1 & 1 \\ 1 & 1 & \cdots & 1 & 1 & 1 \\ 0 & 1 & \cdots & 1 & 1 & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 1 & 1 \\ 0 & 0 & \cdots & 0 & 1 & 1 \end{bmatrix}. \tag{3}$$

We note for later use that each submatrix of T_n of order $n-1$ has all 1s on its main diagonal.

We now obtain a bound on the number of nonzero entries of S in a SNS-matrix pair (S,C) in terms of the degree of the indicator polynomial. We denote the strictly upper triangular (0,1)-matrix of order m with all 1s above the main diagonal by U_m. The all 1s matrix of size m by p is denoted by $J_{m,p}$.

THEOREM 2.5. *Let (S,C) be a* SNS*-matrix pair of order n whose indicator polynomial has degree k. Then the number of nonzero entries of S is at most*

$$\binom{n}{2} + 2k - 1 \quad \text{if } k \neq 0,$$

$$\binom{n}{2} \quad \text{if } k = 0.$$

If equality occurs, then there exists an integer r with $0 \le r \le n-k$ and permutation matrices P and Q such that

$$|S| = P \begin{bmatrix} U_r & J_{r,k} & J_{r,n-k-r} \\ O & T_k & J_{k,n-k-r} \\ O & O & U_{n-k-r} \end{bmatrix} Q. \tag{4}$$

Moreover, for each k with $0 \le k \le n$ there exists a SNS*-matrix pair (S,C) such that $|S|$ satisfies* (4).

Proof. If $k = 0$, then the theorem follows from Theorem 2.3.[1] For the remainder of the proof we assume that $k \neq 0$.

[1] The matrices T_k and U_{n-k-r} are vacuous.

First suppose that $k = n$. By Theorem 2.1 all the nonzero coefficients of the indicator polynomial have the same sign, in particular the coefficients of the monomials of degree n have the same sign. It follows that S is a SNS-matrix, and hence by Theorem 2.4 the theorem holds in this case.[2]

Next suppose that $0 < k < n$. Without loss of generality, we assume that

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}, \tag{5}$$

where A_{11} is a matrix of order k whose main diagonal entries have degree equal to 1, and $\det A_{22}$ is a nonzero scalar. We assume that S, C, and X are correspondingly partitioned into blocks S_{ij}, C_{ij} and X_{ij}, respectively. By Theorem 2.3 S_{22} has at most $(n-k)(n-k-1)/2$ nonzero entries. It follows that (S_{11}, C_{11}) is a SNS-matrix pair of order k whose indicator polynomial has degree k, and hence, as above, S_{11} is a SNS-matrix. By Theorem 2.4 the number of nonzero entries of S_{11} is at most $(k^2+3k-2)/2$. Since A_{22} is nonsingular, we may assume that the minors of its main diagonal entries are nonzero. We claim that diagonally opposite entries of S_{12} and S_{21} cannot both be nonzero, and hence the total number of nonzero entries in S_{12} and S_{21} is at most $k(n-k)$. Suppose to the contrary that there exist integers p and q with $1 \le p \le k < q \le n$ such that $s_{pq}s_{qp} \neq 0$. Then $x_{pq}x_{qp}x_{11}\cdots x_{p-1,p-1}x_{p+1,p+1}\cdots x_{kk}$ appears in a nonzero term of degree at least $k+1$ in the indicator polynomial of (S, C), contradicting the assumption that the indicator polynomial has degree k. Putting these three bounds together we obtain the first assertion of the theorem.

Suppose that S has exactly

$$\binom{n}{2} + 2k - 1$$

nonzero entries. Then S_{11} has exactly $(k^2+3k-2)/2$ nonzero entries and hence by Theorem 2.4 we may assume that $|S_{11}| = T_k$. Also S_{22} has exactly $(n-k)(n-k-1)/2$ nonzero entries and hence by Theorem 2.3 we may assume that $S_{22} = U_{n-k}$ and A_{22} is an upper triangular matrix with nonzero scalars on its main diagonal. Moreover, $|S_{12}| + |S_{21}^t| = J_{k,n-k}$. We claim that row q of S_{21} and column q of S_{12} cannot both contain a nonzero entry. Suppose to the contrary that there exist integers p_1, p_2, and q with $1 \le p_1, p_2 \le k < q \le n$ such that $s_{p_1q}s_{qp_2} \neq 0$. The property of T_n noted following its definition implies that the minor of the (p_1, p_2) entry of A_{11} has degree $k-1$. The minor of the $(q-k)$th diagonal entry of A_{22} is a nonzero scalar. Hence there is a nonzero term of degree $k+1$ in the indicator polynomial of A—a contradiction. Thus there exists an integer r such that $|S_{21}|$ has r rows of 1s with corresponding rows of 0s in $|S_{12}^t|$, and $k-r$ rows of 0s with corresponding rows of 1s in $|S_{12}^t|$. It is now easy to see that there are permutation matrices P and Q such that (4) holds. The theorem now follows. □

Let (S, C) be a SNS-matrix pair of order n. If the degree of the indicator polynomial is n, then as shown in the proof of Theorem 2.5, S is a SNS-matrix. If the degree of the indicator polynomial is 0, then it follows from Theorem 2.3 that there exists a SNS-matrix S' which can be obtained from S by replacing n 0s with 1s. We do not know whether it is always possible to find a SNS-matrix S^* which can be obtained from S by replacing some of its 0s with ± 1s.

[2] The matrices U_r and U_{n-k-r} are vacuous.

REFERENCES

[1] R. A. BRUALDI AND B. L. SHADER, *On sign-nonsingular matrices and the conversion of the permanent into the determinant*, in Applied Geometry and Discrete Mathematics, The Victor Klee Festschrift, P. Gritzmann and B. Sturmfels, eds., American Mathematical Society, Providence, RI, 1991, pp. 117–134.

[2] J. DREW, C. R. JOHNSON, AND P. VAN DEN DRIESSCHE, *Strong forms of nonsingularity*, Linear Algebra Appl., 162 (1992), to appear.

[3] P. M. GIBSON, *Conversion of the permanent into the determinant*, Proc. Amer. Math. Soc., 27 (1971), pp. 471–476.

[4] V. KLEE, R. LADNER AND R. MANBER, *Signsolvability revisited*, Linear Algebra Appl., 59 (1984), pp. 131–157.

[5] K. MUROTA, LU-*decomposition of a matrix with entries of different kinds*, Linear Algebra Appl., 49 (1983), pp. 275–283.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 41–45, January 1992

005

ON THE SUM OF THE LARGEST EIGENVALUES OF A SYMMETRIC MATRIX*

MICHAEL L. OVERTON† AND ROBERT S. WOMERSLEY‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. The sum of the largest k eigenvalues of a symmetric matrix has a well-known extremal property that was given by Fan in 1949 [*Proc. Nat. Acad. Sci.*, 35 (1949), pp. 652–655]. A simple proof of this property, which seems to have been overlooked in the vast literature on the subject and its many generalizations, is discussed. The key step is the observation, which is neither new nor well known, that the convex hull of the set of projection matrices of rank k is the set of symmetric matrices with eigenvalues between 0 and 1 and summing to k. The connection with the well-known Birkhoff theorem on doubly stochastic matrices is also discussed. This approach provides a very convenient characterization for the subdifferential of the eigenvalue sum, described in a separate paper.

Key words. eigenvalue sum, convex hull, projection matrix, Fan's theorem

AMS(MOS) subject classifications. 15A42, 15A45

Let A be an n by n real symmetric matrix, with eigenvalues

$$\lambda_1 \geqq \cdots \geqq \lambda_n$$

and a corresponding orthonormal set of eigenvectors $q_1, \cdots, q_n$; thus

$$A = Q\Lambda Q^T, \qquad Q^TQ = I_n,$$

where $\Lambda = \text{Diag}(\lambda_1, \cdots, \lambda_n)$ and $Q = [q_1, \cdots, q_n]$. In 1949, the following theorem was proved by Fan [Fan].

THEOREM 1.

$$(1) \qquad \sum_{i=1}^{k} \lambda_i = \max_{X^TX = I_k} \text{tr } X^TAX.$$

Here I_k is the identity matrix of order k, and hence X is a matrix whose columns are k orthonormal vectors in $\mathscr{R}^n$. (All matrices are assumed to be real, but extension to the case where A is complex Hermitian is straightforward.)

In the case $k = 1$, the theorem reduces to the Rayleigh principle; in the case $k = n$, it states only that the trace of a square matrix is the sum of its eigenvalues. Cases $1 < k < n$ are natural generalizations of the Rayleigh principle and are also reminiscent of the Courant–Fischer theorem, which first appeared in 1905 [Fis]. (The Courant and Fischer versions differ in the infinite-dimensional case.) The Courant–Fischer theorem may be stated succinctly as follows.

THEOREM 2.

$$\lambda_k = \max_{X^TX = I_k} \min_{v^Tv = 1} v^TX^TAXv.$$

* Received by the editors March 21, 1991; accepted for publication (in revised form) July 15, 1991.

† Computer Science Department, Courant Institute of Mathematical Sciences, New York University, New York, New York 10012 (overton@cs.nyu.edu). The work of this author was supported in part by National Science Foundation grant CCR-9101640.

‡ School of Mathematics, University of New South Wales, Kensington, New South Wales, Australia (rsw@hydra.maths.oz.au). The work of this author was supported in part by U.S. Department of Energy contract DEFG0288ER25053 while the author was visiting New York University.

For both Theorems 1 and 2, it is immediately clear that the right-hand side is greater than or equal to the left by taking $X = [q_1, \cdots, q_k]$. The proof of the Courant–Fischer theorem is completed by the following argument: for any X, take v to be orthogonal to the first $k-1$ rows of Q^TX; then $v^TX^TAXv \leqq \lambda_k$. Completing the proof of Fan's theorem is somewhat harder, however. Clearly, letting $Y = Q^TX$, the result is equivalent to the following:

$$\sum_{i=1}^{k} \lambda_i = \max_{Y^TY = I_k} \operatorname{tr} Y^T\Lambda Y.$$

Since Λ is diagonal,

$$\operatorname{tr} Y^T\Lambda Y = \sum_{j=1}^{k} \sum_{i=1}^{n} \lambda_i y_{ij}^2, \tag{2}$$

and one might suppose the rest of the proof to be straightforward. Only a few lines of inequalities are in fact required, but deriving these is not a completely trivial exercise; for the details, see Fan's original proof. Obtaining Theorem 1 as a consequence of Theorem 2 does not seem to be any easier. Another approach uses properties of doubly stochastic matrices and will be described below. Therefore, although it is hard to imagine that Rayleigh, Fischer, or Courant would have been surprised by Fan's result, it is quite plausible that they were not aware of it. Actually, Marshall and Olkin [MO] point out that Theorem 1 is a special case of a much more general but less well known result of von Neumann dating to 1937 [vN]:

$$\sum_{i=1}^{n} \sigma_i\tau_i = \max_{U^TU = I_n, V^TV = I_n} \operatorname{tr} UBVC,$$

where $\sigma_1 \geqq \cdots \geqq \sigma_n$ and $\tau_1 \geqq \cdots \geqq \tau_n$ are, respectively, the singular values of the n by n nonsymmetric matrices B and C. See [MO, Chap. 20] for the proof.

There is a vast literature on various inequalities for sums and products of eigenvalues of symmetric matrices; see particularly [BB, Chap. 2], [Bel, Chap. 8], [MM, Part II], [Fri], and references therein. However, we are not aware that any of the many results available in the literature are particularly relevant to the discussion here, except as noted below.

The purpose of this note is to describe an easy but interesting result, which then leads to a trivial proof of Fan's theorem. This result, while not new, is so beautifully simple that its obscurity is surprising.

THEOREM 3. *Let*

$$\Omega_1 = \{ YY^T : Y^TY = I_k \}$$

and

$$\Omega_2 = \{ W : W = W^T, \operatorname{tr} W = k, 0 \leqq W \leqq I \}.$$

Here Y has dimension n by k, so that YY^T is a projection matrix of order n and rank k, and W has dimension n by n, with the last condition meaning that W and $I - W$ are both positive semidefinite. Then Ω_2 is the convex hull of Ω_1, and Ω_1 is the set of extreme points of Ω_2.

Theorem 1 then follows as a consequence, because

$$\operatorname{tr} Y^T\Lambda Y = \operatorname{tr} YY^T\Lambda, \tag{3}$$

and maximizing this linear function over $YY^T \in \Omega_1$ is equivalent to maximizing it over $W \in \Omega_2$. Since

$$\operatorname{tr} W\Lambda = \sum_{i=1}^{n} \lambda_i w_{ii},$$

the fact that the maximum value is $\sum_{i=1}^{k} \lambda_i$ follows trivially from the conditions on W. Equivalently,

$$\sum_{i=1}^{k} \lambda_i = \max_{V \in \Omega_2} \operatorname{tr} VA, \tag{4}$$

as $V \in \Omega_2$ if and only if $W = Q^T V Q \in \Omega_2$.

We derived Theorem 3 before we found it in the literature. Our proof is as follows. The fact that any convex combination of elements of Ω_1 lies in Ω_2 is immediate. Also, using the spectral decomposition of W, which has eigenvalues lying between 0 and 1 that sum to k, it is clear that any element of Ω_2 with rank greater than k is not an extreme point. The only candidates for extreme points, then, are those with rank k, i.e., the elements of Ω_1. But it is not possible that some rank k elements are extreme points and others are not, since the definition of Ω_2 does not in any way distinguish between different rank k elements. Since a compact convex set must have extreme points, and is in fact the convex hull of its extreme points, the proof is complete.

A slightly different proof of this theorem was given in 1971 by Fillmore and Williams [FW], a paper whose existence was recently brought to our attention by H. Woerdeman and C.-K. Li. The paper is primarily concerned with the numerical range of a matrix, which in the symmetric case is simply the line segment $[\lambda_n, \lambda_1]$, together with generalizations of this notion, and is referenced in the subsequent literature on generalized numerical ranges, e.g., [GS], [Poo]. However, [FW] does not seem to be well known in the general linear algebra community. We do not know of an explicit statement of Theorem 3 that appeared before 1971. S. Friedland has pointed out that the result may be obtained, in fact in a more general form, by using Theorem 4 below in conjunction with the classical technique of majorization [MO] and that, furthermore, the result is related to a 1950 theorem of Lidskii [Kat, p. 145]; however, such approaches to Theorem 3 are considerably more complicated than the trivial proof just given. It seems very surprising that Theorem 3 is so little known, especially given its resemblance to the following famous theorem.

THEOREM 4. *Let Ω_3 be the set of n by n permutation matrices, and let Ω_4 be the set of n by n doubly stochastic matrices, i.e., nonnegative matrices whose row sums and column sums are one. Then Ω_4 is the convex hull of Ω_3, and Ω_3 is the set of extreme points of Ω_4.*

This theorem is usually attributed to Birkhoff, who discovered it in 1946, although Chvátal [Chv, Chap. 20] notes that it was given by König in 1936 [Kön, p. 381]. It has been rediscovered, reproved in various ways, and generalized by many authors; see [MO, Chap. 2] for an extensive discussion.

Theorem 4 has also been used as the basis for proving Fan's theorem, e.g., [RV, Chap. 6] and [MO, Chap. 20]. The former reference proves Theorem 1 as a consequence of some general inequalities proved using Theorem 4; the latter gives a more direct proof as follows. We have already noted that the maximization objective in Theorem 1 may be written in the forms (2) and (3); another equivalent form is

$$\lambda^T Z e, \tag{5}$$

where e is the k-dimensional vector with all elements equal to one, $\lambda = [\lambda_1, \cdots, \lambda_n]^T$, and Z is the matrix of dimension n by k whose elements are defined by $z_{ij} = y_{ij}^2$. Since Y has orthonormal columns, the column sums of Z are one and the row sums are less than or equal to one. Clearly, there exists an n by n doubly stochastic matrix whose first k columns are the columns of Z (simply extend Y to a square orthogonal matrix). Consequently, maximizing (5) over permissible values of Z cannot give a larger value than maximizing

$$\lambda^T S f \tag{6}$$

over all n by n doubly stochastic matrices S, where f is the vector with first k elements equal to one and last $n - k$ elements equal to zero. By Theorem 4, this is equivalent to maximizing (6) over the permutation matrices, giving an upper bound of $\sum_{i=1}^{k} \lambda_i$ for the maximization objective and completing the proof of Theorem 1. (The last step is actually slightly different in [MO], using majorization instead of Theorem 4.) Note, by the way, that not all doubly stochastic matrices can be obtained using such a construction ([MO, Chap. 2]). This does not affect the proof since, as noted at the beginning of the discussion, the lower bound for the maximization objective is immediate.

The parallels and differences in the two proofs of Theorem 1 given above are quite striking. The first proof used Theorem 3, writing the maximization over the nonconvex set Ω_1 and noting that maximizing instead over its convex hull Ω_2 led to the desired conclusion. The second proof used Theorem 4, writing the maximization over the convex set Ω_4 and noting that maximizing instead over its extreme points Ω_3 led to the desired conclusion. (For a third proof, see [RW].)

It is a well-known consequence of Fan's theorem that, because the left-hand side of (1) is the pointwise maximum of the linear functions on the right-hand side, the sum of the first k eigenvalues of a matrix is a convex function of its elements. The same property is also deduced from (4). We show in [OW] that formulas for the subdifferential of the eigenvalue sum may be derived from either of these two max formulations, but that the latter is particularly useful, for a reason related to the discussion in the preceding paragraph; Ω_2 is convex, while Ω_3 is not. It follows from either derivation that the sum of the first k eigenvalues is a smooth function of the matrix elements, i.e., the subdifferential reduces to a gradient, if and only if $\lambda_k > \lambda_{k+1}$. Our interest in this subject arose from consideration of minimizing sums of eigenvalues of matrices that are specified only in part; see [CDW] for some applications and [OW] for further details. For the important special case $k = 1$, see [Ove]. Finally, [OW] also addresses extremal properties of sums of the largest eigenvalues in absolute value, giving some interesting generalizations of the results discussed here.

REFERENCES

[BB] E. F. BECKENBACK AND R. BELLMAN, *Inequalities*, Springer-Verlag, New York, 1971.

[Bel] R. BELLMAN, *Introduction to Matrix Analysis*, Second Edition, McGraw-Hill, New York, 1970.

[Chv] V. CHVÁTAL, *Linear Programming*, W. H. Freeman, New York, 1980.

[CDW] J. CULLUM, W. E. DONATH, AND P. WOLFE, *The minimization of certain nondifferentiable sums of eigenvalues of symmetric matrices*, Math. Programming Stud., 3 (1975), pp. 35–55.

[Fan] K. FAN, *On a theorem of Weyl concerning the eigenvalues of linear transformations*, Proc. Nat. Acad. Sci. U.S.A., 35 (1949), pp. 652–655.

[FW] P. A. FILLMORE AND J. P. WILLIAMS, *Some convexity theorems for matrices*, Glasgow Math. J., 12 (1971), pp. 110–117.

[Fis] E. FISCHER, *Über quadratische Formen mit rellen Koeffizienten*, Monatsh. Math. Physik, 16 (1905), pp. 234–249.

[Fri] S. FRIEDLAND, *Convex spectral functions*, Linear and Multilinear Algebra, 9 (1981), pp. 299–316.

[GS] M. GOLDBERG AND E. G. STRAUS, *Elementary inclusion relations for generalized numerical ranges*, Linear Algebra Appl., 18 (1977), pp. 1–24.

[Kat] T. KATO, *A Short Introduction to Perturbation Theory for Linear Operators*, Springer-Verlag, New York, 1982.

[Kön] D. KÖNIG, *Theory of Finite and Infinite Graphs*. Birkhäuser, Boston, 1990 (translation of *Theorie der endlichen und unendlichen Graphen*, Akademische Verlagsgesellschaft Leipzig, 1936).

[MM] M. MARCUS AND H. MINC, *A Survey of Matrix Theory and Matrix Inequalities*, Allyn and Bacon, Boston, 1964.

[MO] A. W. MARSHALL AND I. OLKIN, *Inequalities: Theory of Majorization and Its Applications*, Academic Press, New York, 1979.

[Ove] M. L. OVERTON, *Large-scale optimization of eigenvalues*, SIAM J. Optim., 1992, to appear.

[OW] M. L. OVERTON AND R. S. WOMERSLEY, *Optimality conditions and duality theory for minimizing sums of the largest eigenvalues of symmetric matrices*, Computer Science Department Report 566, New York University, New York, NY, 1991; submitted to Math. Programming.

[Poo] Y.-T. POON, *Another proof of a result of Westwick*, Linear and Multilinear Algebra, 9 (1980), pp. 35–37.

[RW] F. RENDL AND H. WOLKOWICZ, *Applications of parametric programming and eigenvalue maximization to the quadratic assignment problem*, Math. Programming, to appear.

[RV] A. W. ROBERTS AND D. E. VARBERG, *Convex Functions*, Academic Press, New York, 1973.

[vN] J. VON NEUMANN, *Some matrix inequalities and metrization of matrix space*, Tomsk Univ. Rev., 1 (1937), pp. 286–300; Reprinted in John v. Neumann: Collected Works, Vol. IV, A. H. Taub, ed., Macmillan, New York, 1962, pp. 205–219.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 46–67, January 1992

006

REGULARIZATION OF DESCRIPTOR SYSTEMS BY DERIVATIVE AND PROPORTIONAL STATE FEEDBACK*

ANGELIKA BUNSE-GERSTNER†, VOLKER MEHRMANN‡, AND NANCY K. NICHOLS§

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. For linear multivariable time-invariant continuous or discrete-time singular systems it is customary to use a proportional feedback control in order to achieve a desired closed loop behaviour. Derivative feedback is rarely considered. This paper examines how derivative feedback in descriptor systems can be used to alter the structure of the system pencil under various controllability conditions. It is shown that derivative and proportional feedback controls can be constructed such that the closed loop system has a given form and is also regular and has index at most 1. This property ensures the solvability of the resulting system of dynamic-algebraic equations. The construction procedures used to establish the theory are based only on orthogonal matrix decompositions and can therefore be implemented in a numerically stable way. The problem of pole placement with derivative feedback alone and in combination with proportional state feedback is also investigated. A computational algorithm for improving the "conditioning" of the regularized closed loop system is derived.

Key words. differential-algebraic systems, singular systems, controllability, regularizability, numerical stability, optimal conditioning

AMS(MOS) subject classifications. 93B10, 93B05, 93B40, 93B52, 93B35, 65F35

1. Introduction.

We consider linear time-invariant continuous or discrete-time dynamical systems of the form

$$E\dot{x} := E dx/dt = Ax(t) + Bu(t), \qquad x(t_0) = x_0, \tag{1}$$

$$y(t) = Cx(t), \tag{2}$$

or

$$Ex_{k+1} = Ax_k + Bu_k, \qquad x_0 \text{ given}, \tag{3}$$

$$y_k = Cx_k, \tag{4}$$

where $E, A \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$, $C \in \mathbb{R}^{p,n}$, and $\operatorname{rank}(B) = m \leqq n$, $\operatorname{rank}(C) = p \leqq n$. Here $x(t)$ or $x_k \in \mathbb{R}^n$ is the state, $y(t)$ or $y_k \in \mathbb{R}^p$ is the output, and $u(t)$ or $u_k \in \mathbb{R}^m$ is the input or control of the system. Such systems are called *descriptor* or *generalized state-space* systems. In the case $E = I$, the identity matrix, we refer to (1), (2) and (3), (4) as *standard* systems.

Descriptor systems arise naturally in a variety of circumstances [19], [13] and have recently been investigated in a number of papers [18], [4]–[7], [10]–[12], [14], [16], [17], [20]–[26]. The response of a descriptor system can be described in terms of the eigenstructure of the matrix pencil

$$\alpha E - \beta A. \tag{5}$$

* Received by the editors February 25, 1991; accepted for publication (in revised form) July 31, 1991. This research was supported in part by FSP Mathematisierung. The research of the first two authors was supported in part by SFB 343, Diskrete Strukturen in der Mathematik.

† Fachbereich Mathematik und Informatik, Universität Bremen, Postfach 330440, D-2800, Bremen 33, Germany (abg@informatik.uni-bremen.de).

‡ Fakultät für Mathematik, Universität Bielefeld, Postfach 8640, D-4800, Bielefeld 1, Germany (umatf108@dbiunill.bitnet). Present address, Institut für Geometrie und Praktische Mathematik, RWTH Aachen, Templergraben 55, D-5100, Aachen, Germany.

§ Department of Mathematics, University of Reading, Box 220, Reading RG6 2AX, United Kingdom (smsnicho@am.rdg.ac.uk).

In order to alter the behaviour of the system, it is customary to use proportional state or output feedback to modify the matrix A. The closed loop system pencil then becomes

$$\alpha E - \beta(A + BFC), \tag{6}$$

where the control is taken to be $u = Fy + v$ or $u_k = Fy_k + v_k$. In the theory of matrix pencils, the roles of E and A are interchangeable, but the analogous use of derivative state or output feedback in multivariable systems has not been investigated much in the literature. Derivative feedback modifies the matrix E, and the closed loop system pencil then becomes

$$\alpha(E + BGC) - \beta A, \tag{7}$$

where the control is taken to be $u = -G\dot{y} + v$ or $u_k = -Gy_{k+1} + v_k$.

Derivative information has long been used in the practical design of PD controllers. Recently it has been applied in the construction of a discrete-time observer using both current and past output data in the current state estimation [18]. This leads to a system for the error with a matrix pencil of the form

$$\alpha(E + GC) - \beta(A + FC). \tag{8}$$

Even for nonsingular E the use of the output derivative information is valuable, and it is shown in [18] that choosing G such that the condition number of $E + GC$ is small gives improved state estimates.

Theoretical aspects of derivative feedback for descriptor systems are studied in a few recent papers [4], [16], [21], [26]. A control of the restricted form $u = F(\alpha x - \dot{x}) + v$ is discussed in [4], [21], [26]. In [16] a full state feedback of the form $u = -G\dot{x} + Fx + v$ is studied for the pole placement problem. In these papers the main task of the derivative feedback is to transform E into a nonsingular matrix $E + BG$. Complete controllability and regularity of the system pencil (5) is assumed.

In this paper we investigate both derivative and proportional state feedback and examine the properties that can be achieved with these types of feedback under various controllability conditions. Applications to pole placement are also considered. Detailed proofs of results previously presented in [2] are given and new results on strongly controllable systems are derived.

The principal aim of this paper is to provide *numerically stable* methods for constructing the feedback controllers based on orthogonal matrix decompositions [9]. Parts of the mathematical theory developed here have been derived concurrently by Dai [6]. Additional assumptions are required in [6], however, and the techniques used for constructing the feedback matrices in [6] are not suitable for numerical computation. It is assumed in [6] that the matrix pencil (5) associated with the system (1), (2) or (3), (4) is regular. This assumption is not required to establish the results presented here. Furthermore, in [6] it is necessary to transform the system into separate "fast" and "slow" subsystems in order to obtain the feedback controls. This transformation is well known to be computationally unreliable [22]. The proofs given here do not require this transformation; and it is shown specifically how to select a feedback in a numerically stable way so as to ensure that the closed loop system is regular and that the controllability (observability) properties of the system are preserved.

In the next section of the paper we introduce notation and examine how the response of the system depends on the eigenstructure of the associated matrix pencil. Definitions of *complete* and *strong* controllability are given and the significance of these conditions is discussed.

In § 3 we summarize the system properties that can be achieved by derivative and proportional *state* feedback under the different controllability conditions. It is shown

that a system that is *completely* controllable can be transformed into a *standard* system by derivative feedback. It is shown, furthermore, that a system that is *strongly* controllable can be transformed into a regular system of index at most 1 (that is, a system in which impulses are excluded) by either proportional *or* derivative state feedback. Derivative feedback can be used, however, to increase the explicit degrees of freedom defining the solution space (*reachable subspace*) of the system. The construction of the required feedback matrices is obtained by reducing the system pencil to an equivalent "canonical" form using orthogonal transformations that are numerically stable [9]. Most but not all of the conclusions of this section can also be achieved by *output* derivative and proportional feedback. Preliminary results are presented in [1] and [2].

In § 4 applications to the pole placement problem are discussed. The extent to which the poles can be assigned by derivative and/or proportional state feedback whilst retaining regularity is examined under the different controllability conditions.

In the final section we discuss a numerical technique for regularizing the dynamical part of a descriptor system by a derivative feedback which optimizes the conditioning of $E + BG$. The results of the paper are then summarized, and concluding remarks are given.

2. Definitions and properties. The system equations (1) and (3) are said to be *solvable* if and only if the system pencil (5) is *regular*, that is,

$$\det(\alpha E - \beta A) \not\equiv 0 \quad \forall (\alpha, \beta) \in \mathbb{C}^2 \setminus \{0,0\}. \tag{9}$$

For solvable systems there exist unique solutions for any *sufficiently smooth* input and any admissible initial conditions corresponding to an admissible input [3], [25]. The behaviour of the system response is then governed by the eigenstructure of the system pencil. In the next section we examine the eigenstructure of generalized state-space systems and in the following section we define conditions that ensure the controllability (observability) of the system.

2.1. Eigenstructure of descriptor systems. For a regular pencil *generalized eigenvalues* are defined to be pairs $(\alpha_j, \beta_j) \in \mathbb{C}^2$ such that

$$\det(\alpha_j E - \beta_j A) = 0, \qquad j = 1, 2, \cdots, n. \tag{10}$$

Observe that pairs (α_j, β_j) and $(t\alpha_j, t\beta_j)$, $t \in \mathbb{C} \setminus \{0\}$ are identified. Eigenvalue pairs (α_j, β_j) where $\beta_j \neq 0$ are said to be *finite* and, without loss of generality, can be taken to have the "value" $\lambda_j = \alpha_j/\beta_j$. Pairs where $\beta_j = 0$ are said to be *infinite* eigenvalues. The maximum number of finite eigenvalues that a pencil can have is less than or equal to the rank of E. (For a pencil that is *not* regular, the generalized eigenvalues are similarly defined as pairs (α_j, β_j) such that the pencil loses rank.)

For regular pencils the solution of the system equations can be characterized in terms of the Kronecker canonical form (KCF) [8]. In this case there exist nonsingular matrices X and Y (representing the right and left generalized eigenvectors and principal vectors of the system pencil, respectively) which transform E and A into the KCF:

$$Y^T E X = \begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix}, \qquad Y^T A X = \begin{bmatrix} J & 0 \\ 0 & I \end{bmatrix}. \tag{11}$$

Here J is a Jordan matrix corresponding to the finite eigenvalues of the pencil and N is a nilpotent Jordan matrix such that $N^m = 0$, $N^{m-1} \neq 0$, corresponding to the infinite eigenvalues. The *index* of the system, denoted by $\operatorname{ind}_\infty(E, A)$, is defined to be equal to the degree m of nilpotency. (For pencils that are not regular, the KCF can also be defined and the index is then given similarly by the dimension of the largest nilpotent block in the KCF. See [1] and [8].)

We observe that if a descriptor system is regular then it is of index 0 if and only if E is nonsingular. In this case the system can be reformulated as a standard system and the usual theory applies. In practice the reduction to standard form can be numerically unstable, however, if E is ill conditioned with respect to inversion. Hence, even for index 0 systems, it may be preferable to work directly with the generalized state-space form.

We observe that if a descriptor system is regular then it is of index at most 1 if and only if it has exactly $q = \operatorname{rank}(E)$ finite eigenvalues. Conditions for the system to be regular and of index less than or equal to 1 are given in the following lemma [10]. (Here and in the following we denote the nullspace of a matrix M by $\mathcal{N}(M)$.)

LEMMA 1. *Let $E, A \in \mathbb{R}^{n,n}$. Let S_∞ and T_∞ be full rank matrices whose columns span the null spaces $\mathcal{N}(E)$ and $\mathcal{N}(E^T)$, respectively. Then the following are equivalent*:

(i) $\alpha E - \beta A$ *is regular and* $\operatorname{ind}_\infty(E, A) \leqq 1$,

(ii) $\operatorname{rank}([E, AS_\infty]) = n$,

(iii) $$\operatorname{rank}\left(\begin{bmatrix} E \\ T_\infty^H A \end{bmatrix}\right) = n.$$

For systems that are regular and of index at most 1, there exists a unique solution for all admissible controls with consistent initial conditions. Such systems separate into purely dynamical and purely algebraic parts, and in theory the algebraic part can be eliminated to give a reduced-order *standard* system. The reduction process, however, may not be numerically stable [15].

For higher-index systems, if the control is not sufficiently smooth, impulses can arise in the response of the system and the system can lose causality [23], [1]. It is desirable, therefore, to use a feedback control that ensures that the closed loop system is regular and of index less than or equal to 1, if possible. In the next sections we show that this can be achieved under certain "controllability" ("observability") conditions.

2.2. Controllability and observability of descriptor systems. The definitions of controllability and observability for standard control systems can be extended to descriptor systems. However, various types of controllability/observability can be identified [25]. Here we investigate the properties of the generalized state-space system (1), (2) and (3), (4) under the following conditions:

$$\begin{aligned} &\text{C0: } \operatorname{rank}([\alpha E - \beta A, B]) = n \quad \forall (\alpha, \beta) \in \mathbb{C}^2 \setminus \{(0,0)\}; \\ (12) \qquad &\text{C1: } \operatorname{rank}([\lambda E - A, B]) = n \quad \forall \lambda \in \mathbb{C}; \\ &\text{C2: } \operatorname{rank}([E, AS_\infty, B]) = n \quad \text{where the columns of } S_\infty \text{ span } \mathcal{N}(E). \end{aligned}$$

For systems that are *regular*, these conditions characterize the controllability of the system. We have the following definition.

DEFINITION 2. Let $\alpha E - \beta A$ be a regular pencil. Then the triple (E, A, B) and the corresponding descriptor system are said to be *completely controllable* (C-controllable) if and only if condition C0 holds.

We remark that a descriptor system satisfies C0, i.e., is completely controllable, only if

$$\operatorname{rank}([E, B]) = n. \tag{13}$$

Complete controllability ensures that for any given initial and final states $x_0, x_f \in \mathbb{R}^n$ of the system, there exists an admissible control that transfers the system from x_0 to x_f in finite time [25]. Hence, descriptor systems that are completely controllable can be expected to have properties similar to those of standard systems.

A weaker definition of controllability is given by the following.

DEFINITION 3. Let $\alpha E - \beta A$ be a regular pencil. Then the triple (E, A, B) and the corresponding descriptor system are said to be strongly controllable (S-controllable) if and only if C1 and C2 hold.

We remark that C-controllability implies S-controllability. Clearly C1 follows from C0 for $\beta \neq 0$ and $\lambda = \alpha/\beta$. Condition C2 follows from (13), but is weaker. In the literature, regular systems that satisfy C2 are often described as "controllable at infinity" or "impulse controllable" [5], [10], [23]. For these systems "impulsive modes" can be excluded. A descriptor system that has a regular pencil of index less than or equal to 1 is always controllable at infinity, since by Lemma 1 we have $\operatorname{rank}([E, AS_\infty]) = n$.

The controllability conditions are preserved under certain transformations of the system. Specifically, C0, C1, and C2, are all preserved under nonsingular "equivalence" transformations of the pencil and under proportional state feedback. With the exception of C2, these same conditions are also preserved under derivative state feedback. The following lemma summarizes these results.

LEMMA 4. *Let* (E, A, B) *satisfy* C0 *or* C1 *or* C2. *Then for any nonsingular* P *and* $Q \in \mathbb{R}^{n,n}$ *and for any* $F \in \mathbb{R}^{m,n}$, *the system* $(\tilde{E}, \tilde{A}, \tilde{B})$, *where*

$$\tilde{E} = PEQ, \quad \tilde{A} = PAQ, \quad \tilde{B} = PB \tag{14}$$

or

$$\tilde{E} = E, \quad \tilde{A} = A + BF, \quad \tilde{B} = B, \tag{15}$$

also satisfies these conditions.

Furthermore, for any matrix $G \in \mathbb{R}^{m,n}$, *the system* $(\tilde{E}, \tilde{A}, \tilde{B})$, *where*

$$\tilde{E} = E + BG, \quad \tilde{A} = A, \quad \tilde{B} = B, \tag{16}$$

also satisfies these conditions with the exception of C2.

Proof. In case (14), for all $(\alpha, \beta) \in \mathbb{C}^2 \setminus \{(0, 0)\}$ we have

$$\begin{aligned} \operatorname{rank}([\alpha E - \beta A, B]) &= \operatorname{rank}\left(P[\alpha E - \beta A, B] \begin{bmatrix} Q & 0 \\ 0 & I \end{bmatrix} \right) \\ &= \operatorname{rank}([\alpha \tilde{E} - \beta \tilde{A}, \tilde{B}]) \end{aligned} \tag{17}$$

and

$$\begin{aligned} \operatorname{rank}([E, AS_\infty, B]) &= \operatorname{rank}\left(P[E, AQQ^{-1}S_\infty, B] \begin{bmatrix} Q & 0 & 0 \\ 0 & I & 0 \\ 0 & 0 & I \end{bmatrix} \right) \\ &= \operatorname{rank}([\tilde{E}, \tilde{A}\tilde{S}_\infty, \tilde{B}]), \end{aligned} \tag{18}$$

where $\tilde{S}_\infty = Q^{-1}S_\infty$ spans $\mathcal{N}(\tilde{E})$. Therefore, C0, C1, and C2 are preserved under the transformation (14).

In case (15) we have

$$\begin{aligned} \operatorname{rank}([\alpha E - \beta A, B]) &= \operatorname{rank}\left([\alpha E - \beta A, B] \begin{bmatrix} I & 0 \\ -\beta F & I \end{bmatrix} \right) \\ &= \operatorname{rank}([\alpha \tilde{E} - \beta \tilde{A}, \tilde{B}]) \end{aligned} \tag{19}$$

and

$$\text{rank}([E, AS_\infty, B]) = \text{rank}\left([E, AS_\infty, B] \begin{bmatrix} I & 0 & 0 \\ 0 & I & 0 \\ 0 & FS_\infty & I \end{bmatrix} \right) \tag{20}$$

$$= \text{rank}([\tilde{E}, \tilde{A}\tilde{S}_\infty, \tilde{B}]),$$

where $\tilde{S}_\infty = S_\infty$, since $\tilde{E} = E$. Therefore, C0, C1, and C2 are all retained.

In case (16) the proof that C0 and C1 are preserved is shown analogously to case (15). Condition C2 is not necessarily preserved, however, since in the case (16), the nullspace S_∞ is altered by the feedback and $\tilde{S}_\infty \neq S_\infty$. □

An example is given in [1] demonstrating that C2 is not necessarily preserved under derivative feedback. If derivative feedback is used to change the system dynamics, it is therefore necessary to be careful not to lose controllability at infinity. In the next section we investigate the use of derivative feedback to make the system regular and of index at most 1. Thus, the resulting system is always controllable at infinity. Regularity of the original system is not needed to achieve this result.

Observability conditions for the time-invariant systems (1), (2) and (3), (4) can be defined as the dual of the controllability conditions. Specifically, a system represented by the triple (E, A, C) is said to satisfy conditions O0, O1, O2 if and only if the dual system, represented by the triple (E^T, A^T, C^T), satisfies C0, C1, C2, respectively. A regular system is defined to be *completely observable* (C-observable) if and only if O0 is satisfied and *strongly observable* (S-observable) if and only if O1 and O2 hold.

In the following sections we derive numerically stable techniques for constructing feedback controllers to achieve particular objectives. By duality these techniques can also be used in the construction of state estimators and observer-based controllers.

3. Derivative and proportional feedback for descriptor systems. In this section we discuss conditions under which we can alter the structure of the system pencil (5) by the use of derivative and/or proportional state feedback. We show that if the triple (E, A, B) satisfies C0, i.e., is C-controllable, then the system (1) or (3) can be transformed into a *completely controllable standard* system by derivative feedback [1]. We show also that if a system satisfies C1 and C2, then a closed loop system that is *strongly controllable, regular, and of index at most* 1 can be obtained by derivative or proportional feedback. With derivative feedback, however, the explicit degrees of freedom describing the reachable subspace of the system (corresponding to the number of finite poles of the closed loop system) can be increased to a maximum equal to $\text{rank}([E, B])$. Previously it has been shown that proportional state feedback can be used to obtain a regular closed loop system of index at most 1 and simultaneously to place $q = \text{rank}(E)$ poles [10]. Here we describe a simpler numerical procedure for constructing a regular closed loop system of index at most 1 by proportional state feedback. This procedure does not guarantee that the closed loop poles take specified values. In § 4 techniques for pole placement are discussed.

In the first part of this section we give basic theorems that form the core of the numerical construction techniques. Subsequently, the C-controllable and S-controllable cases are each examined, and finally, the combined use of both derivative and proportional feedback is discussed. Throughout the development we make extensive use of the singular value decomposition (SVD) of a matrix $M \in \mathbb{R}^{m,n}$, e.g., [9]. In the usual notation the SVD is given by

$$M = U \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} V^T, \tag{21}$$

where U and V are $m \times m$ and $n \times n$ orthogonal matrices, respectively, and Σ is a $\operatorname{rank}(M) \times \operatorname{rank}(M)$ diagonal matrix with positive diagonal entries. Here we also refer to the orthogonal reduction of M to diagonal form

$$U^T M V = \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \tag{22}$$

as an SVD of M, because we always need it in this form.

3.1. Preliminary theory. The first lemma serves as a basic tool and provides a "canonical" form for the system (1) or (3), which can be obtained in a numerically stable way.

LEMMA 5. *Let $E \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$, and* $\operatorname{rank}(B) = m \leqq n$. *There exist orthogonal matrices Q, U, and V such that*

$$QEU = \begin{bmatrix} \Sigma_1 & 0 & 0 \\ E_{21} & E_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix} \quad \text{and} \quad QBV = \begin{bmatrix} 0 \\ \Sigma_B \\ 0 \end{bmatrix}, \tag{23}$$

where Σ_1 and Σ_B are $l \times l$ and $m \times m$ diagonal matrices, respectively, with positive diagonal entries, and E_{22} is an $m \times s$ matrix with full column rank. The partitioning in QEU and QBV is conformable.

Proof. Let

$$\tilde{P} B V = \begin{bmatrix} \Sigma_B \\ 0 \end{bmatrix} \tag{24}$$

be an SVD of B. Let

$$P = \begin{bmatrix} 0 & I_{n-m} \\ I_m & 0 \end{bmatrix} \tilde{P}. \tag{25}$$

Then we obtain

$$PBV = \begin{bmatrix} 0 \\ \Sigma_B \end{bmatrix}, \qquad PE = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix}, \tag{26}$$

with a compatible partitioning. Let

$$W E_1 Z_1 = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix} \tag{27}$$

be an SVD of E_1, where Σ_1 is an $l \times l$ diagonal matrix with positive diagonal entries. Then

$$\begin{bmatrix} W & 0 \\ 0 & I_m \end{bmatrix} PEZ_1 = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \\ E_{21} & \tilde{E}_{22} \end{bmatrix}, \tag{28}$$

where $[E_{21}, \tilde{E}_{22}]$ is a compatible partitioning of $E_2 Z_1$. Let Z_2 be an orthogonal matrix that does a "column compression"

$$\tilde{E}_{22} Z_2 = [E_{22}, 0] \tag{29}$$

on $\tilde{E}_{22}$ such that E_{22} has full column rank. The matrix Z_2 could, for example, be derived from an RQ-decomposition of $\tilde{E}_{22}$ (e.g., [9]),

$$\tilde{E}_{22} = [R, 0] Z_2^T. \tag{30}$$

Then from (26), (28), and (29) we get the desired transformation as

$$\begin{bmatrix} I_l & 0 & 0 \\ 0 & 0 & I_m \\ 0 & I_{n-m-l} & 0 \end{bmatrix} \begin{bmatrix} W & 0 \\ 0 & I_m \end{bmatrix} PEZ_1 \begin{bmatrix} I_l & 0 \\ 0 & Z_2 \end{bmatrix} = \begin{bmatrix} \Sigma_1 & 0 & 0 \\ E_{21} & E_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{31}$$

and

$$\begin{bmatrix} I_l & 0 & 0 \\ 0 & 0 & I_m \\ 0 & I_{n-m-l} & 0 \end{bmatrix} \begin{bmatrix} W & 0 \\ 0 & I_m \end{bmatrix} PBV = \begin{bmatrix} 0 \\ \Sigma_B \\ 0 \end{bmatrix}. \tag{32}$$

□

In the next theorem we establish conditions that guarantee that there exist matrices F and G such that the matrix pencil $\alpha(E + BG) - \beta(A + BF)$ is regular and of index at most 1 and such that $\operatorname{rank}(E + BG) = r$, where r can be chosen to be any integer satisfying $q - m \leqq r \leqq q = \operatorname{rank}([E, B])$.

THEOREM 6. *Let $E, A \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$ with $\operatorname{rank}(B) = m \leqq n$, and let S_∞ be a full rank matrix whose columns span $\mathcal{N}(E)$. If $\operatorname{rank}([E, AS_\infty, B]) = n$ and $r \in \mathbb{N}$ such that $0 \leqq l = q - m \leqq r \leqq q = \operatorname{rank}([E, B])$, then there exist matrices $F, G \in \mathbb{R}^{m,n}$ such that the matrix pencil $\alpha(E + BG) - \beta(A + BF)$ is regular, $\operatorname{ind}_\infty(E + BG, A + BF) \leqq 1$, and $\operatorname{rank}(E + BG) = r$.*

Proof. By Lemma 5 there exist orthogonal matrices Q, U, and V such that (23) holds and we may choose

$$S_\infty = U \begin{bmatrix} 0 \\ 0 \\ I_{n-l-s} \end{bmatrix}. \tag{33}$$

Partitioning QAU compatibly with QEU we have

$$QAU = \begin{bmatrix} A_{11} & \tilde{A}_{12} & \tilde{A}_{13} \\ A_{21} & \tilde{A}_{22} & \tilde{A}_{23} \\ A_{31} & \tilde{A}_{32} & \tilde{A}_{33} \end{bmatrix} \tag{34}$$

and then

$$n = \operatorname{rank}([E, AS_\infty, B]) = \operatorname{rank}\left(\begin{bmatrix} QEU, \begin{bmatrix} \tilde{A}_{13} \\ \tilde{A}_{23} \\ \tilde{A}_{33} \end{bmatrix}, QBV \end{bmatrix} \right) \tag{35}$$

implies that $\tilde{A}_{33}$ must have full row rank, that is, $\operatorname{rank}(\tilde{A}_{33}) = n - l - m$.

Without loss of generality we may assume that the last $n - m - l$ columns of $\tilde{A}_{33}$ are linearly independent. If this is not the case, we can achieve this property by a "column compression" of $\tilde{A}_{33}$ to the right using an RQ-decomposition of $\tilde{A}_{33}$ or with an SVD.

From (23) we see that $\operatorname{rank}(E) = l + s$ and $q = \operatorname{rank}([E, B]) = l + m$. Let

$$\tilde{G} = [G_1, G_2, G_3] \tag{36}$$

and choose $G_1 = -\Sigma_B^{-1} E_{21}$ and G_2, G_3 such that

$$[E_{22} + \Sigma_B G_2, \Sigma_B G_3] = [\mathscr{E}_2, 0], \tag{37}$$

where $\mathscr{E}_2$ is an $m \times (r - l)$ matrix of full column rank. For instance, if $r > l + s$, we may select $G_2 = 0$ and

$$G_3 = \left[\Sigma_B^{-1} \hat{E}_{22} \begin{bmatrix} I_{r-l-s} \\ 0 \end{bmatrix}, 0 \right], \tag{38}$$

where $\hat{E}_{22}$ forms a basis for the orthogonal complement of E_{22}; if $r < l + s$, then we may choose $G_3 = 0$ and

$$G_2 = \left[0, -\Sigma_B^{-1} E_{22} \begin{bmatrix} 0 \\ I_{l+s-r} \end{bmatrix}\right]; \tag{39}$$

and if $r = l + s = \operatorname{rank}(E)$ then we may choose G_3, $G_2 = 0$. (The matrix $\hat{E}_{22}$ can be obtained in practice from the RQ-decomposition (30) used in the reduction of Lemma 5.) Then $QEU + QBV\tilde{G}$ has rank equal to r precisely and its nullspace is spanned by

$$\begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix}. \tag{40}$$

Now let $\tilde{Z}$ be an orthogonal matrix that gives a column compression of the last $n - r$ columns of $[\tilde{A}_{32}, \tilde{A}_{33}]$ to the right; that is, such that

$$[\tilde{A}_{32}, \tilde{A}_{33}] \begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix} \tilde{Z} = [0, A_{34}], \tag{41}$$

where A_{34} is a nonsingular $(n - l - m) \times (n - l - m)$ matrix. This is achievable by our assumption that the last $n - l - m$ columns of $\tilde{A}_{33}$ are linearly independent. Then with

$$Z = \begin{bmatrix} I_r & 0 \\ 0 & \tilde{Z} \end{bmatrix} \tag{42}$$

we obtain

$$(QEUZ + QBV\tilde{G}) = \begin{bmatrix} \Sigma_1 & 0 & 0 & 0 \\ 0 & \mathscr{E}_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \tag{43}$$

and

$$QAUZ = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & 0 & A_{34} \end{bmatrix}. \tag{44}$$

Now let

$$\tilde{F} = [F_1, F_2, F_3, F_4], \tag{45}$$

partitioned conformably with $QAUZ$, and choose F_3 such that the $m \times m$ matrix $[\mathscr{E}_2, A_{23} + \Sigma_B F_3]$ is of full rank. For instance, if $r < l + m$, we may select $F_3 = \Sigma_B^{-1}(\hat{\mathscr{E}}_2 - A_{23})$, where $\hat{\mathscr{E}}_2$ spans the orthogonal complement of $\mathscr{E}_2$. (If $\tilde{G}$ is as previously suggested, $\hat{\mathscr{E}}_2$ is easily constructed from E_{22} and $\hat{E}_{22}$.)

If $r = l + m = \operatorname{rank}([E, B])$, then we may select $\tilde{F} = 0$. Finally, with $G = V\tilde{G}Z^T U^T$, $F = V\tilde{F}Z^T U^T$, we find that the nullspace of $E + BG = E + BV\tilde{G}Z^T U^T$ is spanned by

$$\hat{S}_\infty = UZ \begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix} \tag{46}$$

and it follows that

$$\begin{aligned} &\operatorname{rank}([E + BG, (A + BF)\hat{S}_\infty]) \\ &\quad = \operatorname{rank}\left(\left[QEUZ + QBV\tilde{G}, (QAUZ + QBV\tilde{F}) \begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix}\right]\right) \\ &\quad = \operatorname{rank}\left(\begin{bmatrix} \Sigma_1 & 0 & A_{13} & A_{14} \\ 0 & \mathscr{E}_2 & A_{23} + \Sigma_B F_3 & A_{24} + \Sigma_B F_4 \\ 0 & 0 & 0 & A_{34} \end{bmatrix}\right) = n. \end{aligned} \tag{47}$$

By Lemma 1, the pencil $\alpha(E + BG) - \beta(A + BF)$ is therefore regular and has index less than or equal to 1. □

An immediate consequence of Theorem 6 is the following corollary.

COROLLARY 7. *Let* $E, A \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$ *with* $\operatorname{rank}(B) = m \leqq n$, *and let* S_∞ *be a full rank matrix whose columns span* $\mathcal{N}(E)$. *If* $\operatorname{rank}([E, AS_\infty, B]) = n$, *then the following hold*:

(i) *There exists a matrix* $G \in \mathbb{R}^{m,n}$ *such that the matrix pencil* $\alpha(E + BG) - \beta A$ *is regular, has index at most* 1, *and* $\operatorname{rank}(E + BG) = \operatorname{rank}([E, B])$;

(ii) *There exists a matrix* $F \in \mathbb{R}^{m,n}$ *such that the matrix pencil* $\alpha E - \beta(A + BF)$ *is regular and has index at most* 1.

Proof. The first result follows directly from the construction of F and G in Theorem 6 in the case where $r = \operatorname{rank}([E, B])$. The second result, where $r = \operatorname{rank}(E)$, also follows as in Theorem 6, with the exception that $\tilde{G} = 0$ is selected and F_3 is constructed such that the $r \times r$ matrix

$$\begin{bmatrix} \Sigma_1 & 0 & A_{13} \\ E_{21} & E_{22} & A_{23} + \Sigma_B F_3 \end{bmatrix} \tag{48}$$

is of full rank. The feedback F_3 could, for instance, be taken as

$$F_3 = \Sigma_B^{-1}(\hat{E}_{22} - A_{23} + E_{21}\Sigma_1^{-1}A_{13}), \tag{49}$$

where $\hat{E}_{22}$ gives a basis for the orthogonal complement of E_{22}. □

We remark that the decomposition (43) of Theorem 6 reveals the extent to which the structure of $E + BG$ can be controlled by a derivative feedback G. In a later section we discuss techniques for selecting G to give a "well-conditioned" regularization of the descriptor system. Lemma 5, Theorem 6, and Corollary 7 provide the key steps in the proofs of the following theorems. We note that these results can also be achieved using the generalized singular value decomposition (see [9]), but the full reduction to this decomposition is not needed here and it is preferable to use the decomposition (23), which requires only orthogonal transformations, for numerical stability.

3.2. C-controllable systems: Derivative feedback. We now show that if the triple (E, A, B) satisfies condition C0, then systems (1) and (3) can be transformed into completely controllable *standard* systems by derivative state feedback. These results have also been established in [16] and [6], but here numerically stable techniques for constructing the feedback are provided. Regularity of the system (E, A, B) is not required.

The main theorem is given as follows.

THEOREM 8. *There exists a real feedback control* $u = -G\dot{x} + v$ *or* $u_k = -Gx_{k+1} + v_k$ *such that the system defined by the triple* $(E + BG, A, B)$ *is* C*-controllable and the matrix* $E + BG$ *is nonsingular if and only if the triple* (E, A, B) *satisfies* C0, *that is*, $\operatorname{rank}([\alpha E - \beta A, B]) = n$ *for all* $(\alpha, \beta) \in \mathbb{C}^2 \setminus \{(0, 0)\}$.

Proof. Condition C0 implies that $\operatorname{rank}([E, B]) = n$, and hence $\operatorname{rank}([E, AS_\infty, B]) = n$. Therefore, by Corollary 7, there exists $G \in \mathbb{R}^{m,n}$ such that $\operatorname{rank}(E + BG) = \operatorname{rank}([E, B]) = n$. Now by Lemma 4, the condition C0 is preserved under derivative state feedback and the theorem follows immediately from the definition of C-controllability. □

We remark that the condition $\operatorname{rank}([E, B]) = n$ is both necessary and sufficient to find G such that $E + BG$ is nonsingular. Sufficiency follows from Corollary 7 and necessity from the observation that

$$E + BG = [E, B]\begin{bmatrix} I \\ G \end{bmatrix}. \tag{50}$$

From Theorem 8 we conclude that systems (1) or (3), which satisfy C0, can be transformed into standard systems by derivative feedback. If the system matrix $S = E + BG$ is nonsingular, then the corresponding closed loop system is equivalent to the standard system

$$\dot{x} = \tilde{A}x + \tilde{B}v \tag{51}$$

or

$$x_{k+1} = \tilde{A}x_k + \tilde{B}v_k, \tag{52}$$

where $\tilde{A} = S^{-1}A$, $\tilde{B} = S^{-1}B$. Transformation to this standard form may not be numerically reliable, however, if $S = E + BG$ is ill conditioned with respect to inversion. The decomposition (43) of Theorem 6 reveals the extent to which the conditioning of S can be controlled by an appropriate choice of G. In a later section of this paper we discuss techniques for selecting G to provide an optimally conditioned "regularization" of the system.

The following example illustrates the regularization of a very simple system (given in [23]).

Example 1. Let

$$E = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{53}$$

Then (1) gives the equation of a simple electrical circuit, where x_1 is the current and x_2 is the potential of the capacitor. The system is C-controllable. Let $G = [g_1, g_2]$ with $g_1 \neq 0$; then

$$E + BG = \begin{bmatrix} 0 & 1 \\ g_1 & g_2 \end{bmatrix} \tag{54}$$

is nonsingular, and choosing $u = -G\dot{x} + v$ transforms the system into the system

$$\begin{aligned} \dot{x}_2 &= x_1, \\ g_1\dot{x}_1 + g_2\dot{x}_2 &= x_2 + v, \end{aligned} \tag{55}$$

which is equivalent to the completely controllable standard system

$$\begin{aligned} \dot{x}_1 &= -\frac{g_2}{g_1}x_1 + \frac{1}{g_1}x_2 + \frac{1}{g_1}v, \\ \dot{x}_2 &= x_1. \end{aligned} \tag{56}$$

If g_1 is taken to be very small, the conditioning of $E + BG$ is very poor and the standard system (56) may be very sensitive to perturbations. Selecting $g_1 = 1$, $g_2 = 0$ optimizes the conditioning of $E + BG$ and ensures that the system (56) is robust.

We have established here that a C-controllable descriptor system can be transformed by derivative state feedback into a completely controllable standard system of full order. By duality, the analogous results hold for C-observable systems.

We remark that the transformation to standard form cannot be achieved with *proportional* state feedback alone. In the next sections we show that under the weaker S-controllability condition, a closed loop system that is regular and of index at most 1 can be achieved by either derivative or proportional state feedback. Such systems are equivalent to reduced-order standard systems and are completely controllable within a subspace of less than full dimension.

3.3. S-controllable systems: Derivative feedback. We now show that a system (E, A, B) that satisfies C1 and C2 can be transformed by derivative state feedback into a system $(E + BG, A, B)$, which is regular and has index at most 1, has system matrix $E + BG$ of maximal rank equal to rank $([E, B])$, and is S-controllable. The main theorem is given as follows.

THEOREM 9. *There exists a real feedback control* $u = -G\dot{x} + v$ *or* $u_k = -Gx_{k+1} + v_k$ *such that the continuous or discrete closed loop system defined by the triple* $(E + BG, A, B)$ *is S-controllable and the system pencil* $\alpha(E + BG) - \beta A$ *is regular,* $\operatorname{ind}_\infty(E + BG, A) \leqq 1$, *and* $\operatorname{rank}(E + BG) = \operatorname{rank}([E, B])$ *if the triple* (E, A, B) *satisfies* C1 *and* C2, *that is,* $\operatorname{rank}([\lambda E - A, B]) = n$ *for all* $\lambda \in \mathbb{C}$ *and* $\operatorname{rank}([E, AS_\infty, B]) = n$, *where* S_∞ *forms a basis for* $\mathcal{N}(E)$.

Proof. By Corollary 7, C2 ensures the existence of a matrix G such that $\alpha(E + BG) - \beta A$ is regular, $\operatorname{ind}_\infty(E + BG, A) \leqq 1$, and $\operatorname{rank}(E + BG) = \operatorname{rank}([E, B])$. The triple $(E + BG, A, B)$ therefore also satisfies C2 and by Lemma 4, C1 is preserved under derivative feedback. Thus, the closed loop system is S-controllable. □

We remark that the converse of Theorem 9 does not hold. The condition $\operatorname{rank}([E, AS_\infty, B]) = n$ is not necessarily preserved under derivative feedback, since S_∞ is altered. The condition $\operatorname{rank}([E, AS_\infty, B]) = n$ is therefore sufficient, but *not* necessary to obtain a regular pencil $\alpha(E + BG) - \beta A$ of index at most 1 with $\operatorname{rank}(E + BG) = \operatorname{rank}([E, B])$. An example is given in [1].

From Theorem 9 we conclude that systems that satisfy C1 and C2, or are S-controllable, can be transformed by derivative feedback into completely controllable, reduced-order, standard systems with maximal dimension equal to the dimension of the reachable subspace of the original system. By Lemma 5 and Theorem 6, C2 ensures that there exist orthogonal matrices Q, U, V, and Z such that (23), (43), and (44) hold, where $r = \operatorname{rank}([E, B]) = m + l$, A_{34} is nonsingular, and

$$E_R := \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \mathscr{E}_2 \end{bmatrix} \tag{57}$$

is also nonsingular. The last $(n - l - m)$-block of algebraic equations of the equivalent system $(Q(E + BG)UZ, QAUZ, QBV)$ can thus be solved and the corresponding variables can be eliminated from the first $(l + m)$-block of equations, leaving a purely dynamical descriptor system of the form (E_R, A_R, B_R), with E_R nonsingular. This reduced-order system has dimension $l + m = \operatorname{rank}([E, B])$ and is equivalent to the completely controllable, standard system

$$\dot{z} = E_R^{-1} A_R z + E_R^{-1} B_R v \tag{58}$$

or

$$z_{k+1} = E_R^{-1} A_R z_k + E_R^{-1} B_R v_k. \tag{59}$$

The reachable subspace of this system is thus of dimension $l + m$ and the degrees of freedom are all explicit in the initial conditions. To illustrate this result, consider the following example.

Example 2. Let

$$E = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}. \tag{60}$$

This system is S-controllable but not C-controllable, and is already in decomposed form (37) and (43). The solutions to this system are given by $x_1 = -u$, $x_2 = -\dot{u}$, and

$x_3 = 0$. For a specific choice of control, there are no degrees of freedom in the initial state of the system. The reachable subspace over all possible choices of the control has dimension 2 and is given by

$$\operatorname{span}\left(\begin{bmatrix} I_2 \\ 0 \end{bmatrix}\right).$$

If we now let $G = [g_1, g_2, 0]$ with $g_2 \neq 0$, then the feedback $u = -G\dot{x} + v$ transforms the system into

$$\begin{aligned} \dot{x}_1 &= x_2, \\ g_1\dot{x}_1 + g_2\dot{x}_2 &= x_1 + v, \\ 0 &= x_3. \end{aligned} \tag{61}$$

The last variable can be eliminated and the remaining dynamical system can be transformed into standard form by inverting the matrix

$$\begin{bmatrix} 1 & 0 \\ g_1 & g_2 \end{bmatrix} \tag{62}$$

to obtain the completely controllable system

$$\begin{aligned} \dot{x}_1 &= x_2, \\ \dot{x}_2 &= \frac{1}{g_2}x_1 - \frac{g_1}{g_2}x_2 + \frac{1}{g_2}v. \end{aligned} \tag{63}$$

These equations can be initiated from any state with any control and the dimension of the reachable subspace is precisely 2.

We remark that the reduction to standard form of systems that are regular and have index at most 1 may not be numerically reliable if the matrices A_{34} and E_R obtained from the decompositions (43) and (44) are not well conditioned for inversion. The conditioning of E_R is influenced by the selection of the derivative feedback G; the matrix A_{34} is not affected by the feedback G.

We have now established that an S-controllable descriptor system can be transformed by derivative state feedback into a reduced-order, completely controllable standard system with explicit degrees of freedom in the initial conditions equal to the dimension of the reachable subspace. By duality, the analogous results hold for S-observable systems. In the next sections we examine what can be achieved with proportional feedback alone and in combination with derivative state feedback.

3.4. S-controllable systems: Proportional feedback and other results. We next show that a system (E, A, B) that satisfies C1 and C2 can be transformed by proportional state feedback into a system $(E, A + BF, B)$ that is regular, has index at most 1, and is S-controllable. This result has been established (implicitly) in [5], [6], [7], and [10] using various approaches. Here we give another proof, based on the decomposition of Lemma 5, which allows for the construction of the required feedback in a numerically stable manner.

The main theorem is given as follows.

THEOREM 10. *There exists a real feedback control $u = Fx + v$ or $u_k = Fx_k + v_k$ such that the continuous or discrete system defined by the triple $(E, A + BF, B)$ is* S*-controllable and the system pencil $\alpha E - \beta(A + BF)$ is regular and* $\operatorname{ind}_\infty(E, A + BF) \leqq 1$ *if and only if the triple (E, A, B) satisfies* C1 *and* C2, *that is,* $\operatorname{rank}([\lambda E - A, B]) = n$ *for all $\lambda \in \mathbb{C}$ and* $\operatorname{rank}([E, AS_\infty, B]) = n$, *where S_∞ forms a basis for $\mathcal{N}(E)$.*

Proof. By Corollary 7, C2 ensures the existence of a matrix F such that $\alpha E - \beta(A + BF)$ is regular and $\operatorname{ind}_\infty(E, A + BF) \leqq 1$. The remainder of the theorem is established by applying Lemma 4, which ensures that C1 and C2 are both preserved under proportional state feedback. □

We remark that the condition $\operatorname{rank}([E, AS_\infty, B]) = n$ is both necessary and sufficient to find F such that $\alpha E - \beta(A + BF)$ is regular and of index at most 1. This follows because E and therefore S_∞ are not changed by proportional state feedback.

From Theorem 10 we see that systems that satisfy C1 and C2 can also be transformed by proportional state feedback into regular systems of index at most 1 and hence into completely controllable reduced-order *standard* systems. The order of the standard system is minimal, however, being equal to $\operatorname{rank}(E)$. The degrees of freedom in the reduced-order dynamical system thus do not reflect the dimension of the solution space (i.e., reachable subspace) of the original descriptor system.

As an illustration consider again Example 2.

Example 3. Let (E, A, B) be given as in Example 2 by (60). Let $F = [f_1, f_2, 0]$. The feedback $u = Fx + v$ transforms the system into

$$\begin{aligned} \dot{x}_1 &= x_2, \\ 0 &= (1 + f_1)x_1 + f_2 x_2 + v, \\ 0 &= x_3, \end{aligned} \tag{64}$$

which is regular and of index 1, provided $f_2 \neq 0$. The last two equations of (64) can then be solved explicitly and eliminated from the first to give

$$\dot{x}_1 = \frac{1 + f_1}{f_2} x_1 - \frac{1}{f_2} v, \tag{65}$$

a standard system of order 1. The solution space of the system (64) is in fact of dimension 2, and hence the degrees of freedom in the reduced-order system (65) do not explicitly describe the solution space of the original system.

We conclude that although proportional state feedback can be used to eliminate impulses by controlling poles at infinity, it cannot be used to regularize a descriptor system completely. Using proportional feedback in combination with derivative feedback, on the other hand, can provide good design techniques that are computationally reliable. A suitable strategy is to use derivative feedback to obtain a well-conditioned regularization of the dynamic-algebraic equations and then to apply proportional feedback to achieve further objectives, such as pole assignment or stable reduction to a reduced-order system. This approach is particularly attractive, since proportional feedback cannot make the system lose regularity, once the rank of E has been maximized so that $\operatorname{rank}(E) = \operatorname{rank}([E, B])$. We have the following theorem.

THEOREM 11. *If* $\operatorname{rank}(E) = \operatorname{rank}([E, B])$ *and* $\operatorname{rank}([E, AS_\infty]) = n$, *then for any* $F \in \mathbb{R}^{m,n}$, $\operatorname{rank}([E, (A + BF)S_\infty]) = n$. *Here* S_∞ *defines a basis for* $\mathcal{N}(E)$.

Proof. Suppose there exists $z \neq 0$ such that $z^T[E, (A + BF)S_\infty] = 0$. Then $z^T E = 0$ and $z^T AS_\infty = -z^T BFS_\infty$. But since $\operatorname{rank}(E) = \operatorname{rank}([E, B])$, it follows that $z^T B = 0$, and thus $z^T AS_\infty = 0$. But then $z^T[E, AS_\infty] = 0$, which contradicts the assumption that $\operatorname{rank}([E, AS_\infty]) = n$. □

If a system (E, A, B) satisfies C1 and C2, it follows that there exists a derivative feedback G such that the system $(E + BG, A, B)$ satisfies $\operatorname{rank}([E + BG, B]) = \operatorname{rank}([E, B]) = \operatorname{rank}(E + BG)$, is regular and of index at most 1, and is S-controllable. By Lemma 1, then, $\operatorname{rank}([E + BG, A\hat{S}_\infty]) = n$, where $\hat{S}_\infty$ gives a basis for $\mathcal{N}(E + BG)$. Theorem 11 then guarantees that for any choice of F the system triple

$(E + BG, A + BF, B)$ satisfies $\operatorname{rank}([E + BG, (A + BF)\hat{S}_\infty]) = n$, and hence the system remains regular with index at most 1.

We have shown here that an S-controllable system can be transformed by proportional state feedback into a regular system of index at most 1, and hence into a reduced-order, controllable standard system. By duality, analogous results hold for S-observable systems. Of more practical significance, however, we have established that if a system has already been transformed into a regular system of index at most 1 by a derivative feedback that maximizes the dimension of the dynamic part of the system, that is, the system has been fully "regularized" by derivative feedback, then no proportional feedback can cause the system to lose regularity.

We complete this part of the paper by examining the results that can be obtained in general with a combination of derivative and proportional feedback.

3.5. Combined derivative and proportional feedback. We now summarize the results that can be achieved by using both derivative and proportional state feedback together. We show that for a system (E, A, B) that satisfies C1 and C2, a closed loop system can be obtained such that the system pencil $\alpha(E + BG) - \beta(A + BF)$ is regular and of index at most 1, and such that $\operatorname{rank}(E + BG) = r$, where r is any integer between $l = q - m$ and $q = \operatorname{rank}([E, B])$. (Here $m = \operatorname{rank}(B)$.) We have the following theorem, which follows directly from Theorem 6.

THEOREM 12. *There exists a real feedback control* $u = Fx - G\dot{x} + v$ *or* $u_k = Fx_k - Gx_{k+1} + v_k$ *such that the continuous or discrete time system defined by the triple* $(E + BG, A + BF, B)$ *is* S*-controllable and the system pencil* $\alpha(E + BG) - \beta(A + BF)$ *is regular,* $ind_\infty(E + BG, A + BF) \leqq 1$, *and* $\operatorname{rank}(E + BG) = r$ *with* $l \leqq r \leqq q$, *where* $q = \operatorname{rank}([E, B])$, $m = \operatorname{rank}(B)$, *and* $l = q - m$, *if the triple* (E, A, B) *satisfies* C1 *and* C2, *that is,* $\operatorname{rank}([\lambda E - A, B]) = n$ *for all* $\lambda \in \mathbb{C}$ *and* $\operatorname{rank}([E, AS_\infty, B]) = n$, *where* S_∞ *forms a basis for* $\mathcal{N}(E)$.

Proof. The existence of F and G such that $\alpha(E + BG) - \beta(A + BF)$ is regular and of index at most 1, and $\operatorname{rank}(E + BG) = r$ follows from C2 and Theorem 6. Then the transformed system given by $(E + BG, A + BF, B)$ must also satisfy C2, and by Lemma 4 C1 is preserved under both derivative and proportional state feedback, which establishes the theorem. □

We include Theorem 12 here primarily for completeness. It essentially shows that if C1 and C2 hold, then we can transform the system (1) or (3) by derivative and proportional state feedback into a regular system of index at most 1 with precisely r *finite* poles, where r is between $\operatorname{rank}([E, B])$ and $\operatorname{rank}([E, B]) - \operatorname{rank}(B)$. We emphasize that regularity of the original system is *not* required. Moreover, the feedback matrices F and G that achieve the result can be constructed in a *numerically stable* manner, using only orthogonal transformations.

Since the transformed system is regular and of index at most 1, it can be further transformed into a completely controllable, reduced-order, standard system of precise order r. For this reduction, however, the feedback matrices F and G must be selected with care.

In the next section we examine how derivative and proportional state feedback can be used to place the poles of the system in prescribed locations. In the final section we derive a computational algorithm for optimizing the conditioning of regularized dynamical systems obtained by derivative and proportional state feedback.

4. Eigenvalue assignment in descriptor systems. We now examine the consequences of the theory of § 3 for the problem of eigenvalue assignment. The conclusions follow directly from the "regularizability" results of Theorems 8 and 9. We begin by stating the pole assignment problem.

PROBLEM 1. *Given a triple of real matrices* (E, A, B) *and a set* $\mathcal{L} = \{(\alpha_1, \beta_1), (\alpha_2, \beta_2), \cdots, (\alpha_n, \beta_n)\}$, *where* $(\alpha_j, \beta_j) \in \mathbb{C}^2$ *and* $(\alpha_j, \beta_j) \in \mathcal{L}$ *implies* $(\bar{\alpha}_j, \bar{\beta}_j) \in \mathcal{L}$ *for* $j = 1, \cdots, n$, *find* $F, G \in \mathbb{R}^{m,n}$ *such that all pairs in* $\mathcal{L}$ *are generalized eigenvalues of the matrix pencil* $\alpha(E + BG) - \beta(A + BF)$ *and such that*

$$\det(\alpha(E + BG) - \beta(A + BF)) \neq 0 \quad \text{for some } (\alpha, \beta) \notin \mathcal{L} \cup \{(0, 0)\}. \tag{66}$$

The condition (66) ensures that the closed loop system obtained by the feedback $u = Fx - G\dot{x}$ or $u_k = Fx_k - Gx_{k+1}$ in system (1) or (3), respectively, is *regular*. In assigning a set of eigenpairs by feedback, it is always possible for the closed loop system to lose regularity, even if the original system is regular. It is important, therefore, in assigning eigenpairs, to ensure that (66) holds.

The problem of pole assignment by proportional feedback alone has been treated in [7] and [10]. In this case for systems that satisfy C1 and C2, at most $r = \operatorname{rank}(E)$ finite generalized eigenvalues (α_j, β_j), $\beta_j \neq 0, j = 1, 2, \cdots, r$, can be assigned such that the closed loop pencil is regular. The remaining $n - r$ infinite eigenvalues $(\alpha_j, 0), j = n - r + 1, \cdots, n$ cannot be reassigned. By exchanging the role of E and A in the system pencil, it can be seen that under analogous conditions, at most $s = \operatorname{rank}(A)$ *nonzero* eigenvalues (α_j, β_j), $\alpha_j \neq 0$, $j = 1, 2, \cdots, s$ (including infinite eigenvalues) can be assigned with derivative feedback alone. It might, therefore, be expected that with both derivative and proportional feedback, a full set of n eigenpairs could be assigned. This is, in fact, the case if and only if the system satisfies C0. We note that no assumptions are needed about the regularity of the system. We have the following theorem.

THEOREM 13. *For any arbitrary set* $\mathcal{L}$ *of* n *self-conjugate poles there exists a pair of real matrices* F *and* G *solving the pole placement problem, Problem* 1, *if and only if the triple of real matrices* (E, A, B) *satisfies* C0, *that is*,

$$\operatorname{rank}([\alpha E - \beta A, B]) = n \quad \forall (\alpha, \beta) \in \mathbb{C}^2 \setminus \{(0,0)\}. \tag{67}$$

Proof. Since the triple (E, A, B) satisfies C0, the triple (A, E, B) also satisfies this condition. Therefore, by Theorem 8 there exists a feedback matrix $F_1 \in \mathbb{R}^{m,n}$ such that $A + BF_1$ is nonsingular and the standard system $(I, (A + BF_1)^{-1}E, (A + BF_1)^{-1}B)$ is completely controllable. It follows that there exists $G \in \mathbb{R}^{m,n}$ such that G assigns k, $1 \leqq k \leqq n$, zero poles to this standard system, and, therefore, such that the pencil $\alpha(E + BG) - \beta(A + BF_1)$ has k infinite eigenvalues $(\alpha_j, 0)$, $j = 1, 2, \cdots, k$. Let $P, Q \in \mathbb{C}^{n,n}$ be nonsingular matrices that transform this pencil into Kronecker canonical form:

$$P(\alpha(E + BG) - \beta(A + BF_1))Q = \alpha \begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix} - \beta \begin{bmatrix} J & 0 \\ 0 & I \end{bmatrix}. \tag{68}$$

Partition $PB = \begin{bmatrix} B_1 \\ B_2 \end{bmatrix}$ analogously. The new triple

$$\left(\begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix}, \begin{bmatrix} J & 0 \\ 0 & I \end{bmatrix}, \begin{bmatrix} B_1 \\ B_2 \end{bmatrix} \right) \tag{69}$$

still satisfies C0, by Lemma 4, and hence the triple (I, J, B_1) is completely controllable. Thus, there exists $F_2 \in \mathbb{R}^{m,(n-k)}$ such that the eigenvalues of $J + B_1F_2$ are the finite eigenvalues (α_j, β_j), $\beta_j \neq 0, j = n - k + 1, \cdots, n$, belonging to $\mathcal{L}$. Let $F = [F_2, 0]Q^{-1} + F_1$. Then the pencil

$$\alpha(E + BG) - \beta(A + BF) = P^{-1} \left(\alpha \begin{bmatrix} I & 0 \\ 0 & N \end{bmatrix} - \beta \begin{bmatrix} J + B_1F_2 & 0 \\ B_2F_2 & I \end{bmatrix} \right) Q^{-1} \tag{70}$$

has the required eigenvalues.

Conversely, if there exist $F, G \in \mathbb{R}^{m,n}$ such that the pencil $\alpha(E + BG) - \beta(A + BF)$ has arbitrary generalized eigenvalues, then there exist F and G such that the pencil has arbitrary *finite* eigenvalues, that is, such that $E + BG$ is nonsingular and the eigenvalues of $(E + BG)^{-1}(A + BF)$ are arbitrary. The standard system $(I, (E + BG)^{-1}(A + BF), (E + BG)^{-1}B)$ must therefore be controllable. The triple $(E + BG, A + BF, B)$ must then satisfy C0, and by Lemma 4 the triple (E, A, B) also satisfies this condition. □

The construction of feedback matrices F, G in the proof of Theorem 13 requires a reduction to Kronecker canonical form, which in general is not a numerically reliable technique. Furthermore, the poles of the closed loop pencil obtained by this construction are not in general robust with respect to perturbations in the system matrices. In order to assign an arbitrary number of infinite poles to the closed loop system, the pencil must be allowed to have index greater than 1. Such systems are necessarily less robust than systems of index less than or equal to 1. Moreover, due to the Jordan form of the nilpotent part of the system, ill-conditioned transformations cannot be avoided.

In practice, it is not generally desirable to assign finite poles to infinite positions. If the number of infinite poles to be prescribed is limited, then the feedback matrices F and G can be constructed such that the closed loop pencil is not only regular and has the required finite poles, but also has index at most 1. Up to n finite eigenvalues can be assigned if and only if the triple (E, A, B) satisfies C0. Under the weaker assumptions C1 and C2, up to $q = \operatorname{rank}([E, B])$, finite poles can be prescribed. These results follow directly from Theorem 12. We have the following general result.

THEOREM 14. *For any arbitrary set $\mathcal{L}$ of r self-conjugate finite poles (α_j, β_j), $\beta_j \neq 0$, $j = 1, \cdots, r$, and $n - r$ infinite poles $(\alpha_j, 0)$, $j = r + 1, \cdots, n$, where $q = \operatorname{rank}([E, B]) \geqq r \geqq q - \operatorname{rank}(B)$, there exists a pair of real matrices F and G solving the pole placement problem, Problem 1, such that the pencil $\alpha(E + BG) - \beta(A + BF)$ is regular and $\operatorname{ind}_\infty(E + BG, A + BF) \leqq 1$ if the triple of real matrices (E, A, B) satisfies* C1 *and* C2, *that is,* $\operatorname{rank}([\lambda E - A, B]) = n$ *for all* $\lambda \in \mathbb{C}$ *and* $\operatorname{rank}([E, AS_\infty, B]) = n$, *where S_∞ forms a basis for $\mathcal{N}(E)$.*

Proof. By Theorem 12, there exist matrices G and F_1 such that the pencil $\alpha(E + BG) - \beta(A + BF_1)$ is regular and of index at most 1 and $\operatorname{rank}(E + BG) = r$, where $l = q - m \leqq r \leqq q = \operatorname{rank}([E, B])$, $m = \operatorname{rank}(B)$. The system $(E + BG, A + BF_1, B)$ is, moreover, S-controllable. It follows that there exists F_2 which assigns to this system up to $r = \operatorname{rank}(E + BG)$ *finite* poles and such that the closed loop system $(E + BG, A + BF, B)$, with $F = F_1 + F_2$, is regular and of index at most 1. (See [5], [7], [10].) By definition, this system has precisely $n - r$ infinite poles, which establishes the theorem. □

Conditions C1 and C2 are sufficient but not necessary for the results of Theorem 14 to hold. If it is required to assign precisely n *finite* poles, then C0 is both necessary and sufficient. Sufficiency follows directly from Theorem 14, since C0 implies C1 and C2 and $\operatorname{rank}([E, B]) = n$. Necessity follows from Theorem 13.

In order to assign precisely n *finite* eigenvalues (assuming C0 holds) we may select G such that $E + BG$ is nonsingular, by Theorem 8, and then select F to assign the prescribed poles to the equivalent standard system $(I, (E + BG)^{-1}A, (E + BG)^{-1}B)$. For this strategy to be computationally reliable, it is important to ensure that $E + BG$ is well conditioned for inversion. In the next section we describe a technique for selecting G to optimize the conditioning of $E + BG$. (In practice, it may not be possible to ensure that $E + BG$ is nicely conditioned; in this case the techniques of [10] can be applied to the generalized state-space system $(E + BG, A, B)$ to assign the n prescribed finite poles as robustly as possible.)

If the weaker conditions C1 and C2 hold, but C0 does not, then it is possible to assign a maximum of precisely $q = \operatorname{rank}([E, B]) < n$ *finite* poles. In this case, by

Theorem 9 we may select G such that $\operatorname{rank}(E + BG) = \operatorname{rank}([E, B]) = q$ and the pencil is regular and of index 1. As demonstrated in § 3.3, the corresponding closed loop system can then be transformed into a reduced-order, completely controllable system (E_R, A_R, B_R) of dimension q, where E_R is nonsingular. It is then possible to choose F_R to assign the required finite poles to the standard system $(I, E_R^{-1}A_R, E_R^{-1}B_R)$, and hence to construct F such that the pencil $\alpha(E + BG) - \beta(A + BF)$ has the required finite eigenvalues. By Theorem 11, this pencil is regular and of index 1. For this strategy to be numerically stable, it is necessary for E_R to be well conditioned, and also for A_{34} (defined in § 3.1) to be well conditioned in order for the reduction to the lower-order system to be computationally reliable.

A similar approach can be used for constructing the solution to the general problem of assigning *r finite* poles, where $q - m \leqq r \leqq q$, first applying Theorem 12 to obtain a regular S-controllable system $(E + BG, A + BF, B)$, where $\operatorname{rank}(E + BG) = r$, and then using a reduction to a lower-order standard form. In practice, however, this "reduced-order" approach may not be as efficient or as reliable as applying a direct procedure such as that of [10] to the "regularized" descriptor system in order to assign the poles.

In the next section, we develop techniques for "regularizing" the descriptor system so as to ensure that the dynamic part of the closed loop system is as well conditioned as possible.

5. Algorithm for regularizing a descriptor system. In previous sections we have examined conditions under which the descriptor systems (1) and (3) can be "regularized" by derivative and proportional state feedback, that is, conditions that ensure that a closed loop system can be constructed which is regular and of index at most 1, and is S-controllable. Regularity of the original system is not required, and the construction procedures are based on numerically stable techniques.

It has been shown in general that it is desirable in constructing a closed loop system of the form $(E + BG, A + BF, B)$ to ensure that $E + BG$ is "well conditioned" in some sense. In this final section of the paper we present a computational technique for generating a feedback G in such a way as to control the conditioning of the system matrix $E + BG$. In addition it is desirable to ensure that $A + BF$ is chosen such that the transformed descriptor system can be reduced to a standard system in a numerically stable way. A technique is also described for achieving this result. It is assumed that the system (E, A, B) satisfies C1 and C2.

In order for the matrix $E + BG$ to be well conditioned (with respect to inversion of the nonsingular part), it is necessary for the ratio $\sigma_{\max}/\sigma_{\min}$ of the largest singular value $\sigma_{\max}$, to the smallest *nonzero* singular value $\sigma_{\min}$ of $E + BG$, to be minimal. Now by Theorem 6, there exist orthogonal transformations Q, U, V, and Z and a feedback G such that $Q(E + BG)UZ$ is of form (43); moreover, G can be chosen such that $\mathscr{E}_2$, defined in (37), is of the form

$$\mathscr{E}_2 = \begin{bmatrix} \Sigma_2 \\ 0 \end{bmatrix}, \tag{71}$$

where Σ_2 is an $r \times r$ diagonal matrix with positive diagonal components and $q - \operatorname{rank}(B) \leqq r \leqq q = \operatorname{rank}([E, B])$. It follows that the singular values of $E + BG$ are given by the diagonal components of Σ_1 and Σ_2. Since Σ_1 arises from the decomposition (23) of E and cannot be altered by feedback, we find that the minimal possible condition number is $\sigma_{\max}/\sigma_{\min} = \|\Sigma_1\|_2 \|\Sigma_1^{-1}\|_2$. This value is attained provided the diagonal components of Σ_2 are selected to lie between the smallest and largest diagonal components of Σ_1.

In the case $r = q = \operatorname{rank}([E, B])$, the system generated by this procedure is regular and of index at most 1. In the case $r < q$, in order to obtain a system that is guaranteed to have these properties, it is necessary to use both derivative and proportional feedback. The proportional feedback matrix F must be selected, by Theorem 6, such that (47) holds. It is desirable also to select F such that the last $(n-r)\times(n-r)$ principal submatrix of $Q(A+BF)UZ$ is well conditioned with respect to inversion. As indicated in previous sections, the reduction of the descriptor system $(E+BG, A+BF, B)$ to a lower-order *standard* system is then expected to be computationally reliable.

From Theorem 6 it can be seen that if $\mathscr{E}_2$ is of the form (71), then (47) holds if we select

$$F_3 = \Sigma_B^{-1}\left(\begin{bmatrix} 0 \\ \Sigma_3 \end{bmatrix} - A_{23}\right), \qquad F_4 = -\Sigma_B^{-1}A_{24}, \tag{72}$$

where Σ_3 is an $(m+l-r)\times(m+l-r)$ diagonal matrix with positive diagonal elements. Then

$$Q(A+BF)UZ\begin{bmatrix} 0 & 0 \\ 0 & I_{n-r} \end{bmatrix} = \begin{bmatrix} \Sigma_3 & 0 \\ 0 & A_{34} \end{bmatrix} \tag{73}$$

has singular values given by the singular values of A_{34} and the diagonal components of Σ_3. To optimize the conditioning of (73), we must therefore select the components of Σ_3 to lie between $\|A_{34}^{-1}\|_2^{-1}$ and $\|A_{34}\|_2$.

If we let

$$W_2 A_{34} Z_3 = \Sigma_4 \tag{74}$$

be an SVD of A_{34} and define

$$\tilde{Q} = \begin{bmatrix} I_{l+m} & 0 \\ 0 & W_2 \end{bmatrix} Q, \qquad \tilde{U} = UZ\begin{bmatrix} I_{l+m} & 0 \\ 0 & Z_3 \end{bmatrix}, \tag{75}$$

then the pencil $\alpha(E+BG)-\beta(A+BF)$ constructed in this way is orthogonally equivalent to the pencil

$$\tilde{Q}[\alpha(E+BG)-\beta(A+BF)]\tilde{U} =: \alpha\begin{bmatrix} \Sigma_R & 0 \\ 0 & 0 \end{bmatrix} - \beta\begin{bmatrix} A_1 & A_2 \\ A_3 & \Sigma_A \end{bmatrix}, \tag{76}$$

where

$$\Sigma_R = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{bmatrix}, \qquad \Sigma_A = \begin{bmatrix} \Sigma_3 & 0 \\ 0 & \Sigma_4 \end{bmatrix}, \tag{77}$$

and Σ_R, Σ_A are as well conditioned as possible. The transformed descriptor system given by the triple $(E+BG, A+BF, B)$ can therefore be reduced to the standard system

$$\dot{z} = \tilde{A}z + \tilde{B}v \tag{78}$$

or

$$z_{k+1} = \tilde{A}z_k + \tilde{B}v_k, \tag{79}$$

where the system matrix $\tilde{A}$ is given by

$$\tilde{A} = \Sigma_R^{-1}(A_1 - A_2\Sigma_A^{-1}A_3). \tag{80}$$

The sensitivity of this computation to round-off errors then depends on the conditioning of Σ_1 and Σ_4, which are determined by E, A, and B.

We have established here a stable numerical technique for constructing a "regularized" descriptor system $(E + BG, A + BF, B)$ that is as well conditioned as possible. It is assumed that (E, A, B) satisfies conditions that correspond to S-controllability, but regularity of the original system pencil $\alpha E - \beta A$ is *not* needed. The computational algorithm for determining the required derivative and proportional state feedback matrices G and F is summarized in full in the Appendix. This procedure can also be extended to the problem of regularizing the systems (1), (2) and (3), (4) by *output* feedback. This topic is currently under investigation. Preliminary results are given in [1].

6. Conclusions. We investigate here the use of derivative and proportional feedback in descriptor, or generalized state-space, systems. We define various conditions for controllability (observability) and demonstrate to what extent the system can be altered by derivative and/or proportional state feedback under these conditions.

It is established that systems that satisfy conditions ensuring *complete* controllability can be transformed into *standard* systems (of full dimension) by a combination of derivative and proportional state feedback. It is shown, furthermore, that in this case, with state feedback, all of the poles of the system can be assigned to prescribed positions.

It is also established that systems that satisfy conditions ensuring *strong* controllability can be transformed by derivative and proportional state feedback into systems that are regular and of index at most 1 and have precisely r *finite* poles, where r lies between $q = \operatorname{rank}([E, B])$ and $q - \operatorname{rank}(B)$. Moreover, it is shown that these r poles can be assigned to arbitrary (finite) locations. Such systems are "impulse controllable" and can be transformed into reduced-order *standard* systems of precise dimension r.

The proofs of these results do not require regularity of the original system. Furthermore, the procedure for constructing the feedback matrices which regularize the closed loop system are based on orthogonal matrix decompositions and are numerically stable. In practice it is desirable not only that the closed loop descriptor system is regular, but also "well conditioned," in the sense that the reduction to standard form is computationally reliable. We show here that the feedback matrices that regularize the system can also be chosen to optimize the "conditioning" of the closed loop system, and a computational algorithm for achieving this result is presented.

7. Appendix: Algorithm for regularizing a descriptor system.

Step 1. Find orthogonal matrices $\tilde{P}$, V such that

$$\tilde{P}BV = \begin{bmatrix} \Sigma_B \\ 0 \end{bmatrix},$$

using the singular value decomposition of B.

Step 2. Let

$$P = \begin{bmatrix} 0 & I_{n-m} \\ I_m & 0 \end{bmatrix}$$

and partition

$$P\tilde{P}E = \begin{bmatrix} E_1 \\ E_2 \end{bmatrix}$$

compatibly with

$$P\tilde{P}BV = \begin{bmatrix} 0 \\ \Sigma_B \end{bmatrix}.$$

Step 3. Find orthogonal matrices W, Z_1 such that

$$WE_1Z_1 = \begin{bmatrix} \Sigma_1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad \Sigma_1 = \operatorname{diag}(\sigma_1, \cdots, \sigma_l), \tag{81}$$

by the singular value decomposition of E_1.

Step 4. Partition $E_2Z_1 = [E_{21}, \tilde{E}_{22}]$ compatibly with $W_1E_1Z_1$ and find an orthogonal matrix Z_2 such that $\tilde{E}_{22}Z_2 = [E_{22}, 0]$, where E_{22} is of full column rank. This can, for example, be achieved by an RQ-decomposition of E_{22}.

Step 5. Let

$$Q = \begin{bmatrix} I_l & 0 & 0 \\ 0 & 0 & I_m \\ 0 & I_{n-l-m} & 0 \end{bmatrix} \begin{bmatrix} W & 0 \\ 0 & I_m \end{bmatrix} P\tilde{P}, \qquad U = Z_1 \begin{bmatrix} I_l & 0 \\ 0 & Z_2 \end{bmatrix}. \tag{82}$$

Step 6. Select r such that $q = \operatorname{rank}([E, B]) \geqq r \geqq q - \operatorname{rank}(B)$. Find orthogonal matrices $\tilde{W}$, $\tilde{Z}$ such that

$$\tilde{W}[0, I_{n-l-m}]QAU\hat{U}\begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix}\tilde{Z} = [0, \Sigma_4], \qquad \Sigma_4 = \operatorname{diag}(\sigma_{l+m+1}, \cdots, \sigma_n) \tag{83}$$

by the singular value decomposition of $[0, I_{n-l-m}]QAU\hat{U}[\begin{smallmatrix} 0 \\ I_{n-r} \end{smallmatrix}]$, where $\hat{U}$ is chosen such that the lower right $(n-l-m) \times (n-l-m)$ block of $QAU\hat{U}$ is nonsingular. The matrix $\hat{U}$ can, for example, be found by an RQ-decomposition of the lower right $(n-l-s) \times (n-l-m)$ block of QAU, which is of full rank.

Step 7. Let

$$\tilde{Q} = \begin{bmatrix} I_{l+m} & 0 \\ 0 & \tilde{W} \end{bmatrix} Q, \qquad \tilde{U} = U\hat{U}\begin{bmatrix} I_r & 0 \\ 0 & \tilde{Z} \end{bmatrix}. \tag{84}$$

Step 8. Select

$$\Sigma_2 = \operatorname{diag}(\sigma_{l+1}, \cdots, \sigma_r), \qquad \Sigma_3 = \operatorname{diag}(\sigma_{r+1}, \cdots, \sigma_{l+m}), \tag{85}$$

where

$$\begin{aligned} \|\Sigma_1^{-1}\|_2^{-1} \leqq \sigma_j \leqq \|\Sigma_1\|_2, \qquad & j = l+1, \cdots, r, \\ \|\Sigma_4^{-1}\|_2^{-1} \leqq \sigma_j \leqq \|\Sigma_4\|_2, \qquad & j = r+1, \cdots, l+m. \end{aligned} \tag{86}$$

Step 9. Select

$$G = V[G_1, G_2, G_3, 0]\tilde{U}^T, \qquad F = V[0, 0, F_3, F_4]\tilde{U}^T, \tag{87}$$

where

$$\begin{aligned} G_1 &= -\Sigma_B^{-1}E_{21}, \qquad [G_2, G_3] = \Sigma_B^{-1}\left(\begin{bmatrix} \Sigma_2 & 0 \\ 0 & 0 \end{bmatrix} - [E_{22}, 0]\right), \\ F_3 &= \Sigma_B^{-1}\left(\begin{bmatrix} 0 \\ \Sigma_3 \end{bmatrix} - A_{23}\right), \qquad F_4 = -\Sigma_B^{-1}A_{24}, \end{aligned} \tag{88}$$

with

$$[A_{23}, A_{24}] = [0, I_m, 0]\tilde{Q}AU\begin{bmatrix} 0 \\ I_{n-r} \end{bmatrix}. \tag{89}$$

REFERENCES

[1] A. BUNSE-GERSTNER, V. MEHRMANN, AND N. K. NICHOLS, *Derivative feedback for descriptor systems*, FSP Mathematisierung, Universität Bielefeld, Bielefeld, Germany, Materialien LVIII, 1989, presented at the IFAC Workshop on System Structure and Control: State-space and Polynomial Methods, Prague, September 1989.

[2] ———, *On derivative and proportional feedback design for descriptor systems*, in Proc. Internat. Symposium on Mathematical Theory of Networks and Systems '89, M. A. Kaashoek, J. H. Van Schuppen, and A. C. M. Ran, eds., Vol. III, Birkhäuser, Basel, 1990, pp. 437–446.

[3] S. L. CAMPBELL, *Singular Systems of Differential Equations*, Pitman, San Francisco, 1980.

[4] M. CHRISTODOULOU, *Decoupling in the design and synthesis of singular systems*, Automatica, 22 (1986), pp. 245–249.

[5] D. J. COBB, *Feedback and pole placement in descriptor variable systems*, Internat. J. Control, 33 (1981), pp. 1135–1146.

[6] L. DAI, *Singular Control Systems*, Lecture Notes in Control and Inform. Sci., 118, Springer-Verlag, Berlin, 1989.

[7] L. R. FLETCHER, J. KAUTSKY, AND N. K. NICHOLS, *Eigenstructure assignment in descriptor systems*, IEEE Trans. Automat. Control, 31 (1986), pp. 1138–1141.

[8] F. R. GANTMACHER, *Theory of Matrices*, Vols. I, II, Chelsea, New York, 1959.

[9] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, North Oxford Academic, Oxford, 1983.

[10] J. KAUTSKY, N. K. NICHOLS, AND E. K.-W. CHU, *Robust pole assignment in singular control systems*, Linear Algebra Appl., 121 (1989), pp. 9–37.

[11] V. KUCERA AND D. ZAGALAK, *Fundamental theorem of state feedback for singular systems*, Automatica, 24 (1988), pp. 653–658.

[12] F. L. LEWIS, *Descriptor systems: Fundamental matrix, reachability and observability matrices, subspaces* in Proc. 23rd Conference on Decision and Control, Las Vegas, NV, December 1984, pp. 293–298.

[13] D. G. LUENBERGER, *Dynamic equations in descriptor form*, IEEE Trans. Automat. Control, 22 (1977), pp. 312–321.

[14] V. MEHRMANN, *Existence, uniqueness and stability of solutions to singular linear quadratic control problems*, Linear Algebra Appl., 121 (1989), pp. 291–331.

[15] ———, *The autonomous linear quadratic control problem: Theory and numerical algorithms*, Lecture Notes in Control and Inform. Sci., submitted.

[16] R. MUKUNDAN AND W. DAYAWANSA, *Feedback control of singular systems—Proportional and derivative feedback of the state*, Internat. J. Systems Sci., 14 (1983), pp. 615–632.

[17] K. OZCALDIRAN, *Geometric notes on descriptor systems*, Proc. 27th IEEE Conference on Decision and Control, Los Angeles, CA, 1987.

[18] D. W. PEARSON, M. J. CHAPMAN, AND D. N. SHIELDS, *Partial singular value assignment in the design of robust observers for discrete time descriptor systems*, IMA J. Math. Control Inform., 5 (1988), pp. 203–213.

[19] H. H. ROSENBROCK, *Structural properties of linear dynamic systems*, Internat. J. Control, 20 (1974), pp. 191–202.

[20] M. A. SHAYMAN, *Pole placement by dynamic compensation for descriptor systems*, Automatica, 24 (1988), pp. 279–282.

[21] M. A. SHAYMAN AND Z. ZHOU, *Feedback control and classification of generalized linear systems*, IEEE Trans. Automat. Control, 32 (1987), pp. 483–494.

[22] P. VAN DOOREN, *The generalized eigenstructure problem in linear system theory*, IEEE Trans. Automat. Control, 6 (1981), pp. 111–129.

[23] G. C. VERGHESE, B. C. LÉVY, AND T. KAILATH, *A general state space for singular systems*, IEEE Trans. Automat. Control, 26 (1981), pp. 811–831.

[24] G. C. VERGHESE, P. VAN DOOREN, AND T. KAILATH, *Properties of the system matrix of a generalized state space system*, Internat. J. Control, 30 (1979), pp. 235–243.

[25] E. L. YIP AND R. F. SINCOVEC, *Solvability, controllability and observability of continuous descriptor systems*, IEEE Trans. Automat. Control, 26 (1981), pp. 702–707.

[26] Z. ZHOU, M. A. SHAYMAN, T.-J. TARN, *Singular systems: A new approach in the time domain*, IEEE Trans. Automat. Control, 32 (1987), pp. 42–50.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 68–89, January 1992

007

SEPARABLE NONLINEAR LEAST SQUARES WITH MULTIPLE RIGHT-HAND SIDES*

LINDA KAUFMAN† AND GARRETT SYLVESTER‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. One of the significant problems in modal analysis, that of extracting modal parameters from experimental data, can be written as a separable nonlinear least-squares problem in which a linear combination of nonlinear functions is fit to data in many data sets. The linear parameters are to be specific to each data set but the nonlinear parameters have to minimize the least-squares function for all the data sets. Golub and LeVeque have devised a method for problems like the modal analysis problem where there are multiple data sets. For determining the Jacobian J of the problem, their algorithm essentially computes a decomposition as if there were only one data set and then uses this decomposition for all the data sets. Many general nonlinear least-squares solvers compute an orthogonal decomposition of the Jacobian and after the Golub–LeVeque algorithm has been applied, this decomposition is the most time-consuming portion of the method. This paper proposes using the orthogonal decomposition of a specific derivative matrix for one data set to reduce the work in determining the orthogonal decomposition of the Jacobian for all the data sets. For general minimizers which require J^TJ, rather than J itself, the paper shows how to compute J^TJ quickly to take advantage of the structure of the Jacobian in the multiple data set case. For the modal analysis problem, the nonlinear variables in the model also separate, which gives the model additional structure that is exploited to further reduce the construction time. We find that with the economies of speed and storage described below, nonlinear parameter fitting to extract modal parameters from large data sets becomes computationally feasible.

Key words. nonlinear least squares, variable projection

AMS(MOS) subject classifications. 65K10, 90C06, 93E24

1. Introduction.

In many mathematical modeling problems one wishes to fit data $\mathbf{y}$ in a least-squares sense by a model $\boldsymbol{\eta}$ of parameters to be determined. Often this model can be formulated as a linear combination of nonlinear functions as in the exponential model

$$c_1 e^{\alpha_1 t} + c_2 e^{\alpha_2 t} + c_3 \approx y(t) \tag{1.1}$$

or the Gaussian model

$$c_1 e^{-\alpha_1 t} + c_2 e^{-\alpha_2 (t-\alpha_3)^2} + c_3 e^{-\alpha_4 (t-\alpha_5)^2} \approx y(t) \tag{1.2}$$

or the pole finding problem where one has complex data and the model

$$c_1/(i\omega - \alpha_1) + c_2/(i\omega - \alpha_2) \approx y(\omega). \tag{1.3}$$

Notice that all these models have linear parameters $\mathbf{c}$, a vector of length l, nonlinear parameters $\boldsymbol{\alpha}$, and a vector of length p to be determined, and have the general form

$$\boldsymbol{\eta}(\mathbf{c}, \boldsymbol{\alpha}) = \sum_{j=1}^{l} c_j \boldsymbol{\phi}_j(\boldsymbol{\alpha}) + f(\boldsymbol{\alpha}). \tag{1.4}$$

* Received by the editors December 3, 1990; accepted for publication (in revised form) July 23, 1991.

† AT&T Bell Laboratories, Room 2C-461, 600 Mountain Avenue, Murray Hill, New Jersey 07974 (lck@research.att.com).

‡ AT&T Bell Laboratories, Room 2A-207, 67 Whippany Road, Whippany, New Jersey 07981-0903.

In separable nonlinear least-squares problems one has data $\mathbf{y} \in R^n$ and one wishes to determine the unknowns in a model having the form of (1.4) so that

$$\|\mathbf{y} - \boldsymbol{\eta}(\mathbf{c}, \boldsymbol{\alpha})\|_2 = \|\mathbf{y} - (\Phi(\boldsymbol{\alpha})\mathbf{c} + \mathbf{f}(\boldsymbol{\alpha}))\|_2 \tag{1.5}$$

is minimized.

One could give a problem of the form of (1.5) to a general nonlinear least-squares solver, but it might be more economical to take advantage of the structure of (1.4) and determine the c's implicitly as in Golub and Pereyra [5]. They use the fact that for any given $\boldsymbol{\alpha}$, the optimal $\mathbf{c}$ that minimizes (1.5) is given by

$$\mathbf{c} = \Phi^+(\mathbf{y} - \mathbf{f}) \tag{1.6}$$

where Φ^+ is the Moore–Penrose generalized inverse of Φ (see [7]). Golub and Pereyra show that the value of $\boldsymbol{\alpha}$ that minimizes

$$\|(\Phi(\boldsymbol{\alpha})\Phi^+(\boldsymbol{\alpha}) - I)(\mathbf{f} - \mathbf{y})\| \tag{1.7}$$

also minimizes (1.5). They use the value of $\boldsymbol{\alpha}$ that minimizes (1.7) to determine $\mathbf{c}$ from (1.6). Since effective nonlinear least-squares solvers usually demand the specification of the Jacobian matrix, i.e., the matrix of first partials, Golub and Pereyra derive a formula for the columns of the Jacobian matrix J of (1.7), which can be easily computed using the derivative of the Φ matrix and the QR decomposition of the Φ matrix. The advantages of using the Golub and Pereyra approach are several-fold. In the first place, the approach reduces the number of parameters in the problem. If $\boldsymbol{\alpha}$ is a vector of length p and $\mathbf{c}$ of length l, then the Golub and Pereyra approach reduces the problem from $p + l$ parameters to p parameters. Secondly, one does not have to supply initial guesses for the linear parameters. Thirdly, the linear parameters and nonlinear parameters often are of different orders of magnitude, which can play havoc with unsophisticated nonlinear least-squares solvers. This problem is avoided with the Golub and Pereyra approach.

This paper was motivated by a sequence of problems arising in modal analysis and discussed in more detail in § 4. The simplest of these problems had 80 data sets each with 2400 observations that were to be fit with the same general model which had 288 nonlinear parameters and 84 linear parameters. The linear parameters could vary over the data sets, which meant there were really 80×84 or 6720 linear parameters, but the nonlinear parameters were supposed to be the same across all the data sets. Treating the problem as a separable problem rather than a standard one reduced the number of unknowns by a factor of 21. However, the Φ matrix still had 192,000 rows and 6720 columns, unpleasantly large. Fortunately, as pointed out by Golub and LeVeque [4] while trying to do a biological data analysis problem which had similar constraints, the Φ matrix has the form

$$\Phi = \begin{pmatrix} G & & & \\ & G & & \\ & & \ddots & \\ & & & G \\ & & & & G \end{pmatrix} \tag{1.8}$$

where G represents the model for one data set. In the modal analysis problem G had 2400 rows and 84 columns. As pointed out in [4], the QR decomposition of Φ, which was the main ingredient in the Jacobian calculations needed by the underlying general nonlinear least-squares solvers, could be easily computed from the QR decomposition of G, a less daunting calculation.

In § 2 of this paper we will consider some further enhancements of the ideas in [4] to handle the multiple data set problem. The algorithm in [4] treated only the problem of setting up the Jacobian and did not consider using the Jacobian. Many nonlinear least-squares solvers require the QR decomposition of the Jacobian and this often becomes the most time-consuming portion of the computation. We show how to decrease this portion of the computation when **f** is a function of very few of the nonlinear variables; for most of the data sets provided there are more than about five nonlinear variables.

In § 3 we consider the case in which J^TJ is requested by the underlying nonlinear least-squares solver, rather than the Jacobian itself. If each element of α appears in only a few columns of G, using this approach is very attractive. Theoretically in the small version of the modal analysis problem, using the techniques of § 3 reduced the operation count by a factor of 2 over the techniques of § 2. In a larger problem with about 2.7 million observations, 30,000 linear parameters and about 1,300 nonlinear parameters, there was a reduction of a factor of 8 in number of operations.

Section 4 gives a description of the modal analysis problem. Applying the techniques of §§ 2 and 3 to a small version of the problem requires a reasonable amount of computational resources. However, these techniques are not sufficient to handle a physically realistic version of the problem. Fortunately, further algebraic structure can be used: the nonlinear variables themselves separate. One can partition α^T into $(\psi_1, \psi_2, \cdots, \psi_D, \mathbf{s})^T$, and the G matrix itself has the following structure:

$$G = \begin{pmatrix} B & & & A(\mathbf{s})S_1(\psi_1) \\ & B & & A(\mathbf{s})S_2(\psi_2) \\ & & \ddots & \vdots \\ & & & \vdots \\ & & B & A(\mathbf{s})S_{D-1}(\psi_{D-1}) \\ & & & B \quad A(\mathbf{s})S_D(\psi_D) \end{pmatrix}. \tag{1.9}$$

The matrix B in (1.9) is independent of the nonlinear variables. In § 4 we take advantage of the fact that the matrix A is repeated and not dependent on the ψ variables.

Soo and Bates [9] have recently discussed a problem of assigning knots for B-splines, which reduces to a separable nonlinear least-squares problem where there is a further separation of the nonlinear variables. Their G matrix of (1.9) has the same zero structure but the diagonal blocks vary and the right-hand column blocks are just considered as general matrices. They use a two-stage algorithm for finding the QR decomposition of G that is similar to ours but must take into consideration the generality of their blocks. Moreover, in our case because the right-hand column blocks all begin with the same matrix A, the work in forming the Jacobian for all the ψ variables is much less and the Jacobian matrix contains huge blocks of zero rows for these variables.

The techniques of §§ 2 and 3 reduce the work of a multiple data set problem to essentially that of one data set. In § 4, each data set consists of several data sets but the unknowns are all nonlinear. The techniques of handling the innermost data sets are slightly different from those given in §§ 2 and 3 so that § 4 gives more than a numerical application of the techniques of §§ 2 and 3. The methods of §§ 2 and 3 reduce the problem in (1.7) to one essentially involving only the G matrix in (1.8). The methods in § 4 reduce the problem to one involving little more than the A matrix in (1.9). The saving in space means that one can fit the problem in a modern computer, and essentially reduces computational months to an hour.

2. Multiple data sets when the Jacobian J is requested. In this section we present a detailed description of the separable nonlinear least-squares algorithm for multiple data

sets when the underlying nonlinear least-squares solver requests that the user provide the Jacobian matrix J. At each iteration most nonlinear least-squares solvers solve a linear least-squares problem with the Jacobian by performing a QR decomposition of J. Golub and LeVeque [4] have shown that the Jacobian for the multiple data set problem has special structure. We will show in this section that this structure can be used to reduce the number of nonzero rows that the general nonlinear rows thinks are in the Jacobian. Thus the work involved in computing the QR decomposition of the Jacobian is reduced. For the big example given in § 4, using this structure while determining the QR decomposition decreased the total computation time by a factor of 7.

Many algorithms for minimizing

$$\|\mathbf{r}(\alpha)\|_2 \tag{2.1}$$

require the user to provide a subroutine for computing $\mathbf{r}$ and the Jacobian J whose kth column is given by

$$\mathbf{j}_k = \frac{\partial \mathbf{r}(\alpha)}{\partial \alpha_k}. \tag{2.2}$$

In the separable least-squares case

$$\mathbf{r}(\alpha) = (\Phi\Phi^+ - I)(\mathbf{f} - \mathbf{y}) \equiv P_\Phi^\perp(\mathbf{y} - \mathbf{f}) \tag{2.3}$$

and the derivatives of $P_\Phi^\perp$ are

$$\frac{\partial P_\Phi^\perp(\alpha)}{\partial \alpha_k}(\mathbf{y} - \mathbf{f}) = -\left(P_\Phi^\perp(\alpha)\frac{\partial \Phi(\alpha)}{\partial \alpha_k}\Phi^+(\alpha) + \left(P_\Phi^\perp(\alpha)\frac{\partial \Phi(\alpha)}{\partial \alpha_k}\Phi^+(\alpha)\right)^T\right)(\mathbf{y} - \mathbf{f}). \tag{2.4}$$

The easiest way to compute (2.3) and its Jacobian involves the QR decomposition of $\Phi(\alpha)$ given by

$$Q(\alpha)\Phi(\alpha)Z(\alpha) = \begin{pmatrix} R(\alpha) & R_1(\alpha) \\ 0 & 0 \end{pmatrix} \tag{2.5}$$

where Q is an orthogonal matrix, Z is a permutation matrix, and if Φ is an $n \times l$ matrix of rank m, R is an $m \times m$ upper triangular matrix. In (2.3) $\mathbf{r}$ is simply given by

$$\mathbf{r} = Q^T\begin{pmatrix} 0 & \\ & I_{n-m} \end{pmatrix}Q(\mathbf{y} - \mathbf{f}). \tag{2.6}$$

The matrix Q is seldom explicitly formed. It is usually composed of a sequence of Householder transformations and only the information needed to apply these transformations to a vector such as $\mathbf{y} - \mathbf{f}$ is stored. This requires $nm + O(m)$ space and often the Φ matrix is overwritten to store this information.

Kaufman [6] pointed out that if one partitions $Q(\alpha)$ as

$$Q(\alpha) = \begin{pmatrix} Q_1(\alpha) \\ Q_2(\alpha) \end{pmatrix} \tag{2.7}$$

where Q_2 has $n - m$ rows, then

$$\|\mathbf{r}\|_2 = \|Q_2(\alpha)(\mathbf{y} - \mathbf{f})\| \equiv \|\hat{\mathbf{r}}\|_2, \tag{2.8}$$

and she suggested working with $Q_2(\alpha)$ rather than using $\Phi\Phi^+ - I$ as in Golub and Pereyra [5]. Assume Z is partitioned as $Z = (Z_1 \mid Z_2)$, where Z_1 has m columns. In [6] it is shown that the derivative

$$\begin{aligned}\frac{\partial Q_2(\alpha)}{\partial \alpha_k}(\mathbf{y}-\mathbf{f}) &= \left(-Q_2(\alpha)\frac{\partial \Phi(\alpha)}{\partial \alpha_k}\Phi^+ + H\right)(\mathbf{y}-\mathbf{f}) \\ &\equiv \left(-Q_2(\alpha)\frac{\partial \Phi(\alpha)}{\partial \alpha_k}Z_1 R^{-1}Q_1 + H\right)(\mathbf{y}-\mathbf{f}),\end{aligned} \tag{2.9}$$

where H is dependent on the exact representation chosen for Q_2. (Note that although Q_1 is unique up to a diagonal unitary matrix, Q_2 has more freedom to vary.) Moreover, for Gauss–Newton methods and Marquardt procedures, which are the most widely used general nonlinear least-squares solvers, the H matrix can be neglected because of orthogonality considerations (see [6]). Again there is no need to form the Q_2 matrix explicitly. Only the information needed to apply the Householder transformations is needed. The subroutine NSG in PORT [3] implements the separable nonlinear least squares algorithm suggested in [6] using the general nonlinear least-squares algorithm described in [1].

As shown in [4], for the multiple data set problem, where Φ has the form (1.8), one needs only the QR decomposition of G given by

$$\tilde{Q}G\tilde{Z} = \begin{pmatrix} \tilde{R} & \tilde{R}_1 \\ 0 & 0 \end{pmatrix} \tag{2.10}$$

where if there are u data sets in the problem so that G is $(n/u) \times (l/u)$, then $\tilde{R}$ is a nonsingular upper triangular matrix of rank $\tilde{m}$, and $\tilde{Q}$ may be partitioned as

$$\tilde{Q} = \begin{pmatrix} \tilde{Q}_1 \\ \tilde{Q}_2 \end{pmatrix} \tag{2.11}$$

where $\tilde{Q}_2$ has $n/u - \tilde{m}$ rows, and Z may be partitioned as $Z = (Z_1, Z_2)$ where Z_1 has $\tilde{m}$ columns.

If one ignored the structure of Φ, computing its QR decomposition would require $nl^2 + O(nl)$ multiplications. Taking advantage of its structure reduces the operation count to $nl^2/u^3 + O(nl/u^2)$ multiplications. In a small modal analysis example, ignoring the structure of Φ meant that more than 8570 billion multiplications would be required to compute its QR decomposition, while taking advantage of its structure reduced the number of multiplications to about 17 million, a speedup of 510,000. More significantly it reduced the space requirements from over a billion words to about 3 million words.

Let $\tilde{n} = n/u$ and $\tilde{l} = l/u$. If one designated the data for the ith data set as $\mathbf{y}_i$, and the nonlinear term in the model as $\mathbf{f}_i$, then $\mathbf{r}$ of (2.6) is given by

$$\mathbf{r} = \begin{pmatrix} \tilde{Q}^T \begin{pmatrix} 0 & \\ & I_{\tilde{n}-\tilde{m}} \end{pmatrix} \tilde{Q}(\mathbf{y}_1 - \mathbf{f}_1) \\ \vdots \\ \tilde{Q}^T \begin{pmatrix} 0 & \\ & I_{\tilde{n}-\tilde{m}} \end{pmatrix} \tilde{Q}(\mathbf{y}_u - \mathbf{f}_u) \end{pmatrix} \tag{2.12}$$

and $\hat{\mathbf{r}}$ of (2.8) is given by

$$\hat{\mathbf{r}} = \begin{pmatrix} \tilde{Q}_2(\alpha)(\mathbf{y}_1 - \mathbf{f}_1) \\ \vdots \\ \tilde{Q}_2(\alpha)(\mathbf{y}_u - \mathbf{f}_u) \end{pmatrix}. \tag{2.13}$$

Let

$$B = \begin{pmatrix} 0 & \\ & I_{\tilde{n}-\tilde{m}} \end{pmatrix} \tilde{Q} \frac{\partial G(\alpha)}{\partial \alpha_k} \tilde{Z}_1 (\tilde{R}^{-1} : 0) \begin{pmatrix} I_{\tilde{m}} & \\ & 0 \end{pmatrix}. \tag{2.14}$$

Then the equivalent of (2.4) is

$$\frac{\partial P_{\Phi}^{\perp}(\alpha)}{\partial \alpha_k}(\mathbf{y} - \mathbf{f}) = \begin{pmatrix} -\tilde{Q}^T(B + B^T)\tilde{Q}(\mathbf{y}_1 - \mathbf{f}_1) \\ \vdots \\ -\tilde{Q}^T(B + B^T)\tilde{Q}(\mathbf{y}_u - \mathbf{f}_u) \end{pmatrix}. \tag{2.15}$$

Moreover, the equivalent of (2.9), neglecting H, is

$$\frac{\partial Q_2(\alpha)}{\partial \alpha_k}(\mathbf{y} - \mathbf{f}) = \begin{pmatrix} -\tilde{Q}_2(\alpha)\frac{\partial G(\alpha)}{\partial \alpha_k}\tilde{Z}_1\tilde{R}^{-1}\tilde{Q}_1(\mathbf{y}_1 - \mathbf{f}_1) \\ \vdots \\ -\tilde{Q}_2(\alpha)\frac{\partial G(\alpha)}{\partial \alpha_k}\tilde{Z}_1\tilde{R}^{-1}\tilde{Q}_1(\mathbf{y}_u - \mathbf{f}_u) \end{pmatrix}. \tag{2.16}$$

Thus the QR decomposition of G and the derivative of G are used s times during the computation of the Jacobian. This reuse of data is evident in the operation counts.

Normally finding the Jacobian using (2.9), but ignoring H, one might proceed as follows.

K algorithm.

(1) Form $\mathbf{u} = Q(\mathbf{y} - \mathbf{f})$

(2) Partition $\mathbf{u} = \begin{pmatrix} \mathbf{v} \\ \mathbf{r} \end{pmatrix}$

(3) Solve $R\mathbf{w} = \mathbf{v}$. Set $\mathbf{c} = Z_1\mathbf{w}$

(4) For $k = 1, \cdots, p$

Form the columns $\mathbf{e}_k = -\dfrac{\partial \Phi(\alpha)}{\partial \alpha_k}\mathbf{c} - \dfrac{\partial \mathbf{f}}{\partial \alpha_k}$ of E

(5) Form $J = \begin{pmatrix} 0 \\ I_{n-m} \end{pmatrix} QE$

Assuming that Q is not explicitly formed but saved as a sequence of Householder transformations, steps (1) and (5) of the above algorithm together require $m(2n - m)(p + 1) + O(m(p + 1))$ multiplications. Step 3 requires $m^2/2$ operations and assuming that there are d derivative columns, step (4) requires nd operations.

In the multiple data set case it makes more sense to apply $\tilde{Q}$ to the derivative matrices of G and $\mathbf{f}$ directly rather than to the equivalent of E as in the last step of the above algorithm. Thus to implement the equivalent of Kaufman's algorithm using (2.16), one would proceed as follows.

Golub–LeVeque-K algorithm.

(1) For $i = 1, \cdots, u$

Form $\mathbf{u}_i = Q(\mathbf{y}_i - \mathbf{f}_i)$

Partition $\mathbf{u}_i = \begin{pmatrix} \mathbf{v}_i \\ \mathbf{r}_i \end{pmatrix}$

Solve $\tilde{R}\mathbf{w}_i = \mathbf{v}_i$. Set $\mathbf{c}_i = \tilde{Z}_1\mathbf{w}_i$.

(2) Let D = the nonzero columns of all the derivatives of the columns of G taken with respect to all the elements of α. D should have d columns.
(3) Form $M=-\begin{pmatrix}0\\ I_{\tilde{n}-\tilde{m}}\end{pmatrix}\tilde{Q}D$.
(4) For $k=1, \cdots, p$
Let M_k be those columns of M corresponding to derivatives with respect of α_k with columns of zeros inserted so that M_k has $\tilde{l}$ columns.
The kth column of the Jacobian is given by

$$\begin{pmatrix} M_k\mathbf{c}_1+\begin{pmatrix}0\\ I_{\tilde{n}-\tilde{m}}\end{pmatrix}\tilde{Q}\dfrac{\partial \mathbf{f}_1}{\partial \alpha_k} \\ \vdots \\ M_k\mathbf{c}_u+\begin{pmatrix}0\\ I_{\tilde{n}-\tilde{m}}\end{pmatrix}\tilde{Q}\dfrac{\partial \mathbf{f}_u}{\partial \alpha_k} \end{pmatrix}.$$

In the above algorithm step (1) requires about $2n\tilde{m}-u\tilde{m}^2/2$ multiplications and step (3) requires $2\tilde{n}\tilde{m}d$ multiplications. In step (4) M_k will probably have several columns, which are all 0. If one takes advantage of these columns while working with M_k, then step (4) requires nd multiplications, which means that there is a substantial saving, as Golub and LeVeque indicate.

Implementing the Golub–Pereyra algorithm using (2.15) in the multiple data set case is slightly more complicated and one would proceed as follows.

Golub–LeVeque-GP algorithm.
(1)–(3) Proceed as in the Golub–LeVeque-K algorithm.
(4) For $k=1, \cdots, p$
Let M_k be those columns of M corresponding to derivatives with respect to α_k with columns of zeros inserted so that it has $\tilde{l}$ columns.
Let $M_k^{(2)}$ be the last $\tilde{n}-\tilde{m}$ rows of M_k.
For $i=1, \cdots, s$
Form $\mathbf{u}_{ik} = M_k^T\mathbf{r}_i$
Solve $\tilde{R}^T\mathbf{w}_{ki} = Z_1^T\mathbf{u}_{ki}$.
The kth column of the Jacobian is given by

$$\begin{pmatrix} \tilde{Q}^T\begin{pmatrix}\mathbf{w}_{k1}\\ M_k^{(2)}\mathbf{c}_1\end{pmatrix}+\tilde{Q}^T\begin{pmatrix}0\\ I_{\tilde{n}-\tilde{m}}\end{pmatrix}\tilde{Q}\dfrac{\partial \mathbf{f}_1}{\partial \alpha_k} \\ \vdots \\ \tilde{Q}^T\begin{pmatrix}\mathbf{w}_{ks}\\ M_k^{(2)}\mathbf{c}_u\end{pmatrix}+\tilde{Q}^T\begin{pmatrix}0\\ I_{\tilde{n}-\tilde{m}}\end{pmatrix}\tilde{Q}\dfrac{\partial \mathbf{f}_u}{\partial \alpha_k} \end{pmatrix}.$$

Because of the size of our Jacobian, we needed an underlying nonlinear least-squares solver that accepted groups of rows of the Jacobian and did not require the whole Jacobian at once. Thus we could not consider the general nonlinear least squares solver in MINPACK [8]. We chose to use RN2G in PORT [3]. For our problem it is very natural to compute that part of the Jacobian corresponding to a specific data set and reuse the Jacobian space. Thus only $O(np/u)$ space is needed, which means that it is usually possible to solve most problems with multiple data sets on even modest machines using the Golub–LeVeque algorithm. However in some circumstances one can further enhance

the Golub–LeVeque algorithm by looking at the intended use of the Jacobian by the general nonlinear least-squares solver.

A general nonlinear least-squares solver usually solves a linear least-squares system with the Jacobian it has obtained. Thus after the Jacobian is computed using the Golub–LeVeque-K algorithm, the QR decomposition of the Jacobian will be determined at the cost of $u(\tilde{n}-\tilde{m})p^2 + O(u(\tilde{n}-\tilde{m})p)$ multiplications. When p is greater than say 5, the cost of this additional QR decomposition can be significant. Excluding the cost of computing G and its derivatives, 10 percent of the computation time for our small modal analysis problem was spent in forming the Jacobian with the Golub–LeVeque-K algorithm and 90 percent of the computation time was spent in obtaining its QR decomposition by the general nonlinear least-squares solver.

If the underlying least-squares program is only going to compute the QR decomposition of the Jacobian and perform no further computations with the Jacobian, there is no need to apply the $\tilde{Q}^T$ in step (4) of the Golub–LeVeque-GP algorithm. This eliminates most of the excess work of the Golub–LeVeque-GP algorithm over the Golub–LeVeque-K algorithm. It also suggests a mechanism for decreasing the number of nonzero rows in the Jacobian if the nonlinear term is not present (i.e., $\mathbf{f} = \mathbf{0}$) in both the K and GP versions of the Golub–LeVeque algorithm.

If the nonlinear term is absent in the model and if $\tilde{n} - \tilde{m} > d$, the number of nonzero rows in the Jacobian can be decreased by performing a QR decomposition on the last $\tilde{n} - \tilde{m}$ rows of the M matrix of the Golub–LeVeque algorithm. Thus one finds an upper trapezoidal matrix U of rank m_d, an orthogonal matrix Q_d, and a permutation matrix Z_d such that

$$Q_d M Z_d = \begin{pmatrix} M_1 \\ U \\ 0 \end{pmatrix} \tag{2.17}$$

where M_1 represents the first $\tilde{m}$ rows of the M matrix. Now U is substituted in step (4) of the Golub–LeVeque algorithm for M and each residual block is multiplied by Q_d. Because columns of the M matrix appear in all of the data sets, by doing one decomposition we decrease the number of nonzero rows in the Jacobian for all the data sets.

Thus instead of giving the general nonlinear least-squares solver a Jacobian that has $u \times (\tilde{n} - \tilde{m})$ nonzero rows, the general nonlinear least-squares solver should be given a Jacobian that has $m_d \times u$ nonzero rows. The idea costs $(\tilde{n} - \tilde{m})m_d(2d - m_d)$ multiplications to perform the decomposition in (2.17) and $2u(\tilde{n} - \tilde{m})m_d$ multiplications to apply Q_d to the residual, but it reduces the cost in step (4) of the Golub–LeVeque-K algorithm and in the QR decomposition of the Jacobian from $(u(\tilde{n} - \tilde{m}))(d + p^2)$ multiplications to $m_d u(d + p^2)$ multiplications. In the small modal analysis problem where d is about $2p$ and m_d is about p, one could save about 298 billion multiplications by performing 13.5 billion multiplications. Moreover, one can apply Q_d to the u residuals simultaneously and take even further advantage of a multiprocessing machine.

Let $\lambda = m_d/(\tilde{n} - \tilde{m})$. Incorporating (2.17) into the Golub–LeVeque-K algorithm speeds up the computation of the QR decomposition of the Jacobian by approximately

$$\frac{(p-1)pu}{2m_d u + \lambda u(p-1)p + m_d(2d - m_d)}. \tag{2.18}$$

In the small modal analysis problem with $p = 288$, $m_d = 288$, $d = 576$, λ approximately .125, and $u = 80$, using (2.17) was certainly beneficial as Table 4 indicates. However for simpler problems the case for using (2.17) is not as apparent. Certainly for the exponential fitting problem in (1.1) with $p = 2$, (2.18) suggests that one would not use (2.17).

TABLE 1
Time for problem (2.19).

u	Time for Golub–LeVeque-K without (2.17)	Time for Golub–LeVeque-K with (2.17)	Ratio	(2.18)
5	2.3	1.7	1.4	1.5
10	3.8	2.6	1.5	1.8
50	15.9	9.4	1.7	2.1

To see whether (2.17) helps in a relatively small example, we considered the problem (commonly called "Osborne 2") given in [5] of fitting Gaussians with an exponential background

$$a_1 e^{-\alpha_1 t} + a_2 e^{-\alpha_2 (t-\alpha_5)^2} + a_3 e^{-\alpha_3 (t-\alpha_6)^2} + a_4 e^{-\alpha_4 (t-\alpha_7)^2}, \tag{2.19}$$

which had $p = 7$. We used the data used by [5] which had 65 observations and repeated those observations for u data sets and varied u. Table 1 shows our results on a Vax 8550 using the Golub–LeVeque-K algorithm using (2.17) and not using (2.17). Time is reported in seconds and has about a 10 percent accuracy. The underlying nonlinear least-squares solver was the one given by Dennis, Gay, and Welsch [1] with fixed scaling and use of the S matrix in their quadratic model disabled. The inner product routine was replaced by one that did not check for underflows each multiplication. Table 1 shows the total time for the problem so one could ascertain the global effect of using (2.17). The computations all required 16 function evaluations and 13 derivative evaluations for convergence. The data corroborates our theory that the QR factorization of the Jacobian dominates the computation and that (2.18) is a fairly good predictor of the behavior of the total algorithm.

If the same nonlinear term appeared in all data sets, i.e., all the $\mathbf{f}_i$'s were identical, one could augment the M matrix (2.13) with the nonzero derivative columns of $\mathbf{f}_1$ and use the same technique. In the problem suggested by Sylvester [10] for about 95 percent of the data sets the nonlinear term was missing. However, as explained in § 4.1.3, the nonlinear term had such a special form that (2.17) could be used even to handle the nonlinear terms.

In this section we have shown that with multiple data sets one can reduce the computation beyond that given by the Golub–LeVeque algorithm by reducing the number of nonzero rows that the general nonlinear least-squares solver thinks are in the Jacobian.

3. Multiple data sets when J^TJ is required. In the previous section we considered the multiple data set problem when the Jacobian J is requested and at each iteration the underlying nonlinear least-squares solver solves a linear least-squares problem using the QR decomposition of the $n \times p$ Jacobian J given by

$$Q_J J = \begin{pmatrix} R_J \\ 0 \end{pmatrix}, \tag{3.1}$$

where Q_J is an orthogonal matrix and R_J is an upper triangular matrix with p rows. In this section we consider nonlinear least-squares solvers which at each iteration solve a linear least-squares problem by determining the Cholesky factor of $\Lambda = J^TJ$. In infinite precision arithmetic the two methods produce the same result. In the multiple data set problem, the matrix Λ has special structure and, as we will show in this section, computing

Λ and its Cholesky factorization can be much more efficient than going through the QR route. The speedup depends on the number of terms in the model in which each nonlinear variable occurs. In our big modal analysis problem the speedup in the overall problem was a factor of 8.

In the QR approach of the previous section the matrix Q_J is rarely saved and all that is needed is R_J and $Q_J\mathbf{x}$ for some vector $\mathbf{x}$. However, theoretically, neglecting roundoff error, these quantities can be computed in a different manner. If $\Lambda = J^TJ$ and if the Cholesky factorization of Λ is

$$\Lambda = R_\Lambda^T R_\Lambda, \tag{3.2}$$

where R_Λ is an upper triangular matrix, then, up to the sign of the rows, $R_\Lambda = R_J$. Moreover, if

$$\mathbf{x}_\Lambda = R_\Lambda^{-T} J\mathbf{x}, \tag{3.3}$$

then $\mathbf{x}_\Lambda = Q_J\mathbf{x}$ up to the corresponding signs of the elements. Thus theoretically one could compute Λ and use (3.2) and (3.3) rather than computing (3.1). The main reason that (3.1) is preferred is stability, especially when J is nearly singular. It is easy to construct examples in which J is nonsingular, but because of roundoff error Λ will be numerically singular (see [7]). Moreover, the error in the computation of R_J depends on the condition number of J, while that of R_Λ depends on the square of the condition number of J. When n is approximately p the number of multiplications required by (3.1) is approximately that expended for determining Λ and R_Λ. When $n \gg p$ the price of using (3.1) is about double that for computing Λ and R_Λ, but it is usually considered a modest price because of the information about singularity which one gleans, and because in many nonlinear least-squares problems computing (3.1) is not the most time-consuming part of the problem compared to determining the Jacobian.

In our situation with multiple right-hand sides, the case for computing J^TJ and using (3.2) and (3.3) is somewhat stronger. First of all, redundant terms in the model will be revealed in the QR decomposition of the model matrix. Thus the information which is usually obtained from the Jacobian about singularity can be obtained to a certain extent elsewhere. Secondly, there are problems in which the computation of the QR decomposition is the major portion of the time for the total algorithm, and by going through the normal equations approach of (3.2) one can gain factors much larger than 2 by taking advantage of the fact that this is a multiple right-hand side problem. Note that we are not advocating that an approach similar to (3.2) be taken for the model matrix Φ; we are only advocating it for the Jacobian.

Let $J^{(i)}$ represent the Jacobian for the ith data set. Then

$$J^TJ = \sum_{i=1}^{u} J^{(i)T} J^{(i)}. \tag{3.4}$$

If the nonlinear term $\mathbf{f}_i$ is missing from the ith data set, then referring to the Golub–LeVeque-K algorithm in § 2, the kth column of $J^{(i)}$ has the form

$$\mathbf{j}_k^{(i)} = M_k\mathbf{c}_i. \tag{3.5}$$

If $\Lambda^{(i)} = J^{(i)T}J^{(i)}$, then its (k, u) component is

$$\lambda_{ku}^{(i)} = \mathbf{c}_i^T(M_k^TM_u)\mathbf{c}_i. \tag{3.6}$$

Since each parameter might appear only a few times in a model, only a few columns of M_k might be nonzero. In the case of (2.19) the matrix $M_k^TM_u$ will have only one nonzero element and thus can be precomputed easily and used for all data sets.

As in the Golub–LeVeque-K algorithm, let M be the concatenation of the nonzero derivative columns of G which have been multiplied by Q_2. Let

$$\hat{M} = M^T M. \tag{3.7}$$

Assume each of the p nonlinear variables appears in v columns of G so that $\hat{M}$ has vp rows and columns. Assuming that $\hat{M}$ has been precomputed, then computing $\Lambda^{(i)}$ by multiplying first by $\mathbf{c}_i$ on the right and then the result by $\mathbf{c}_i^T$ on the left requires $p^2(v^2+v)/2$ multiplications. Computing Λ requires $p^2u(v^2+v)/2$ multiplications and computing R_Λ in (3.2) requires $p^3/6$ multiplications. In comparison, if M is of rank m_d, computing the QR decomposition (3.1) using (2.17) requires about um_dp^2 multiplications. Thus if $m_d = p$ and $v = 1$, as in (2.19), the time required for (3.1) with (2.17) divided by the time required to find R_Λ would be

$$p/(1+p/(6u)), \tag{3.8}$$

a healthy speedup for large values of p. In the modal analysis problem in which $v = 2$ and $m_d = p$, the speedup is about

$$p/(3+p/(6u)). \tag{3.9}$$

For the largest problems with $p = 1500$ and $d = 260$, (3.9) comes to about 378, which makes using J^TJ very attractive.

Of course there is the initial overhead of computing $\hat{M}$ of (3.7) and $J^T\mathbf{x}$ in (3.3). Computing $\hat{M}$ requires $\tilde{n}d^2/2$ multiplications. However when using (3.1) one would want to compute the QR decomposition in (2.17). When using (3.2) this is no longer necessary. Computing (2.17) requires $(\tilde{n}-\tilde{l})m_d(2d-m_d)$ multiplications. Thus when $m_d = d$, computing $\hat{M}$ is about half the cost of computing the decomposition of (2.17) and if $m_d = d/2$, they both require about the same amount of effort. If one uses the normal equation approach, the J matrix itself need not be computed or stored. If one does not use (2.17) when using the QR approach, the cost of forming $J^T\mathbf{x}$ is approximately that of forming J, once M and the $\mathbf{c}$'s have been determined. If one uses (2.17) with the QR approach, then the cost of actually forming the Jacobian is reduced, but one has to apply Q_d to the residual which again is similar to forming $J^T\mathbf{x}$. Thus the additional overhead for using the normal equations approach is no more than the overhead of using (2.17) and the QR decomposition approach. For problems in which after using (2.17), computing (3.1) is the dominant user of time, then using (3.2) and (3.3) is more cost-effective.

Fortunately, in the PORT library the subroutine MNH, a general minimizer in which the user provides the Hessian matrix of second partials, delivers the same iterates when given J^TJ as the Hessian as N2G does when the S approximation to the Hessian is set to the 0 matrix. Thus one is able to assess the speedup computationally. Table 2

TABLE 2
Time for problem (2.19).

u	Time for Golub–LeVeque-K without (2.17)	Time for Golub–LeVeque-K with (2.17)	Time for normal equations
5	2.3	1.7	1.2
10	3.8	2.6	1.6
50	15.9	9.4	4.7

shows the computation time for (2.19) for the Golub–LeVeque-K algorithm with and without (2.17) and for MNH using J^TJ as the Hessian. For this problem (3.1) was a small part of the total computation and thus a large speedup is not noticed. In the next section, where we give a more detailed accounting of the modal analysis problem, we show that for the large problem, using the normal equations approach versus the QR of the Jacobian with (2.17) gives an overall speedup of about 8 for the largest problem. Thus, the normal equations approach is not simply one to be used if the appropriate software is available. It is definitely the method of choice.

4. The modal analysis problem. In this section we describe a sequence of separable nonlinear least-squares problems arising in the decomposition of multichannel transfer function data into complex natural modes [2]. Applying the Golub–LeVeque algorithm to the largest problem in the sequence reduces the numbers of multiplications per iteration by a factor of about 500 and reduces the space requirements from about 100 billion words to about 50 million words, a huge decrease but still insufficient. Applying the algorithm given in § 3 reduces the operation count by another factor of 50 but still does not decrease the space requirements significantly. In this section we use the fact that the G matrix of § 2 has the structure of (1.9) and that the nonlinear variables *separate* further. Our ideas reduce the operation count by another factor of 15 and reduce the space requirements to about 7 million words for the largest problem, an amount that is tolerable for the Multiflow machine, where the algorithm has been implemented.

Using engineering-oriented notation, the modal analysis problem can be stated as follows: Fit the complex data y_{krd}, $k \leqq N_K$, $r \leqq N_R$, $d \leqq N_D$, $d \leqq r$, in a least-squares sense to the model

$$\eta_{krd} = \sum_{m=1}^{N_M} \left(\frac{\psi_{rm}\psi_{dm}}{i\omega_k - s_m} + \frac{\psi_{rm}^*\psi_{dm}^*}{i\omega_k - s_m^*} \right) + \sum_{\mu=-N_L}^{N_U} \beta_{rd\mu}(i\omega_k)^\mu, \tag{4.1}$$

where $i = (-1)^{1/2}$, $*$ indicates conjugation, and the ω's are known sample frequencies. The physical interpretation of the fitting problem (4.1) is as follows. A body, for example an airplane, is driven by time-harmonic forces of angular frequency ω applied at N_D points, indexed by d. The resulting velocity (in some direction) is measured at N_R "receive" points, indexed by r. As the frequency is stepped through a set of N_K samples in some measurement band, one obtains transfer functions y_{krd} from applied force to velocity response. The resonant characteristics of the body can be investigated by identifying the complex poles s_m and residues $\psi_{rm}\psi_{dm}$ of the N_M in-band modes describing the transfer function data. The out-of-band modes are collectively modeled by the last term $\sum_{\mu=-N_L}^{N_U} \beta_{rd\mu}(i\omega k)^\mu$ in (4.1).

A sequence of large-scale problems of the type (4.1) has been investigated, and, in one of the largest, N_M is 61, N_D is 10, N_R is 261, N_K is 520, and $v = N_U + N_L + 1$ is about 3. Thus there are $520 \times 261 \times 10 = 1{,}357{,}200$ complex data points. The unknowns ψ_{rm} for $N_D < r \leqq N_R$ appear linearly. There are $251 \times 61 = 15{,}311$ of them and they are complex. The β's, which we call background coefficients, also appear linearly. They are real and there are only about 7830 of them, a small number compared to the other numbers in the problem. There are 61 complex s's which appear strictly nonlinearly, and there are $61 \times 10 = 610$ complex ψ_{dm}, $d \leqq N_D$, which appear quadratically, and one may wish to treat them slightly differently than the s's. Obviously the problem has many unknowns, which appear linearly and nonlinearly, and much data.

One way to organize the problem is to consider that there are N_R data sets. The first N_D have a strict linear term with the β's and a strict nonlinear term, involving the Ψ_{dm} and s_m, with no linear coefficients. The last $N_R - N_D$ data sets each have $N_K \times N_D =$

5200 complex observations and the model may be viewed as trying to find a linear combination of nonlinear terms as in a separable nonlinear least-squares problem. To help us describe the problem, let $\mathbf{y}_r$ be the $N_K \times \min(r, N_D)$-element data vector for the rth data set, let $\mathbf{s}$ be a vector containing all the s's, let ψ_r be a vector containing the ψ_{rm}'s, let β_r be a vector containing all the $\beta_{rd\mu}$'s, and let $\psi = (\psi_1^T, \cdots, \psi_{N_D}^T)^T$. Assume the elements of a complex $N_K \times v$ matrix B are given by

$$b_{kj} = (i\omega_k)^{j-1-N_L} \tag{4.2}$$

and let G_r be a block diagonal matrix containing r copies of the B matrix along its diagonal so that $G_r = I_r \otimes B$. Then for data set r, $r \leqq N_D$, there exists a vector $\mathbf{f}_r(\mathbf{s}, \psi)$ which takes into consideration the first term in (4.1) so that the least-squares problem for this data set can be expressed as minimizing

$$\|G_r\beta_r + \mathbf{f}_r(\mathbf{s}, \psi) - \mathbf{y}_r\|. \tag{4.3}$$

For $r > N_D$ the model for (4.1) will have more linear variables and there is no need to have a separate function $\mathbf{f}$. For these data sets one can form a matrix A which is only a function of the s's and matrices S_d that are only a function of ψ_d and put them into a matrix G which has the form of (1.9). Moreover the S matrices in complex arithmetic would be diagonal.

Obviously the problem is a separable nonlinear least-squares problem with multiple right-hand sides. Since the columns of G_r for $r < N_D$, are submatrices of G, one can use the Golub–LeVeque algorithm easily and at least for the $N_R - N_D$ data sets, i.e., for 95 percent of the data sets, (2.17) is applicable.

In Table 3 we give approximate storage requirements for the various algorithms discussed in this section for the large modal analysis problem and a smaller test problem which had about 80 data sets, each with 1200 complex observations, and (referring to (4.1)) had $N_D = 3$, $N_M = 42$. The storage requirement for the routines requiring the explicit Jacobian was based on subroutine RN2G in the PORT library [3], which is based on the algorithm given in [1]. This subroutine was chosen because it does not require the whole Jacobian at once but accepts groups of rows. We chose to deliver a block at a time. The storage requirement for the algorithm supplying J^TJ was based on MNH in PORT. Table 3 indicates that the storage requirements are still a major concern and it is this concern that will be addressed in this section.

Table 4 gives an approximate multiplication count for each iteration for the various sections of the various algorithms in §§ 2 and 3 for the two modal analysis problems covered by Table 3. No attempt was made to take advantage of the structure of G in (1.9) in any of the estimates. The table shows how using the ideas of Golub–LeVeque decreased the cost of the total algorithm and left that part of the problem, which seemed inconsequential at first, as the most time-consuming operation. It shows how the idea of

TABLE 3
Comparison of methods—Storage requirements.

	Small problem	Big problem
Variable projection algorithm [6]	1400×10^6	936×10^8
Golub–LeVeque algorithm	2.6×10^6	$.49 \times 10^8$
Normal equations	2.1×10^6	$.38 \times 10^8$

TABLE 4
Comparison of methods—Number of multiplications.

	Small problem	Big problem
Variable projection [6]:		
Decomposition of Φ	857×10^{10}	274×10^{13}
Determining Jacobian	74×10^{10}	23×10^{13}
QR of Jacobian	1.6×10^{10}	$.5 \times 10^{13}$
Total	933×10^{10}	298×10^{13}
Golub–LeVeque algorithm:		
Decomposition of Φ	1.7×10^{7}	$.2 \times 10^{9}$
Determining Jacobian	37.3×10^{7}	15×10^{9}
QR of Jacobian	1590×10^{7}	488×10^{10}
Total	1629×10^{7}	490×10^{10}
Golub–LeVeque with (2.17):		
Decomposition of Φ	1.7×10^{7}	$.2 \times 10^{9}$
Determining Jacobian	85×10^{7}	71×10^{9}
QR of Jacobian	191×10^{7}	631×10^{9}
Total	287×10^{7}	702×10^{9}
Using J^TJ:		
Decomposition of Φ	1.7×10^{7}	$.2 \times 19^{9}$
Determining $\bar{M}$ and gradient	117×10^{7}	90×10^{9}
Determining Λ and R_Λ	2.4×10^{7}	1.8×10^{9}
Total	121×10^{7}	92×10^{9}

computing the QR decomposition of M in (2.17) made a significant dent in the problem, and that the normal equations approach of using J^TJ yields further substantial savings. Now the biggest part of the problem again involves the matrices that make up the Jacobian.

Let us consider the G matrix of (1.9) generated from the modal analysis problem. Although the elements in the modal analysis problem are complex, we can implement the problem in real arithmetic by doubling the number of observations and doubling the number of columns in the A and S matrices. The S matrices will be block diagonal with diagonal blocks of order 2×2. Assume B in (1.9) is a $t \times v$ matrix and A is a $t \times w$ matrix where $t > (v + w)$. Thus G will have $N_D \times t$ rows. We shall assume B has full rank and thus there is no need for column pivoting for stability when dealing with B. As we shall show, one can reduce the problem of obtaining the QR decomposition of G to one involving the QR decomposition of $(B \mid A)$ followed by another decomposition of a matrix that has $N_D \times w$ rows and considers the information in the S matrices. Specifically, consider the QR decomposition given by

$$Y(B \mid A)\begin{pmatrix} I_v & 0 \\ 0 & Z_A \end{pmatrix} = \begin{pmatrix} V & W \\ 0 & T \\ 0 & 0 \end{pmatrix}, \tag{4.4}$$

where Y is an orthogonal matrix, Z_A is a permutation matrix, V is a $v \times v$ upper triangular matrix, W is a $v \times w$ matrix, and T is an $m_A \times w$ trapezoidal matrix of rank m_A. Let

$$\tilde{G} = (I_{N_D} \otimes Y)G\begin{bmatrix} I_{N_D \times v} & 0 \\ 0 & Z_A \end{bmatrix}.$$

From (1.9) we see that there must exist a permutation matrix Z such that

$$Z\tilde{G}=\begin{pmatrix} V & & & WS_1 \\ & \ddots & & \vdots \\ & & V & WS_{N_D} \\ & & & TS_1 \\ & & & \vdots \\ & & & TS_{N_D} \\ & 0 & & \end{pmatrix}, \tag{4.5}$$

which is almost in upper triangular form. Let C be the matrix with $N_D \times m_A$ rows and w columns

$$C=\begin{pmatrix} TS_1 \\ \vdots \\ TS_{N_D} \end{pmatrix}. \tag{4.6}$$

Assume C has rank m_A. Let $\tilde{P}$ be the sequence of Householder transformations, which reduces C to the upper trapezoidal matrix $(K{:}K_1)$ where K is upper triangular and has m_A rows so that

$$\tilde{P}C=\begin{pmatrix} K & K_1 \\ 0 & 0 \end{pmatrix}. \tag{4.7}$$

When $\tilde{P}$ is applied to rows $vN_D + 1$ through $N_D(v + m_A)$ of $Z\tilde{G}$ of (4.5) the upper triangular matrix $\tilde{R}$ of (2.10) is produced.

Since the S matrices are block 2×2 matrices, this decomposition requires far less storage space than neglecting the algebraic structure of G. Applying $\tilde{Q}$ to a vector $\mathbf{x}$ involves

(1) N_D applications of Y in (4.4) to appropriate subvectors.

(2) Applying $\tilde{P}$ of (4.7) to the appropriate elements of $I_{N_D} \otimes Y\mathbf{x}$.

Assuming the matrices are of full rank, obtaining the QR decomposition of G without taking advantage of its structure requires $O(N_D t(N_D v + w)^2 - (N_D v + w)^3/3)$ multiplications. Applying the transformations that make up the $\tilde{Q}$ matrix to a vector requires about another $2N_D t(N_D v + w) - (N_D v + w)^2$ multiplications. If one takes advantage of the structure of G as outlined above, then $t(v + w)^2 + N_D w^3 - (v + w)^3/3 - w^3/3$ multiplications are required to obtain the information for the decomposition and another $2t(v + w) + 2N_D w^2 - (v + w)^2 - w^2$ multiplications to apply the information to a vector. Thus the cost of computing the decomposition is greatly reduced, but the cost of applying the transformations to a random vector, such as the data for all the data sets, is slightly more. Because most of the columns in the derivative of G have the same algebraic structures as the matrices of G, applying $\tilde{Q}$ to these columns using Y and P is cheap as we shall see.

4.1. When a Jacobian is required. In this section we discuss the structure of the Jacobian for the modal analysis problem and show how to use this structure to decrease the work involved in obtaining its QR decomposition. The information in § 4.1.1, which discusses the columns of J corresponding to the ψ variables, is also relevant to computing

J^TJ, in § 4.2. We show that one can reduce the number of nonzero rows of the Jacobian corresponding to the ψ variables without using the trick in (2.17). In § 4.1.2, we consider the columns of J corresponding to the s variables. Here we use a trick akin to (2.17) which is not relevant to § 4.2. In § 4.1.3 we consider the nonlinear terms and show that they fit into the general framework.

4.1.1. The ψ variables. The fact that the decomposition of G is separated into two decompositions helps in holding down the cost of forming the Jacobian and the space required to store its components. Recall that the elements of ψ can be indexed as ψ_{dm} and that for any m, ψ_{dm} is found only in S_d. Thus $\partial G/\partial\psi_{dm}$ looks like

$$\frac{\partial G}{\partial \psi_{dm}} = \begin{pmatrix} 0 \\ \cdot \\ 0 \\ A\dfrac{\partial S_d}{\partial \psi_{dm}} \\ 0 \\ \cdot \\ 0 \end{pmatrix}. \tag{4.8}$$

The only significant computation and the only nonzero elements in $\tilde{Q}_2(\partial G/\partial\psi_{dm})$ come from applying $\tilde{P}$ of (4.7) to

$$\begin{pmatrix} 0 \\ \cdot \\ 0 \\ T\dfrac{\partial S_d}{\partial \psi_{dm}} \\ 0 \\ \cdot \\ 0 \end{pmatrix}, \tag{4.9}$$

which contributes to at most $(N_D - 1)m_A$ rows of the Jacobian for any data set. This means that for each data set the Jacobian will have the structure

$$\begin{pmatrix} 0 \\ D \\ 0\ E \end{pmatrix}. \tag{4.10}$$

In the large problem the matrix of (4.10) is 10,400 rows by 1220 columns, but D accounts for only 1098 of the rows and E for 122 columns.

In addition, because of the zeros in (4.9), one can construct $\tilde{P}$ so that D is easy to compute and D will have special structure. Since the matrix T occurs for each variable in (4.9), one can form D of (4.10) by applying $\tilde{P}$ to the N_D block diagonal matrix

$$\tilde{T} = \begin{pmatrix} T & & & \\ & T & & \\ & & T & \\ & & & T \end{pmatrix}.$$

Let $\Theta = \tilde{P}\tilde{T}$ and partition

$$\Theta = \begin{pmatrix} \Theta_1 \\ \cdot \\ \cdot \\ \Theta_{N_D} \end{pmatrix} = \begin{pmatrix} \Theta_{11} & \Theta_{12} & \Theta_{1N_D} \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \Theta_{N_D1} & \Theta_{N_D2} & \Theta_{N_DN_D} \end{pmatrix}. \tag{4.11}$$

If $\tilde{P}$ is formed by first eliminating the last block of (4.6) using the next to the last block and working upwards, then $\Theta_{i,j} = 0$ for $i > j + 1$. Moreover, one can show that the D matrix of (4.10) will have the following block structure:

$$\begin{pmatrix} X & X & \cdot & \cdot & X & X \\ 0 & X & \cdot & \cdot & \cdot & \cdot \\ \cdot & 0 & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & X & X & X \\ 0 & \cdot & \cdot & 0 & X & X \end{pmatrix}. \tag{4.12}$$

Since most of the work in the total problem is involved in getting the QR decomposition of the Jacobian and since that work grows quadratically with the number of nonzero columns in each block handed to the nonlinear least-squares solver, using the block structure of C in (4.6) to determine $\tilde{P}$ of (4.7) to produce a matrix with the structure of (4.12) lowers the operation count of the largest part of the problem. More specifically for $N_D = 4$ one would witness a decrease of about 30 percent in the total computation time and for $N_D = 10$ nearly 60 percent.

Since about 90 percent of the nonlinear variables are in ψ, our two-stage decomposition forces most of the Jacobian matrix to be zero without using the trick in (2.17). Moreover, the major contributor to the storage requirements of this algorithm has been sharply reduced.

4.1.2. The s variables. Unfortunately the columns of the Jacobian corresponding to the s variables require more computation and have more nonzeros. Assume that there are N_s elements in the vector $\mathbf{s}$. To apply $\tilde{Q}$ to $\partial G/\partial s_k$ one would first apply Y to $\partial A/\partial s_k$ to obtain

$$Y\frac{\partial A}{\partial s_k} = \begin{pmatrix} A_{1k} \\ A_{2k} \\ A_{3k} \end{pmatrix}, \tag{4.13}$$

where A_1 has v rows, A_2 has m_A rows, and A_3 has $t - (m_A + v)$ rows. We thus have

$$\begin{pmatrix} 0 \\ \tilde{Q}_2 \end{pmatrix}\frac{\partial G}{\partial s_k} = \begin{pmatrix} 0 \\ \hline \begin{pmatrix} 0_{m_A} & \\ & I \end{pmatrix}\tilde{P}\begin{pmatrix} A_{2k}S_1 \\ \cdot \\ \cdot \\ A_{2k}S_{N_D} \end{pmatrix} \\ \hline A_{3k}S_1 \\ \cdot \\ \cdot \\ A_{3k}S_{N_D} \end{pmatrix}. \tag{4.14}$$

The bottom part of (4.14) corresponds to the E matrix in (4.10) and a trick similar to that given in § 2 can eliminate most of the nonzero rows in that matrix.

Let N represent the nonzero columns of all concatenations of all the $N_s A_{3k}$ matrices and assume N has d columns and is of rank m_N. Let L be an orthogonal matrix and Z_N be a permutation matrix such that

$$LNZ_N = \begin{pmatrix} M \\ 0 \end{pmatrix}, \tag{4.15}$$

where M is an upper trapezoidal matrix of rank m_N. Let us redefine $\tilde{Q}_2$ as

$$\begin{pmatrix} 0 \\ \tilde{Q}_2 \end{pmatrix} = \begin{pmatrix} 0 \\ I \end{pmatrix} \begin{pmatrix} I_{N_D \times v} & \tilde{P} \\ & I_{N_D} \otimes L \end{pmatrix} \tilde{Z} Z (I_{N_D} \otimes Y),$$

where $\tilde{Z}$ is a permutation matrix which would move all the zero rows of $I_{N_D} \otimes \binom{M}{0}$ to the bottom. Now the E matrix of (4.10) really has the form

$$\begin{pmatrix} \tilde{E} \\ 0 \end{pmatrix},$$

where $\tilde{E}$ accounts for $N_D \times m_N$ of the $N_D \times (t - v - m_A)$ rows of E. For our large problem, $\tilde{E}$ would account for 1220 rows of the 10,400 rows in the Jacobian for each data set.

Combining the results of §§ 4.1.1 and 4.1.2 we see that for the large problem for each data set, only 2440 rows of the original 10,400 rows of the Jacobian have nonzeros. Moreover only 122 columns of the original 1220 may be considered as dense. Because of the structure of our Jacobian, we needed an underlying nonlinear least-squares solver that accepted groups of rows of the Jacobian and did not require the whole Jacobian at once. We chose to use RN2G in PORT [3], which processes the portion of rows of the Jacobian one column at a time and skips transformations whenever there is a zero column.

4.1.3. Nonlinear function. The presence of a strictly nonlinear term in the model would seem to hurt our efforts to decrease the number of rows and columns in the Jacobian matrix. In the modal analysis problem about 5 percent of the data sets had a nonlinear term and it was first assumed that for these data sets one would pay the full price of obtaining the QR of the Jacobian. However, on second glance it was realized that the nonlinear terms had a structure similar to that given in (1.9). For those data sets with a nonlinear term, the last column block was missing and in its place one had a nonlinear term of the form

$$\mathbf{f}_i(\alpha) = \begin{pmatrix} A(\mathbf{s})\mathbf{b}_{i1}(\psi_i, \psi_1) \\ A(\mathbf{s})\mathbf{b}_{i2}(\psi_i, \psi_2) \\ \vdots \\ A(\mathbf{s})\mathbf{b}_{ii}(\psi_i) \end{pmatrix}, \tag{4.16}$$

where the matrix A is the same one that appears in (1.9) and (4.4) and the vector $\mathbf{b}_{ij}$ is some vector specific to the data set. For these data sets $\mathbf{r} = Q_{2,i}\,\mathbf{f}_i$, where $Q_{2,i}$ comes from the QR decomposition of $I_i \otimes B$ and is independent of α.

Because the variables in (4.16) separate and have the same basic structure as the last column block of (1.9), the Jacobian for the ith data set, J_i, looks similar to (4.10) except that D is $i \times m_A$. As in § 4.1.2 the number of nonzero rows in the E portion of the matrix can be drastically reduced by multiplying it by $I_i \otimes L$, where L is the matrix described in (4.15). Thus some of the machinery generated for the separable problem

could be used for the nonlinear problem and handling the nonlinear term required no extra work.

4.2. When J^TJ is required. In § 3 we mentioned that, for problems like the modal analysis problem, rather than computing the Jacobian itself explicitly it is much more efficient to compute $\Lambda = J^TJ$. Much of the machinery of § 4.1 is still relevant when Λ is computed and the Jacobian itself is not needed. The main exception is that the decomposition in (4.15) to help decrease the number of rows for the s variables is no longer needed. For the modal analysis problem the matrix $\hat{M}$ of (3.7) would be constructed as follows:

Form $\hat{M}^{(1)}_{i,j} = \hat{M}^{(1)}_{j,i} = \Theta_i^T\Theta_j$ for $i = 1, \cdots, N_D$, and $j = 1, \cdots, i$, where Θ_i is given in (4.11).
For $i = 1, 2, \cdots, N_s$

$$\text{Form } H_i = \tilde{P}\begin{pmatrix} A_{2i}S_1 \\ \cdot \\ \cdot \\ A_{2i}S_{N_D} \end{pmatrix}.$$

Form $\hat{M}^{(2)}_{i,j} = \Theta_i^T H_j$ for $i = 1, \cdots, N_D$, and $j = 1, \cdots, N_s$
Form $\hat{M}^{(3)}_{i,j} = \hat{M}^{(3)}_{j,i} = H_i^T H_j + A_{3i}^T A_{3j}$ for $i = 1, \cdots, N_s$ and $j = 1, \cdots, i$, where A_{3i} comes from (4.13).

To obtain the matrix Λ we do not have to construct J but we can proceed as in § 3 using the $\hat{M}$ matrices:

Set $\Lambda = 0$
For $i = 1, \cdots, N_D$,

If the kth column of the Jacobian corresponds to $\psi_{t,k'}$ and the jth column of the Jacobian corresponds to $\psi_{t',j'}$, then set

$$\lambda_{kj} = \lambda_{kj} + \left(\mathbf{c}_i^T \frac{\partial S_t}{\partial \psi_{t,k'}^T}\right)\left(\hat{M}^{(1)}_{t,t'}\left(\frac{\partial S_{t'}}{\partial \psi_{t',j'}}\mathbf{c}_i\right)\right).$$

If the kth column of the Jacobian corresponds to $\psi_{t,k'}$ and the jth column of the Jacobian corresponds to $s_{j'}$, then set

$$\lambda_{kj} = \lambda_{kj} + \left(\mathbf{c}_i^T \frac{\partial S_t}{\partial \psi_{t,k'}^T}\right)(\hat{M}^{(2)}_{i,j'}\mathbf{c}_i).$$

If the kth column of the Jacobian corresponds to $s_{k'}$ and the jth column corresponds to $s_{j'}$, then set

$$\lambda_{kj} = \lambda_{kj} + \mathbf{c}_i(\hat{M}^{(3)}_{k',j'}\mathbf{c}_i^T).$$

Since $\Theta_{tj} = 0$ for $t > j + 1$, the cost of computing $\hat{M}^{(1)}$ is cut by a factor of 3 if a block $\tilde{P}$ is used. Similarly, the cost of computing $\hat{M}^{(2)}$ is divided in half if block $\tilde{P}$ is used.

Table 5 gives a comparison of the operation counts of the various portions of the various algorithms given in this section with those that did not use the fact that G has special structure and that α separates into $\mathbf{s}$ and ψ. Table 6 gives a comparison of the storage requirements. The problems are exactly those given in Table 3. From Table 6 it is obvious that the idea of separating variables within each block was absolutely necessary in order to store the large problem. Using the separation of variables technique does not greatly affect the operation counts when only the Jacobian is required, but when Λ rather

TABLE 5
Comparison of methods—Number of multiplications.

	Small problem	Big problem
Golub–LeVeque with (2.17):		
Decomposition of Φ	1.7×10^7	$.2 \times 10^9$
Determining Jacobian	95×10^7	71×10^9
QR of Jacobian	191×10^7	631×10^9
Total	287×10^7	702×10^9
Using J^TJ:		
Decomposition of Φ	1.7×10^7	$.2 \times 10^9$
Determining $\hat{M}$ and gradient	117×10^7	90×10^9
Determining Λ and R_Λ	2.4×10^7	1.8×10^9
Total	121×10^7	92×10^9
Structured *G–L–K*, General $\tilde{P}$:		
Decomposition of Φ	$.6 \times 10^7$	$.3 \times 10^8$
Determining Jacobian	10×10^7	23×10^8
QR of Jacobian	105×10^7	521×10^9
Total	116×10^7	524×10^9
Structured *G–L–K*, Block $\tilde{P}$:		
Decomposition of Φ	$.6 \times 10^7$	$.3 \times 10^8$
Determining Jacobian	10×10^7	20×10^8
QR of Jacobian	84×10^7	242×10^9
Total	95×10^7	245×10^9
Structured J^TJ, Block $\tilde{P}$:		
Decomposition of Φ	$.6 \times 10^7$	$.3 \times 10^8$
Determining $\hat{M}$ and gradient	18×10^7	45×10^8
Determining Λ and R_Λ	2×10^7	14×10^8
Total	21×10^7	60×10^8

than J is used, we see a drop in the operation count of another order of magnitude with the implementation of the ideas in this section.

The algorithms in this section were implemented on a Multiflow machine. At first only the small problem was considered. As predicted the change in computation time

TABLE 6
Storage requirements of various methods.

	Small problem	Big problem
Golub–LeVeque	26×10^5	49×10^6
Normal equations	21×10^5	38×10^6
Structured *G–L–K* with General $\tilde{P}$	6.4×10^5	8.7×10^6
Structured *G–L–K* with block $\tilde{P}$	6.3×10^5	7.5×10^6
Structured normal with block $\tilde{P}$	5.8×10^5	6.5×10^6

TABLE 7
Percentage of time for structured J^TJ on multiflow.

Applying Y of (4.4)	22.4
Determining Λ from $\hat{M}$'s	17.7
Determining R_Λ from Λ	15.4
Handling nonlinear terms	12.7
Applying $\tilde{P}$ of (4.7)	11.6
Determining the $\hat{M}$'s	9.2
Determining gradients other than above	6.8
Other (including user and decompositions)	4.2

from the general $\tilde{P}$ algorithm using the Jacobian to the one with block $\tilde{P}$ was about 20 percent. The QR of the Jacobian required about 80 percent of the computation time for each iteration, slightly lower than predicted, because of the effort that the second author expended to adapt that part of the algorithm to the Multiflow machine. When the algorithm was changed from determining J to J^TJ, we witnessed a factor of 5 decrease in the computation time for the small problem, just about the amount that Table 5 suggests. Besides computing J^TJ, one had to also compute the gradient $J^T\mathbf{r}$ for the underlying minimizer. The Multiflow machine is particularly adept at the inner product calculations required by the subroutines forming the gradient. The subroutine computing Λ required a larger portion of the computation time than estimated in Table 5. The compiler directives did not work as well as they did for the subroutine computing R_Λ, primarily because the former had a bit of indirect addressing and the latter was mainly straight inner products. On the Multiflow using the structured J^TJ algorithm each iteration for the small problem requires about 40 seconds and for the big problem about 1200 seconds. The ratio is similar to that predicted by our estimates. Table 7 gives the percentage of work done by various parts of the algorithm for the big problem on the Multiflow. It should be obvious that fine tuning any one part of the algorithm will not make a major difference.

The initial estimates of the nonlinear parameters were made using a frequency-domain polyreference Prony-type algorithm [10]. In principle, the Prony-type code is exact for noiseless data of the form (4.1), but as is well known, the performance of such codes degrades significantly in the presence of noise. The nonlinear fitting algorithm we have described is intended to ameliorate this performance loss, and, as we have demonstrated, is computationally feasible on large-scale problems.

Acknowledgment. The authors would like to thank David Gay for many helpful discussions. The second author would like to acknowledge the contributions of a number of colleagues: Art, Frank, Gary, and Les.

REFERENCES

[1] J. E. DENNIS, D. M. GAY, AND R. E. WELSCH, *An adaptive nonlinear least-squares algorithm*, ACM Trans. Math. Software, 7 (1981), pp. 348–368.

[2] D. J. EWINS, *Modal Testing: Theory and Practice*, John Wiley, New York, 1984.
[3] P. A. FOX, ED., *The PORT Mathematical Subroutine Library*, AT&T Bell Laboratories, Murray Hill, NJ, 1984.
[4] G. H. GOLUB AND R. LEVEQUE, *Extensions and uses of the variable projection algorithm for solving nonlinear least-squares problems*, Computer Science Report SU 326, Stanford University, Stanford, CA, 1978.
[5] G. H. GOLUB AND V. PEREYRA, *The differentiation of pseudo-inverses and nonlinear least-squares problems whose variables separate*, SIAM J. Numer. Anal., 10 (1973), pp. 413–432.
[6] L. KAUFMAN, *A variable projection method for solving separable nonlinear least-squares problems*, BIT, 15 (1975), pp. 49–57.
[7] C. L. LAWSON AND R. J. HANSON, *Solving Least-Squares Problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974.
[8] J. J. MORE, B. S. GARBOW, AND K. E. HILLSTROM, *User Guide for MINPACK-1*, Report ANL-80-74, Argonne National Laboratory, Argonne, IL, 1980.
[9] Y.-W. SOO AND D. M. BATES, *Loosely coupled nonlinear least squares*, in Comput. Statist. Data Anal., to appear, 1992.
[10] G. S. SYLVESTER, *Structure and performance of a polyreference frequency-domain prony algorithm*, in preparation.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 90–101, January 1992

008

DISTRIBUTED AND SHARED MEMORY BLOCK ALGORITHMS FOR THE TRIANGULAR SYLVESTER EQUATION WITH sep^{-1} ESTIMATORS*

BO KÅGSTRÖM† AND PETER POROMAA†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Coarse grain message passing and shared memory algorithms for solving the quasi-triangular Sylvester equation are discussed. The basic algorithm is of block type, i.e., rich in matrix-matrix operations. The focus is on computing reliable estimates of the sep^{-1} function (a natural condition number for the Sylvester equation and the invariant subspace problem). Estimators based on the Frobenius norm and the 1-norm, respectively, are presented. Accuracy, efficiency, and reliability results are presented. The applicability of the estimators to both the shared memory and distributed memory paradigms are discussed. Some performance results of the parallel block algorithms with condition estimators are also presented. The reliability of both estimators are very good. The Frobenius norm–based estimator is much more efficient in both sequential and parallel settings (on average between four to five times). Further, it is applicable to both the standard and generalized problems.

Key words. Sylvester equation, parallel algorithms, condition number estimation

AMS(MOS) subject classifications. 65F05, 65W05

1. Introduction. In [16]–[18] and [21] we present and discuss parallel block algorithms for solving the triangular Sylvester equation

$$AX + XB = C, \tag{1.1}$$

where the unknown X is m-by-n and A, B, and C are given m-by-m, n-by-n, and m-by-n matrices, respectively, with real entries and A, B are upper triangular (possibly upper quasi-triangular[1]). This paper gives a brief survey of distributed and shared memory block algorithms and focuses on methods for computing reliable estimates of the sep^{-1} function that are based on effective triangular Sylvester solvers. Equation (1.1) has a unique solution if and only if A and $-B$ have disjoint spectra. The sep function (introduced later in § 2) measures the distance (separation) between the spectra of A and $-B$.

If A and B are upper triangular, the unknown X in (1.1) can be solved for by elementwise identification as follows:

$$x_{ij} = \frac{c_{ij} - \sum_{k=i+1}^{m} a_{ik}x_{kj} - \sum_{k=1}^{j-1} x_{ik}b_{kj}}{a_{ii} + b_{jj}}. \tag{1.2}$$

First we solve for x_{m1}, then x_{m-11} or x_{m2}, and so on diagonal by diagonal. The parallel block algorithms are all based on the *sequential block algorithm* TSYLV_B in Fig. 1.1.

Let M and N be the block sizes for A and B, respectively. Solving for the first M-by-N block of X (i.e., $X_{nb\_A,1}$) results in solving a "small" upper triangular Sylvester equation (in the following, denoted by an M-by-N subsystem). The solution of this subsystem is then used to update the remaining blocks in the first block column and the last block row of the right-hand side C with respect to A and B, respectively. Then we can solve a similar system for the next block of X in the first block column. To get the

* Received by the editors February 11, 1991; accepted for publication (in revised form) June 14, 1991.

† Institute of Information Processing, University of Umeå, S-901 87 Sweden (bokg@cs.umu.se and peterp@cs.umu.se). Financial support has been received from Swedish Board of Technical Development (STU) contracts STU-712-89-02578 and STU-726-90-02307.

[1] A quasi-triangular matrix is a block triangular matrix with possible 2-by-2 blocks on the diagonal corresponding to pairs of complex conjugate eigenvalues.

```
for j = 1 to nb_B {nb_B is no. of diagonal blocks in B}
  for i = nb_A downto 1 {nb_A is no. of diagonal blocks in A}
    {solve for X_ij block in (i, j)th subsystem}
    A_ii*X_ij + X_ij*B_jj = C_ij
    {substitute X_ij block into remaining equations}
    for k = 1 to i - 1 {update block column j of C}
      C_kj = C_kj - A_ik*X_ij
    for k = j + 1 to nb_B {update block row i of C}
      C_ik = C_ik - X_ij*B_jk
```

FIG. 1.1. *Block algorithm* TSYLV_B.

complete X we have to solve $m/M * n/N$ triangular Sylvester equations. Each, except the last one, is followed by an updating of C with respect to A, B, and the recently computed block of X. By choosing $M = 1$ and $N = n$ in TSYLV_B we get a *row-oriented* level-2 algorithm. Similarly, by choosing $N = 1$ and $M = m$ the block algorithm reduces to a *column-oriented* level-2 algorithm. These are used for solving the M-by-N subsystems.

Sequential and numerically stable algorithms for solving the general Sylvester equation are presented in [2] and [11]. The problem of transforming A and B to Schur form is not emphasized here. The solution of the triangular Sylvester equation is the second main step of the algorithm in [2]. A fine grained message passing algorithm is described in [20]. From an algorithmic point of view the triangular Sylvester equation is a natural extension of the solution of triangular systems of equations, a problem that is inherently fine grained and sequential [13]. If $m = 1$ or $n = 1$ then A or B are scalars and the Sylvester equation reduces to a triangular system. We find applications in control theory for the Sylvester equation, e.g., pole assignment for linear systems [3], [6].

The rest of the paper is outlined as follows. In § 2 the separation between two matrices is defined and its relation to the Sylvester equation is shown. Error bounds for the invariant subspace problem and the Sylvester equation, which motivate reliable estimates of the sep^{-1} function, are also reviewed. In § 3 sep^{-1} estimators based on the Frobenius norm and the 1-norm, respectively, are presented. Accuracy, efficiency, and reliability results are also presented. In § 4 this qualitative comparison is complemented by an evaluation of the applicability of the estimators to the shared memory and distributed memory MIMD paradigms. Section 5 presents some performance results of the parallel block algorithms with condition estimator. Finally, some conclusions are summarized in § 6.

1.1. Notation used in the paper. $\|A\|_2$ and $\|A\|_F$ denote the spectral norm and the Frobenius norm of a matrix A, respectively. $\sigma_{\min}(A)$ denotes the smallest singular value of A. A_{ij} denotes the (i, j) block of A and a_{ij} denotes the (i, j) element of A. $A \otimes B$ denotes the Kronecker product of the matrices A and B. vec (A) denotes an ordered stack of the columns of A.

2. Separation between two matrices and Sylvester's equation. One objective for studying the triangular Sylvester equation is its relation to the problem of computing well-conditioned invariant subspaces, which we illustrate below:

$$S^{-1}MS = \begin{bmatrix} I & -X \\ 0 & I \end{bmatrix} \begin{bmatrix} A & -C \\ 0 & -B \end{bmatrix} \begin{bmatrix} I & X \\ 0 & I \end{bmatrix} = \begin{bmatrix} A & AX+XB-C \\ 0 & -B \end{bmatrix}. \tag{2.1}$$

The block upper triangular matrix M in Schur canonical form is block diagonalized by S if and only if the (1, 2) block of $S^{-1}MS$ is zero. For a given M and a specification of the spectrum of the (1, 1) block we can order the eigenvalues of M along the block main diagonal giving A, B, and C in the Sylvester equation. The sensitivity of the invariant subspaces spanned by the two block columns of S is proportional to the inverse of $\operatorname{sep}(A, -B)$, the *separation between the matrices A and* $-B$ [22]:

$$\operatorname{sep}(A,-B)=\min_{X\neq 0}\frac{\|AX+XB\|_F}{\|X\|_F}=\|Z^{-1}\|_2^{-1}=\sigma_{\min}(Z), \tag{2.2}$$

where $Z = I_n \otimes A + B^T \otimes I_m$. Below we summarize the perturbation theory for invariant subspaces and clusters of eigenvalues that will lead to computable error bounds (for details, see [7] and [22]). Let $\mathbf{X}$ be the m-dimensional right invariant subspace corresponding to the (1, 1)-block A of M. The *spectral projector* belonging to the eigenvalues of A is defined by

$$P=\begin{bmatrix} I_m & X \\ 0 & 0 \end{bmatrix}, \qquad \|P\|_2=(1+\|X\|_2^2)^{1/2}, \tag{2.3}$$

where X satisfies $AX + XB = C$ in (2.1). Now, let $\mathbf{X}'$ be the right invariant subspace of $M + \Delta M$, where ΔM is an mn-by-mn general perturbation of M. Then, if $\|\Delta M\|_F$ satisfies

$$\operatorname{sep}^{-1}(A,-B)\|P\|_2\|\Delta M\|_F<\tfrac{1}{4}, \tag{2.4}$$

the m eigenvalues of the (1, 1)-block of $M + \Delta M$ will remain disjoint from the eigenvalues outside the cluster [22] and the *maximum angle between the m-dimensional invariant subspaces* $\mathbf{X}$ and $\mathbf{X}'$ can be bounded as [7]

$$\Theta_{\max}(\mathbf{X},\mathbf{X}')\leqq\arctan\left(\operatorname{sep}^{-1}(A,-B)\left[\frac{2\|\Delta M\|_F}{1-4\operatorname{sep}^{-1}(A,-B)\|P\|_2\|\Delta M\|_F}\right]\right). \tag{2.5}$$

Moreover, the absolute value of the difference between the average of the eigenvalues of the unperturbed and perturbed clusters can be bounded as [7]

$$|\lambda_{\text{avg}}-\lambda'_{\text{avg}}|\leqq 2\|E\|_F\|P\|_2. \tag{2.6}$$

Note that $\operatorname{sep}^{-1}(A, -B)$ is also the natural condition number for the Sylvester equation itself [11]. Let ΔA, ΔB, and ΔC be general perturbations of A, B, and C, respectively. Now, if the perturbations satisfy

$$\operatorname{sep}^{-1}(A,-B)(\|\Delta A\|_F+\|\Delta B\|_F)\leqq\tfrac{1}{2}, \tag{2.7}$$

then the relative error in the solution of the perturbed Sylvester equation can be bounded as [11]:

$$\frac{\|\Delta X\|_F}{\|X\|_F}\leqq 4\operatorname{sep}^{-1}(A,-B)(\|\Delta A\|_F+\|\Delta B\|_F). \tag{2.8}$$

When backward stable algorithms are used to solve the invariant subspace problem or the Sylvester equation, the norm of the perturbations is bounded by $O(macheps)$ times the norm of the input matrix (*macheps* denotes the relative machine precision). To conclude, $\|P\|_2$ and $\operatorname{sep}^{-1}(A, -B)$ are the crucial quantities in the error bounds.

In (2.4)–(2.6) we may replace $\|P_2\|$ by $(1 + \|X\|_F^2)^{1/2}$ to get easier-to-compute but somewhat weaker bounds [7], [22]. However, the smallest singular value of $Z = I_n \otimes A + B^T \otimes I_m$ is costly to compute exactly (it requires $O(m^3n^3)$ flops). The next section presents methods to compute an estimate of $\|Z^{-1}\|_2 = 1/\sigma_{\min}(Z)$ to much lower cost.

3. sep^{-1} estimators. It is possible to compute lower bounds of $\text{sep}^{-1}(A, -B)$ by solving the triangular Sylvester equation. Here we describe two estimators, one based on the Frobenius norm (see [5], [19], [18], [23]) and one based on estimating the 1-norm of a matrix (see [12], [14], [15]).

3.1. A Frobenius norm–based estimator. By assuming that A and $-B$ have disjoint spectra (i.e., sep $(A, -B)$ is nonzero), it is straightforward to derive the following bound from $AX + XB = C$:

$$\frac{\|X\|_F}{\|C\|_F} \leqq \text{sep}^{-1}(A, -B). \tag{3.1}$$

By choosing the right-hand C such that the left-hand side of (3.1) is as large as possible, we get a lower bound for $\text{sep}^{-1}(A, -B)$. We have implemented both distributed and shared memory block algorithms for one of the estimators proposed in [19] (see § 4 below). The cost of computing the condition estimator is $O(m^2n + mn^2)$ flops, i.e., the same cost as solving a triangular Sylvester equation.

First we note that TSYLV_B can be written in a more compact form, showing that (1.1) is equivalent to

$$A_{ii}X_{ij} + X_{ij}B_{jj} = C_{ij} - \left(\sum_{k=i+1}^{nb\_A} A_{ik}X_{kj} + \sum_{k=1}^{j-1} X_{ik}B_{kj} \right) \tag{3.2}$$

for $j = 1, \cdots, nb\_B$, $i = 1, \cdots, nb\_A$. For each pair (i, j) the subsystem (3.2) is a system of the form $Z_{ij}x = y\,(=h - f)$, where $Z_{ij} = I \otimes A_{ii} + B_{jj}^T \otimes I$ and h and f are vector notation for C_{ij} and the block matrix sum, respectively (e.g., $h = \text{vec}(C_{ij})$). Since each M-by-N subsystem (3.2) solves a Sylvester equation with diagonal blocks of size 1-by-1 or 2-by-2 all these sub-subsystems can be written in a compact form similar to (3.2). Dropping subscripts, we denote a sub-subsystem by $Zx = y\,(=h - f)$ and proceed as follows: Each element of h is chosen to $+1$ or -1. The sign of h is chosen with a look-ahead procedure similar to the Linpack estimator [9], except that our right-hand sides are loaded with substitutions from earlier computations (f above). Given a LU decomposition of Z we use the algorithm BSOLVE in [19] when solving for y (see Fig. 3.1).

Finally, we have to make a backward substitution with U to get $x = U^{-1}(L^{-1}y)$. We get a contribution to the estimator from each sub-subsystem and the local choice of

---

```
y = − f
for k = 1 to N{N = dim (Z), 1, 2 or 4}
  {add ±1 to the kth element of y}
  y_{k+} = 1 + y_k, y_{k−} = −1 + y_k
  S^+ = |y_{k+}| + Σ_{k+1}^N |y_j − l_{jk} y_{k+}|
  S^− = |y_{k−}| + Σ_{k+1}^N |y_j − l_{jk} y_{k−}|
  {compute the kth element of L^{-1}y}
  if {S^+ ≧ S^−) then y_k = y_{k+} else y_k = y_{k−}
  {compute the remaining right-hand sides}
  for j = k + 1 to N
    y_j = y_j − l_{jk} y_k
```

FIG. 3.1. *Algorithm* BSOLVE.

right-hand sides yields a global right-hand side C where all elements are ± 1. Thus our *lower bound based on the Frobenius norm* is

$$s_F := \frac{\|X\|_F}{\|C\|_F} = \frac{\|X\|_F}{(mn)^{1/2}}. \tag{3.3}$$

We gain some performance by having separate code segments for the upper triangular and quasi-triangular cases, respectively.

3.2. A 1-norm–based estimator. In [12] a method for estimating $\|A^{-1}\|_1$ is presented that only requires $A^{-1}x$ and $A^{-T}x$ (A^{-1} or A^{-T} are not needed). The algorithm is improved in [14] where one code is described that can serve as a condition estimator for all linear equation solvers. We can apply this to $AX + XB = C$ by remembering that solving (1.1) is equivalent to solving $Zx = c$ where $Z = I_n \otimes A + B^T \otimes I_m$, $x = \text{vec}(X)$, and $c = \text{vec}(C)$ and for a given c we have to supply $Z^{-1}c$ or $Z^{-T}c$. We can use the same algorithm (code) for $Z^{-T}c$ by simply using C^T as the right-hand side and exchanging A and B in the call to the triangular Sylvester solver. Finally, we have to transpose the solution to get $Z^{-T}c$. Our Fortran-like algorithm for computing the lower bound `est` of $\|Z^{-1}\|_1$ is outlined in Fig. 3.2. Now by using the norm inequality $\|F\|_1/(n)^{1/2} \leqq \|F\|_2$, for any matrix F of size n-by-n, we convert the lower bound `est` of $\|Z^{-1}\|_1$ to a lower bound of $\|Z^{-1}\|_2$. We obtain the following 1-*norm–based lower bound* of $\|Z^{-1}\|_2$:

$$s_1 := \frac{\texttt{est}}{(mn)^{1/2}}. \tag{3.4}$$

The cost for computing (3.4) is about `ITER` times the cost of one Sylvester solve. The 1-norm–based estimator has so far only been implemented for the shared memory block algorithm (see § 4). A similar approach is proposed to be incorporated in LAPACK [1], [4].

3.3. Accuracy, efficiency, and reliability results. In this section we present some results that show and compare the efficiency, accuracy, and reliability of the two estimators. The tables show the exact value of $\text{sep}^{-1}(A, -B)$, together with the products $p_1 = s_1\sigma_{\min}(Z)$ and $p_F = s_F\sigma_{\min}(Z)$. We also show the product $q_1 = \texttt{est}/\|Z^{-1}\|_1$ where `est` is the lower bound of $\|Z^{-1}\|_1$ in (3.4). `ITER` is the number of Sylvester solves needed to compute s_1. T_1/T_F shows the ratio between the execution times for computing s_1 and

```
      kase = 0; ITER = 0; mn = m*n
      compute a new estimate
10    call donest (mn, v, c, isgn, est, kase)
      if (kase .ne. 0) then
         ITER = ITER + 1
         if (kase .eq. 1) then
c           compute Z^-1 c (solve AX + XB = C)
         else (kase .eq. 2)
c           compute Z^-T c (solve BX^T + X^T A = C^T)
         endif
         goto 10
      endif
```

FIG. 3.2. *Algorithm for* $\|Z^{-1}\|_1$ *estimator.*

TABLE 3.1
Accuracy and reliability results for examples 1, 2, *and* 3.

Ex.	m	n	p_1	q_1	ITER	T_1/T_F	p_F	$\sigma_{\min}^{-1}(Z)$
1	1	1	1.0	1.0	1	0.7	1.0	0.5
	2	2	0.298	0.45	5	6.6	0.907	0.699
	4	4	0.226	0.65	5	4.3	0.630	0.701
	8	8	0.693e − 1	0.39	5	4.9	0.410	0.701
	10	10	0.451e − 1	0.32	5	4.8	0.351	0.701
	32	32	0.426e − 2	0.10	5	4.7	0.144	0.701
2	10	6	0.130e − 1	0.14	7	6.8	0.510e − 1	0.130e + 6
	10	6	0.103e − 1	0.25e − 1	5	5.2	0.152	0.746e + 5
	12	12	0.847e − 1	0.63	11	10.5	0.190	0.538e + 5
	12	12	0.661e − 1	0.84e − 1	5	5.1	0.132	0.208e + 9
	32	12	0.537e − 2	0.35e − 1	5	5.1	0.631e − 1	0.109e + 6
	32	12	0.461e − 3	0.47e − 2	5	5.1	0.559e − 1	0.181e + 9
	64	2	0.496e − 1	0.45	5	5.5	0.172	0.565e + 5
3	10	6	0.142e − 1	0.10	9	7.2	0.246	0.758e + 3
	10	6	0.979e − 1	0.36	9	7.3	0.317	0.341e + 2
	12	12	0.179e − 1	0.11	5	4.0	0.145	0.197e + 4
	12	12	0.285	0.81	5	4.1	0.156	0.287e + 5
	32	12	0.950e − 5	0.21e − 3	9	7.3	0.104	0.249e + 8
	32	12	0.694e − 2	0.37e − 1	5	4.0	0.102	0.221e + 3
	64	2	0.195e − 1	0.96e − 1	5	4.2	0.170	0.303e + 3

s_F, respectively, and measures the relative efficiencies of the two estimators. The reliability of the estimators is proved if $0 < p_1, p_F \leqq 1$ and their accuracy is determined by how close the products are to 1 (p_1 or p_F equal 1 when we have equality in the lower bounds (3.3)–(3.4)). The product q_1 (which should also satisfy $0 < q_1 \leqq 1$) is displayed in order to make the qualitative comparison between the estimators more fair. For example, a small value of p_1 and a value of q_1 close to 1 show that the lower bound of $\|Z^{-1}\|_1$ is accurate while the 1-norm–based lower bound of $\|Z^{-1}\|_2$ is poor. The different qualitative behaviour illustrates cases when the inequality $\|Z^{-1}\|_1/(mn)^{1/2} \leqq \|Z^{-1}\|_2$ is weak.

In Table 3.1, some of the results for the following three examples are shown. Example 1 shows $a_{ij} = j$, $b_{ij} = i$. Example 2 shows upper triangular matrices A and B with the nonzero elements chosen randomly in $[-0.5, 0.5]$. Example 3 shows strict quasi-triangular matrices A and B with the nonzero elements chosen randomly in $[-0.5, 0.5]$. Table 3.2

TABLE 3.2
Accuracy and reliability results for example 4.

m	n	α	p_1	q_1	ITER	T_1/T_F	p_F	$\sigma_{\min}^{-1}(Z)$
4	4	0.5	0.383	1.0	4	3.0	0.408	0.291e + 4
10	10	0.5	0.155	1.0	4	3.8	0.166	0.29e + 11
6	3	0.25	0.297	1.0	4	3.9	0.302	0.143e + 7
6	4	0.125	0.230	1.0	4	4.1	0.231	0.76e + 10
6	6	0.0625	0.192	1.0	4	4.0	0.192	0.49e + 16
16	16	0.6	0.105	1.0	4	3.9	0.115	0.14e + 16
32	32	1.0	3.5e − 1	1.0	7	7.5	9.2e − 1	0.82e + 18
32	32	0.99	2.7e + 1	1.0	7	6.9	0.530	0.39e + 17
32	32	0.95	3.0e + 2	1.0	4	4.0	0.385	0.44e + 17

shows some of the results for example 4: A and B are upper triangular [24]: $A = J_m(1, -1)$, $B = J_n(-1 + \alpha, 1)$ where $J_n(d, s)$ denotes a Jordan block of dimension n with d and s as diagonal and superdiagonal elements, respectively, and α real > 0. Example 4 is an ill-conditioned problem with multiple eigenvalues (A and $-B$ have the eigenvalues 1 and $1 - \alpha$, respectively). The uniprocessor results in Tables 3.1 and 3.2 are computed on an ALLIANT FX/2816 system.

For the examples below the horizontal line in Table 3.2 we have computed the Frobenius norm-based estimator as $\|X\|_F/\|C\|_F$. The reason is that $\|X\|_F/(mn)^{1/2}$ overestimates the true value of $\text{sep}^{-1}(A, -B)$. This can only happen when we have a very ill conditioned problem ($\sigma_{\min}(Z)$ is close to zero) giving solutions of some subsystems with very large components and when we get cancellation of terms in the updating and solving phase of the system related to the sep^{-1} estimator. We also see that the 1-norm-based estimator overestimates, too. Since the condition number sep^{-1} nearly equals 1/*macheps*, the "true" values of sep^{-1} and $\|Z^{-1}\|_1$ are likely to be inaccurate. Note that in all these cases the inverse of both estimators is zero to machine precision and signals the extreme ill-conditioning correctly. Of the 20 results presented in Table 3.1, the Frobenius norm-based estimator gave somewhat better results in 19 cases ($p_F \geqq p_1$). In 14 of 20 results, $p_F \geqq q_1$. Of the first seven results presented in Table 3.2, $p_F \geqq p_1$ in all cases, while $p_F \leqq q_1$.

Accuracy and reliability results for large problems are difficult to compute. One problem is that it is very expensive to compute the exact value of the smallest singular value of $Z = I_n \otimes A + B^T \otimes I_m$. For example, if $m = n = 1024$, then Z requires 8796 Gbytes of memory to store and about 10^{18} flops to get the exact value of $\text{sep}^{-1}(A, -B)$. Therefore, we have to rely on examples where we know or can "easily" estimate the true value of $\text{sep}^{-1}(A, -B)$. In Table 3.3 we show some results for example 5. Let $a_{ii} = b_{ii} = 1.0$ for all i, and all nonzero $a_{ij} = b_{ij} = 1.0/(mn)$ for $i \neq j$. Then $\sigma_{\min}(Z)$ tends to 2 as m and n tend to infinity. A and B tend to the identity matrix I at the same time. The uniprocessor results in Table 3.3 are computed on an IBM 3090 VF/600J system. For this problem, up to as large as we could handle ($m = n = 1024$), the Frobenius norm-based estimate of $\text{sep}^{-1}(A, -B)$ is exact to two decimals precision ($p_F = 1$). The 1-norm-based estimate of $\text{sep}^{-1}(A, -B)$ differs approximately with a factor $(mn)^{1/2}$ from the predicted value, which is due to the fact that $\|Z^{-1}\|_1 \approx \|Z^{-1}\|_2$ for this example. This means that $p_F \approx q_1$ and that $\|Z^{-1}\|_1$ is accurately estimated, but the accuracy decreases as a function of m and n when we estimate $\|Z^{-1}\|_2$ with s_1 (3.4), which is the quantity required by the error bounds in § 3.

Note that since both estimators are based on heuristics, there is no guarantee that they will always compute a good estimate of $\text{sep}^{-1}(A, -B)$. See also the counterexamples

TABLE 3.3
Accuracy and reliability results for example 5.

Ex.	m	n	p_1	ITER	T_1/T_F	p_F	p_F/p_1	$\sigma_{\min}^{-1}(Z)$
5	1	1	0.100e + 1	1	1.0	0.100e + 1	1.0	0.500
	8	8	0.137	5	2.0	0.100e + 1	7.3	0.504
	32	32	0.322e − 1	5	3.1	0.100e + 1	31.1	0.500
	64	64	0.159e − 1	5	4.8	0.100e + 1	62.9	0.500(es)
	128	128	0.787e − 2	5	5.0	0.100e + 1	127.1	–"–
	256	256	0.392e − 2	5	4.9	0.100e + 1	255.1	–"–
	512	512	0.195e − 2	5	5.0	0.100e + 1	512.8	–"–
	1024	1024	0.977e − 3	5	4.8	0.100e + 1	1023.5	–"–

in [5] and [19] and the discussion in [14] and [15]. Recently, the preliminary version of the code for estimating $\text{sep}^{-1}(A, -B)$ in LAPACK [1] became available to us. We have repeated all tests presented here with that code and received similar results as with our 1-norm–based estimator.

4. Shared memory versus distributed memory block algorithms. In order to efficiently compute bounds (discussed in § 2) for large problems on high performance computers, the $\text{sep}^{-1}(A, -B)$ estimators have been implemented in our shared memory and distributed memory block algorithms for solving the triangular Sylvester equation. Besides the qualitative comparison of the 1-norm–based and Frobenius norm–based estimators in § 3, we evaluate the applicability of the estimators with respect to the two most common paradigms of commercially available multiprocessor systems: distributed memory multicomputers (DMM) and shared memory multiprocessors (SMM). As background, we shortly review the characteristics of our parallel block algorithms.

The level-3 algorithm TSYLV_B (see Fig. 1.1) has an *inherent parallelism* at the block level similar to the elementwise algorithm (1.2): (i) X-blocks on the same diagonal can be computed in parallel, (ii) the updating of blocks in C can be done in parallel (for details, see [16] and [17]). The parallelization of TSYLV_B differs somewhat for the shared memory and distributed memory target MIMD machines (ALLIANT FX/8 and Intel iPSC/2 hypercube, respectively). The implementation languages used are C for iPSC/2 (dynamic allocation of arrays makes it easy to handle messages and storage allocation on the local processors) and Fortran 77 for ALLIANT FX/8 (optimized level-3 BLAS [8] exists). Since two-dimensional arrays are stored rowwise in C and columnwise in Fortran, we choose the orientation of the block algorithms accordingly. Since most of the work are matrix-matrix operations (updatings of C) or matrix-vector operations (solve subsystems), processors are preferably equipped with vector facilities (i.e., of SIMD type).

In the *shared memory block algorithm* the two updating loops in TSYLV_B are merged into one loop, which then is parallelized. Subsystems are solved sequentially and block columnwise. Inside a subsystem we can utilize vectorization, vector concurrency, or COVI-mode (concurrent outer vector inner; for details, see [16]). If we both compute a lower bound for $\text{sep}^{-1}(A, -B)$ and solve the Sylvester equation, we can concurrently solve the M-by-N subsystems belonging to the Sylvester equation and the $\text{sep}^{-1}(A, -B)$ estimator, respectively. Separate codes are handling the triangular and the quasi-triangular cases. The code handling the quasi-triangular case can with some loss of efficiency be used on triangular problems.

In the *distributed memory block algorithms* the two outermost loops in TSYLV_B are exchanged and then the outermost loop is parallelized. Subsystems are solved diagonalwise (starting at the (nb_A, 1)-block of X) and possibly concurrently. No other parallelism, except possibly vectorization, is performed inside a subsystem. The solution of subsystems can be seen as a wavefront starting at the southwest corner of X and propagating to the northeast corner. Block row and block column updates are performed concurrently over the two updating loops. Since block rows are wrap-mapped onto the processors, a specific block row will be updated by a single processor, while a block column will be updated by possibly all processors (see [17] for details).

As mentioned above, data are distributed by row-block–wrap-mapping, i.e., contiguous blocks of rows are assigned to the nodes in the usual wraparound fashion. The distributed memory algorithms are: DRB_AC, DRB_ABC, and a ring variant of DRB_AC. The first two algorithms are of *fan-out type*, i.e., messages are broadcasted to all processors by utilizing the direct connect module of the iPSC/2 (cut-through rout-

ing). In DRB_AC, matrices A and C are distributed by row-block–wrap-mapping and all nodes store B. However, if there is not enough memory on the local processors, we may also distribute B over the processors, which is done in DRB_ABC. The arrival of X-blocks in the fan-out algorithms is to some extent nondeterministic and requires some extra logic and checking. In the ring variant of DRB_AC the arrival of X-blocks is deterministic, which makes the algorithm easier to understand and its behaviour predictable. Moreover, it is possible to implement a ring-oriented distributed algorithm on a shared memory MIMD machine. Performance results show that there is no loss of efficiency over 5 percent for the ring variant of DRB_AC compared to the fan-out version, even though the balancing of communication is worse for the ring variant [21].

The scalar (SX) and vector (VX) versions of our distributed implementations are textually similar and handle both the triangular and the quasi-triangular cases. The only difference is what subroutine library we use and what memory space for arrays is allocated. Theoretical performance models of the distributed algorithms show good agreement between predicted and real performance [17], [21] and are used to predict near-to-optimal block sizes M and N. We have also investigated different strategies for *variable blocking*, but our experience is that the *static choice of block sizes* is preferable [21].

All parallel block algorithms have the option to compute a lower bound of $\text{sep}^{-1}(A, -B)$. The Frobenius norm–based estimator s_F is straightforwardly and efficiently implemented in both the shared memory and distributed block algorithms. For example, the node program of the distributed algorithms is extended with algorithm BSOLVE (see Fig. 3.1) for computing local contributions to the estimator. The data flow and parallelism are similar for s_F and the parallel triangular Sylvester solvers and s_F can therefore be computed almost as efficiently as solving one triangular Sylvester equation (see the results in § 5).

This is not the case for the 1-norm–based estimator. First of all, computing s_1 involves the solution of several (ITER) triangular Sylvester equations and since all processors must have access to the vector c after each Sylvester solve (see Fig. 3.2), this imposes a synchronization bottleneck for any parallel algorithm that implements s_1. Further, some of the iterations correspond to solving a transposed Sylvester equation, which has different access patterns of the matrices A, B, and C, respectively. This is not a big problem for SMM but it is for DMM. Let us illustrate the intricacies for the ring variant of DRB_AC. Here B is stored on each node and A and C are row-block–wrap-mapped, as described earlier. In order to obtain a similar distributed block algorithm for the transposed problem, A^T must also be distributed similarly, i.e., we also require A to be column-block–wrap-mapped. This requires more local memory and extra communication overhead proportional to the message traffic involved by A in the ring variant of DRB_AC (for details, see [17] and [19]).

5. Performance results. For performance results of the distributed memory and shared memory block algorithms we refer to [17]–[19] and [21]. Here we only give a brief *performance comparison* between the shared memory implementations (ALLIANT FX/8) and the distributed memory implementations (iPSC/2 hypercube). Table 5.1 displays execution times in seconds for the scalar (SX) and vector (VX) versions of the distributed algorithm DRB_AC (using 64 and 16 processors, respectively) and for the shared memory implementation of TSYLV_B (using eight processors). When running the SX and VX versions of DRB_AC on one processor, we have seen that the VX version is at most a factor 3.125 times faster. The results in Table 5.1 show that there is a relative gain in using vector boards on the iPSC/2 hypercube.

TABLE 5.1
Timing results for distributed and shared memory block algorithms.

$m = n$	DRB_AC(SX) 64P	DRB_AC(VX) 16P	TSYLV_B 8P
256	4.57	4.54	6.9
256*	5.96	5.75	16.32
512	9.02	10.38	21.37
1024†	249.7	—	102.8

* This is a strict quasi-triangular problem, i.e., the block diagonal of A and B consist of $m/2$ and $n/2$ 2-by-2 blocks, respectively.

† DRB_ABC is used since DRB_AC could not solve such a large problem. Each node stores one row block of B. Extra communication cost: each node needs all p row blocks of Bnb_A/p times.

For an increasing number of processors, there is a *relative gain in both computing a Frobenius norm–based bound for* $\text{sep}^{-1}(A, -B)$ *and solving the Sylvester equation* at the same time (illustrated in Table 5.2). The top line shows an example run on the ALLIANT FX/8 with eight processors. $m = n = 256$ and each third and fourth diagonal block in A and B is 2-by-2. The results beyond the horizontal line correspond to DRB_AC (SX) run on the iPSC/2 with $m = n = 256$, A and B strict upper triangular, and $p = 8$, 16, and 64. All times T are in seconds. Note the relatively bad performance of the parallel shared memory version of TSYLV_B in the "condition estimator" case. The explanation is that the code for solving the upper triangular Sylvester equation uses the optimized level-2 BLAS routine DTRSV (triangular solve), while the Frobenius norm–based estimator uses BSOLVE. This effect almost disappears when we both perform a Sylvester solve and compute a sep^{-1} estimator.

6. Conclusions. Block algorithms are appropriate for solving the triangular Sylvester equation (1.1) on distributed memory multicomputers (DMM) as well as on shared memory multiprocessors (SMM) with memory hierarchy. The choice of block sizes is very much architecture-dependent. The communication/computation cost ratio t_C/t_A is an important characteristic that determines the overall efficiency of distributed algorithms. Here t_C denotes the cost of communication between adjacent nodes in a message passing environment (350 μs (microseconds) for the iPSC/2) and t_A denotes the cost for one flop (1.54 μs for the iPSC/2 with SX boards). A DMM with large ratio (227 for the iPSC/2) requires a more coarse grain block algorithm. Theoretical performance models for the distributed block algorithms predict near-to-optimal block sizes [17], [21]. We have seldom seen more than 10 percent better performance on $p = 32$ and 64 processors.

TABLE 5.2
Timing comparisons for Sylvester solver and sep^{-1} *estimator.*

$m = n$	p	$T(AX + XB = C)$	$T(\text{sep}^{-1}(A, -B))$	T (both)	gain in %
256	8	15.61	22.77	24.16	37.1
256	8	23.61	23.71	47.59	−0.6
256	16	12.51	12.56	24.91	0.6
256	64	6.85	6.86	10.57	22.9

The communication cost for a corresponding SMM ratio includes the cost of transferring data between different levels in the memory hierarchy (e.g., a global main memory, a global cache memory, and vector registers as for the ALLIANT FX/8) [10]. The choice of block size is now determined so that data can be reused as much as possible at the top of the hierarchy (fit the size of cache memory and match the length of the vector registers).

The distributed memory and shared memory block algorithms have successfully been used to compute reliable estimates of $\text{sep}^{-1}(A, -B)$. Two estimators, one based on the Frobenius norm (s_F) and one based on the 1-norm (s_1), have been described. By replacing $\text{sep}^{-1}(A, -B)$ by one of the estimators s_F and s_1, respectively, we obtain computable error bounds for the invariant subspace problem and for the Sylvester equation itself (see also [1]). Uniprocessor results show that the 1-norm–based estimator is, on average, between four and five times as expensive as the Frobenius norm–based estimator. The estimators give very accurate results, on average, within a factor 5 and seldom worse than within a factor 100 (1000 for large problems) of the true value of $\text{sep}^{-1}(A, -B)$ for the problems we have studied. In the evaluation of the estimators with respect to the DMM and SMM parallel paradigms, we conclude that s_F can be efficiently implemented in our parallel triangular solvers, while, e.g., a distributed implementation of s_1 both requires extra memory and induces extra communication overhead. In summary, the reliability of both estimators is very good but, overall, the Frobenius norm–based estimator is to be preferred since it has a much better efficiency, both in sequential and parallel settings.

As the sep^{-1} function is the natural condition number for the invariant subspace problem $M - \lambda I$, the dif^{-1} function is the corresponding condition number for the deflating subspace problem of $M - \lambda N$ [22]. Now block diagonalizing the pair (M, N) as in (3.1) corresponds to solving for R and L in the generalized Sylvester equation $(AR + LB, DR + LE) = (C, F)$. Generalized Schur methods for solving the generalized Sylvester equation with Frobenius norm–based dif^{-1} estimators are presented in [19]. The distributed and shared memory block algorithms presented here can straightforwardly be extended to the generalized Sylvester equation. Note that $Z^{-T}c$ (see Fig. 3.2), where now Z is a matrix representation of the generalized triangular Sylvester equation, does not generally correspond to solving a transposed system as in the standard case. So it is not clear that a generalized triangular Sylvester solver can be used to compute $Z^{-T}c$. The applicability of the Frobenius norm–based estimator to both the standard and generalized problems makes it even more attractive.

Acknowledgments. The authors are grateful for suggestions from the referees that made the efficiency and accuracy comparison between the estimators more complete.

REFERENCES

[1] Z. BAI, J. W. DEMMEL, AND A. MCKENNEY, *On the conditioning of the nonsymmetric eigenproblem: Theory and software*, LAPACK Working Note 13, CS-89-96, Computer Science Department, University of Tennessee, Knoxville, TN, October 1989.

[2] R. BARTELS AND G. W. STEWART, *Solution of the matrix equation $AX + XB = C$*, Comm. ACM, 15 (1972), pp. 820–826.

[3] S. P. BHATTACHARYYA AND E. DE SOUZA, *Pole assignment via Sylvester's equation*, Systems Control Lett., 1 (1982), pp. 261–263.

[4] C. H. BISHOF, J. W. DEMMEL, J. J. DONGARRA, J. DU CROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY, AND D. SORENSEN, *LAPACK provisional contents*, LAPACK Working Note 5, Math-

ematics and Computer Science Report ANL-88-38, Argonne National Laboratory, Argonne, IL, 1988.

[5] R. BYERS, *A LINPACK-style condition estimator for the equation* $AX - XB^T = C$, IEEE Trans. Automat. Control, AC-29 (1984), pp. 926–928.

[6] B. N. DATTA, *Parallel and large-scale matrix computations in control: Some ideas*, Linear Algebra Appl., 121 (1989), pp. 243–264.

[7] J. W. DEMMEL, *Computing stable eigendecompositions of matrices*, Linear Algebra Appl., 79 (1986), pp. 163–193.

[8] J. J. DONGARRA, J. DU CROZ, I. DUFF, AND S. HAMMARLING, *A proposal of level* 3 *basic linear algebra subprograms*, ACM Trans. Math. Software, 16 (1990), pp. 1–17, 18–32.

[9] J. J. DONGARRA, C. MOLER, J. BUNCH, AND G. STEWART, *Linpack User's Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[10] K. A. GALLIVAN, R. J. PLEMMONS, AND A. H. SAMEH, *Parallel algorithms for dense linear algebra computations*, SIAM Rev., 32 (1990), pp. 54–135.

[11] G. H. GOLUB, S. NASH, AND C. F. VAN LOAN, *A Hessenberg Schur method for the problem* $AX + XB = C$, IEEE Trans. Automat. Control, AC-24 (1979), pp. 909–913.

[12] W. W. HAGER, *Condition estimates*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 311–316.

[13] M. HEATH AND C. ROMINE, *Parallel solution of triangular systems on distributed memory computers*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 558–588.

[14] N. J. HIGHAM, *Fortran codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation*, ACM Trans. Math. Software, 14 (1988), pp. 381–396.

[15] ———, *Experience with a matrix norm estimator*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 804–809.

[16] B. KÅGSTRÖM, L. NYSTRÖM, AND P. POROMAA, *Parallel shared memory algorithms for solving the triangular Sylvester equation*, in Vector and Parallel Computing, J. Dongarra, I. Duff, P. Gaffney, and S. McKee, eds., Ellis Horwood, Chichester, U.K., 1989, pp. 229–240.

[17] B. KÅGSTRÖM AND P. POROMAA, *Distributed block algorithms for the triangular Sylvester equation with condition estimator*, in Hypercube and Distributed Computers, F. Andre and J. P. Verjus, eds., North-Holland, Amsterdam, 1989, pp. 233–248.

[18] ———, *Distributed and shared memory block algorithms for the triangular Sylvester equation with* sep^{-1} *estimators*, presented at the Fourth SIAM Conference on Parallel Processing for Scientific Computing, Chicago, IL, Dec. 11–13, 1989.

[19] B. KÅGSTRÖM AND L. WESTIN, *Generalized Schur methods with condition estimators for solving the generalized Sylvester equation*, IEEE Trans. Automat. Control, 34 (1989), pp. 745–751.

[20] D. O'LEARY AND G. W. STEWART, *Data-flow algorithms for parallel matrix computations*, Comm. ACM, 28 (1985), pp. 840–853.

[21] P. POROMAA, *Design and implementation aspects of parallel algorithms for the triangular Sylvester equation with condition estimators*, Report UMINF-188.90, Institute of Information Processing, University of Umeå, S-901 87 Umeå, Sweden, November 1990.

[22] G. W. STEWART, *Error and perturbation bounds for subspaces associated with certain eigenvalue problems*, SIAM Rev., 15 (1973), pp. 727–764.

[23] C. F. VAN LOAN, *On estimating the condition of eigenvalues and eigenvectors*, Linear Algebra Appl., 88/89 (1987), pp. 715–732.

[24] J. M. VARAH, *On the separation of two matrices*, SIAM J. Numer. Anal., 16 (1979), pp. 216–222.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 102–120, January 1992

009

BEST CYCLIC REPARTITIONING FOR OPTIMAL SUCCESSIVE OVERRELAXATION CONVERGENCE*

SOFOKLIS GALANIS† AND APOSTOLOS HADJIDIMOS‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. In this paper, the successive overrelaxation (SOR) method for the solution of a linear system whose matrix coefficient A is block p-cyclic consistently ordered is discussed. In recent works, many researchers considered some "natural" assumptions on the spectrum $\sigma(J_p)$ of the block Jacobi matrix J_p associated with A and answered the following question: What is the repartitioning of A into a block q-cyclic form ($2 \leqq q \leqq p$) which yields the best optimal SOR method for the solution of the given system? In this paper, the same question is answered in the most general case considered so far, that is, under the assumption $\sigma(J_p^p) \subset [-\alpha^p, \beta^p] \subset \mathbb{R} \setminus \{[1, \infty)\}$, $\alpha, \beta \geqq 0$. It is also shown that the results in all previous works are recovered as particular subcases of the case considered here.

Key words. iterative solution of linear systems, successive overrelaxation (SOR) iterative method, block p-cyclic consistently ordered matrices, optimal relaxation factor

AMS(MOS) subject classification. 65F10

C.R. subject classification. 5.14

1. Introduction and preliminaries.

For the solution of the nonsingular linear system

$$Ax = (D - L - U)x = b, \tag{1.1}$$

where $A \in \mathbb{C}^{n,n}$ is partitioned into a $p \times p$ block form, $D := \operatorname{diag}(A_{11}, A_{22}, \cdots, A_{pp})$, with diagonal blocks square and nonsingular, and L and U strictly lower and strictly upper triangular matrices, respectively, we consider the block successive overrelaxation (SOR) method

$$x^{(m+1)} = \mathcal{L}_\omega x^{(m)} + \omega(D - \omega L)^{-1} b, \qquad m = 0, 1, 2, \cdots, \tag{1.2}$$

$$\mathcal{L}_\omega := (D - \omega L)^{-1}[(1 - \omega)D + \omega U], \tag{1.3}$$

with $x^{(0)}$ arbitrary and the relaxation factor $\omega \in (0, 2)$. As is known (see, e.g., Varga [17] or Young [20]), a measure of the asymptotic rate of convergence of (1.2)–(1.3) is $\rho(\mathcal{L}_\omega)$, where $\rho(\cdot)$ denotes spectral radius. For convergence, $\rho(\mathcal{L}_\omega) < 1$ constitutes a necessary and sufficient condition and it holds that the smaller $\rho(\mathcal{L}_\omega)(<1)$ is, the faster the asymptotic convergence is. So the problem of the determination of the optimal ω, namely, that which minimizes $\rho(\mathcal{L}_\omega)$, is of vital importance.

To solve the problem in question, we usually assume a given structure for the spectrum $\sigma(T)$ of the Jacobi matrix

$$T := D^{-1}(L + U) \tag{1.4}$$

associated with A in (1.1). This problem has not been solved in the general case. However, if A possesses Young's "property A" (Young [19]; see also [17] or [20]) or, more generally,

* Received by the editors November 19, 1990; accepted for publication (in revised form) February 28, 1991.

† Department of Mathematics, University of Ioannina, GR-451 10 Ioannina, Greece (galanis@grioanun.bitnet).

‡ Computer Sciences Department, Purdue University, West Lafayette, Indiana 47907 (hadjidim@cs.purdue.edu). The work of this author was supported in part by National Science Foundation grant CCR-8619817 and by Air Force Office of Scientific Research grant 8810243.

Varga's "block p-cyclic consistently ordered" property ([16]; see also [17] or [20]), then under some further assumptions on $\sigma(T^p)$, for example, "$\sigma(T^p)$ is nonnegative" and "$\rho(T) < 1$," we can determine optimal values for ω and $\rho(\mathscr{L}_\omega)$, denoted from now on by ω_p and ρ_p, respectively. In Varga's work [16], as well as in other researchers' work that followed (see, e.g., [8], [10], [5], [18]), ω_p and ρ_p were found under various assumptions on $\rho(T)$.

In this paper, it is assumed that A possesses the block p-cyclic consistently ordered property (or A is "p-cyclic" for short) and is of the form

$$A = \begin{bmatrix} A_{11} & 0 & 0 & \cdots & A_{1p} \\ A_{21} & A_{22} & 0 & \cdots & 0 \\ 0 & A_{32} & A_{33} & \cdots & 0 \\ & \ddots & & \ddots & \vdots \\ & 0 & \ddots & \ddots & \vdots \\ & & \ddots & \ddots & \vdots \\ & & & A_{p,p-1} & A_{pp} \end{bmatrix}, \tag{1.5}$$

so that its associated Jacobi matrix J_p is given by

$$J_p := T = \begin{bmatrix} 0 & 0 & 0 & \cdots & T_1 \\ T_2 & 0 & 0 & \cdots & 0 \\ 0 & T_3 & 0 & \cdots & 0 \\ & \ddots & & & \vdots \\ & 0 & \ddots & & \vdots \\ & & \ddots & & \vdots \\ & & & T_p & 0 \end{bmatrix}, \tag{1.6}$$

where $T_j := -A_{jj}^{-1}A_{jk}$, $k = p$ for $j = 1$, and $k = j - 1$ for $j = 2(1)p$. From now on, the SOR method (1.2)–(1.3), with A being of the form (1.5), will be referred to as the "p-cyclic SOR."

Markham, Neumann, and Plemmons [9] were the first to consider the problem of repartitioning a 3-cyclic matrix A into a 2-cyclic one for the solution of a least squares problem (see [1] and [14]). Based on optimal results obtained previously by Kredell [8] for $p = 2$ and by Niethammer, de Pillis, and Varga [10] for $p = 3$, in the cases of nonpositive spectra $\sigma(J_2^2)$ and $\sigma(J_3^3)$, and by observing that $\sigma(J_3^3)\setminus\{0\} \equiv \sigma(J_2^2)\setminus\{0\}$, they came up with two important results. First, the optimal 2-cyclic SOR was better than the optimal 3-cyclic one, and second, the former method was convergent for any $\rho(J_3) < \infty$ [8] while the latter one was convergent only for $\rho(J_3) < 3$ [10]. Motivated by their work and a similar result in the case of a class of monoparametric p-step Euler methods [4], Galanis, Hadjidimos, and Noutsos [5] obtained, among other results, the following more general one. A repartitioning of the p-cyclic matrix A in (1.5) into a $[(p+1)/2]$-cyclic form (with $[s]$ denoting the integer part of the real number s) in the cases of nonnegative and nonpositive spectra $\sigma(J_p^p)$ yielded a better optimal cyclic SOR. So, a repetitive application of their conclusion leads to the obvious one that a repartitioning of A into a 2-cyclic form yields, eventually, an optimal 2-cyclic SOR better than the original optimal p-cyclic one. Furthermore, it is noticed that when we go down, via the aforementioned successive repartitionings, from the p- to the 2-cyclic SOR, we obtain better and better optimal q-cyclic SORs, but apparently not all the intermediate values of $q = 3(1)p - 1$ are covered. However, the basic idea of repartitioning was being questioned because there was the impression that whatever was gained by improving on the

convergence was lost due to the extra arithmetic required per iteration. Then came Pierce [12] with a very surprising result: By repartitioning a p-cyclic matrix A into a q-cyclic one ($q < p$), the arithmetic required for one iteration of either the p- or the q-cyclic SOR is exactly the same! This is true for any q-cyclic repartitioning, $q = 2(1)p - 1$, of the p-cyclic matrix A. For example, if $p = 7$ in (1.5), the 3-cyclic repartitioning of it indicated below by the solid lines

$$A = \left[\begin{array}{cc|ccc|cc} A_{11} & & & & & & A_{17} \\ A_{21} & A_{22} & & & & & \\ \hline & A_{32} & A_{33} & & & & \\ & & A_{43} & A_{44} & & & \\ & & & A_{54} & A_{55} & & \\ \hline & & & & A_{65} & A_{66} & \\ & & & & & A_{76} & A_{77} \end{array}\right] \tag{1.5'}$$

yields a 3-cyclic SOR that requires exactly the same arithmetic per iteration as the original 7-cyclic SOR.

Pierce, Hadjidimos, and Plemmons [13], based on Pierce's work [12], settled the question of the strict decrease of the optimal spectral radii of the q-cyclic SOR obtained from the repartitioning into a q-cyclic form of the original p-cyclic matrix for all values of $q = 2(1)p$. More specifically, they showed that for $\sigma(J_p^p)$ nonnegative and under the assumption $\rho(J_p) < 1$,

$$\rho_2 < \rho_3 < \cdots < \rho_{p-1} < \rho_p < 1, \tag{1.7}$$

while for $\sigma(J_p^p)$ nonpositive and under *no* restriction on $\rho(J_p)$, they showed that there exists a unique $k \in \{2, \cdots, p\}$ such that

$$\left(\frac{k+1}{k-1}\right)^{(k+1)/p} \leqq \rho(J_p) < \left(\frac{k}{k-2}\right)^{k/p}$$

(with $k/(k-2) = \infty$ for $k = 2$) so that

$$\rho_2 < \rho_3 < \cdots < \rho_k < 1 \leqq \rho(\mathscr{L}_{\omega_q}), \tag{1.8}$$

where $q = k + 1(1)p$. Immediately after the result in [13] was established, Eiermann, Niethammer, and Ruttan [2] raised, indirectly, the question as to what the best repartitioning would be if, instead of considering a nonnegative or a nonpositive spectrum for J_p^p, one considered a real spectrum. In [2] a numerical example was presented for a 4-cyclic matrix with $\sigma(J_4^4)$ real, $\beta = 0.9$, and $\alpha \in [0, 2]$, where $-\alpha^4$, β^4 were the two extreme eigenvalues of J_4^4. It was shown there that any ordering in terms of increasing magnitude of ρ_2, ρ_3, ρ_4 (the optimal spectral radii of the q-cyclic SORs corresponding to 2-, 3-, and 4-cyclic form repartitionings) was obtained as α was allowed to vary in $[0, 2]$. Also, a particular theoretical example was considered, where for a p-cyclic matrix with $\rho(J_p) = \beta < 1$ and $\sigma(J_p^p)$ real it was shown that $\rho_p < \rho_2$.

The object of this paper is to give the solution to the problem in the general case of a p-cyclic consistently ordered matrix of the form (1.5) under the assumption $\sigma(J_p^p) \subset \mathbb{R} \setminus \{[1, \infty)\}$. In § 2 we present and state the main results followed by a number of comments and remarks that clarify some basic points. Also, a discussion is presented, an illustrative table is given, and numerical examples are presented which help one "decide" how to partition A for the best optimal SOR convergence. Finally in § 3 we give

the proofs of the statements of § 2, preceded by a number of statements necessary for the analysis that follows.

Before we close this section, we would like to note the following. First, it is worth mentioning that the results of the present paper can be directly applied for the computation of the steady state distribution vectors of large-scale finite Markov chains that possess p-cyclic infinitesimal generators. (For this the reader is referred to the most recent work by Kontovasilis, Plemmons, and Stewart [7].) Second, it should be said that it would be worth investigating whether the idea of the cyclic repartitioning discussed in this paper applies to the case where instead of minimizing $\rho(\mathscr{L}_\omega)$, we are interested in minimizing $\|\mathscr{L}_\omega^m\|_2$, with the integer m being fixed, as has recently been considered by Golub and de Pillis [6].

2. Main results, comments, and discussion.

2.1. The main results. We begin with the statement of the following two theorems, the second of which includes only known results given in a compact form. The proof of the first theorem can be found in § 3.

THEOREM 2.1 (of the best repartitioning). *Let J_p be the block Jacobi matrix* (1.6) *associated with the linear system* (1.1), *where A has the p-cyclic consistently ordered form* (1.5), $p \geqq 3$, *and let* $\sigma(J_p^p) \subset [-\alpha^p, \beta^p]$, *with* $-\alpha^p, \beta^p \in \sigma(J_p^p)$, *where* $0 \leqq \beta < 1$ *and* $0 \leqq \alpha < \infty$. *Assume that A is repartitioned into a block q-cyclic consistently ordered form* ($2 \leqq q < p$) *and denote by ω_q and ρ_q the relaxation factor and the spectral radius of the optimal q-cyclic SOR* ($q = 2(1)p$). *Then the following hold*:

(I) *If*

$$0 \leqq \frac{\alpha}{\beta} < \frac{p-2}{p}, \tag{2.1}$$

then there exists a unique integer $l \in \{2, 3, \cdots, p-1\}$ *satisfying*

$$\left(\frac{l-2}{l}\right)^{l/p} \leqq \frac{\alpha}{\beta} < \left(\frac{l-1}{l+1}\right)^{(l+1)/p} \tag{2.2}$$

and such that

$$\rho_l < \rho_{l-1} < \cdots < \rho_2 < 1 \tag{2.3a}$$

and

$$\rho_{l+1} < \rho_{l+2} < \cdots < \rho_p < 1. \tag{2.3b}$$

Furthermore, to each $\beta \in (0, 1)$ *there corresponds a unique value of* α, *namely*,

$$\alpha_{l,l+1} := \alpha(\beta) \in \left[\left(\frac{l-2}{l}\right)^{l/p}\beta, \left(\frac{l-1}{l+1}\right)^{(l+1)/p}\beta\right)$$

given by

$$\alpha_{l,l+1} = \left(\frac{2\rho^{1/l} - (1+\rho)\beta^{p/l}}{1-\rho}\right)^{l/p}, \tag{2.4}$$

where ρ is the unique root, in $(0, 1)$, *of the equation*

$$\beta^p(l+\rho)^{l+1} - (l+1)^{l+1}\rho = 0, \tag{2.5}$$

such that

$$\rho_l < \rho_{l+1} \quad \text{for } \left(\frac{l-2}{l}\right)^{l/p} \beta \leqq \alpha < \alpha_{l,l+1}, \tag{2.6a}$$

$$\rho_l = \rho_{l+1} \quad \text{for } \alpha = \alpha_{l,l+1}, \tag{2.6b}$$

$$\rho_l > \rho_{l+1} \quad \text{for } \alpha_{l,l+1} < \alpha < \left(\frac{l-1}{l+1}\right)^{(l+1)/p} \beta. \tag{2.6c}$$

Consequently, in case (2.6a) *the l-cyclic SOR gives the best optimal one, in* (2.6c) *the* $(l+1)$*-cyclic SOR gives the best one, while in* (2.6b) *both the l- and the* $(l+1)$*-cyclic SOR give the best optimal SOR method.*

(II) *If*

$$\frac{p-2}{p} \leqq \frac{\alpha}{\beta} < 1, \tag{2.7}$$

then

$$\rho_p < \rho_{p-1} < \cdots < \rho_3 < \rho_2 < 1 \tag{2.8}$$

and the original p-cyclic form gives the best optimal cyclic SOR.

(III) *If*

$$\frac{\alpha}{\beta} = 1, \tag{2.9}$$

where the case $\alpha = \beta = 0$ *is also included, then*

$$\rho_p = \rho_{p-1} = \cdots = \rho_3 = \rho_2 = \beta^p < 1 \tag{2.10}$$

and any one of the q-cyclic Gauss–Seidel methods, $q = 2(1)p$*, is the best optimal cyclic SOR.*

(IV) *If*

$$1 < \frac{\alpha}{\beta} \leqq \frac{p}{p-2}, \tag{2.11}$$

the conclusions are identical to those in case (II).

(V) *If*

$$\frac{p}{p-2} < \frac{\alpha}{\beta} \leqq \infty, \tag{2.12}$$

where $\alpha/\beta = \infty$ *corresponds to the case* $\alpha > 0$, $\beta = 0$, *then there exist a unique integer* $k \in \{2, \cdots, p\}$, *such that*

$$\left(\frac{k+1}{k-1}\right)^{(k+1)/p} < \rho(J_p) \leqq \left(\frac{k}{k-2}\right)^{k/p}, \tag{2.13}$$

and a unique integer $l \in \{2, \cdots, \min(p-1, k)\}$, *satisfying*

$$\left(\frac{l+1}{l-1}\right)^{(l+1)/p} < \frac{\alpha}{\beta} \leqq \left(\frac{l}{l-2}\right)^{l/p} \tag{2.14}$$

such that

$$\rho_l < \rho_{l-1} < \cdots < \rho_3 < \rho_2 < 1 \tag{2.15}$$

and

$$\rho_{l+1} < \rho_{l+2} < \cdots < \rho_{k-1} < \rho_k \leqq 1 \leqq \rho(\mathscr{L}_{\omega_q}), \tag{2.16}$$

with $q = k + 1(1)p$ *and with equality holding in* $\rho_k \leqq 1$ *if and only if* $k > 2$ *and equality holds in* (2.13). *Furthermore, for* $l = k$, *the l-cyclic SOR gives the best optimal one. For* $l < k$ *to each* $\alpha \in (0, ((l+1)/(l-1))^{(l+1)/p})$, *there corresponds a unique value of* β, *namely*

$$\beta_{l,l+1} := \beta(\alpha) \in \left[\left(\frac{l-2}{l}\right)^{l/p}\alpha, \left(\frac{l-1}{l+1}\right)^{(l+1)/p}\alpha\right],$$

given by

$$\beta_{l,l+1} = \left(\frac{2\rho^{1/l} - (1-\rho)\alpha^{p/l}}{1+\rho}\right)^{l/p}, \tag{2.17}$$

where ρ *is the unique root, in* $(0, 1)$, *of the equation*

$$\alpha^p(l-\rho)^{l+1} - (l+1)^{l+1}\rho = 0 \tag{2.18}$$

such that

$$\rho_l < \rho_{l+1} \quad \text{for } \left(\frac{l-2}{l}\right)^{l/p}\alpha \leqq \beta < \beta_{l,l+1}, \tag{2.19a}$$

$$\rho_l = \rho_{l+1} \quad \text{for } \beta = \beta_{l,l+1}, \tag{2.19b}$$

$$\rho_l > \rho_{l+1} \quad \text{for } \beta_{l,l+1} < \beta < \left(\frac{l-1}{l+1}\right)^{(l+1)/p}\alpha. \tag{2.19c}$$

Consequently, in case (2.19a) *the l-cyclic SOR gives the best optimal one, in case* (2.19c) *it is the* $(l+1)$*-cyclic SOR, while in case* (2.19b) *both the l- and* $(l+1)$*-cyclic SOR give the best optimal cyclic SOR, respectively.*

THEOREM 2.2 (of optimal and best optimal values). *Under the assumptions of Theorem* 2.1 *let* ω_r *and* ρ_r *denote the relaxation factor and the spectral radius of either the best optimal or any optimal r-cyclic SOR (provided it converges) corresponding to an r-cyclic repartitioning of A in* (1.5) *with* $r \in \{2, 3, \cdots, p\}$. *Then* ω_r *and* ρ_r *are given from the equations*

$$\left(\frac{(\alpha_r+\beta_r)}{2}\omega\right)^r - \frac{(\alpha_r+\beta_r)}{(\beta_r-\alpha_r)}(\omega-1) = 0 \tag{2.20}$$

and

$$\rho_r = \frac{(\alpha_r+\beta_r)}{(\beta_r-\alpha_r)}(\omega_r - 1) = \left(\frac{(\alpha_r+\beta_r)}{2}\omega_r\right)^r, \tag{2.21}$$

where ω_r *is the unique positive root of* (2.20) *in*

$$\left(\min\left\{1, 1+\frac{\beta_r-\alpha_r}{\beta_r+\alpha_r}\right\}, \max\left\{1, 1+\frac{\beta_r-\alpha_r}{\beta_r+\alpha_r}\right\}\right)$$

and where

$$\alpha_r=\frac{r-2}{r}\beta^{p/r},\quad \beta_r=\beta^{p/r}\quad \text{iff}\ \frac{\alpha}{\beta}\leqq\left(\frac{r-2}{r}\right)^{r/p}, \tag{2.22a}$$

$$\alpha_r=\alpha^{p/r},\quad \beta_r=\beta^{p/r}\quad \text{iff}\left(\frac{r-2}{r}\right)^{r/p}\leqq\frac{\alpha}{\beta}\leqq\left(\frac{r}{r-2}\right)^{r/p}, \tag{2.22b}$$

$$\alpha_r=\alpha^{p/r},\quad \beta_r=\frac{r-2}{r}\alpha^{p/r}\quad \text{iff}\left(\frac{r}{r-2}\right)^{r/p}\leqq\frac{\alpha}{\beta}. \tag{2.22c}$$

2.2. Comments and remarks. After having stated Theorems 2.1 and 2.2, it is necessary to make a number of comments and remarks.

(i) It should be stressed that some of the relations in Theorem 2.1 must be interpreted in a broader sense. For example, in case (V) if $k = 2$, then the right-hand side of (2.13) should be considered as being ∞ and the corresponding inequality as a strict one, and since $l = 2$ in this case, the same interpretation applies to (2.14). Moreover, (2.15) and (2.16) will be equivalent to $\rho_2 < 1$ and $\rho_2 < 1 \leqq \rho(\mathscr{L}_{\omega_q})$, $q = 3(1)p$, respectively.

(ii) As is seen in Theorem 2.1, the decisive factor relating to which out of all possible cyclic repartitionings gives the best optimal r-cyclic SOR depends mainly on the relative position of the ratio α/β with respect to the numbers $(p-2)/p$, 1, and $p/(p-2)$.

(iii) In cases (I) and (V) of Theorem 2.1 the ratio α/β alone does not specify which one out of the two possible successive repartitionings gives the best optimal r-cyclic SOR. Then the value of α in the former case and both values of α and β in the latter one are the additional decisive factors.

(iv) The particular cases (a) $\alpha = 0$, $\beta > 0$, (b) $\alpha = \beta = 0$, and (c) $\alpha > 0$, $\beta = 0$, which were given here as special subcases, are the ones that were considered and solved in [13]. By just looking at the results given in cases (I) and (V) of Theorem 2.1 it is concluded that those obtained in [13] for $\sigma(J_p^p)$ nonnegative hold also in case (I) for $l = 2$ and $\alpha < \alpha_{2,3}$, while those for $\sigma(J_p^p)$ nonpositive hold in case (V) for $l = k = 2$ and for $l = 2 < k$ and $\beta < \beta_{2,3}$.

(v) The numerical example in [2], mentioned in the Introduction, where $p = 4$, $\beta = 0.9$, and α was taking values in $[0, 2]$, is in agreement with the theoretical results of Theorem 2.1. Also, the already mentioned theoretical example in [2], in which it was shown that $\rho_p < \rho_2$, is a very special case of case (II) of Theorem 2.1.

(vi) Although it is not explicitly stated in Theorem 2.1 we can easily decide about the relative magnitude of ρ_{q_1} and ρ_{q_2} corresponding to any q_1- and q_2-cyclic repartitionings of A even if this is not directly presented here. For example, if the assumptions of case (I) of the theorem are satisfied, where $q_1 \in \{2, \cdots, l\}$, $q_2 \in \{l+1, \cdots, p\}$, α_{q_1,q_2} can be found in a way quite analogous to that in which $\alpha_{l,l+1}$ was found and a decision along the lines suggested by (2.6) can be made. The same applies to case (V) where for $l < k$ the numbers β_{q_1,q_2} play similar roles to those of α_{q_1,q_2} in case (I). In fact, based on this observation, we can devise a simple algorithm to order all ρ_q's, $q = 2(1)p$, in increasing order of magnitude in both cases (I) and (V).

(vii) As was already mentioned, Theorem 2.2 is not a new one. It is simply adjusted to be in agreement with the notation of Theorem 2.1, and is a compact form of all the known theorems concerned with optimal values for the r-cyclic SOR, $r = 2(1)p$, when $\sigma(J_p^p) \subset \mathbb{R}\setminus\{[1, \infty)\}$. It follows a pattern already presented in [3]. So, the classical results of Young [19], $\sigma(J_2^2) \geqq 0$, $\rho(J_2) < 1$, Varga [16], $\sigma(J_p^p) \geqq 0$, $p \geqq 3$, $\rho(J_p) < 1$, and also of Kredell [8], $\sigma(J_2^2) \leqq 0$, Niethammer, de Pillis, and Varga [10], $\sigma(J_3^3) \leqq 0$, $\rho(J_3) < 3$, Wild and Niethammer [18], and Galanis, Hadjidimos, and Noutsos [5],

covering both the nonnegative and the nonpositive cases for $\sigma(J_p^p)$, can be easily recovered. Furthermore, Theorems 1 and 2 of Eiermann, Niethammer, and Ruttan [2] are included in Theorem 2.2, which is also important.

(viii) The methods used in § 3 to prove Theorem 2.1 are "elementary" algebraic methods. We did not try to use "directly" complex analysis statements and the "elegant" geometric approach involving hypocycloids of cusped and shortened type as was done in [11], [4], [18], [5], and especially in [2]. This was because it "seemed" that we could not always have a particular region strictly contained within another, as, e.g., in [2, Thm. 3]. So we could not draw conclusions as to the relative magnitude of the corresponding spectral radii of the optimal SORs involved, while these conclusions were drawn "easily" by using algebraic methods.

2.3. How can one "decide" how to repartition? Given a p-cyclic ($p \geqq 3$) matrix A of the form (1.5), the practical question, which arises naturally, is: How can one "decide" how to repartition A so that the best possible optimal SOR convergence can be achieved? Or, in other words, how can one make use of the findings of Theorem 2.1 and determine the cyclicity r that gives the best repartitioning?

We answer this question under the assumption that $\sigma(J_p^p)$ is real. In such a case we estimate lower and upper bounds for the spectrum in question by considering any of the diagonal blocks of J_p^p, preferably those of the smallest possible order. Let these bounds be $\underline{b}$ and $\bar{b}$, respectively. If $\bar{b} \geqq 1$, then *no* optimal convergent SOR method can be obtained. If $\bar{b} < 1$, then set

$$-\alpha^p := \min\{0, \underline{b}\}, \qquad \beta^p := \max\{\bar{b}, 0\}, \tag{2.23}$$

with $\alpha, \beta \geqq 0$, and form the ratio α/β. The value of this ratio indicates the basic case (or subcase) in Table 1 in which the cyclicity r of the best repartitioning is to be sought. In subcase I(ii) (V(i)) the integer l and the real $\alpha_{l,l+1}$ (the integers k, l and, if $l < k$, the real $\beta_{l,l+1}$) have to be determined in order to distinguish further subcases. Having obtained r from Table 1, the optimal parameters of the r-cyclic SOR can be determined by applying (2.20)–(2.21) of Theorem 2.2, with α_r, β_r being given from the corresponding case of (2.22), essentially indicated by the ratio α/β (or β/α) considered previously.

2.4. Numerical examples. In this section 14 numerical examples are given in order to show how easy it is to make use of Table 1. These examples cover all possible cases and subcases except the two trivial ones I(iib) ($\alpha = \alpha_{l,l+1}$) and V(iBb) ($\beta = \beta_{l,l+1}$) and are presented in the self-explained Table 2. The cyclicity of the matrix coefficient A in (1.5) is considered to be $p = 4$, $\beta = 0.8$ (<1) in all but example 14, where $\beta = 0$, and α is given 14 values in [0, 5]. In examples 1 and 14 we are obviously in subcases I(i) and V(ii), respectively, so the cyclicity $r = 2$ of the best repartitioning is immediately implied from Table 1. Since $(p-2)/p = 0.5$ and $p/(p-2) = 2$, we readily see from Table 1 that examples 6, 7, and 8 are in cases II, III, and IV, respectively, hence the corresponding values for r follow directly, while examples 2–4 are in subcase II(ii) and examples 9–13 in V(i). The integer l is to be found in all the latter examples (examples 2–4, 9–13) while k is to be found only when $\alpha/\beta > p/(p-2) = 2$, that is, in examples 9–13. In examples 2 and 3, $l = 2$ and therefore $\alpha_{2,3}$ is determined from the corresponding equations (2.4)–(2.5). Since, as is seen, $\alpha < \alpha_{2,3}$ and $\alpha > \alpha_{2,3}$, respectively, the former example is in subcase I(iia), while the latter one is in I(iic), and the best cyclicities $r = 2$ and 3 are readily implied. The situation in examples 4 and 5 is similar except that $l = 3$. In examples 9 and 10, due to the fact that $l = 3 < k = 4$, $\beta_{3,4}$ has to be determined using (2.17)–(2.18). Obviously, there are two different values for $\beta_{3,4}$ because of the two different values for α. It is then readily found out that in example 9 we are in subcase V(iBc),

TABLE 1
Cyclicity r of the best repartitioning.

Case	Value or domain of the ratio α/β	Values of l and $\alpha_{l,l+1}$ (or $\beta_{l,l+1}$) if further subcases have to be considered	Further subcases	Cyclicity r of the best repartitioning
I	(i) 0 $(0 = \alpha < \beta < 1)$	—	—	2
	(ii) $(0, (p-2)/p)$	Determine the largest integer $l \in \{2, 3, \cdots, p-1\}$: $((l-2)/l)^l \leqq (\alpha/\beta)^p$ and then $\alpha_{l,l+1}$ from (2.4)–(2.5)	(a) $((l-2)/l)^{l/p}\beta \leqq \alpha < \alpha_{l,l+1}$	l
			(b) $\alpha = \alpha_{l,l+1}$	$l, l+1$*
			(c) $\alpha_{l,l+1} < \alpha < ((l-1)/(l+1))^{(l+1)/p}\beta$	$l+1$
II	$[(p-2/p), 1)$	—	—	p
III	1 $(0 \leqq \alpha = \beta < 1)$	—	—	$2, 3, \cdots, p$†
IV	$(1, p/(p-2)]$	—	—	p
V	(i) $(p/(p-2), \infty)$	Determine the largest integers $k \in \{2, 3, \cdots, p\}$: $((k-2)/k)^k \leqq (1/\alpha^p)$, $l \in \{2, 3, \cdots, \min(p-1, k)\}$: $((l-2)/l)^l \leqq (\beta/\alpha)^p$ and, if $l < k$, then $\beta_{l,l+1}$ from (2.17)–(2.18)	(A) $l = k$	l
			(B) $l < k$	
			(a) $((l-2)/l)^{l/p}\alpha \leqq \beta < \beta_{l,l+1}$	l
			(b) $\beta = \beta_{l,l+1}$	$l, l+1$*
			(c) $\beta_{l,l+1} < \beta < ((l-1)/(l+1))^{(l+1)/p}\alpha$	$l+1$
	(ii) ∞ $(\alpha > 0, \beta = 0)$	—	—	2

* Either will do.
† Any will do. The optimal SOR is the Gauss–Seidel method.

TABLE 2
Cyclicity r of the best repartitioning for p = 4.

Example	(α, β)	$\frac{\alpha}{\beta}$	k	l	$\alpha_{l,l+1}$, $\beta_{l,l+1}$	Case or subcase in Table 1	r
1	(0, 0.8)	0	—	2*	—	I(i)	2
2	(0.15, 0.8)	0.1875	—	2	$\alpha_{2,3} = 0.2183632$	I(iia)	2
3	(0.3, 0.8)	0.375	—	2	$\gg$	I(iic)	3
4	(0.36, 0.8)	0.45	—	3	$\alpha_{3,4} = 0.3701637$	I(iia)	3
5	(0.38, 0.8)	0.475	—	3	$\gg$	I(iic)	4
6	(0.5, 0.8)	0.625	—	—	—	II	4
7	(0.8, 0.8)	1	—	—	—	III	2, 3, 4
8	(1.5, 0.8)	1.875	—	—	—	IV	4
9	(1.62, 0.8)	2.025	4	3	$\beta_{3,4} = 0.7946596$	V(iBc)	4
10	(1.7, 0.8)	2.125	4	3	$\beta_{3,4} = 0.8374480$	V(iBa)	3
11	(1.9, 0.8)	2.375	4	2	$\beta_{2,3} = 0.7861522$	V(iBc)	3
12	(2.0, 0.8)	2.5	4	2	$\beta_{2,3} = 0.8419411$	V(iBa)	2
13	(3.5, 0.8)	4.375	2	2	—	V(iA)	2
14	(5.0, 0)	∞	2*	2*	—	V(ii)	2

* No need to be given!

while in example 10 we are in V(iBa). Examples 11 and 12 are similar to the two previous ones except that $l = 2$. Finally, in example 13 it is found out that $l = k = 2$; we are therefore in subcase V(iA), and the conclusion $r = l = 2$ readily follows from Table 1.

In connection with examples 12, 13, and 14 it would be worth noting the following: In all three of them and in view of the fact that $\alpha \geqq p/(p-2) = 2$, the original 4-cyclic SOR diverges for any value of ω. However, in example 12 both the 3-cyclic and the 2-cyclic repartitionings lead to convergent SORs and the best out of the two optimal ones happens to be the 2-cyclic SOR. On the other hand, in examples 13 and 14 and because of the additional fact that $\alpha^{4/3} > 3/(3-2) = 3$, even the 3-cyclic repartitioning fails to produce an SOR that converges for some ω. So, in the last two examples it is only the 2-cyclic repartitioning that (always) produces convergent SORs and leads, therefore, to the best SOR method.

3. Proof of Theorem 2.1.

3.1. Introductory basic material. The key to the proof of Theorem 2.1 is to prove case (II) first. For this and for all the other results, the analysis is mainly based on the recent works by Wild and Niethammer [18]; Galanis, Hadjidimos, and Noutsos [5]; Pierce, Hadjidimos, and Plemmons [13]; and Eiermann, Niethammer, and Ruttan [2]. We begin our analysis by stating and proving four lemmas that will be very useful in the sequel.

LEMMA 3.1. *The function*

$$y(x) := \left(\frac{x-2}{x}\right)^x, \qquad x \in [2, \infty), \tag{3.1}$$

is strictly increasing.

Proof. On differentiating (3.1), we obtain

$$dy/dx = y(x)z(x), \tag{3.2}$$

where

$$z(x) := \ln\left(\frac{x-2}{x}\right) + \frac{x}{x-2} - 1. \tag{3.3}$$

$z(x)$ is well defined and differentiable in $(2, \infty)$. Thus, it is readily found out that

$$dz/dx \sim -4 < 0, \qquad x \in (2, \infty), \tag{3.4}$$

where for convenience we use the notation "$\sim$" to denote equality of sign between two expressions. Equations (3.4) imply that $z(x)$ strictly decreases with x and therefore

$$\inf_{x \in (2,\infty)} z(x) = \lim_{x \to \infty} z(x) = 0,$$

in view of (3.3). So, $z(x) > 0$ and from (3.2), by virtue of (3.1), it is concluded that $y(x)$ strictly increases in $(2, \infty)$, taking on positive values. Noting that $y(x)$ is continuous on $[2, \infty)$ and that $y(2) = 0$ completes the proof. □

COROLLARY 3.1. *For any integer* $p \geqq 3$, *there holds*:

$$y(2) < y(3) < \cdots < y(p-1) < y(p), \tag{3.5}$$

with y *being the function in* (3.1).

LEMMA 3.2. *Let $p \geqq 3$ be a given integer and a, b be positive constants satisfying*

$$\frac{a}{b} \geqq \left(\frac{p-2}{p}\right)^p$$

with $b < 1$. Then the function

$$g(x) = 1 - \frac{x}{2}(b^{1/x} - a^{1/x}), \tag{3.6}$$

defined for all real $x \in [2, p]$, takes on only positive values.

Proof. For $a \geqq b$ the validity of the statement is obvious. For $a < b\ (<1)$, it is

$$g(x) \sim \frac{2}{x} - \left(1 - \left(\frac{a}{b}\right)^{1/x}\right) b^{1/x}. \tag{3.7}$$

To prove that the right-hand side of (3.7) is positive, since $0 < b < 1$, is equivalent to proving that

$$\frac{a}{b} \geqq \left(\frac{x-2}{x}\right)^x. \tag{3.8}$$

But

$$\frac{a}{b} \geqq \left(\frac{p-2}{p}\right)^p \geqq \left(\frac{x-2}{x}\right)^x, \tag{3.9}$$

with the first inequality in (3.9) holding by our assumptions on a and b and the second one by virtue of Lemma 3.1. This proves that $g(x)$ in (3.6) is always positive. □

LEMMA 3.3. *Let c and d be distinct constants satisfying*

$$0 < c < 1, \qquad 0 < d. \tag{3.10}$$

Then the function

$$\begin{aligned} f(\rho) := {} & [(c-d)\rho + (c+d)] \ln \left(\tfrac{1}{2}[(c-d)\rho + (c+d)]\right) \\ & + (-c \ln c + d \ln d)\rho + (-c \ln c - d \ln d), \qquad \rho \in [0, 1], \end{aligned} \tag{3.11}$$

is strictly increasing, taking on only negative values and the value zero at $\rho = 1$.

Proof. $f(\rho)$ is a well defined and twice continuously differentiable function on [0, 1]. It is easy to derive that

$$f'(\rho) = (c-d)\{1 + \ln (\tfrac{1}{2}[(c-d)\rho + (c+d)])\} + (-c \ln c + d \ln d) \tag{3.12}$$

and

$$f''(\rho) = \frac{(c-d)^2}{(c-d)\rho + (c+d)} > 0 \quad \forall \rho \in [0, 1]. \tag{3.13}$$

Thus $f'(\rho)$ strictly increases with ρ in [0, 1]. Hence

$$\begin{aligned} \min_{\rho \in [0,1]} f'(\rho) = f'(0) &\sim \frac{c}{d}\left\{1 + \ln\left[\frac{1}{2}\left(1 + \frac{d}{c}\right)\right]\right\} - \left\{1 + \ln\left[\frac{1}{2}\left(1 + \frac{c}{d}\right)\right]\right\} \\ &= x\left[1 + \ln\left(\frac{x+1}{2x}\right)\right] - \left[1 + \ln\left(\frac{x+1}{2}\right)\right] =: z(x), \end{aligned} \tag{3.14}$$

where we have set $x = c/d$. The function $z(x)$ in (3.14) is twice continuously differentiable. So it can be obtained that

$$\frac{dz}{dx} = \frac{x-1}{x+1} + \ln\left(\frac{x+1}{2x}\right)$$

and

$$\frac{d^2z}{dx^2} \sim x - 1. \tag{3.15}$$

Consequently, if $c > d$, then $x > 1$ and from (3.15) it is concluded that dz/dx strictly increases, as a function of $x \in [1, \infty)$, and thus

$$\min_{x\in[1,\infty)} \frac{dz}{dx} = \frac{dz}{dx}\bigg|_{x=1} = 0.$$

Therefore, dz/dx is always positive in $(1, \infty)$, implying that $z(x)$ strictly increases with x. So from (3.12) and (3.14) it is obtained that

$$\min_{\rho\in[0,1]} f'(\rho) = f'(0) > d \inf_{x\in(1,\infty)} z(x) = 0. \tag{3.16}$$

If $c < d$, then $x < 1$ and from (3.15) we have that dz/dx strictly decreases with x increasing in $(0, 1]$, and

$$\min_{x\in(0,1]} \frac{dz}{dx} = \frac{dz}{dx}\bigg|_{x=1} = 0.$$

Therefore we arrive at the same relations (3.16) as before. However, the result (3.16) and the strictly increasing nature of $f'(\rho)$ in $[0, 1]$, proved previously, imply that $f(\rho)$ is strictly increasing in $[0, 1]$. On the other hand,

$$\max_{\rho\in[0,1]} f(\rho) = f(1) = 0,$$

as is readily checked, which proves the lemma. □

LEMMA 3.4. *Let the relation*

$$F(x,\rho) := \rho - \left(\tfrac{1}{2}[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})]\right)^x = 0, \tag{3.17}$$

in x and ρ be given, where $p \geqq 3$ is a known integer, and a and b (<1) positive constants satisfying

$$\frac{a}{b} \geqq \left(\frac{p-2}{p}\right)^p.$$

Then ρ is a well-defined function of x for all $x \in [2, p]$ and x is a well-defined function of $\rho \in [0, 1]$. Moreover, ρ is a strictly decreasing function of $x \in [2, p]$.

Proof. Obviously, F as well as

$$F_x = -\left(\tfrac{1}{2}[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})]\right)^x$$

$$\cdot\Bigg\{\ln\left(\tfrac{1}{2}[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})]\right)$$

$$+ x\,\frac{\tfrac{1}{2}[(-(1/x^2)b^{1/x}\ln b + (1/x^2)a^{1/x}\ln a)\rho + (-(1/x^2)b^{1/x}\ln b - (1/x^2)a^{1/x}\ln a)]}{\tfrac{1}{2}[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})]}\Bigg\}$$

and

$$F_\rho = 1 - x(\tfrac{1}{2}[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})])^{x-1}\tfrac{1}{2}(b^{1/x} - a^{1/x}),$$

which can be rewritten by using (3.17) as

$$\begin{aligned} F_x = -\tfrac{1}{2}\rho^{1-1/x}\{&[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})] \\ &\cdot \ln(\tfrac{1}{2}[(b^{1/x} - a^{1/x})\rho + (b^{1/x} + a^{1/x})]) \\ &+ (-b^{1/x}\ln b^{1/x} + a^{1/x}\ln a^{1/x})\rho \\ &+ (-b^{1/x}\ln b^{1/x} - a^{1/x}\ln a^{1/x})\} \end{aligned} \tag{3.18}$$

and

$$F_\rho = 1 - \frac{x}{2}(b^{1/x} - a^{1/x})\rho^{1-1/x}, \tag{3.19}$$

respectively, are continuous functions of x and ρ for all $x \in [2, p]$ and $\rho \in [0, 1]$. Furthermore and for all the aforementioned values of x and ρ, $F_x \geqq 0$ (with equality holding for $\rho = 1$) by virtue of Lemma 3.3, with $c = b^{1/x}$ and $d = a^{1/x}$, while $F_\rho > 0$ by virtue of Lemma 3.2. Therefore, by the Implicit Function Theorem (see, e.g., [15, Thm. 14.1]), ρ is a well-defined function of $x \in [2, p]$ and x is a well-defined function of $\rho \in [0, 1]$. Moreover, $d\rho/dx$ exists, is continuous, and is given by

$$\rho' = -\frac{F_x}{F_\rho} \leqq 0, \tag{3.20}$$

with equality holding for $\rho = 1$. So ρ is a strictly decreasing function of $x \in [2, p]$ and the proof is complete. □

Now we have all the necessary tools to prove case (II) of Theorem 2.1 and subsequently all the other cases of the theorem in question.

3.2. Proof of case (II) of Theorem 2.1. Let q refer to any q-cyclic repartitioning (as well as to p itself) of the p-cyclic matrix in (1.5) and let the assumptions of case (II) of Theorem 2.1 hold. As is known from [13], there holds

$$\sigma(J_q^q)\setminus\{0\} \equiv \sigma(J_p^p)\setminus\{0\}, \qquad q = 2(1)p. \tag{3.21}$$

Then, if $-\alpha_q^q$ and β_q^q denote the extreme eigenvalues of J_q^q, where $\alpha_q, \beta_q > 0$, it will be

$$\alpha_q = a^{1/q}, \qquad \beta_q = b^{1/q} \tag{3.22}$$

where we have set

$$a = \alpha^p, \qquad b = \beta^p. \tag{3.23}$$

In our case we have

$$\frac{p-2}{p} \leqq \frac{\alpha}{\beta} < 1,$$

implying

$$1 > \frac{\alpha^p}{\beta^p} \geqq \left(\frac{p-2}{p}\right)^p.$$

Now from (3.22)–(3.23) and by Corollary 3.1, we readily obtain that

$$1 > \frac{\alpha_q^q}{\beta_q^q} \geqq \left(\frac{q-2}{q}\right)^q, \qquad q = 2(1)p,$$

or equivalently,

$$\frac{q-2}{q} \leqq \frac{\alpha_q}{\beta_q} < 1, \qquad q = 2(1)p. \tag{3.24}$$

Thus the optimal parameters of the q-cyclic SOR will be determined from (2.20) and (2.21) of Theorem 2.2, with $r = q$ and α_r and β_r being given by (2.22b), where in view of (3.22),

$$\alpha_r = a^{1/q}, \qquad \beta_r = b^{1/q}.$$

Consequently, ω_q is the unique positive root in

$$\left(1, 1 + \frac{b^{1/q} - a^{1/q}}{b^{1/q} + a^{1/q}}\right)$$

of the equation

$$\left(\frac{(a^{1/q} + b^{1/q})}{2}\omega\right)^q - \frac{(a^{1/q} + b^{1/q})}{(b^{1/q} - a^{1/q})}(\omega - 1) = 0, \tag{3.25}$$

while ρ_q is given by

$$\rho_q = \frac{(a^{1/q} + b^{1/q})}{(b^{1/q} - a^{1/q})}(\omega_q - 1) = \left(\frac{(a^{1/q} + b^{1/q})}{2}\omega_q\right)^q. \tag{3.26}$$

Dropping the index q from ρ_q, for simplicity, and eliminating ω_q from the two equations in (3.26), we obtain

$$\rho = \{\tfrac{1}{2}[(b^{1/q} - a^{1/q})\rho + (b^{1/q} + a^{1/q})]\}^q, \qquad q = 2(1)p. \tag{3.27}$$

Now we embed the set of integers $\{2, 3, \cdots, p\}$ into the set of real numbers of the closed interval $[2, p]$. We then observe that the difference of the two members of (3.27) set equal to zero is nothing but the relation $F(x, \rho) = 0$ in (3.17) of Lemma 3.4, where the real q now plays the role of the real x in (3.17). As can be readily checked, all other assumptions of Lemma 3.4 hold. Therefore, all the conclusions of this lemma hold true. Consequently, ρ is a strictly decreasing function of $x \in [2, p]$, which effectively proves (2.8). □

3.3. Proof of case (I) of Theorem 2.1. This time condition (2.1) is satisfied or, equivalently,

$$0 \leqq \frac{\alpha^p}{\beta^p} < \left(\frac{p-2}{p}\right)^p.$$

From these inequalities, in view of (3.22)–(3.23) and Corollary 3.1 and because

$$0 = \left(\frac{2-2}{2}\right)^2,$$

we are readily led to the conclusion that there exists a unique integer $l \in \{2, 3, \cdots, p-1\}$ such that

$$0 \leqq \left(\frac{l-2}{l}\right)^l \leqq \frac{\alpha_l^l}{\beta_l^l} = \frac{\alpha^p}{\beta^p} = \frac{\alpha_{l+1}^{l+1}}{\beta_{l+1}^{l+1}} < \left(\frac{l-1}{l+1}\right)^{l+1} \leqq \left(\frac{p-2}{p}\right)^p, \tag{3.28}$$

from which (2.2) follows. But now from (3.28) the following inequalities are implied:

$$0 \leqq \frac{l-2}{l} \leqq \frac{\alpha_l}{\beta_l},$$

which are of the same type as those in (2.7), with l instead of p. Thus the conclusion of § 3.2 holds for all $q = 2(1)l$, implying the validity of (2.3a). On the other hand, for all $q = l + 1(1)p$, the analysis of [13] for the nonnegative case applies directly, leading to the conclusion (2.3b). It remains to compare ρ_l and ρ_{l+1}, which are to be determined by Theorem 2.2. For the optimal l-cyclic SOR, having in mind from (3.28) that

$$\frac{l-2}{l} \leqq \frac{\alpha_l}{\beta_l} < 1,$$

formulas (2.20), (2.21) are applied with (2.22b). In other words ω_l is the unique root in

$$\left(1, 1+\frac{\beta^{p/l}-\alpha^{p/l}}{\beta^{p/l}+\alpha^{p/l}}\right)$$

of

$$\left(\frac{(\alpha^{p/l}+\beta^{p/l})}{2}\omega\right)^l - \frac{(\alpha^{p/l}+\beta^{p/l})}{(\beta^{p/l}-\alpha^{p/l})}(\omega-1)=0, \tag{3.29}$$

while ρ_l is given by

$$\rho_l = \frac{(\alpha^{p/l}+\beta^{p/l})}{(\beta^{p/l}-\alpha^{p/l})}(\omega_l-1) = \left(\frac{(\alpha^{p/l}+\beta^{p/l})}{2}\omega_l\right)^l. \tag{3.30}$$

For the optimal $(l+1)$-cyclic SOR it is

$$0 \leqq \frac{\alpha_{l+1}}{\beta_{l+1}} < \frac{l-1}{l+1}$$

and formulas (2.20), (2.21) with (2.22a) apply. Thus ω_{l+1} is the unique root, in $(1, (l+1)/l)$, of

$$(\beta^{p/(l+1)}\omega)^{l+1} - \frac{(l+1)^{l+1}}{l^l}(\omega-1)=0 \tag{3.29'}$$

while ρ_{l+1} is given by

$$\rho_{l+1} = \left(\frac{l}{l+1}\beta^{p/(l+1)}\omega_{l+1}\right)^{l+1} = l(\omega_{l+1}-1). \tag{3.30'}$$

Eliminating ω_l and ω_{l+1} from the equations in (3.30) and (3.30'), respectively, we obtain

$$\rho_l = \left\{\frac{1}{2}[(\beta^{p/l}-\alpha^{p/l})\rho_l + (\beta^{p/l}+\alpha^{p/l})]\right\}^l \tag{3.31}$$

and

$$\rho_{l+1} = \beta^p\left(\frac{l+\rho_{l+1}}{l+1}\right)^{l+1}. \tag{3.31'}$$

It is known, and can be very easily checked, that for a fixed β (p and l are fixed) provided

$$\frac{\alpha}{\beta} = \left(\frac{\alpha_l}{\beta_l}\right)^{l/p} \in \left[\left(\frac{l-2}{l}\right)^{l/p}, 1\right)$$

and

$$\frac{\alpha}{\beta} = \left(\frac{\alpha_{l+1}}{\beta_{l+1}}\right)^{(l+1)/p} \in \left[0, \left(\frac{l-1}{l+1}\right)^{(l+1)/p}\right),$$

ρ_l in (3.31) (or in (3.30)) is a strictly increasing function of α, while ρ_{l+1} in (3.31′) remains fixed. Since from the analysis so far it follows that for

$$\frac{\alpha_l}{\beta_l}=\frac{l-2}{l}, \qquad \rho_l<\rho_{l+1},$$

while for

$$\frac{\alpha_{l+1}}{\beta_{l+1}}=\frac{l-1}{l+1}, \qquad \rho_{l+1}<\rho_l,$$

it is implied that for

$$\frac{\alpha}{\beta}\in\left(\left(\frac{l-2}{l}\right)^{l/p},\left(\frac{l-1}{l+1}\right)^{(l+1)/p}\right)$$

there exists a unique value of α (β is kept fixed in (0, 1)), denoted by $\alpha_{l,l+1}$, such that $\rho_l = \rho_{l+1}$. To determine $\rho := \rho_l = \rho_{l+1}$, we find the unique value of $\rho \in (0, 1)$, which satisfies (3.31′), namely,

$$\rho=\beta^p\left(\frac{l+\rho}{l+1}\right)^{l+1}. \tag{3.32}$$

(Note: To show that a unique value of ρ exists, rewrite (3.32) as

$$h(\rho):=\beta^p(l+\rho)^{l+1}-(l+1)^{l+1}\rho=0 \tag{3.33}$$

and verify that

$$h(0)=\beta^p l^{l+1}>0, \qquad h(1)=(l+1)^{(l+1)}(\beta^p-1)<0$$

and

$$\begin{aligned}h'(\rho)&=(l+1)\beta^p(l+\rho)^l-(l+1)^{l+1}\\&\sim\beta^p(l+\rho)^{l+1}-(l+1)^l(l+\rho)\sim(l+1)\rho-(l+\rho)=l(\rho-1),\end{aligned}$$

if (3.33) is used, and so $h'(\rho) < 0$ for all $\rho \in (0, 1)$.) Next, having found ρ, we solve (3.31) for α to obtain

$$\alpha_{l,l+1}=\left(\frac{2\rho^{1/l}-(1+\rho)\beta^{p/l}}{1-\rho}\right)^{l/p}. \tag{3.34}$$

Consequently, the conclusions in (2.6) readily follow and the proof of case (I) is complete. □

3.4. Proofs of cases (III)–(V) of Theorem 2.1. For the proof of case (III) we note that the particular case $\alpha = \beta = 0$ was presented in [13]. In case $0 < \alpha = \beta < 1$, we may follow a limiting process argument by assuming that $\alpha \to \beta^-$ continuously. In such a case the results of case (II) apply since then for values of α very close to β it will be

$$1>\frac{\alpha_q}{\beta_q}=\frac{\alpha^{p/q}}{\beta^{p/q}}>\left(\frac{p-2}{p}\right)^{p/q}\geqq\frac{q-2}{q}, \qquad q=2(1)p.$$

So the optimal values for each q will be found from (3.25)–(3.26), which, in view of (3.23), can be rewritten as follows:

$$(\beta^{p/q}-\alpha^{p/q})\left(\frac{(\alpha^{p/q}+\beta^{p/q})}{2}\omega\right)^q-(\alpha^{p/q}+\beta^{p/q})(\omega-1)=0 \tag{3.25′}$$

and

$$\rho_q=\left(\frac{(\alpha^{p/q}+\beta^{p/q})}{2}\omega_q\right)^q. \tag{3.26'}$$

For $\alpha \to \beta^-$, (3.25') gives in the limit $\omega_q = 1$ for all $q = 2(1)p$, while from (3.26') we have $\rho_q = \beta^p$ for all $q = 2(1)p$.

For the proof of case (IV) a process analogous to the one for case (II) is followed. This time the function

$$y(x)=\left(\frac{x}{x-2}\right)^x, \qquad x\in(2,\infty),$$

is considered assuming that for $x = 2$, $y(2) = \infty$, $y(x)$ is the inverse of the function considered in Lemma 3.1 and Corollary 3.1 and is thus strictly decreasing for $x \in [2, \infty)$. Consequently,

$$1<\frac{\alpha_q}{\beta_q}=\frac{\alpha^{p/q}}{\beta^{p/q}}\leqq\left(\frac{p}{p-2}\right)^{p/q}\leqq\frac{q}{q-2}, \qquad q=2(1)p \tag{3.24'}$$

analogous to (3.24). So the optimal parameters will be given again by (3.25) and (3.26), the only difference being that

$$\omega_q\in\left(1+\frac{b^{1/q}-a^{1/q}}{b^{1/q}+a^{1/q}},1\right).$$

As a consequence, we have that (3.27) holds even in this present case. The remainder of the proof is a repetition of the argumentation of the proof of case (II) and shows the validity of (2.8).

For the proof of case (V) we begin (i) by noting that the case $\alpha > 0$, $\beta = 0$ was given in [13], and (ii) with a basic observation made in [13]. More specifically, if

$$\rho(J_p)=\alpha<\frac{p}{p-2}$$

is not valid, that is, if

$$\rho(J_p)=\alpha\geqq\frac{p}{p-2},$$

then the p-cyclic SOR will diverge for any value of ω. But the possibility of a q-cyclic repartitioning leading to a convergent optimal SOR is not excluded. For example, for $q = 2$, it will be

$$\left(\frac{p}{p-2}\right)^p<\frac{\alpha^p}{\beta^p}=\frac{\alpha_2^2}{\beta_2^2}<\infty\left(:=\left(\frac{2}{2-2}\right)^2!\right),$$

as was pointed out in the Introduction (see also remark (i) in § 2.2), and therefore convergence for the optimal 2-cyclic SOR will be guaranteed. So it is clear from (2.12) that, in view of the decreasing nature of the function $(x/(x-2))^x$, there exists a unique integer

$$k\in\{2,\cdots,p\}$$

satisfying (2.13). Thus for $q = k + 1(1)p$, if this set is not empty, the optimal q-cyclic SOR, whatever this optimal is, will diverge and this is stated in the rightmost inequality in (2.16). For all other values of $q\in\{2,\cdots,\min(p-1,k)\}$, a method of proof analogous

to that of the proof of case (I) is followed. In other words, we again claim the existence of a unique integer l satisfying (2.14), which is true due to the decreasing nature of $(x/(x-2))^x$. Obviously, for all $q = l + 1(1)k$, provided this set is not empty, there will hold

$$\left(\frac{q}{q-2}\right)^{q/p} \leqq \left(\frac{l+1}{l-1}\right)^{(l+1)/p} < \left(\frac{\alpha_q}{\beta_q}\right)^{q/p} = \frac{\alpha}{\beta}\left(\leqq \left(\frac{l}{l-2}\right)^{l/p}\right),$$

and a reasoning along the lines of [13] in the nonpositive case shows the validity of the strict inequalities in (2.16). For $q = 2(1)l$, the proof follows exactly the same pattern of proof as in the corresponding part of case (IV). It is obvious that for $l = k$, the analysis so far leads us to the conclusion that the optimal l-cyclic SOR gives the best SOR method. This is because for $l = k$ it is as if we are in case (IV) with l (or k) playing the role of p. For $l < k$ to decide which of ρ_l and ρ_{l+1} is the smallest of the two, we note that for a fixed $\alpha \in (0, ((l+1)/(l-1))^{(l+1)/p})$ and a varying β so that

$$\frac{\alpha}{\beta} \in \left(\left(\frac{l+1}{l-1}\right)^{(l+1)/p}, \left(\frac{l}{l-2}\right)^{l/p}\right),$$

ρ_{l+1} remains fixed while ρ_l strictly increases with β. An analogous analysis to the one before in the proof of case (I) easily yields the results in (2.17)–(2.19) and the proof is complete. □

Acknowledgments. The authors wish to thank Bob Plemmons for his very constructive comments and suggestions on an earlier version of this paper. We also thank Dimitrios Noutsos, who checked all the proofs in the manuscript very carefully and made a number of points. Finally, we express our sincere thanks to the referee for his valuable criticism, which helped us to improve the quality of the paper.

REFERENCES

[1] Y. T. Chen, *Iterative methods for linear least squares problems*, Ph.D. thesis, University of Waterloo, Ontario, Canada, 1975.

[2] M. Eiermann, W. Niethammer, and A. Ruttan, *Optimal successive overrelaxation iterative methods for p-cyclic matrices*, Numer. Math., 57 (1990), pp. 593–606.

[3] S. Galanis and A. Hadjidimos, *On some convergence results of the k-step iterative schemes*, Appl. Numer. Math., 7 (1991), pp. 297–308.

[4] S. Galanis, A. Hadjidimos, and D. Noutsos, *On different classes of monoparametric stationary iterative methods for the solution of linear systems*, Math. Comput. Simulation, 28 (1986), pp. 115–128.

[5] ———, *On the equivalence of the k-step iterative Euler methods and* SOR *methods for k-cyclic matrices*, Math. Comput. Simulation, 30 (1988), pp. 213–230.

[6] G. H. Golub and J. de Pillis, *Toward an effective two-parameter* SOR *method*, in Iterative Methods for Large Linear Systems, D. R. Kincaid and L. Hayes, eds., Academic Press, New York, 1990, pp. 107–118.

[7] K. Kontovasilis, R. J. Plemmons, and W. J. Stewart, *Block cyclic* SOR *for Markov chains with p-cyclic infinitesimal generator*, Tech. Report, Center for Research in Scientific Computing, North Carolina State University, Raleigh, NC, 1990. Linear Algebra Appl., 154/156 (1991), pp. 145–224.

[8] B. Kredell, *On complex successive overrelaxation*, BIT, 2 (1962), pp. 143–152.

[9] T. L. Markham, M. Neumann, and R. J. Plemmons, *Convergence of a direct-iterative method for large-scale least squares problems*, Linear Algebra Appl., 69 (1985), pp. 155–167.

[10] W. Niethammer, J. de Pillis, and R. S. Varga, *Convergence of block-iterative methods applied to sparse least squares problems*, Linear Algebra Appl., 58 (1984), pp. 327–341.

[11] W. Niethammer and R. S. Varga, *The analysis of k-step iterative methods for linear systems from summability theory*, Numer. Math., 41 (1983), pp. 177–206.

[12] D. J. PIERCE, *Parallel least squares computations and related material*, Ph.D. thesis, North Carolina State University, Raleigh, NC, 1987.
[13] D. J. PIERCE, A. HADJIDIMOS, AND R. J. PLEMMONS, *Optimality relationships for p-cyclic* SOR, Numer. Math., 56 (1990), pp. 635–643.
[14] R. J. PLEMMONS, *Adjustments by least squares in geodecy using block iterative methods for sparse matrices*, in Proc. Annual U.S. Army Conference on Numerical Analysis and Computers, 1979, pp. 151–186.
[15] M. H. PROTTER AND C. B. MORREY, *A First Course in Real Analysis*, Springer-Verlag, New York, 1977.
[16] R. S. VARGA, *p-cyclic matrices: A generalization of the Young–Frankel successive overrelaxation scheme*, Pacific J. Math., 9 (1959), pp. 617–623.
[17] ———, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.
[18] P. WILD AND W. NIETHAMMER, *Over- and underrelaxation for linear systems with weakly cyclic Jacobi matrices of index p*, Linear Algebra Appl., 91 (1987), pp. 29–52.
[19] D. M. YOUNG, *Iterative methods for solving partial differential equations of elliptic type*, Trans. Amer. Math. Soc., 76 (1954), pp. 92–111.
[20] ———, *Iterative Solution of Large Linear Systems*, Academic Press, New York, 1971.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 121–137, January 1992

010

PREDICTING THE BEHAVIOR OF FINITE PRECISION LANCZOS AND CONJUGATE GRADIENT COMPUTATIONS*

A. GREENBAUM† AND Z. STRAKOS‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. It is demonstrated that finite precision Lanczos and conjugate gradient computations for solving a symmetric positive definite linear system $Ax = b$ or computing the eigenvalues of A behave very similarly to the exact algorithms applied to any of a certain class of larger matrices. This class consists of matrices $\hat{A}$ which have many eigenvalues spread throughout tiny intervals about the eigenvalues of A. The width of these intervals is a modest multiple of the machine precision times the norm of A. This analogy appears to hold, provided only that the algorithms are not run for huge numbers of steps. Numerical examples are given to show that many of the phenomena observed in finite precision computations with A can also be observed in the exact algorithms applied to such a matrix $\hat{A}$.

Key words. conjugate gradient, Lanczos, finite precision arithmetic

AMS(MOS) subject classifications. 65F10, 65F15

1. Background and introduction.

The Lanczos algorithm for computing eigenvalues and eigenvectors of a symmetric matrix and the conjugate gradient algorithm for solving symmetric positive definite linear systems were introduced in the early 1950s by Lanczos [12] and by Hestenes and Stiefel [11], respectively. It was recognized at that time that the algorithms often failed to behave as they would in exact arithmetic due to the effect of rounding errors. Engeli, Ginsburg, Rutishauser, and Stiefel [5], for example, applied the conjugate gradient method (without a preconditioner) to the biharmonic equation and demonstrated that, for a matrix of order n, convergence did not occur until well after step n (although exact arithmetic theory guarantees that the exact solution is obtained after n steps). For this and other reasons, the algorithms did not gain widespread popularity at that time.

With the idea of preconditioning in the conjugate gradient method, interest in this algorithm was revived in the early 1970s, with several important papers appearing, including work by Reid [16] and by Concus, Golub, and O'Leary [4]. Due largely to the personal efforts of Gene Golub and those that he influenced, news of the effectiveness of the conjugate gradient method as an iterative technique spread quickly throughout the scientific computing community, and the algorithm soon became the most popular method for solving symmetric positive definite linear systems. Although the effect of rounding errors on the conjugate gradient algorithm was not well understood, it was observed numerically that (with a good preconditioner) either the method converged before rounding errors had any significant effect on the iterates, or, whatever the effect of roundoff, it was not catastrophic.

Further attempts were made to understand the effect of rounding errors on these two mathematically equivalent algorithms, and why, if they were run for enough steps

* Received by the editors January 2, 1991; accepted for publication (in revised form) July 29, 1991.

† Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, New York 10012 (greenbaum@nyu.edu). This author's work was supported by the Applied Mathematical Sciences Program of the U.S. Department of Energy under contract DE-AC02-76ER03077.

‡ Czechoslovak Academy of Sciences, Institute of Computer Science, Pod vodarenskou vezi 2, 182 07 Praha 8, Czechoslovakia (na.strakos@na-net.ornl.gov). Part of this work was performed while this author was a visitor at the Courant Institute of Mathematical Sciences, New York, NY.

to be significantly affected by roundoff, the effects were not disastrous. Wozniakowski [20] considered a special version of the conjugate gradient (CG) algorithm and showed, essentially, that a finite precision implementation converged at least as rapidly as the method of steepest descent. More precisely, if the linear system to be solved is $Ax = b$, if $e^k = x - x^k$ denotes the error in the kth iterate, and if κ is the condition number of A, then the A-norm of the error at step k satisfies

$$\|e^k\|_A \leqq (1+2\varepsilon)\left(\frac{\kappa-1}{\kappa+1}\right)\|e^{k-1}\|_A + O(\varepsilon), \tag{1}$$

where ε is the unit roundoff of the machine and $O(\varepsilon)$ denotes terms involving the product of ε with the norm of various powers of A, x^k, and a constant. Wozniakowski also gave a bound on the ultimately attainable accuracy using this special version of the CG algorithm:

$$\begin{aligned} \limsup_{k\to\infty} \frac{\|e^k\|_2}{\|x^k\|_2} &\leqq \varepsilon\kappa^{3/2}C, \\ \limsup_{k\to\infty} \frac{\|b-Ax^k\|_2}{\|x^k\|_2} &\leqq \varepsilon\kappa\|A\|C. \end{aligned} \tag{2}$$

Cullum and Willoughby [3] proved a result similar to (1) for a more standard version of the CG algorithm.

The bound (1) is a large overestimate of the actual error, however. If the CG algorithm really converged as slowly as the method of steepest descent, it would seldom be used. Methods such as the Chebyshev algorithm or Richardson's method would be far superior. The bound (1) was derived by considering *individual* steps of the CG algorithm—assuming only that a particular step k is implemented accurately, it follows that the error at step k is reduced at least as much as it would be by a steepest descent step. Yet, an example due to Crowder and Wolfe [2] shows that unless one considers *all* steps of the CG algorithm, one cannot hope to establish much faster convergence than this. That is, if the initial search direction is chosen incorrectly but all other steps of the algorithm are implemented exactly, then convergence may be almost as slow as the method of steepest descent.

The following simple 3 by 3 example presented in [2] illustrates this phenomenon:

$$A = \begin{pmatrix} .1 & & \\ & 1 & \\ & & 1 \end{pmatrix}, \quad r^0 = \frac{1}{\sqrt{6}}\begin{pmatrix} 1 \\ -\sqrt{5} \\ 0 \end{pmatrix}, \quad p^0 = \frac{1}{4\sqrt{30}}\begin{pmatrix} -10\sqrt{5} \\ 14 \\ -3\sqrt{6} \end{pmatrix}.$$

If r^0 is the initial residual for a linear system with coefficient matrix A, and if, instead of taking the initial search direction p^0 to be r^0 we set p^0 as above, then if the remaining CG formulas,

$$\begin{aligned} r^k &= r^{k-1} - \frac{r^{k-1T}p^{k-1}}{p^{k-1T}Ap^{k-1}}Ap^{k-1} \\ p^k &= r^k - \frac{r^{kT}Ap^{k-1}}{p^{k-1T}Ap^{k-1}}p^{k-1}, \quad k = 1, 2, \cdots, \end{aligned}$$

are implemented exactly, then r^k will satisfy

$$r^k = \tfrac{3}{5} Q r^{k-1}, \quad k = 1, 2, \cdots, \quad \text{where } Q = \begin{pmatrix} 1 & 0 & 0 \\ 0 & -1/5 & -2\sqrt{6}/5 \\ 0 & 2\sqrt{6}/5 & -1/5 \end{pmatrix}.$$

Thus, the residual vector is rotated at each step through the angle $\arccos(-1/5)$ and reduced in size by a factor of $\frac{3}{5}$. Similarly, the A-norm of the error is reduced by a factor of $\frac{3}{5}$ at each iteration. This is somewhat faster than the steepest descent bound,

$$\frac{\kappa - 1}{\kappa + 1} = \frac{9}{11},$$

but it is slower than the Chebyshev method, which would converge at an asymptotic rate

$$\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \approx .52.$$

While most work on the CG algorithm was focusing on individual steps, the effects of rounding errors on the Lanczos algorithm were being studied from a more global point of view—considering the effects of roundoff over the entire course of the computation. Paige [13] wrote the Lanczos recurrence equations in matrix form along with the matrix of perturbations resulting from finite precision arithmetic. Using this formulation, he analyzed the loss of orthogonality among the Lanczos vectors. He showed that loss of orthogonality is only in the direction of converged Ritz vectors. From this it followed that at least one eigenpair must converge by step n. Paige later showed also that Ritz values "stabilize" only to points near eigenvalues of A [14]. The implications of these results as far as the rate of convergence of the Lanczos or CG algorithm were not so clear.

Parlett and Scott [15] used Paige's results to suggest a "fix" for the Lanczos algorithm—selective orthogonalization. This requires saving the Lanczos vectors and orthogonalizing against Ritz vectors as they converge. Further work on reorthogonalization strategies, as well as on understanding the behavior of the algorithms without reorthogonalization, was carried out by Simon [17]. He showed that until approximate orthogonality is lost, the tridiagonal matrix generated by a finite precision Lanczos computation is, indeed, the approximate projection of the matrix A onto the span of the Lanczos vectors (which may or may not be the desired Krylov space). Grcar [8] attempted a forward error analysis of the conjugate gradient algorithm. He showed that under a certain assumption, called the "projection property," the coefficients generated in a finite precision CG computation are within about ε of those that would be generated by the exact algorithm, as long as the vectors remain within about $\sqrt{\varepsilon}$ of the exact ones. Thus the initial deviation from exact arithmetic can be analyzed as if the coefficients were given rather than computed at each step.

In [10] a form of backward error analysis was developed for the Lanczos and CG algorithms. There it was shown that finite precision computations, run for no more than some number J steps, generate the same tridiagonal matrices at each step as the exact algorithms applied to a larger matrix $\bar{A}$, having possibly many more eigenvalues than A, but whose eigenvalues all lie within tiny intervals about the eigenvalues of A. A bound on the size of these intervals was derived in terms of the machine precision ε and the bound J on the number of steps. Using this analogy it was possible, in some cases, to derive more interesting bounds on the convergence rate of the CG and Lanczos algorithms.

For example, assuming that the widths of the intervals containing the eigenvalues of $\bar{A}$ are much smaller than the smallest eigenvalue of A, it follows that the condition number of $\bar{A}$ will be approximately the same as that of A. Hence any exact arithmetic error bound in terms of the condition number κ of A will hold, to a close approximation, for finite precision computations; e.g.,

$$\frac{\|e^k\|_A}{\|e^0\|_A} \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k. \tag{3}$$

This error bound was conjectured in [20], but it could not be proved with the approach used there. The bound derived in [10] on the size of these intervals was, unfortunately, a large overestimate. Thus, while bounds such as (3) were established if the machine precision ε was small enough, in many realistic problems the bound on the interval size was too large to provide useful information. A procedure was given to actually compute the matrix $\bar{A}$, however, and numerical computation of this matrix for many examples indicates that the eigenvalues of $\bar{A}$ are actually contained in much smaller intervals than the proven bound would suggest.

While it was proved in [10] that the behavior of finite precision Lanczos and CG computations is *identical* to that of the exact algorithms applied to a *particular* matrix $\bar{A}$, in this paper we demonstrate numerically that it is also very *similar* to that of the exact algorithms applied to *any* matrix, say, $\hat{A}$, which has many eigenvalues spread throughout tiny intervals about the eigenvalues of A. Thus, the qualitative behavior of a finite precision computation can be understood by understanding the behavior of the exact algorithms applied to such matrices $\hat{A}$. The size of the intervals is a modest multiple of the machine precision. It is not clear if this similarity is maintained if the algorithms are run for huge numbers of steps (say, $\gg 10^5$), but for more realistic computations, the similarity is demonstrated. For test problems, we consider a class of matrices introduced in [18]. There the behavior of the finite precision computations was compared with exact arithmetic theory and shown to give surprising results. In this paper we show why this behavior is to be expected. This similarity can also be used to explain the differences observed in [19] between the actual behavior of incomplete Cholesky and modified incomplete Cholesky preconditioners and that predicted by exact arithmetic theory.

Finite precision CG computations for solving an n by n symmetric positive definite linear system $Ax = b$ sometimes fail to converge after n steps, especially when n is small. In such cases, it is demonstrated that exact CG applied to the corresponding large linear system $\hat{A}\hat{x} = \hat{b}$ also requires more than n iterations to converge. More commonly, finite precision CG computations converge in far fewer than n steps, and the same holds for the exact CG algorithm applied to any matrix $\hat{A}$ whose eigenvalues are clustered in tiny intervals about the eigenvalues of A. Frequently, finite precision CG computations go through several steps at which there is only a modest reduction in the error and then at the next step there is a very sharp decrease in the error. This same behavior is observed in the exact CG algorithm applied to matrices $\hat{A}$ whose eigenvalues are distributed in n tight clusters about the eigenvalues of A.

Related to this phenomenon of slow convergence followed by a sudden drop in the error, is the phenomenon of multiple "copies" of eigenvalues appearing in finite precision Lanczos computations. Finite precision Lanczos computations frequently generate several close approximations to some of the eigenvalues of A before finding any close approximations to some of the other eigenvalues. Analogously, depending on how the clusters of the larger matrix $\hat{A}$ are distributed, the exact Lanczos algorithm applied to $\hat{A}$ may find several eigenvalues within some of the clusters before finding any in some of the other clusters. It is demonstrated that the rate of occurrence of multiple "copies" of eigenvalues

in finite precision Lanczos computations with matrix A is very similar to the rate at which the exact Lanczos algorithm applied to $\hat{A}$ finds different eigenvalues within the same cluster.

Sections 2–4 present numerical examples to demonstrate these phenomena. Implications of this analogy are discussed in § 5.

2. Description of numerical experiments. The matrices considered in this paper were introduced in [18] and have eigenvalues of the form

$$\lambda_i = \lambda_1 + \frac{i-1}{n-1}(\lambda_n - \lambda_1)\rho^{n-i}, \quad i = 2, \cdots, n, \quad \rho \in (0,1), \tag{4}$$

where n, λ_1, and $\kappa = \lambda_n/\lambda_1$ are fixed. For most of our experiments we have taken $n = 24$, $\lambda_1 = .1$, $\kappa = 1000$, and $\rho = .4, .6, .8, .9, 1.0$. These eigenvalues are plotted for each value of ρ in Fig. 1. Eigenvalues that are too close to be distinguished on the horizontal axis have been plotted vertically. For the smaller ρ-values, the eigenvalues are very tightly clustered at the lower end of the spectrum. The minimal difference, $\lambda_2 - \lambda_1$, corresponding to $\rho = .4, .6, .8, .9$, and 1.0 is $7.6e-9$, $5.7e-5$, .032, .43, and 4.3, respectively.

The algorithm used in these experiments for solving a symmetric positive definite linear system $Ax = b$ and computing the eigensystem of A is as follows [11], [12]:

Given an initial guess x^0, compute $r^0 = b - Ax^0$, and set $p^0 = r^0$.
For $k = 1, 2, \cdots$
Compute $\alpha_{k-1} = \dfrac{r^{k-1^T} r^{k-1}}{p^{k-1^T} A p^{k-1}}$.
Set $T_{k,k} = \dfrac{1}{\alpha_{k-1}} + \dfrac{\beta_{k-1}}{\alpha_{k-2}}$.
Take $x^k = x^{k-1} + \alpha_{k-1} p^{k-1}$.
Compute $r^k = r^{k-1} - \alpha_{k-1} A p^{k-1}$.
Compute $\beta_k = \dfrac{r^{k^T} r^k}{r^{k-1^T} r^{k-1}}$.
Set $T_{k,k+1} = T_{k+1,k} = \dfrac{\sqrt{\beta_k}}{\alpha_{k-1}}$.
Take $p^k = r^k + \beta_k p^{k-1}$.

The tridiagonal matrix T generated at step k will be denoted $T^{(k)}$, and its eigenvalues are taken as approximate eigenvalues of A (Ritz values). The eigenvectors of A can also be approximated if the previous residual vectors, $r^0, \cdots, r^{k-1}$, have been saved, but we will not discuss the computation of eigenvectors here. When solving linear systems, we will refer to this algorithm as the conjugate gradient method, or CG, while when using it to compute eigenvalues we will refer to it as the Lanczos algorithm. The equivalence of this method to the usual Lanczos process [12] in exact arithmetic is well known, and arguments in [3], [10] establish similar behavior in finite precision arithmetic as well. Numerical experiments with other variants of this algorithm have yielded similar results, as described in [18].

The above algorithm was applied to matrices A of the form

$$A = U \Lambda U^T,$$

where U is a random orthogonal matrix and $\Lambda = \operatorname{diag}(\lambda_1, \cdots, \lambda_n)$ is defined in (4). In all cases, a random right-hand side vector and a zero initial guess were used. Experiments were carried out on a Sun Sparcstation using double precision arithmetic (about 16

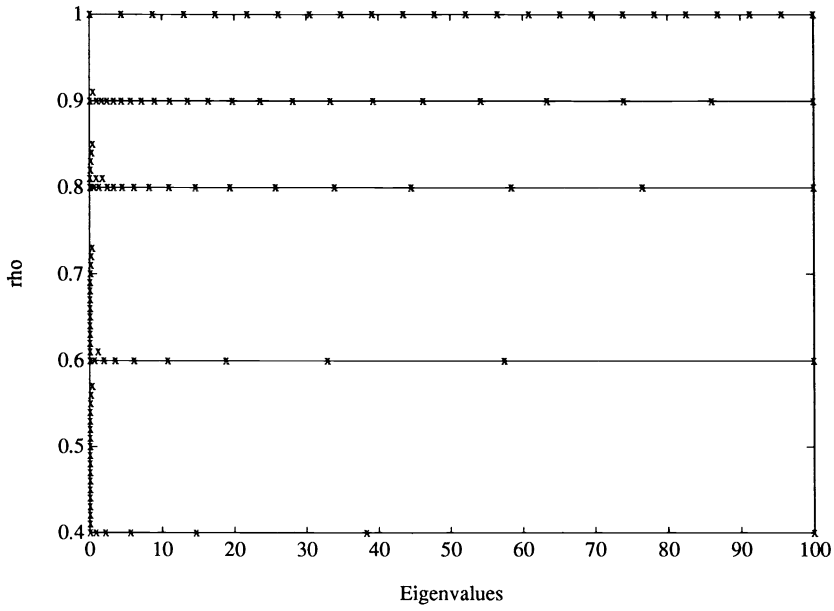


FIG. 1. *Eigenvalue distributions for different ρ-values.*

decimal digits). Most of the experiments were performed using *MATLAB*, making the algorithms relatively simple to implement.

We first compare finite precision computations for solving $Ax = b$ or computing the eigenvalues of A to the exact arithmetic algorithms applied to the same problem. "Exact arithmetic" was simulated by saving all residual vectors and using full reorthogonalization at each step. That is, the formula for r^k becomes

$r^k = r^{k-1} - \alpha_{k-1} A p^{k-1}$
For *kount* = 1, 2,
  For $j = 1, k-1$,

$$r^k = r^k - \frac{r^{k^T} r^j}{r^{j^T} r^j} r^j$$

  Endfor
Endfor

It is shown in [14] that the iterates generated using this modified algorithm do, indeed, resemble those that would be generated by the exact algorithm applied to a slightly different matrix (of the same order) with a slightly different initial vector. Until the size of the vector r^k approaches $\varepsilon \|A\| \max_{j=1,\ldots,k} \|x^j\|$, where ε is the machine precision, the recursively updated r^k is very close to the true residual, $b - Ax^k$ [10].

We also compare the finite precision computations involving the matrix A to "exact arithmetic" (full reorthogonalization) computations involving a larger matrix $\hat{A}$. The matrix $\hat{A}$ was taken to have a total of $11n$ eigenvalues, with eleven eigenvalues uniformly distributed throughout each of n tiny intervals about the eigenvalues of A. Several of the experiments were also performed with a matrix $\hat{A}$ having 21 eigenvalues evenly distributed in each of these same intervals, and the results were indistinguishable when presented in plots such as Figs. 1–9. Most of the experiments were performed with intervals of width 10^{-12}, or approximately

$$50\varepsilon \|A\|,$$

where $\varepsilon = 2^{-52}$ is the unit roundoff of the machine. Some experiments were performed with different size intervals to see the effect of the interval width on the behavior of the algorithms. The finite precision computation for $Ax = b$, with initial guess x^0, or initial residual r^0, was compared to the exact arithmetic computation for $\hat{A}\hat{x} = \hat{b}$, with initial guess $\hat{x}^0$ and initial residual $\hat{r}^0$, where

$$(5) \qquad \hat{A} = \operatorname{diag}(\lambda_{1,1}, \cdots, \lambda_{1,m}, \lambda_{2,1}, \cdots, \lambda_{2,m}, \cdots, \lambda_{n,1}, \cdots, \lambda_{n,m}),$$

$$\lambda_{i,j} = \lambda_i + \frac{j - \frac{m+1}{2}}{m-1} \cdot \delta, \quad j = 1, \cdots, m, \quad m = 11, \quad \delta = 10^{-12} \approx 50\varepsilon \|A\|,$$

$$\hat{r}^0 = \hat{b}, \quad \hat{b}_{i,1} = \hat{b}_{i,2} = \cdots = \hat{b}_{i,m}, \quad \text{and} \quad \sum_{j=1}^{m} (\hat{b}_{i,j})^2 = (U^T b)_i^2, \quad i = 1, \cdots, n,$$

where $\lambda_1, \cdots, \lambda_n$ are the eigenvalues of A and U is the orthonormal matrix of eigenvectors of A.

3. Results of CG computations. Here we show the remarkable similarity between a finite precision CG computation to solve $Ax = b$, with initial residual r^0, and the exact CG algorithm applied to the larger linear system $\hat{A}\hat{x} = \hat{b}$, with initial residual $\hat{r}^0$.

Figure 2(a) shows the convergence of finite precision CG computations applied to the linear system $Ax = b$, for the five different ρ-values listed in the previous section. The right-hand side vector b was taken to have random components, uniformly distributed between 0 and 1, and the initial guess x^0 was taken to be zero. The A-norm of the error at each iteration divided by the A-norm of the initial error

$$\frac{\langle x - x^k, A(x - x^k)\rangle^{1/2}}{\langle x - x^0, A(x - x^0)\rangle^{1/2}}$$

is plotted. (The "exact" solution x was computed as $U\Lambda^{-1}U^T b$.) Note that although exact arithmetic theory ensures that the correct solution is obtained after $n = 24$ steps, the finite precision computation requires significantly more than n steps for some of the ρ-values. For certain values of ρ the computation seems to be considerably more affected by rounding errors than for other values. Convergence slows as ρ goes from .4 to .6 to .8, but then improves as ρ reaches .9 and 1.

For comparison, Fig. 2(b) shows the convergence of the exact CG algorithm applied to the same linear system, with the same initial guess. In contrast to the finite precision computation, there is little difference between the results for $\rho = .8$, .9, and 1.0, with the slowest exact arithmetic convergence rate occurring for $\rho = .9$.

The behavior of the finite precision computations much more closely resembles that of the exact CG algorithm applied to the linear system $\hat{A}\hat{x} = \hat{b}$ (defined in (5)), shown in Fig. 2(c). Here the $\hat{A}$-norm of the error at each iteration divided by the $\hat{A}$-norm of the initial error is plotted. As with the finite precision CG computations, convergence slows as ρ goes from .4 to .6 to .8, but then improves for $\rho = .9$ and 1. The qualitative convergence behavior in Fig. 2(c) is also similar to that of the finite precision CG computations, in that, for certain ρ values, both go through stages at which little improvement is made for several steps and then a sharp drop in the error is seen at a subsequent step.

To see the effect of the interval size on the convergence rate of the exact CG algorithm applied to a matrix $\hat{A}$ with eigenvalues clustered in these intervals, we tried several different interval sizes for the case $\rho = .6$. That is, we considered matrices $\hat{A}$ whose eigenvalues were clustered in intervals of width $\delta = 10^{-13}, 10^{-12}, 10^{-10}$, and 10^{-6} about the eigenvalues

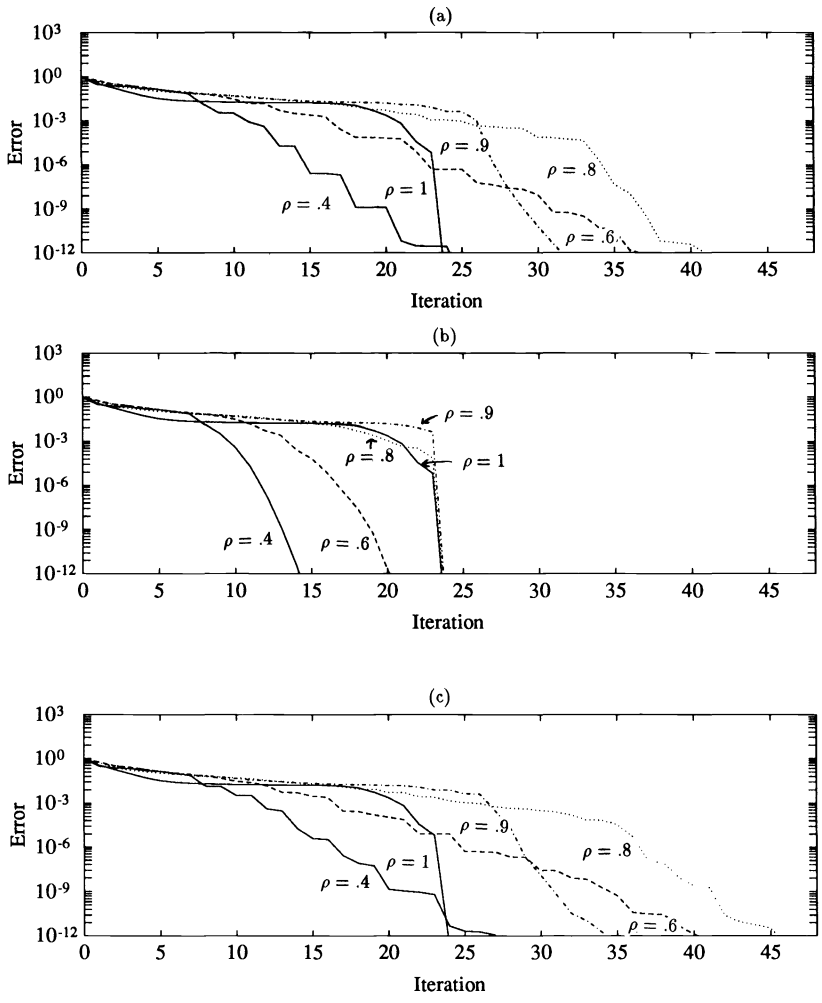


FIG. 2. (a) *Finite precision* CG *for* $Ax = b$. (b) *Exact* CG *for* $Ax = b$. (c) *Exact* CG *for* $\hat{A}\hat{x} = \hat{b}$.

of A. As before, each matrix $\hat{A}$ was taken to have eleven eigenvalues in each interval, uniformly distributed, and the initial residuals $\hat{r}^0$ were set according to (5). Figure 3 shows the convergence of the exact CG algorithm applied to the different problems $\hat{A}\hat{x} = \hat{b}$, along with that of the finite precision computation for $Ax = b$. While the exact computation with the matrix of interval width 10^{-13} most closely resembles the finite precision computation, similar qualitative behavior is reflected in all of these computations, except perhaps the one with interval width 10^{-6}, which is considerably slower to converge. Thus, it appears that a precise estimate of this interval width is not even necessary to predict the qualitative behavior of finite precision CG computations. They resemble exact CG computations for any matrix $\hat{A}$ with eigenvalues spread throughout small intervals about the eigenvalues of A, and the interval size can be anywhere within a rather wide range. The remainder of the CG comparisons will use 10^{-12} as the interval width for the eigenvalues of $\hat{A}$.

While most of our experiments have been performed with very small matrices ($n = 24$), similar phenomena can be observed with larger matrices, for which the CG

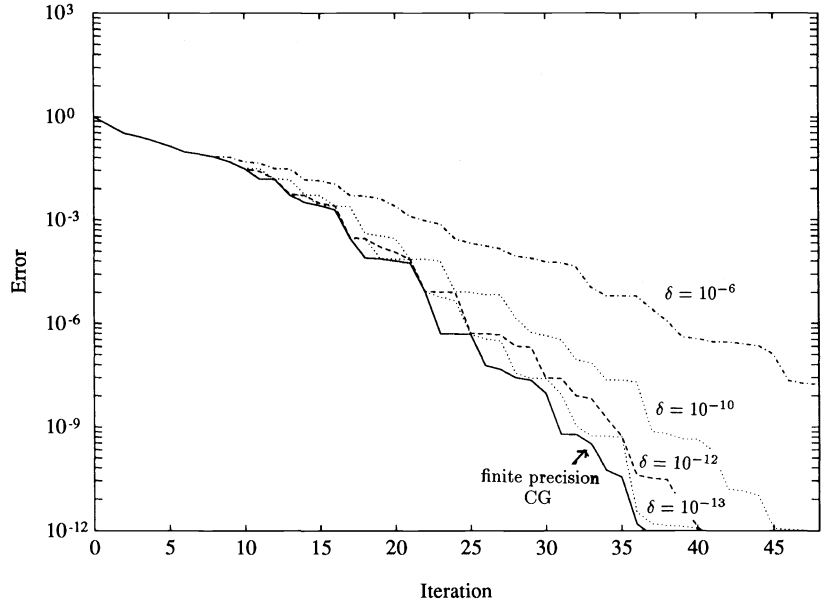


FIG. 3. *Exact* CG *for different interval widths and finite precision* CG *for* $Ax = b$ ($\rho = .6$).

algorithm is more often used. For large matrices, our comparisons with exact arithmetic computations for $\hat{A}$ become quite time- and storage-consuming, however, since $\hat{A}$ is eleven times as large as A and it is necessary to save all residual vectors and reorthogonalize at every step. Still, we have performed one experiment with a matrix A of order 100. The eigenvalues of A are still defined by formula (4), with $n = 100$, $\lambda_1 = .1$, $\kappa = 1000$, and we took $\rho = .8$. Figure 4 shows the convergence of finite precision CG for solving $Ax =$

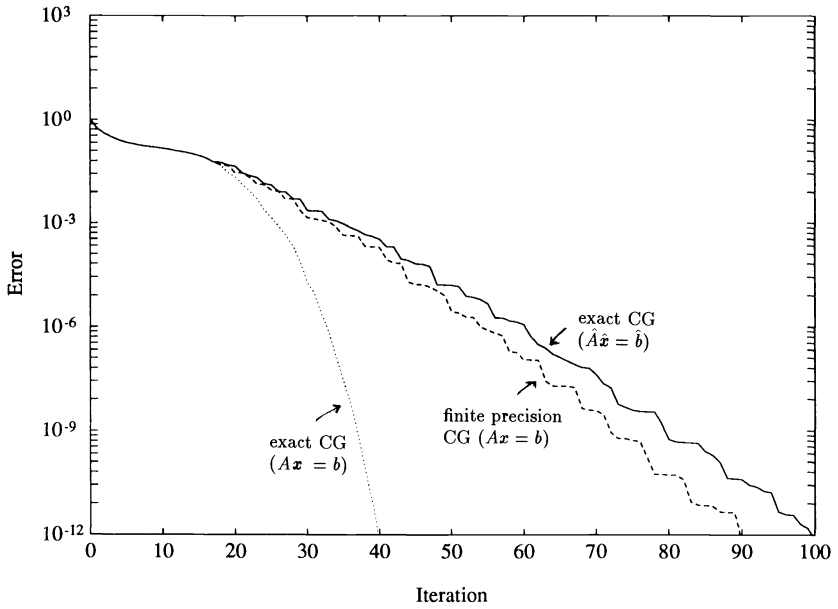


FIG. 4. *Exact and finite precision* CG ($n = 100$, $\rho = .8$).

b, exact CG for solving $Ax = b$, and exact CG for solving the larger problem $\hat{A}\hat{x} = \hat{b}$. Note the close resemblance between the first and last of these curves and the significant differences from the exact CG computation for $Ax = b$. Although this type of behavior is frequently seen in practice, it may not be realized that rounding errors are significantly affecting the convergence rate, since there is no exact arithmetic computation with which to compare.

4. Results of Lanczos computations. Here we show the similarity between the eigenvalue approximations generated at each step of a finite precision Lanczos computation with matrix A and initial vector r^0 and those generated at each step of an exact Lanczos computation with the larger matrix $\hat{A}$ and initial vector $\hat{r}^0$, defined in (5).

Figures 5(a), 6(a), and 7(a) show the eigenvalue approximations generated by a finite precision Lanczos computation with matrix A, for the cases $\rho = .6, .8, 1.0$. Similar results were observed with the other ρ-values. In order to distinguish clustered eigenvalues, we have plotted on the vertical axis not the actual eigenvalues, but the index of each eigenvalue of A, from 1 to n. An approximate eigenvalue that lies, say, $1/r$ of the way between eigenvalue i and eigenvalue $i + 1$ of A, will be plotted on the graph at y-value $i + \frac{1}{r}$. Eigenvalue approximations that are too close to be distinguished on the vertical axis have been plotted horizontally. For clarity, we have plotted the eigenvalue approximations only at every fourth step. In most cases, we see multiple copies of the larger eigenvalues appearing before any close approximations to the smaller, clustered eigenvalues appear. It should be remembered, however, that for the smaller ρ-values, these small eigenvalues are very tightly clustered, and there may be a close approximation to the cluster, even though the individual eigenvalues have not been identified. Only in the case $\rho = 1$ does the finite precision computation find all n eigenvalues by step n.

For comparison, Figs. 5(b), 6(b), and 7(b) show the eigenvalue approximations generated every fourth step of the exact Lanczos algorithm applied to the same matrices, with the same initial vectors. Here no "multiple copies" are observed, and all of the eigenvalues are identified after n steps.

The eigenvalue approximations generated by the finite precision computations much more closely resemble those generated by the exact Lanczos algorithm applied to the matrix $\hat{A}$, with initial vector $\hat{r}^0$, shown in Figs. 5(c), 6(c), and 7(c). In these figures we again see multiple close approximations to the larger eigenvalues appearing before step n, except in the case $\rho = 1$. The rate of appearance of these multiple copies also appears to be similar to that in the finite precision computations with matrix A.

We point out that for $\rho = 1.0$, the effect of roundoff on the Lanczos process is minimal (cf. Figs. 7(a), (b); also Figs. 2(a), (b)). After n iterations the process is "restarted" and it computes all eigenvalues twice in $2n$ iterations. For $\rho < 1.0$, the "restarting" is more frequent and multiple copies of large eigenvalues are computed simultaneously with single approximations to small eigenvalues. It can be observed that if a finite precision Lanczos computation generates a close approximation to each eigenvalue of A at some step, then the error in the corresponding finite precision CG computation drops dramatically at that step. See Figs. 6(a) and 2(a) ($\rho = .8$), between 32 and 36 iteration steps.

Again, to see the effect of the interval width on the eigenvalue approximations generated by an exact Lanczos computation for a matrix $\hat{A}$ with eigenvalues clustered in these intervals, we tried several different interval sizes, for the case $\rho = .6$. That is, we considered matrices $\hat{A}$ whose eigenvalues were clustered in intervals of width $\delta = 10^{-13}$, 10^{-10}, and 10^{-6} about the eigenvalues of A. As before, each matrix $\hat{A}$ was taken to have eleven eigenvalues in each interval, uniformly distributed, and the initial residuals $\hat{r}^0$ were set according to (5). Figures 8(a–c) show the eigenvalue approximations generated

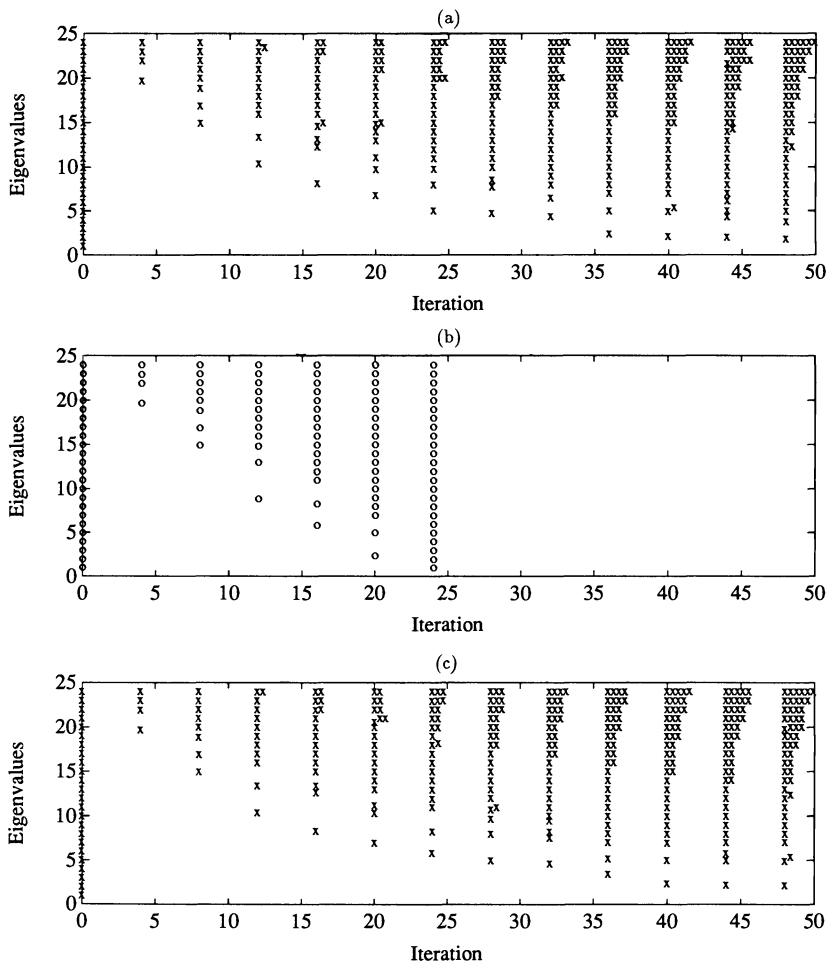


FIG. 5. (a) *Finite precision Lanczos on A* ($\rho = .6$). (b) *Exact Lanczos on A* ($\rho = .6$). (c) *Exact Lanczos on* $\hat{A}$ ($\rho = .6$).

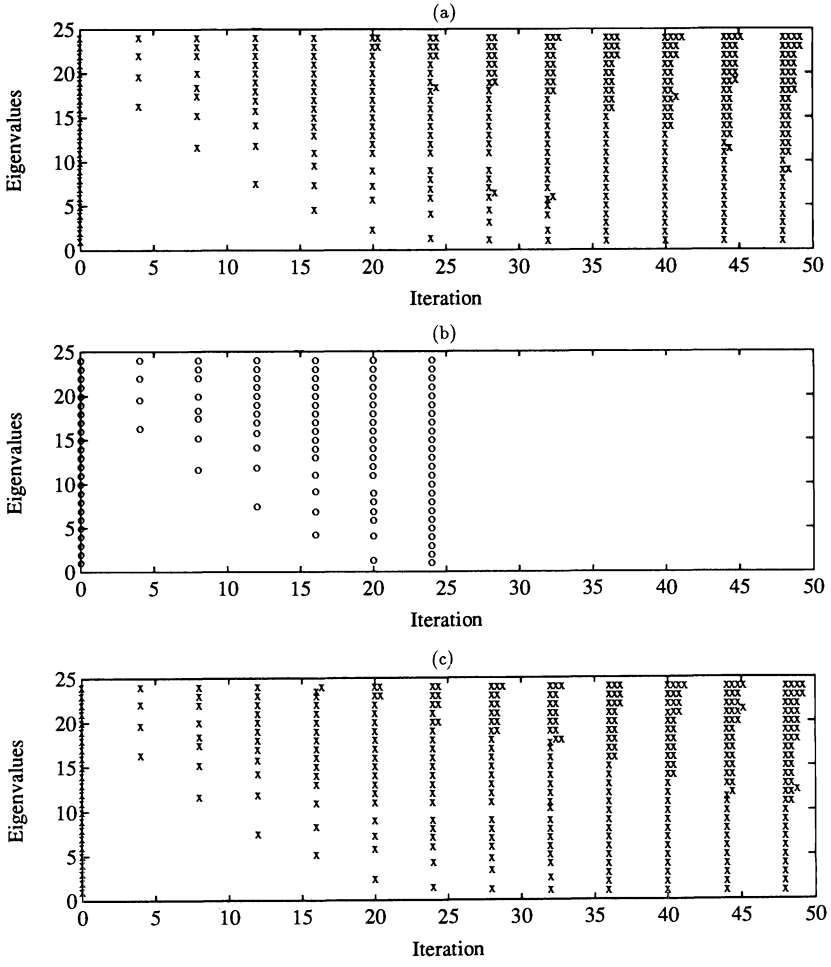


FIG. 6. (a) *Finite precision Lanczos on A* ($\rho = .8$). (b) *Exact Lanczos on A* ($\rho = .8$). (c) *Exact Lanczos on* $\hat{A}$ ($\rho = .8$).

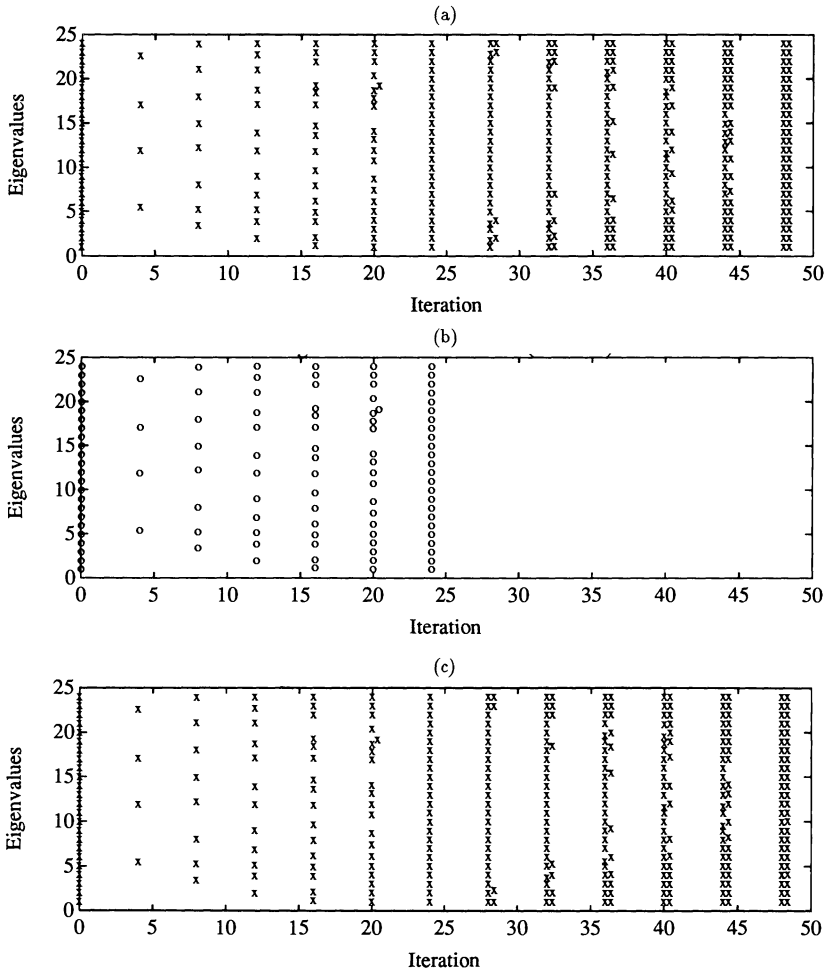


FIG. 7. (a) *Finite precision Lanczos on A* ($\rho = 1$). (b) *Exact Lanczos on A* ($\rho = 1$). (c) *Exact Lanczos on* $\hat{A}$ ($\rho = 1$).

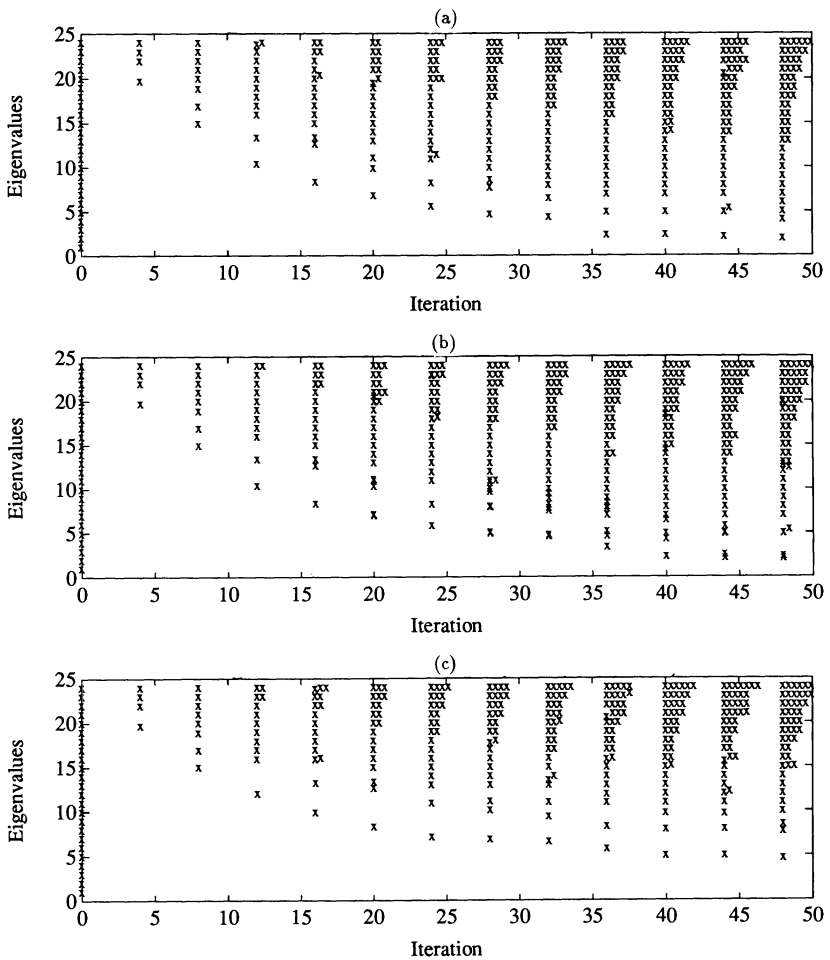


FIG. 8. (a) *Exact Lanczos on* $\hat{A}$ ($\rho = .6$, $\delta = 1.d - 13$). (b) *Exact Lanczos on* $\hat{A}$ ($\rho = .6$, $\delta = 1.d - 10$). (c) *Exact Lanczos on* $\hat{A}$ ($\rho = .6$, $\delta = 1.d - 6$).

at every fourth step by the exact Lanczos algorithm applied to each of these matrices $\hat{A}$. Note the similarity between each of these figures (as well as Fig. 5(c) with $\delta = 10^{-12}$) and Fig. 5(a), showing the eigenvalue approximations generated by a finite precision Lanczos computation for the matrix A. Figure 8(a) ($\delta = 10^{-13}$) shows the closest resemblance to the finite precision computation, while Fig. 8(c) ($\delta = 10^{-6}$) has somewhat more copies of the larger eigenvalues and somewhat fewer approximations to the smaller ones.

Finally, in Figs. 9(a–c), we have plotted results from a larger problem, with $n = 100$, $\rho = .8$. Again, note the similarities between the eigenvalue approximations generated by the finite precision Lanczos computation for the matrix A (Fig. 9(a)) and those generated by the exact arithmetic Lanczos computation for the matrix $\hat{A}$ (Fig. 9(c)). Unlike the exact Lanczos computation for A (Fig. 9(b)), these procedures both generate multiple close approximations to some of the larger eigenvalues before finding any close approximations to some of the smaller ones. The rate at which these multiple approximations appear is also similar in Figs. 9(a) and 9(c).

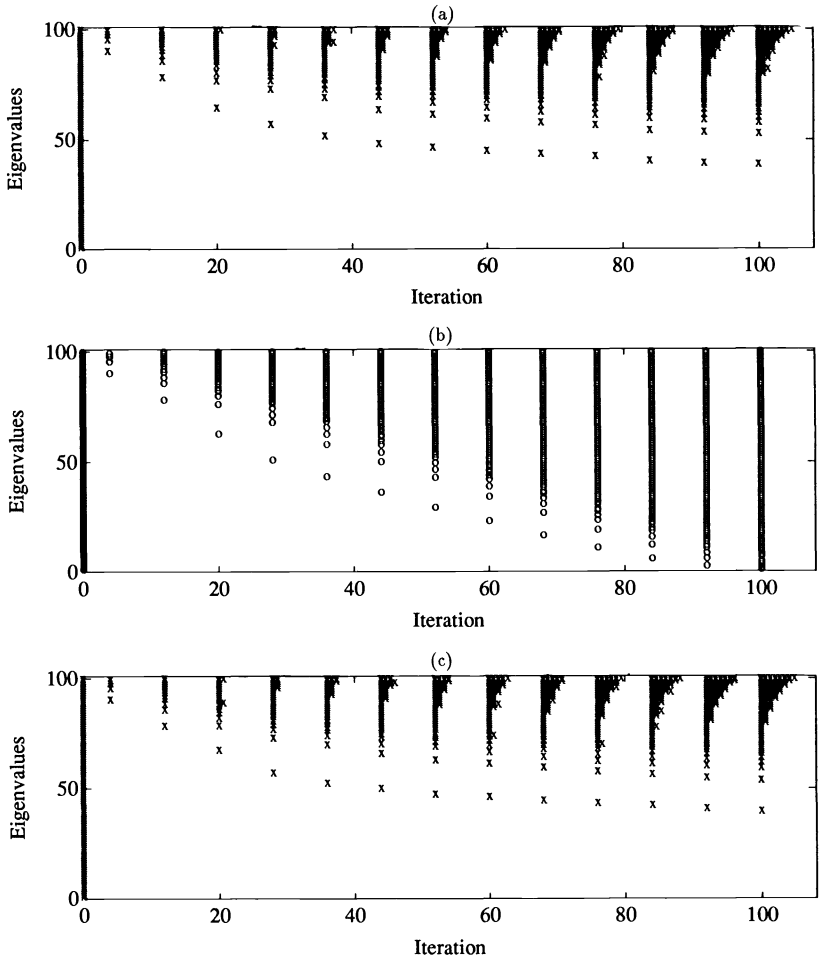


FIG. 9. (a) *Finite precision Lanczos on A* ($\rho = .8$). (b) *Exact Lanczos on A* ($\rho = .8$). (c) *Exact Lanczos on* $\hat{A}$ ($\rho = .8$).

5. Further discussion and open questions. We have demonstrated numerically that the behavior of finite precision Lanczos and CG computations with a matrix A closely resembles that of the exact algorithms applied to matrices $\hat{A}$ that have many eigenvalues spread throughout tiny intervals about the eigenvalues of A. In [10] it was proved that the eigenvalue approximations generated by a finite precision Lanczos computation are *identical* to those generated by the exact algorithm applied to a certain larger matrix $\bar{A}$, and that the A-norm of the error in the equivalent finite precision CG computation is reduced at approximately the same rate as the $\bar{A}$-norm of the error in the exact algorithm. Eigenvalues of the matrix $\bar{A}$ lie in tiny intervals about the eigenvalues of A (provided that the finite precision computation is not run for too many steps), but $\bar{A}$ might have many or only a few eigenvalues in some of these intervals.

Using these analogies, the problem of estimating and bounding the convergence rates of these algorithms in finite precision arithmetic reduces to a problem of estimating or bounding the convergence rates of the exact algorithms applied to certain classes of matrices. If an error bound can be established for the exact algorithm applied to *every* matrix whose eigenvalues lie within these intervals, then it will hold (at least to a close approximation) for the finite precision computation. If an error estimate is good for the exact algorithms applied to every matrix with many eigenvalues spread throughout these intervals, then it will also be a good estimate for the finite precision computation. We will not derive such bounds and estimates here, but it is not difficult to see how they might be derived (as, for example, in (3)), and some examples are given in [10].

A question that is frequently asked is whether a finite precision Lanczos computation eventually finds all eigenvalues of a matrix, or, at least, all well-separated eigenvalues and at least one close approximation to multiple or tightly clustered eigenvalues. Using the analogy developed in [10] between finite precision Lanczos computations run for no more than J steps and the exact algorithm applied to a matrix whose eigenvalues lie within intervals of width, say, δ_J, about the eigenvalues of A, this question can be translated as follows: Is there a J such that the exact Lanczos algorithm applied to every matrix $\tilde{A}$ whose eigenvalues lie within intervals of width δ_J about the eigenvalues of A—with initial vector $\tilde{r}^0$ satisfying

$$\sum_l (\tilde{r}^0, \tilde{u}^{i,l})^2 \approx (r^0, u^i)^2, \qquad i = 1, \cdots, n, \tag{6}$$

where $u^1, \cdots, u^n$ are the eigenvectors of A and $\tilde{u}^{i,l}$, $l = 1, \cdots$, are the eigenvectors of $\tilde{A}$ corresponding to the eigenvalues clustered about λ_i, $i = 1, \cdots, n$—finds at least one eigenvalue from each cluster within J steps? We know of no simple and general sufficient conditions for the existence of such a number J, so this question remains open.

Of course, if the interval widths δ_J could be bounded (with a suitably small bound) independent of J, then it would follow that a finite precision Lanczos computation would eventually find every eigenvalue of A whose eigenvector contained a nonnegligible component in the initial vector. (That is, it would find at least one close approximation to every well-separated eigenvalue and every eigenvalue cluster with a nonnegligible component in the initial vector.) For (6) implies, in this case, that the interval about each eigenvalue has a nonzero weight. From Favard's theorem [6] it would follow that the characteristic polynomials of the tridiagonal matrices generated by a finite precision Lanczos computation were the orthogonal polynomials for a certain measure whose support lies in the union of intervals $\cup_{i=1}^{n} [\lambda_i - \delta, \lambda_i + \delta]$, where δ is the bound on the interval size. But the roots of the orthogonal polynomials are known to converge to all weighted points (see, for example, [1]), so they would converge to at least one point in

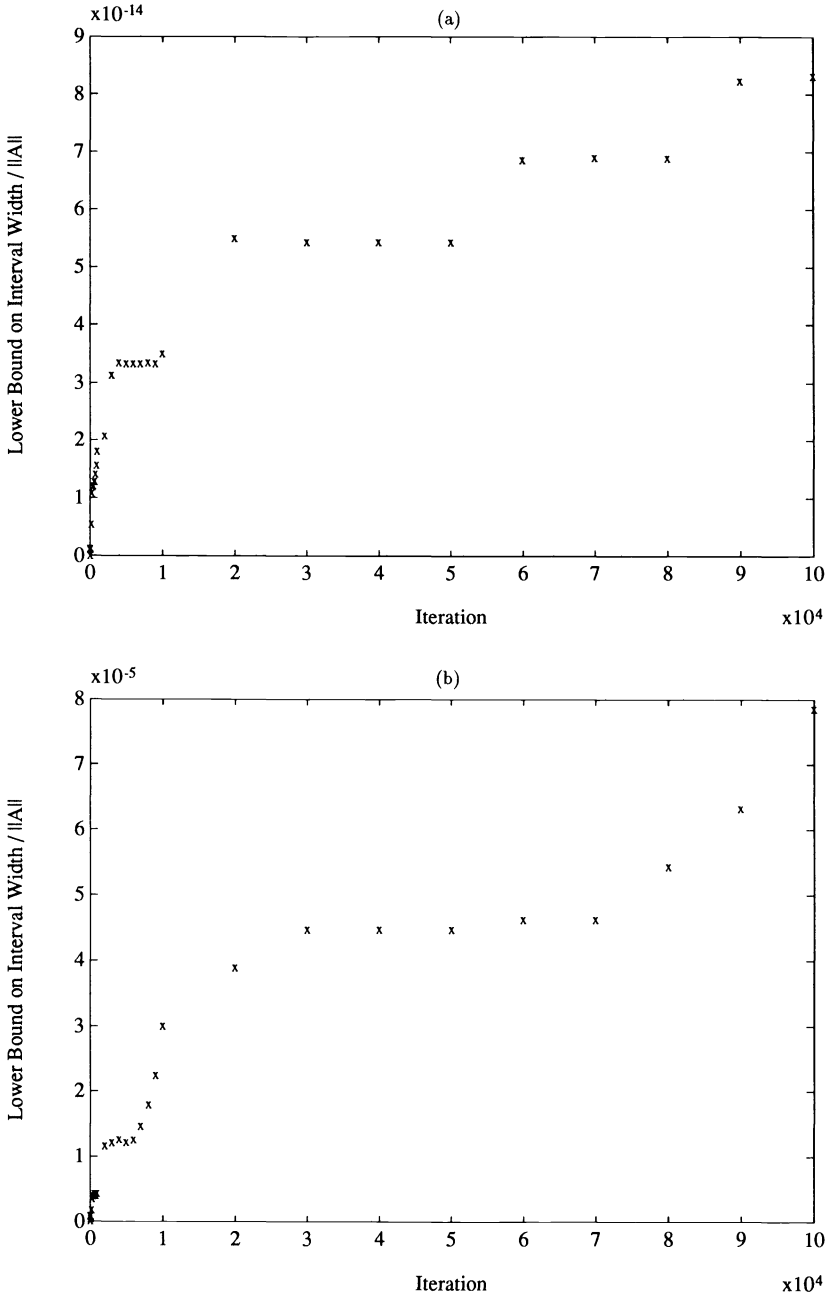


FIG. 10. (a) *Double precision Lanczos* (ρ = .6). *Run for* 10**5 *steps*. (b) *Single precision Lanczos* (ρ = .6). *Run for* 10**5 *steps*.

each of these intervals. Thus, we could then conclude that a finite precision Lanczos computation would eventually find an approximation within δ of each eigenvalue of A.

Whether this interval size can be bounded by a small number δ independent of J is an open question. (Of course, there is some bound that is independent of J. From Gershgorin's theorem and the formulas for the elements of the tridiagonal matrices, it

can be seen that all approximate eigenvalues generated by a finite precision Lanczos computation lie in the interval $[\lambda_{\min} - 2\lambda_{\max}, 3\lambda_{\max}]$. Hence the measure for which the characteristic polynomials are orthogonal has its support in this set. But this is not a very interesting bound.)

To try and determine whether such a bound δ exists, we have run a finite precision Lanczos computation for the case $\rho = .6$, $n = 24$, for 10^5 steps, and we have computed the spread of the eigenvalue approximations clustered about the largest eigenvalue of A. That is, we have taken all eigenvalue approximations that are closer to the largest eigenvalue of A than to any other eigenvalue of A, and we have computed the difference between the largest of these and the second smallest of these. By the interlace theorem, every future tridiagonal matrix will have an eigenvalue greater than the largest of these approximations and an eigenvalue between each pair of these approximations, and so, this difference gives a lower bound on the interval size δ in which the weighted points lie.

Results using double and single precision arithmetic are plotted in Figs. 10(a) and 10(b). For these experiments, we used the standard formulation of the Lanczos algorithm, rather than the CG form presented earlier, to avoid problems with underflow. As can be seen from the figure, this lower bound continues to grow with J at least out to $J = 10^5$, but it is growing *very* slowly. Whether it stops growing at some value significantly less than the distance to the next largest eigenvalue, or whether these eigenvalue approximations would eventually fill the entire Gershgorin interval, we cannot say. This remains an open question.

6. Conclusions. We have found the analogy between finite precision CG/Lanczos computations and the exact algorithms applied to a larger matrix with nearby eigenvalues to be useful in understanding and predicting the behavior of such computations. The proven identity between finite precision computations for A and exact computations for $\bar{A}$ enables one to prove results about finite precision CG/Lanczos computations. The demonstrated similarity between finite precision computations for A and exact computations for matrices $\hat{A}$ enables one to estimate the actual behavior of finite precision CG/Lanczos computations. It provides a nice explanation of the phenomena observed in [18], for example. The proven bound on the size of the intervals containing the eigenvalues of $\bar{A}$ is far from optimal, however, and it is hoped that this might be improved upon. Interesting open questions remain about whether a finite precision Lanczos computation eventually finds all eigenvalues and about how the algorithm behaves if run for huge numbers of steps.

REFERENCES

[1] T. CHIHARA, *An Introduction to Orthogonal Polynomials*, Gordon and Breach, New York, 1978.

[2] H. P. CROWDER AND P. WOLFE, *Linear convergence of the conjugate gradient method*, IBM J. Res. Develop., 16 (1972), pp. 431–433.

[3] J. CULLUM AND R. WILLOUGHBY, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. I. Theory*, Birkhäuser, Boston, 1985.

[4] P. CONCUS, G. H. GOLUB, AND D. P. O'LEARY, *A generalized conjugate gradient method for the numerical solution of elliptic partial differential equations*, in Sparse Matrix Computations, J. R. Bunch and D. J. Rose, eds., Academic Press, New York, 1976.

[5] M. ENGELI, T. GINSBURG, H. RUTISHAUSER, AND E. STIEFEL, *Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems*, Birkhäuser-Verlag, Basel, Stuttgart, 1959.

[6] J. FAVARD, *Sur les Polynomes de Tchebicheff*, C.R. Acad. Sci. Paris, 200 (1935), pp. 2052–2053.

[7] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[8] J. F. GRCAR, *Analyses of the Lanczos algorithm and of the approximation problem in Richardson's method*, Ph.D. thesis, Department of Computer Science, University of Illinois, Urbana, IL, 1981.

[9] A. GREENBAUM, *Comparison of splittings used with the conjugate gradient algorithm*, Numer. Math., 33 (1979), pp. 181–194.

[10] ———, *Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences*, Linear Algebra Appl., 113 (1989), pp. 7–63.

[11] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.

[12] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45 (1950), pp. 225–280.

[13] C. C. PAIGE, *The computation of eigenvalues and eigenvectors of very large sparse matrices*, Ph.D. thesis, Institute of Computer Science, University of London, London, U.K., 1971.

[14] ———, *Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix*, J. Inst. Math. Appl., 18 (1976), pp. 341–349.

[15] B. N. PARLETT AND D. S. SCOTT, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 217–238.

[16] J. K. REID, *On the method of conjugate gradients for the solution of large sparse linear systems*, in Large Sparse Sets of Linear Equations, J. K. Reid, ed., Academic Press, New York, 1971, pp. 231–254.

[17] H. D. SIMON, *The Lanczos algorithm with partial reorthogonalization*, Math. Comp., 42 (1984), pp. 115–136.

[18] Z. STRAKOS, *On the real convergence rate of the conjugate gradient method*, Linear Algebra Appl., 154/156 (1991), pp. 535–549.

[19] H. VAN DER VORST, *The convergence behavior of preconditioned* CG *and* CG-S *in the presence of rounding errors*, in Preconditioned Conjugate Gradient Methods, O. Axelsson and L. Yu Kolotilina, eds., Lecture Notes in Mathematics 1457, Springer-Verlag, Berlin, 1990.

[20] H. WOZNIAKOWSKI, *Roundoff error analysis of a new class of conjugate gradient algorithms*, Linear Algebra Appl., 29 (1980), pp. 507–529.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 138–144, January 1992

011

STOPPING CRITERIA FOR ITERATIVE SOLVERS*

MARIO ARIOLI†‡, IAIN DUFF†§, AND DANIEL RUIZ†

Dedicated to Professor Gene Golub on the occasion of his 60th birthday

Abstract. This paper shows how a theory for backward error analysis can be used to derive a family of stopping criteria for iterative methods and considers particular members of this family. Some theoretical justification is given for why these methods should work well and experimental evidence is presented to justify these claims.

Key words. sparse matrix, sparse equations, iterative methods, stopping criteria, block iterative, Cimmino

AMS(MOS) subject classifications. 65F10, 65F50

1. Introduction. An important aspect of any iterative method for approximating the solution $\mathbf{x}^*$ of the linear equation $\mathbf{Ax} = \mathbf{b}$, where $\mathbf{A}$ is an $n \times n$ real matrix and $\mathbf{b}$ is an n-vector, is to decide at what point to stop the iterations. If the iterations are stopped too early, then a poor answer may be obtained. However, too many iterations may waste computer time without yielding a more accurate answer. We feel it is important to base the stopping criteria on a sound theoretical framework and, in this paper, we use the backward error analysis of Oettli and Prager [7] to develop a family of stopping criteria. We show that this family includes some commonly used criteria and demonstrate that other members of this family at least partially answer the above challenge. We briefly discuss the underlying theory and introduce our criteria in § 2. We illustrate the effect of using our new techniques within a block iterative solver in § 3 and discuss possible extensions to nonlinear systems in § 4. We present some concluding remarks in § 5.

In the following text, $|\mathbf{A}|$ ($|\mathbf{x}|$) will be used to denote the matrix (vector) whose entries are the absolute values of the entries of matrix $\mathbf{A}$ (vector $\mathbf{x}$). Inequalities of the form $\mathbf{E} \geqq \mathbf{0}$ are to be understood componentwise so that all entries of $\mathbf{E}$ are nonnegative.

2. Backward error analysis and stopping criteria. We base our new criteria for stopping iterative methods on the theory of Oettli and Prager [7], which in simple terms can be stated thus:

Let $\hat{\mathbf{x}}$ and $\mathbf{b}$ be any real n-vectors and $\mathbf{A}$ a real $n \times n$ matrix, and define $\mathbf{r}(\hat{\mathbf{x}}) = \mathbf{b} - \mathbf{A}\hat{\mathbf{x}}$. For any matrix $\mathbf{E}$, $\mathbf{E} \geqq 0$, and vector $\mathbf{f}$, $\mathbf{f} \geqq \mathbf{0}$, define

$$\omega = \max_i \frac{|\mathbf{r}(\hat{\mathbf{x}})_i|}{(\mathbf{E}|\hat{\mathbf{x}}| + \mathbf{f})_i}, \tag{2.1}$$

where we take $0/0$ to be 0 and $\rho/0$, for any nonzero ρ, to be ∞. Then if $\omega \neq \infty$, there is a matrix $\delta\mathbf{A}$ and a vector $\delta\mathbf{b}$ with

$$|\delta\mathbf{A}| \leqq \omega\mathbf{E} \quad \text{and} \quad |\delta\mathbf{b}| \leqq \omega\mathbf{f} \tag{2.2}$$

* Received by the editors February 22, 1991; accepted for publication (in revised form) July 9, 1991.

† Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique, 42 Ave G. Coriolis, 31057 Toulouse, France (arioli@cerfacs.fr, isd@ib.rl.ac.uk, and ruiz@japet.cerfacs.fr).

‡ Istituto de Elaborazione della Informazione, Consiglio Nazionale delle Richerche, via S. Maria 46, 56100 Pisa, Italy.

§ Rutherford Appleton Laboratory, Oxon OX11 0QX, England.

such that

$$(\mathbf{A}+\delta\mathbf{A})\hat{\mathbf{x}}=\mathbf{b}+\delta\mathbf{b}. \tag{2.3}$$

Moreover, ω is the smallest number for which such $\delta\mathbf{A}$ and $\delta\mathbf{b}$ exist.

If we therefore compute ω for a given $\mathbf{E}$ and $\mathbf{f}$ we know that, when this quantity is small, we have precisely the solution to a nearby problem.

Our stopping criterion will be to calculate ω from (2.1) and to terminate the iterations when ω reaches a predetermined value or when it is clear that it will not decrease further. Naturally, both the stopping criterion and the associated backward error will be crucially dependent on the choice for $\mathbf{E}$ and $\mathbf{f}$. In the remainder of this section we discuss possible choices and compare them experimentally in the next section.

Skeel [8], [9] chooses $\mathbf{E} = |\mathbf{A}|$ and $\mathbf{f} = |\mathbf{b}|$. When iterative refinement is used in conjunction with Gaussian elimination for full matrices, he can thus obtain a solution that is accurate in a componentwise backward error sense. Arioli, Demmel, and Duff [1] have extended Skeel's work by choosing $\mathbf{f}$ differently to allow the determination of a finite ω when $\mathbf{A}$ is a sparse matrix. We could use the same definition of $\mathbf{E}$ and $\mathbf{f}$ that was used by Skeel, and obtain the componentwise backward error

$$\omega_1 = \max_i \frac{|\mathbf{r}(\hat{\mathbf{x}})_i|}{(|\mathbf{A}||\hat{\mathbf{x}}| + |\mathbf{b}|)_i}. \tag{2.4}$$

Although we examine that possibility experimentally in the next section, we do not think that this is the best choice because it does not reflect the nature of the iterative processes that we are studying. The Skeel choice assumes that the perturbation error in (2.2) and (2.3) is restricted only to the nonzero entries, which was the attractive feature exploited by Arioli, Demmel, and Duff [1] for sparse direct solvers. However, the performance and convergence of most iterative methods is governed by the eigensystem of the iteration matrix or of the original matrix for which a sparse backward error is unknown. Second, the main computation in many iterative methods is the successive multiplication of vectors by the matrix. Powers of $\mathbf{A}$ will be much denser than $\mathbf{A}$ and, in the limit, will be full for irreducible matrices. We would also not expect an iterative method to produce a solution yielding an ω_1 near machine precision, so that we would need to choose some larger threshold to determine termination.

Another choice for $\mathbf{E}$ and $\mathbf{f}$ is to use a normwise backward error by setting $\mathbf{E} = \|\mathbf{A}\|_\infty \mathbf{e}\mathbf{e}^T$ and $\mathbf{f} = \|\mathbf{b}\|_\infty \mathbf{e}$, where $\mathbf{e}$ is a column vector of all ones. This choice gives

$$\omega_2 = \frac{\|\mathbf{r}(\hat{\mathbf{x}})\|_\infty}{\|\mathbf{A}\|_\infty \|\hat{\mathbf{x}}\|_1 + \|\mathbf{b}\|_\infty}. \tag{2.5}$$

We note that the common criterion of using the ratio of the residual to the right-hand side,

$$\omega_3 = \frac{\|\mathbf{r}(\hat{\mathbf{x}})\|_\infty}{\|\mathbf{b}\|_\infty}, \tag{2.6}$$

corresponds to taking a value of $\mathbf{E}$ equal to zero and thus assuming that all backward error is in the right-hand side. This can be misleading in the case when

$$\|\mathbf{b}\|_\infty \ll \|\mathbf{A}\|_\infty \|\mathbf{x}^*\|_1$$

because of bad scaling or cancellation occurring in the product $\mathbf{A}\mathbf{x}^*$ when $\mathbf{x}^*$ has some large entries. In this case, even an $\hat{\mathbf{x}}$ that is a good approximation to $\mathbf{x}^*$ can have a large residual $\mathbf{r}(\hat{\mathbf{x}})$, although ω_2 is small.

We observe that

$$\omega_2 \leqq \omega_1 \quad \text{and} \quad \omega_2 \leqq \omega_3 \tag{2.7}$$

but nothing can be said about the relative values of ω_1 and ω_3. Moreover, ω_1 is row and column scaling independent, whereas ω_3 is dependent on row scaling, and ω_2 is dependent on row and column scaling. However, this is in agreement with the fact that stationary iterative methods depend on the spectral properties of matrices, and these are usually dependent on row and column scaling. In this respect, we prefer ω_2 to ω_3 because ω_2 at least takes into account situations where the ratio $\|\mathbf{A}\|_\infty \|\mathbf{x}^*\|_1 / \|\mathbf{b}\|_\infty$ is large because of bad column scaling. In our experiments, we use the original unscaled data to compute the different criteria. We suggest, however, that it is wise to perform a row scaling of the original problem before applying any iterative or direct solver. In the symmetric case, it would normally be advisable to scale both rows and columns so that symmetry is preserved.

Both the criterion used and the level to which we request that it be reduced should depend on the initial problem. For example, if the operator $\mathbf{A}$ is known exactly (as in the case of some finite difference operators) then it might be appropriate to use ω_3, which only allows perturbations to the right-hand side, whereas for iterative refinement with a direct solver we have already seen [1] that ω_1 is appropriate. We feel that, if little is known about the problem or if the criterion has to be embedded in some general software, the use of ω_2 provides the best compromise and is easy to implement.

Unfortunately, most stationary iterative methods, and also conjugate gradient and Chebychev semi-iterative methods, do not guarantee a priori that ω_1, ω_2, and ω_3 can be driven to machine precision. If we combine a general iterative method with iterative refinement, it is possible to prove under certain conditions [6] that these three criteria will converge to machine precision. However, too many steps of iterative refinement might be necessary, making the overall cost prohibitive (especially for ω_1).

It is reasonable to require that the values of ω_2 and ω_3 are reduced to the same magnitude as the degree of uncertainty in the original data. Let us suppose that the values of the matrix $\mathbf{A}$ and the right-hand side $\mathbf{b}$ come from physical observations and measurements. They will be affected by noise that can be denoted by $\Delta\mathbf{A}$ and $\Delta\mathbf{b}$. If

$$\|\Delta\mathbf{A}\|_\infty \leqq \tau_1 \|\mathbf{A}\|_\infty \quad \text{and} \quad \|\Delta\mathbf{b}\|_\infty \leqq \tau_2 \|\mathbf{b}\|_\infty, \tag{2.8}$$

where τ_1 and τ_2 quantify the level of confidence in the data, then comparing the level of noise on the original data given by (2.8) with the level of accuracy obtained in (2.2), it is not sensible, with our choice for $\mathbf{E}$ and $\mathbf{f}$, to decrease the value of ω_2 to much less than the minimum of τ_1/n and τ_2.

Other choices for $\mathbf{E}$ and $\mathbf{f}$ (and hence ω) that we have considered include replacing the matrix $\mathbf{E} = \|\mathbf{A}\|_\infty \mathbf{e}\mathbf{e}^T$ by $\mathbf{E} = \|\mathbf{A}\|_\infty \mathbf{P}_\mathbf{A}$, where $\mathbf{P}_\mathbf{A}$ is the (0, 1) matrix representing the pattern of $\mathbf{A}$. This might substantially decrease the factor n implicitly included in the infinity norm of $\mathbf{e}\mathbf{e}^T$ but would restrict the backward error to the nonzero entries of $\mathbf{A}$ so that we can still respect the sparsity of the backward error while not imposing such a strict criterion as ω_1. This choice and another of choosing ω as

$$\frac{\|\mathbf{r}(\hat{\mathbf{x}})\|_\infty}{(\tau_2/n\tau_1)\|\mathbf{A}\|_\infty \|\hat{\mathbf{x}}\|_1 + \|\mathbf{b}\|_\infty}$$

in order to balance the weight of the perturbations in $\mathbf{A}$ relative to those in $\mathbf{b}$, might be thought of as having some aesthetic merit. However, we have not found them very competitive on a limited range of experiments and do not discuss them in detail in the experiments that follow.

3. Experimental results. We compare the various stopping criteria on a block iterative method developed by Arioli, Duff, Noailles, and Ruiz [2]. This method is a block generalization of Cimmino's method [5]. It does not require any particular property of the coefficient matrix and so is applicable on a wide range of test problems. We partition the matrix by blocks of rows, solve each resulting underdetermined problem using the direct solution of an augmented system and use the conjugate gradient method to accelerate the convergence. We can also introduce a preconditioning for the conjugate gradient method. It is not relevant to discuss this approach in more detail here and the reader should consult [2] if further information is required.

We illustrate the behaviour of these stopping criteria on two test problems, although the results are typical of those for a far wider range of problems. At each iteration we will compute the three stopping criteria ω_1, ω_2, and ω_3, and compare their relative behaviour. The two experiments use the sparse matrices SHERMAN1 (of order 1000) and LNS511 (of order 511) from the Harwell–Boeing set [4]. SHERMAN1 comes from oil reservoir modelling. LNS511 arises in the solution of linearized Navier–Stokes equations for compressible flow. In both cases, the right-hand side is generated from a given solution vector $\mathbf{x}^*$, where $\mathbf{x}_i^* = n/i$, $i = 1, \cdots, n$. We use these two examples to illustrate some of the main differences between the stopping criteria, both in the case of convergence (SHERMAN1) and in the case of divergence (LNS511).

When there is convergence, as in Fig. 3.1, we first notice that the two stopping criteria ω_2 and ω_3 are much smoother than the componentwise backward analysis criterion (ω_1), which oscillates a lot until a good approximation to the solution is reached. The scaled residual (ω_3) lies between the normwise backward error (ω_2) and the componentwise backward error (ω_1). However, we have observed on some other problems that the curve for ω_3 can cross the curve for ω_1 many times, because of the oscillations in ω_1. This is in agreement with the discussion in (2.7) about the relative values of the three criteria.

On the other hand, when there is failure to converge, as in Fig. 3.2, the normwise backward error (ω_2) oscillates the most, and even increases long before the others do. In the LNS511 problem, for instance, the behaviour of the scaled residual ω_3 does not differ much (at least before iteration 250) from that observed in the first part of the convergence of SHERMAN1 (iterations 1 to 350). This is due to the fact that ω_3 is much less sensitive

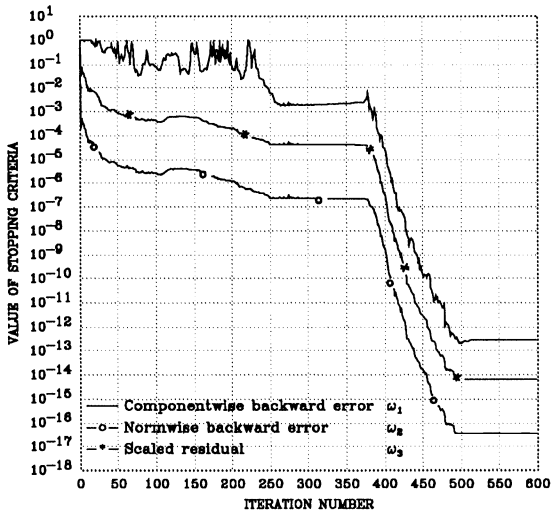


FIG. 3.1. *Behaviour of different stopping criteria on matrix* SHERMAN1.

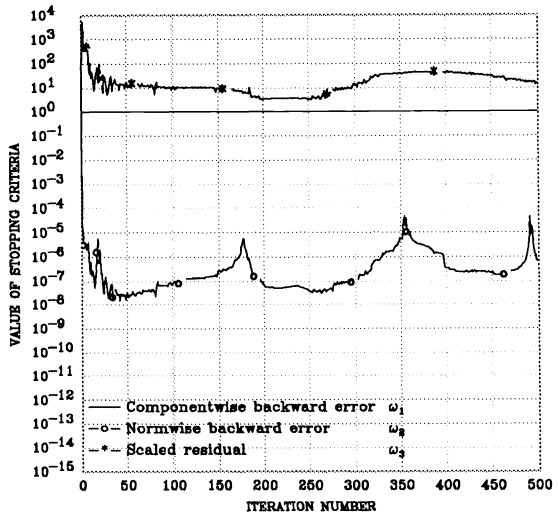


FIG. 3.2. *Behaviour of different stopping criteria on matrix* LNS511.

to variations in $\mathbf{x}$ than ω_2. However, the componentwise criterion ω_1, which stays very close to 1, shows that there are no improvements to the solution. But we must not forget that ω_1 involves the computation of the matrix-vector product $|\mathbf{A}||\mathbf{x}^{(k)}|$ and is thus more costly than the normwise criterion (ω_2), which requires only the infinity norm of the matrix $\mathbf{A}$ and the one-norm of the current iterate.

Note that the detection of a plateau in the behaviour of the convergence of any criterion does not signify the end of the convergence or even the divergence. This can be seen in the SHERMAN1 test problem, for example, where a plateau occurs just before the superlinear convergence of the conjugate gradient acceleration process. Thus it is dangerous to terminate when the ω value becomes nearly constant, so we advocate terminating either on a predetermined value or when an increase in ω_2 is observed.

4. Extensions to nonlinear problems. It is easy to extend the theory of Oettli and Prager [7] to nonlinear systems. More precisely, we can state the following theorem.

THEOREM. *Let* $\hat{\mathbf{E}} \geqq \mathbf{0}$ *and* $\hat{\mathbf{f}} \geqq \mathbf{0}$ *be an* $n \times n$ *matrix and an n-vector, respectively. Let* $\mathbf{F}(\mathbf{x})$ *be a nonlinear function from* R^n *to* R^n, $\mathbf{b}$ *a vector of* R^n, *and* $\tilde{\mathbf{x}}$ *an approximate solution of the problem*

$$\mathbf{F}(\mathbf{x}) = \mathbf{b}. \tag{4.1}$$

There exist $\mathbf{G}$ *and* $\mathbf{g}$ *so that* $\tilde{\mathbf{x}}$ *is a solution of the perturbed problem*

$$\mathbf{F}(\tilde{\mathbf{x}}) + \mathbf{G}\tilde{\mathbf{x}} = \mathbf{b} + \mathbf{g} \tag{4.2}$$

with

$$|\mathbf{G}| \leqq \tilde{\mathbf{E}} \quad \textit{and} \quad |\mathbf{g}| \leqq \hat{\mathbf{f}}, \tag{4.3}$$

if and only if

$$|\mathbf{r}| = |\mathbf{b} - \mathbf{F}(\tilde{\mathbf{x}})| \leqq (\hat{\mathbf{E}}|\tilde{\mathbf{x}}| + \hat{\mathbf{f}}). \tag{4.4}$$

Furthermore, if $\hat{\mathbf{E}} = \omega\mathbf{E}$ *and* $\hat{\mathbf{f}} = \omega\mathbf{f}$, *then the smallest* ω *satisfying the theorem is*

$$\omega = \max_i \frac{|\mathbf{r}_i|}{(\mathbf{E}|\tilde{\mathbf{x}}| + \mathbf{f})_i}. \tag{4.5}$$

Proof. Since

$$\mathbf{r} = \mathbf{b} - \mathbf{F}(\tilde{\mathbf{x}}) = \mathbf{G}\tilde{\mathbf{x}} - \mathbf{g}$$

from (4.1) and (4.2), the "only if" part of the theorem follows immediately.

If we define $\mathbf{R} = \operatorname{diag}(\mathbf{r})$, $\mathbf{D} = \operatorname{diag}(\hat{\mathbf{E}}|\tilde{\mathbf{x}}| + \hat{\mathbf{f}})$ and $\mathbf{S} = \operatorname{diag}(\operatorname{sign}(\tilde{\mathbf{x}}))$, then we can define a matrix $\mathbf{G}$ and vector $\mathbf{g}$ as

$$\mathbf{G} = \mathbf{R}\mathbf{D}^{-1}\hat{\mathbf{E}}\mathbf{S} \quad \text{and} \quad \mathbf{g} = -\mathbf{R}\mathbf{D}^{-1}\hat{\mathbf{f}}.$$

We have, from the hypothesis (4.4), that

$$|\mathbf{R}\mathbf{D}^{-1}| \leqq \mathbf{I}$$

and so $\mathbf{G}$ and $\mathbf{g}$ satisfy conditions (4.3).

Furthermore, we have

$$\begin{aligned}\mathbf{G}\tilde{\mathbf{x}} - \mathbf{g} &= \mathbf{R}\mathbf{D}^{-1}\hat{\mathbf{E}}\mathbf{S}\tilde{\mathbf{x}} + \mathbf{R}\mathbf{D}^{-1}\hat{\mathbf{f}} \\ &= \mathbf{R}\mathbf{D}^{-1}(\hat{\mathbf{E}}|\tilde{\mathbf{x}}| + \hat{\mathbf{f}}) \\ &= \mathbf{R}\mathbf{e},\end{aligned}$$

where $\mathbf{e}$ is the vector of entries equal to one. We have thus proved the "if" part of the theorem.

For nonlinear problems it is more difficult to propose a good choice for $\mathbf{E}$ and $\mathbf{f}$. A possibility is to set the entries in $\mathbf{E}$ to the absolute values of the entries of the Jacobian matrix $\mathbf{J}(\mathbf{x})$ of $\mathbf{F}(\mathbf{x})$ computed at the exact solution $\mathbf{x}^*$ of the problem.

Obviously, this choice is not possible. A more realistic approach might be to set

$$\mathbf{E} = \sup_{\mathbf{x} \in B(\mathbf{x}^*)} \|\mathbf{J}(\mathbf{x})\|_\infty \mathbf{e}\mathbf{e}^T \quad \text{and} \quad \mathbf{f} = \|\mathbf{b}\|_\infty \mathbf{e}$$

with $B(\mathbf{x}^*)$ a ball of centre $\mathbf{x}^*$ in which the algorithm converges. Note that it is not necessary to have the exact value of

$$\sup_{\mathbf{x} \in B(\mathbf{x}^*)} \|\mathbf{J}(\mathbf{x})\|_\infty,$$

but only a good approximation of it.

We do not at the moment have sufficient numerical evidence to support a particular choice for $\mathbf{E}$ or $\mathbf{f}$. We intend to conduct further experiments to determine how best to do this in the nonlinear case.

5. Conclusion. We have established criteria based on backward error analysis for stopping the iterative solution of linear equations and have demonstrated their use in practice. The normwise backward error (ω_2) is always smaller than the other two criteria and is the only one that in our tests reached machine precision when possible. Our proposal is to use the value of ω_2 from (2.5) and to stop when it reaches a predetermined value unless it increases or oscillates significantly. In these cases we terminate immediately. The predetermined value will depend on the problem although in many cases we have found it possible to drive ω_2 to near the machine precision. We recommend that this stopping criterion be used as the standard one in the context of the current efforts to standardize the interface and environment for iterative solvers [3].

REFERENCES

[1] M. ARIOLI, J. W. DEMMEL, AND I. S. DUFF, *Solving sparse linear systems with sparse backward error*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 165–190.

[2] M. ARIOLI, I. S. DUFF, J. NOAILLES, AND D. RUIZ, *A block projection method for sparse matrices*, Report TR/PA/90/31, CERFACS, Toulouse, France, 1990. SIAM J. Sci. Statist. Comput., 13 (1992), to appear.

[3] S. F. ASHBY AND M. K. SEAGER, *A proposed standard for iterative solvers*, Tech. Report 102860, Lawrence Livermore National Laboratory, Livermore, CA, 1990.

[4] I. S. DUFF, R. G. GRIMES, AND J. G. LEWIS, *Sparse matrix test problems*, ACM Trans. Math. Software, 15 (1989), pp. 1–14.

[5] T. ELFVING, *Block-iterative methods for consistent and inconsistent linear equations*, Numer. Math., 35 (1980), pp. 1–12.

[6] M. JANKOWSKI AND H. WOZNIAKOWSKI, *Iterative refinement implies numerical stability*, BIT, 17 (1977), pp. 303–311.

[7] W. OETTLI AND W. PRAGER, *Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides*, Numer. Math., 6 (1964), pp. 405–409.

[8] R. D. SKEEL, *Scaling for numerical stability in Gaussian elimination*, J. Assoc. Comput. Mach., 26 (1979), pp. 494–526.

[9] ———, *Iterative refinement implies numerical stability for Gaussian elimination*, Math. Comp., 35 (1980), pp. 817–832.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 145–161, January 1992

012

NUMERICAL CONSIDERATIONS IN COMPUTING INVARIANT SUBSPACES*

JACK J. DONGARRA†, SVEN HAMMARLING‡, AND JAMES H. WILKINSON§

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. This paper describes two methods for computing the invariant subspace of a matrix. The first method involves using transformations to interchange the eigenvalues. The matrix is assumed to be in Schur form and transformations are applied to interchange neighboring blocks. The blocks can be either one by one or two by two. The second method involves the construction of an invariant subspace by a direct computation of the vectors, rather than by applying transformations to move the desired eigenvalues to the top of the matrix.

Key words. invariant subspaces, eigenvalues, ill-conditioned eigenvalues

AMS(MOS) subject classification. 65F15

1. Introduction. In this paper, we consider the computation of the invariant subspace of a matrix corresponding to some given group of eigenvalues. Potentially, the Schur factorization provides a method for computing such invariant subspaces, with the important numerical property that it provides an orthonormal basis for such spaces. Let us denote the Schur factorization of the real matrix A as

$$A = QTQ^T,$$

where Q is orthogonal and T block upper triangular, with 1×1 and 2×2 blocks on the diagonal, the 2×2 blocks corresponding to complex conjugate pairs of eigenvalues. Since

$$AQ = QT,$$

Q, of course, provides an orthonormal basis for the invariant subspace of the complete eigenvalue spectrum of A. Numerically, Q is a much more satisfactory basis than the eigenvectors and principal vectors of A, which may well be almost linearly dependent. If we partition Q and T as

$$Q = (Q_1\ Q_2), \qquad T = \begin{pmatrix} T_{11} & T_{12} \\ 0 & T_{22} \end{pmatrix},$$

then

$$AQ_1 = Q_1 T_{11},$$

and Q_1 gives an orthonormal basis for the invariant subspace of A corresponding to the eigenvalues contained in T_{11}. It is therefore a common requirement to reorder T so that T_{11} has eigenvalues with some desired property. For example, we might require T_{11} to contain all the stable eigenvalues.

* Received by the editors October 12, 1990; accepted for publication (in revised form) January 22, 1991.

† Department of Computer Science, University of Tennessee, Knoxville, Tennessee 37996-1301, and the Mathematical Sciences Section, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831-8083. The work of this author was supported in part by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy contract DE-AC05-84OR21400, and in part by the Science Alliance, a state-supported program at the University of Tennessee (dongarra@cs.utk.edu).

‡ Numerical Algorithms Group Ltd., Wilkinson House, Jordan Hill Road, Oxford OX2 8DR, United Kingdom (na.hammarling@na-net.ornl.gov).

§ Work on this paper was started as a joint effort with James H. Wilkinson in 1983. After Jim's untimely death, the work lay unfinished for a number of years. The authors recently came across parts of Jim's handwritten manuscript and completed the work.

Unfortunately, unless we know the required group of eigenvalues in advance and accordingly modify the standard shift strategy of the QR algorithm, T_{11} will not normally contain the required eigenvalues on completion of the computation of the Schur factorization. We must therefore perform some further computation to reorder the eigenvalues. Indeed in most applications we perform an initial Schur factorization in order to compute the eigenvalues, which then gives us information on the required grouping.

An example of the application is the computation of matrix functions via the block diagonal form of a matrix. In computing the block diagonal form, it is essential to include "close" eigenvalues in the same diagonal block [3].

To this end, Stewart [9] has described an iterative algorithm for interchanging consecutive 1×1 and 2×2 blocks of the block triangular matrix. The first block is used to determine an implicit QR shift. An arbitrary QR step is performed on both blocks to eliminate the uncoupling between them. Then a sequence of QR steps using the previously determined shift is performed on both blocks. Except in ill-conditioned cases, the two blocks will interchange their positions.

In this paper, we present two other methods for constructing the invariant subspace. The first involves applying transformations directly to interchange the eigenvalues. The second method involves direct computation of the vectors.

2. Interchanging eigenvalues. The reordering of the eigenvalues can be achieved by successively interchanging neighboring blocks in the Schur factor T.

Suppose, in a given T, we have decided to group λ_p, λ_q, λ_r together. We know that there exists a unitary matrix $\tilde{Q}$ such that $\tilde{T} = \tilde{Q}T\tilde{Q}^H$ is still upper triangular but has λ_p, λ_q, λ_r in the first three positions. Such a Q can be readily determined as the product of a finite number of plane rotations. We merely need an algorithm which will enable us to interchange consecutive blocks on the diagonal by means of a plane rotation. Repeated application of this algorithm can then bring any selected set of eigenvalues into the leading positions.

The algorithm we describe could be used on a complex triangular matrix. However, since we are interested here in real matrices, and since complex conjugate eigenvalues will be represented by 2×2 real diagonal blocks, we describe first the algorithm for interchanging two consecutive real eigenvalues.

2.1. Single past single. Suppose λ and μ are in positions p and $p+1$. A similarity rotation in planes p and $p+1$ will alter only rows and columns p and $p+1$ and will retain the triangular form apart from the possible introduction of a nonzero in position $(p+1, p)$. The rotation can be chosen so as to interchange λ and μ while retaining the zero in $(p+1, p)$. Clearly the rotation is determined solely by the 2×2 matrix, which we denote by

$$\begin{pmatrix} \lambda & \alpha \\ 0 & \mu \end{pmatrix}.$$

We have

$$\begin{pmatrix} \lambda & \alpha \\ 0 & \mu \end{pmatrix} \begin{pmatrix} \alpha \\ \mu - \lambda \end{pmatrix} = \mu \begin{pmatrix} \alpha \\ \mu - \lambda \end{pmatrix}, \tag{1}$$

i.e., $(\alpha, \mu - \lambda)^T$ is the eigenvector corresponding to μ. If Q is chosen so that

$$Q \begin{pmatrix} \alpha \\ \mu - \lambda \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}, \tag{2}$$

then

$$Q\begin{pmatrix}\lambda & \alpha\\ 0 & \mu\end{pmatrix}Q^TQ\begin{pmatrix}\alpha\\ \mu-\lambda\end{pmatrix}=\mu Q\begin{pmatrix}\alpha\\ \mu-\lambda\end{pmatrix},$$

and hence, using (2) and dividing by r, we have

$$Q\begin{pmatrix}\lambda & \alpha\\ 0 & \mu\end{pmatrix}Q^T\begin{pmatrix}1\\ 0\end{pmatrix}=\mu\begin{pmatrix}1\\ 0\end{pmatrix}=\begin{pmatrix}\mu\\ 0\end{pmatrix}.$$

This states that the first column of the transformed 2×2 is in the required form. Hence we may write

$$Q\begin{pmatrix}\lambda & \alpha\\ 0 & \mu\end{pmatrix}Q^T=\begin{pmatrix}\mu & \beta\\ 0 & \gamma\end{pmatrix}.$$

Since the trace and Frobenius norm are invariant,

$$\lambda+\mu=\mu+\gamma,\qquad \lambda^2+\mu^2+\alpha^2=\mu^2+\gamma^2+\beta^2,$$

giving

$$\gamma=\lambda \quad\text{and}\quad \beta=\pm\alpha.$$

A rotation giving (2) is defined by

$$\cos\theta=\alpha/r,\qquad \sin\theta=(\mu-\lambda)/r,\qquad r=+[\alpha^2+(\mu-\lambda)^2]^{1/2}, \tag{3}$$

and it will readily be verified that this gives $\beta = +\alpha$.

If the original T has been determined from a matrix A by means of an orthogonal transformation, the matrix defining this transformation must be updated by multiplication with the plane rotations used in the reordering process. Note that in this method, wherever two eigenvalues that we have decided to place in the same group are interchanged, a selected eigenvalue is moved up only past eigenvalues with which it is not to be associated. Moreover, having determined the rotation, we shall apply it to rows and columns p and $p + 1$ but not to the 2×2 itself. There we shall merely interchange λ and μ and do no computation. Moving 1×1 blocks is discussed in [8].

2.2. Single past double. In bringing a selected real eigenvalue to a leading position, we shall, in general, need to pass 2×2 blocks on the diagonal corresponding to complex conjugate pairs. Hence we must be able to interchange a real eigenvalue with a real 2×2 block by means of an orthogonal similarity transformation. Obviously, the transformation is determined by the relevant 3×3 diagonal block which, for simplicity, we write as

$$\left(\begin{array}{cc|c} * & * & b\\ * & * & c\\ \hline 0 & 0 & \lambda_3\end{array}\right)\equiv\left(\begin{array}{c|c} B & b\\ \hline 0\ \ 0 & \lambda_3\end{array}\right). \tag{4}$$

The same principle may be used as in the single past single case. If

$$\begin{pmatrix}x_1\\ x_2\\ 1\end{pmatrix} \tag{5}$$

denotes the eigenvector corresponding to λ_3, then we require a Q such that

$$Q\begin{pmatrix}x_1\\ x_2\\ 1\end{pmatrix}=\begin{pmatrix}r\\ 0\\ 0\end{pmatrix}$$

and then, as before,

$$QTQ^T = \left(\begin{array}{c|cc} \lambda_3 & x & x \\ \hline 0 & x & x \\ 0 & x & x \end{array}\right) = \left(\begin{array}{c|cc} \lambda_3 & x & x \\ \hline 0 & & \\ 0 & & C \end{array}\right). \tag{6}$$

Note that the general principle we are using is the one commonly employed to establish the Schur canonical form by induction. The 2×2 matrix C in the bottom of (6) is not the same as B in (4), but it will, of course, have the same eigenvalues. However, B and C will not, in general, be orthogonally similar.

The matrix Q can be determined as one Householder matrix or as the product of two Givens rotations. Since λ_3 is real and B has complex conjugate eigenvalues, B can have no eigenvalues in common with λ_3; hence, a unique eigenvector of the form (5) will exist. As the two eigenvalues of B approach the real λ_3, their imaginary parts become small, and the eigenvector (5) will have progressively larger components in the first two positions, i.e., the normalized version will have a progressively smaller third component.

2.3. Double past single. When a pair of complex conjugate eigenvalues is included in the selected group, the associated 2×2 diagonal block has to be moved into a leading position on the diagonal. On the way up it will, in general, pass both single eigenvalues and 2×2 blocks with which it is not to be associated. We consider first taking a complex pair past a real eigenvalue. In other words, in terms of the relevant 3×3 matrix, we require an orthogonal Q such that

$$QTQ^T = \left(\begin{array}{c|cc} \lambda_1 & x & x \\ \hline 0 & & \\ 0 & & B \end{array}\right) Q^T = \left(\begin{array}{cc|c} & & x \\ & C & x \\ \hline 0 & 0 & \lambda_1 \end{array}\right).$$

Here the selected eigenvalues are those of B, a complex conjugate pair. The eigenvalues of C will be the same pair, but in general C and B will be different matrices and will not be orthogonally similar. If we think in terms of moving λ_1 to the bottom, we may use much the same principle as before but now we work in terms of a left-hand eigenvector. If

$$y^T T_3 = \lambda y^T \quad \text{with } y^T = (1, y_2, y_3),$$

we determine a Q such that

$$y^T Q = (0, 0, x).$$

Then $Q^T T_3 Q$ has $(0, 0, \lambda_1)$ as its last row, and the objective has been achieved.

2.4. Double past double. Finally, we may need to move a selected 2×2 matrix past an unrelated 2×2. If we denote the relevant 4×4 matrix T_4 by

$$\left(\begin{array}{cc|cc} b_1 & b_2 & x & x \\ b_3 & b_4 & x & x \\ \hline & 0 & c_1 & c_2 \\ & & c_3 & c_4 \end{array}\right) = \left(\begin{array}{c|c} B & X \\ \hline 0 & C \end{array}\right),$$

then we require an orthogonal Q so that

$$\tilde{T}_4 = QT_4Q^T = \left(\begin{array}{c|c} \tilde{C} & \tilde{X} \\ \hline 0 & \tilde{B} \end{array}\right)$$

where B and C have the same eigenvalues as $\tilde{B}$ and $\tilde{C}$, respectively.

The same general principle may be used. We compute generators of the invariant subspace corresponding to C in the form

$$(x,y)=\begin{pmatrix} * & * \\ * & * \\ 1 & 0 \\ 0 & 1 \end{pmatrix}$$

by solving

$$T_4(x,y)=(x,y)C=(x,y)\begin{pmatrix} c_1 & c_2 \\ c_3 & c_4 \end{pmatrix}. \tag{7}$$

This gives us four equations for the four top components in (x, y). If we now determine a Q such that

$$Q(x,y)=\begin{pmatrix} * & * \\ 0 & * \\ 0 & 0 \\ 0 & 0 \end{pmatrix}=\begin{pmatrix} R \\ 0 \end{pmatrix},$$

then QT_4Q^T will be of the required form. Such a Q may be determined as the product of two Householder matrices or four Givens rotations.

To see how $\tilde{C}$ is related to C, we observe that (7) implies that

$$QT_4Q^TQ(x,y)=Q(x,y)C,$$

giving

$$QT_4Q^T\begin{pmatrix} R \\ 0 \end{pmatrix}=\begin{pmatrix} R \\ 0 \end{pmatrix}C,$$

that is,

$$QT_4Q^T\begin{pmatrix} I \\ 0 \end{pmatrix}=\begin{pmatrix} I \\ 0 \end{pmatrix}(RCR^{-1}).$$

This last equation states that the first two columns of QTQ^T are

$$\begin{pmatrix} RCR^{-1} \\ 0 \end{pmatrix},$$

and hence $\tilde{C} = RCR^{-1}$. We shall not, of course, compute $\tilde{C}$ via R!

3. Numerical considerations. In each of the four cases discussed above, we determine either an eigenvector or two independent generators of an invariant subspace.

3.1. Single past single. When taking a single past a single, the formulae giving the components of the vectors are of a particularly simple form. For consistency with the other three cases, the eigenvector in (1) should perhaps have been expressed in the form

$$(\alpha/(\mu-\lambda),\ 1)^T.$$

This emphasizes the fact that when $\mu - \lambda$ is very small compared with α, the first component of the eigenvector is very large, i.e., in the normalized form, the second component is very small. However, in this case λ and μ should almost certainly have been associated together, and we should not be trying to interchange them!

This remark has more force than might be imagined when the full $n \times n$ quasitriangular matrix has been produced from a general matrix A by an orthogonal similarity transformation. In this case, the elements below the diagonal elements are in no sense true zeros. They are at best negligible to working accuracy.

As an example, consider the matrix

$$\begin{pmatrix} 1-\varepsilon & 1 \\ 0 & 1+\varepsilon \end{pmatrix}, \quad \lambda_1 = 1-\varepsilon, \quad \lambda_2 = 1+\varepsilon. \tag{8}$$

A perturbation $-\varepsilon^2$ in the (2, 1) element gives modified eigenvalues $\bar{\lambda}_1 = \bar{\lambda}_2 = 1$, and the matrix is defective. Suppose we are working on a ten-digit computer and $\varepsilon = 10^{-6}$. We may not think of 1 ± 10^{-6} as unduly close, but a perturbation of -10^{-12} gives coincident eigenvalues, and this perturbation is well below the negligible level. If we think in terms of perturbations of order 10^{-10} (i.e., computer noise level), all we can say is that the true eigenvalues are (roughly) in a disk centered on $\lambda = 1$ and of radius 10^{-5}. Thus a perturbation $+10^{-10}$ in (2, 1) gives eigenvalues $1 \pm i(.99)^{1/2}10^{-5}$, while a perturbation of -10^{-10} gives eigenvalues $1 + (1.01)^{1/2}10^{-5}$. To attempt to distinguish between $1 + 10^{-6}$ and $1 - 10^{-6}$, and to interchange them, makes no sense. They have no separate identity, and different rounding errors in the triangularization program giving T might well have led to complex eigenvalues and have a 2×2 block rather than that in (8).

For several moderately close eigenvalues, the remark has even greater force. Thus, if

$$T = \begin{pmatrix} 1-\varepsilon & 1 & 0 \\ & 1 & 1 \\ & & 1+\varepsilon \end{pmatrix},$$

$\lambda_1 = 1 - \varepsilon$, $\lambda_2 = 1$, $\lambda_1 = 1 + \varepsilon$, and $\varepsilon = 10^{-6}$. A perturbation even as small as 10^{-12} in (3, 1) gives three eigenvalues of the form $1 + O(10^{-4})$. This problem is discussed in considerable detail in [10], [12], [13]. Clearly, deciding which eigenvalues should be grouped together cannot be done on the superficial basis of "looking at the separations."

The remarkable fact is that in the single past single case, the $\cos\theta$ and $\sin\theta$ are always given with very low relative errors on a computer with correct rounding or chopping. On such computers, $\mu - \lambda$ is always computed without rounding errors even when severe cancellation takes place. Thus, if

$$\begin{pmatrix} .832567 & .912863 \\ 0 & .832569 \end{pmatrix},$$

we have on a six-digit computer $\mu - \lambda = .000002$, and this has no error. (This will be true even when, e.g., $\lambda = .999999$ and $\mu = 10^1(.100001)$, that is, when close λ and μ have different exponents.) Six-figure floating-point computation using (3) gives

$$\cos\theta = 10^1(.100000), \qquad \sin\theta = 10^{-5}(.219091),$$

and both of these have relative errors on the order of machine precision (10^{-12}) in spite of severe cancellation having taken place. Hence, if we actually do the computation of the 2×2 matrix (in practice, we would not; we could merely insert μ, λ, and α in the appropriate places), we find that the coupled (1, 1), (1, 2) and (2, 2) elements are correct to working accuracy and that the (2, 1) element is well below the negligible level. This is comforting because we shall be applying the transformation to the rest of the matrix.

This is an impressively good result. In many situations, not dissimilar from this, we would have to be satisfied with a matrix that is exactly similar to a T with a perturbation

of order 10^{-6} in its elements and such a matrix could have eigenvalues agreeing with λ and μ in only the first three figures, a disaster from the point of view of effecting an interchange of λ and μ!

3.2. Single past double or double past single. When we turn to the other three cases, the situation is not so simple. Let us consider the algorithm for moving a single past a double. If we denote the eigenvector in (5) by

$$x=(x_1,x_2,1)^T,$$

then $T_3x = \lambda_3 x$ gives

$$\begin{aligned}(t_{11}-\lambda_3)x_1+t_{12}x_2+t_{13}&=0,\\ t_{21}x_1+(t_{22}-\lambda_3)x_2+t_{23}&=0.\end{aligned}\tag{9}$$

The matrix of coefficients $\hat{T}$ of this system of equations is

$$\hat{T}=\begin{pmatrix}t_{11}-\lambda_3 & t_{12}\\ t_{21} & t_{22}-\lambda_3\end{pmatrix},$$

which can be singular only if λ_3 is an eigenvalue of the leading 2×2 matrix of T_3. This possibility is specifically excluded since λ_3 is real and the 2×2 has complex eigenvalues (otherwise we would have triangularized it). When λ_3 is very well separated from the two complex eigenvalues, $\hat{T}$ will be very well conditioned and x_1 and x_2 will not be large; hence, in the normalized version of x the third component will not be small. If we compute the transformation and apply it to the full 3×3 matrix, the top element will be λ_3 to high accuracy, the two complex eigenvalues will be accurately preserved, and the (3, 1) and (3, 2) elements will be negligible. The computed results will be very close to those derived by exact arithmetic.

As λ_3 approaches an eigenvalue of the 2×2 block, however (notice that this means that the imaginary parts of the complex eigenvalues must be small since λ_3 is real, and hence we are really moving towards a triple eigenvalue), the matrix $\hat{T}$ will become progressively more ill conditioned, and in general, x_1 and x_2 will be larger. In the limiting situation, the eigenvector will have a zero third component and will be an eigenvector of the leading 2×2 matrix rather than one corresponding to λ_3 in the 3×3 matrix. The matrix Q is merely a plane rotation in the (1, 2) plane and does not affect λ_3. It is difficult to view this in terms of bringing the (3, 3) element into the leading position! Indeed, we are merely recognizing the fact that the upper 2×2 now has a double real root, and we are triangularizing it. Since the real roots that it has are the same as λ_3, however, the illusion of having moved λ_3 into the leading position is preserved. Thus, if

$$T_3=\left(\begin{array}{cc|c}1 & -1 & 0\\ 1 & -1 & 1\\ \hline 0 & 0 & 0\end{array}\right),\qquad \lambda_1=\lambda_2=\lambda_3=0,$$

the only eigenvector is $(1, 1, 0)^T$; there is no eigenvector of the form $(x, x, 1)^T$. For the rotation in the (1, 2) plane, $\theta = \pi/4$ and the transformed matrix is

$$\begin{pmatrix}0 & -2 & 1/\sqrt{2}\\ 0 & 0 & 1/\sqrt{2}\\ 0 & 0 & 0\end{pmatrix}=\left(\begin{array}{c|cc}\lambda_3 & x & x\\ \hline 0 & & \\ 0 & & C\end{array}\right).$$

The matrix is in the required form, with λ_3 in the leading position, zeros in the first column, and C given by

$$C=\begin{pmatrix}0 & 1/\sqrt{2}\\ 0 & 0\end{pmatrix}, \tag{10}$$

which is similar to the original 2×2, but certainly not orthogonally similar since it has a different Euclidean norm. However, considering how it has come about, it would be perverse to describe it as "bringing λ_3 past the 2×2."

Suppose now that we perturb the (2, 1) entry of the matrix by ε^2 to give

$$\begin{pmatrix}1 & -1 & 0\\ 1+\varepsilon^2 & -1 & 1\\ 0 & 0 & 0\end{pmatrix}, \quad \lambda_1, \lambda_2 = \pm i\varepsilon, \quad \lambda_3 = 0.$$

Then there is an eigenvector x corresponding to λ_3 of the form

$$x^T = (-1/\varepsilon^2, -1/\varepsilon^2, 1) = (-1/\varepsilon^2)(1, 1, -\varepsilon^2).$$

The normalized version of this vector has a very small third component. If we perform our algorithm exactly, it gives a (2, 3) rotation with an angle of order ε^2 (the corresponding matrix is almost the identity matrix) while the (1, 2) rotation has an angle of almost exactly $\pi/4$. The resulting matrix has $\lambda_3 = 0$ in the leading position and the 2×2 matrix C is almost exactly as in (10), but has small perturbations that make its eigenvalues $\pm\varepsilon$.

The simplicity of this discussion is slightly obscured by the use of plane rotations and their introduction of irrationals. If we think in terms of nonorthogonal transformations, then to convert $(1, 1, -\varepsilon^2)$ to (1, 0, 0), we perform a similarity with the unit lower triangular matrix

$$M=\begin{pmatrix}1 & & \\ -1 & 1 & \\ \varepsilon^2 & 0 & 1\end{pmatrix}$$

and obtain as our transformed matrix

$$\left(\begin{array}{c|cc}0 & -1 & 0\\ \hline 0 & 0 & 1\\ 0 & -\varepsilon^2 & 0\end{array}\right).$$

The zero eigenvalue is brought to the top and the eigenvalues $\pm i\varepsilon$ moved to the bottom in a transparently obvious way. When $\varepsilon = 0$, the transformation operates only on row and column 1 and 2, and λ_3 is not involved. Nonetheless, the transformed matrix is

$$\left(\begin{array}{c|cc}0 & -1 & 0\\ \hline 0 & 0 & 1\\ 0 & 0 & 0\end{array}\right),$$

and our "objective" (inappropriate though it is) has been achieved.

The relevance of this discussion to the performance of our algorithm is the following. When we attempt to bring a single past a double having eigenvalues that are fairly close to it, we risk placing too much reliance on the effect achieved by the very small third component in the normalized version of the unique eigenvector corresponding to λ_3. In the analogous single past single case, the solution was determined with considerable accuracy. Here, however, the solution is not nearly as simple. Moreover, when the transformation has been computed, we shall need to apply it to the 3×3 matrix itself, as well

as to the remainder of those relevant rows and columns, since the new 2×2 is not determined in a trivial manner, as were the elements in the single past single case.

Clearly the set of equations must be solved with some care. It is essential that the normalized version of

$$(x_1, x_2, 1), \quad \text{i.e.,} \quad (\tilde{x}_1, \tilde{x}_3, \tilde{x}_3)$$

should be such that

$$(t_{11} - \lambda_3)\tilde{x}_1 + t_{12}\tilde{x}_2 + t_{13}\tilde{x}_3 = \varepsilon_1,$$

$$t_{21}\tilde{x}_1 + t_{22}\tilde{x}_2 + t_{23}\tilde{x}_3 = \varepsilon_2$$

be true with ε_1 and ε_2, which are at noise level relative to the coefficients on the left-hand side (ε_1 and ε_2 would be zero with exact computation). The solution of the system by Gaussian elimination with pivoting ensures just that; it produces x_1 and x_2 with errors that are so correlated that the normalized versions give residuals at noise level.

In place of Gaussian elimination with pivoting, we could use any stable direct method to solve the system, e.g., Givens triangulation. However, if we were to solve the system by an unstable method such as Cramer's rule in standard floating-point arithmetic, we would obtain a computed x_1 and x_2 with errors that are uncorrelated, and the residual corresponding to the normalized vector would not then be at noise level.

Assuming, then, that we have a normalized eigenvector giving negligible residuals, the process is satisfactory. Indeed, it is merely the method of deflation by orthogonal similarity transformations that is used after finding an eigenvector of a general matrix (see, e.g., [11, § 20, Chap. 9]). This is a stable deflation, provided the eigenvector has negligible residuals (independent of its absolute accuracy); the deflated matrix is exactly orthogonally similar to a matrix that differs from the original by a matrix E, which is at noise level relative to it. This is true even when we insert (without computation) the computed eigenvalue in the leading position and zero in the rest of the first column. Such a result is the most we can reasonably expect, though it falls somewhat short of the super-stability of the single past single case.

We have naturally concentrated on the case when we are attempting to move a real eigenvalue λ_3 past a complex conjugate pair, each of which is near λ_3, because numerical stability there needs serious investigation. Of course, when λ_3 is "too close," we usually include all three eigenvalues in the same space. However, when we move a single eigenvalue λ_3 past a complex conjugate pair $\lambda \pm i\mu$ such that $\lambda - \lambda_3$ is not small but μ is small, that pair will be close, and hence, in general, very sensitive to perturbations. The 2×2 block will itself be subjected to a similarity transformation, and small rounding errors will make substantial changes in the eigenvalues. Thus, if we have the matrix

$$\begin{pmatrix} .431263 & .516325 \\ -.000003 & .431937 \end{pmatrix}$$

with the ill-conditioned eigenvalues $.431600 \pm i(.001198)$, and subject it to a plane rotation with angle $\pi/4$, the exact transform gives

$$\begin{pmatrix} .689761 & .258501 \\ -.257827 & .173439 \end{pmatrix}$$

with, of course, precisely the same eigenvalues. If rounding errors produced

$$\begin{pmatrix} .689760 & .258501 \\ -.257827 & .173440 \end{pmatrix}$$

(i.e., changes of -1 and $+1$ in the last figures of the (1, 1) and (2, 2) elements), the eigenvalues become $.431600 \pm i(.001397)$, a substantial change in the imaginary parts. Yet in this example we have used an orthogonal similarity transformation that is favorable to numerical stability. In general, the bypassed matrix will be subjected to a nonorthogonal similarity transformation.

3.3. Double past double. Finally, we turn to the problem of moving a double past a double. Since two pairs of complex conjugate eigenvalues $\lambda_1 \pm \mu_1$ and $\lambda_2 \pm i\mu_2$ are involved, it is not possible for just one eigenvalue in the lower pair to agree with one in the upper pair. If, for convenience, we denote the relevant 4×4 matrix and the invariant subspace by

$$\left(\begin{array}{c|c} T_{11} & T_{12} \\ \hline 0 & T_{22} \end{array}\right) \quad \text{and} \quad \begin{pmatrix} X \\ I \end{pmatrix},$$

respectively, where T_{11}, T_{12}, T_{22}, and X are 2×2 matrices, then we have

$$T_{11}X + T_{12} = XT_{22}.$$

It is well known that if T_{11} and T_{22} have no eigenvalue in common, then this is a nonsingular system.

For the case when T_{11} and T_{22} share an eigenvalue, consider the matrix

$$T = \begin{pmatrix} T_{11} & T_{12} \\ & T_{22} \end{pmatrix} = \left(\begin{array}{cc|cc} 1 & -1 & 0 & 0 \\ 1 & -1 & 1 & 0 \\ \hline & & 0 & 1 \\ & & 0 & 0 \end{array}\right), \quad \lambda_i = 0, \quad i = 1, \cdots, 4.$$

If we try to find an invariant subspace of the form $\binom{X}{I}$, we fail; the elements of X turn out to be infinite. There is no invariant subspace of dimension 2 of the required form. (The particular form chosen for T_{12} is not critical, though, of course, if we take T_{12} to be null, such an invariant subspace does exist with $X = 0$; T is then derogatory.) However,

$$T\begin{pmatrix} I \\ 0 \end{pmatrix} = \begin{pmatrix} T_{11} \\ 0 \end{pmatrix} = \begin{pmatrix} I \\ 0 \end{pmatrix} T_{11},$$

and hence we now have an invariant subspace, which we think of as belonging to T_{11}. But

$$QT_{11}Q^T = \begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix} \quad \text{when} \quad Q = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ -1 & 1 \end{pmatrix} \quad \text{(a rotation)}.$$

Hence

$$T\begin{pmatrix} I \\ 0 \end{pmatrix} Q^T = \begin{pmatrix} I \\ 0 \end{pmatrix} Q^T (QT_{11}Q^T),$$

i.e.,

$$T\begin{pmatrix} Q^T \\ 0 \end{pmatrix} = \begin{pmatrix} Q^T \\ 0 \end{pmatrix}\begin{pmatrix} 0 & -2 \\ 0 & 0 \end{pmatrix} \equiv \begin{pmatrix} Q^T \\ 0 \end{pmatrix} M.$$

But

$$\begin{pmatrix} -\frac{1}{2} & \\ & 1 \end{pmatrix} M \begin{pmatrix} -2 & \\ & 1 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = T_{22}, \quad \text{i.e.,} \quad DMD^{-1} = T_{22}$$

and hence

$$T\begin{pmatrix} Q^T \\ 0 \end{pmatrix} D^{-1} = \begin{pmatrix} Q^T \\ 0 \end{pmatrix} D^{-1}(DMD^{-1}) = \begin{pmatrix} Q^T \\ 0 \end{pmatrix} D^{-1} T_{22},$$

i.e.,

$$T\begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} = \begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} T_{22}.$$

The columns of $\binom{Q^T D^{-1}}{0}$ are orthogonal, but not orthonormal. It looks as though we have an orthogonal basis of an invariant subspace "belonging to T_{22}," but we should not really speak in these terms.

Nevertheless, if we consider

$$T(\varepsilon) = \left(\begin{array}{cc|cc} 1 & -1 & 0 & 0 \\ 1 & -1 & 1 & 0 \\ \hline & & 0 & 1 \\ & & -\varepsilon^2 & 0 \end{array}\right) = \left(\begin{array}{c|c} T_{11} & T_{12} \\ \hline & T_{22}(\varepsilon) \end{array}\right), \quad \lambda_1, \lambda_2 = 0, \quad \lambda_3, \lambda_4 = \pm i\varepsilon,$$

then there is a subspace of the form $\binom{X(\varepsilon)}{I}$ which we could justifiably describe as "belonging to $T_{22}(\varepsilon)$," provided $\varepsilon \neq 0$. The elements of $X(\varepsilon)$ will tend to ∞ as $\varepsilon \to 0$ so that any normalized version of this invariant subspace will have very small components in its lower 2×2 matrix. In fact, since

$$T(\varepsilon)\begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} \equiv T\begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix},$$

we observe that

$$\begin{aligned} T(\varepsilon)\begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} - \begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} T_{22}(\varepsilon) &= T\begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} - \begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} T_{22}(\varepsilon) \\ &= T\begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix} - \begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix}\left(T_{22} + \begin{pmatrix} 0 & 0 \\ -\varepsilon^2 & 0 \end{pmatrix}\right) \\ &= \begin{pmatrix} Q^T D^{-1} \\ 0 \end{pmatrix}\begin{pmatrix} 0 & 0 \\ -\varepsilon^2 & 0 \end{pmatrix}. \end{aligned}$$

When ε is small, this invariant subspace gives negligible residuals "corresponding to $T_{22}(\varepsilon)$."

Can we expect $X(\varepsilon)$ to be $Q^T D^{-1}$ apart from a scale factor? Unfortunately we cannot. In fact, we have

$$\varepsilon^2\begin{pmatrix} X(\varepsilon) \\ I \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 1 & -1 \\ \varepsilon^2 & 0 \\ 0 & \varepsilon^2 \end{pmatrix}.$$

4. A direct method for computing invariant subspaces. In this section, we consider the construction of an invariant subspace by a direct computation of the vectors, rather than by applying transformations to move the desired eigenvalues to the top of the matrix T. We assume that the matrix T is derived from some square general matrix A. Suppose λ_k is the kth eigenvalue along the diagonal of T and T_{kk} is the leading $k \times k$ minor in the matrix T.

If λ_k is a simple eigenvalue, we just solve

$$(T-\lambda_k I)x=0.$$

This gives $x_{k+1}, x_{k+2}, \cdots, x_n = 0$. Next, we take $x_k = 1$ and solve

$$(T_{kk}-\lambda_k I)x=0$$

for $x_{k-1}, x_{k-2}, \cdots, x_2, x_1$, so the vector x will have the form

$$x=(x_1,x_2,\cdots,x_{k-1},1,0,\cdots,0)^T.$$

Now suppose α is a multiple eigenvalue, say a triple, such that

$$\alpha=\lambda_p=\lambda_q=\lambda_r, \qquad (p<q<r).$$

In general, there will be only one eigenvector corresponding to α (unless T is derogatory). First, we find the eigenvector x corresponding to λ_p by solving

$$(T_{pp}-\alpha I)x=0.$$

Next we attempt to find y corresponding to λ_q by taking $y_q = 1$ and attempting to solve

$$(T_{qq}-\lambda_q I)y=0, \quad \text{i.e.,} \quad (T_{qq}-\alpha I)y=0.$$

All is fine until we reach the determination of y_p. We have

$$0y_p+t_{p,p+1}y_{p+1}+\cdots+t_{p,q-1}y_{q-1}+t_{p,q}=0.$$

If we let

$$t_{p,p+1}y_{p+1}+\cdots t_{p,q-1}y_{q-1}+t_{p,q}=d,$$

then

$$0y_p+d=0.$$

If d happens to be zero, then y_p is arbitrary.

It is simplest to take $y_p = 0$. Hence, when $d = 0$, we obtain

$$x=(x_1,x_2,\cdots,x_{p-1},1,0,\cdots,0,0,\cdots,0)^T,$$

$$y=(y_1,y_2,\cdots,y_{p-1},0,y_{p+1},\cdots,y_{q-1},1,\cdots,0)^T.$$

These two vectors are obviously linearly independent. Hence we have two eigenvectors corresponding to α. Both satisfy $(T - \alpha I)x = 0$ and $(T - \alpha I)y = 0$.

If we had taken y_p to be m instead of zero, the solution would have been $y + mx$. This is fine since $y + mx$ is also an eigenvector. We could have chosen $y + mx$ orthogonal to x,

$$x^H(y+mx)=0, \qquad m=(-x^Hy/x^Hx).$$

That the matrix will be derogatory is much less probable than that it will be defective. In fact, even if A were exactly derogatory, T would probably not be, even if it still had exact multiple eigenvalues.

Suppose now $d \neq 0$. To get y_p, we would need to solve

$$0y_p=-d.$$

Hence we cannot get a second eigenvector. Note that if λ_q were $\lambda_p + \varepsilon$ instead of λ_p, we would be solving

$$\varepsilon y_p=-d$$

at this stage, giving an erroneous value of y_p. Obviously, in this case the first p components of y would be essentially $(-d/\varepsilon)x +$ (vector that is not too large). As $\varepsilon \to 0$, the vector y tends to a multiple of x with a relatively negligible amount of interference. In the limit we find that y and x are in exactly the same direction; the last $q - p$ components of y are negligible compared with the rest when q is small, and arbitrarily vanish altogether in the normalized y.

We cannot find a second eigenvector. We can, however, find a vector y such that

$$(T-\lambda_q I)y = dx.$$

Hence the determination of y proceeds as before, from y_q to y_{p+1}, since x is zero in these components. We now have

$$0y_p + t_{p,p+1}y_{p+1} + \cdots + t_{p,q-1}y_{q-1} + t_{p,q} = dx_p = d,$$

so that

$$t_{p,p+1}y_{p+1} + \cdots + t_{p,q-1}y_{q-1} + t_{p,q} = d,$$

giving

$$0y_p = 0.$$

Again y_p is arbitrary, and it is simplest to take y_p to be zero. There are no further problems, and we have

$$x = (x_1, x_2, \cdots, x_{p-1}, 1, 0, \cdots, 0, 0, 0, \cdots, 0)^T,$$

$$y = (y_1, y_2, \cdots, y_{p-1}, 0, y_{p+1}, \cdots, y_{q-1}, 1, 0, \cdots, 0)^T$$

with $(T - \alpha I)x = 0$, $(T - \alpha I)y = dx$, or

$$T(x,y) = (x,y)\begin{pmatrix} \alpha & d \\ & \alpha \end{pmatrix}.$$

Now for the third vector, we shall ignore the possibility of its being derogatory for the moment. We attempt to solve

$$(T_{rr} - \alpha I)z = 0$$

starting with $z_r = 1$. We proceed as usual until we reach z_q. At this stage we have

$$0z_q + t_{q,q+1}z_{q+1} + \cdots + t_{q,r-1}z_{r-1} + t_{q,r} = 0$$

so that

$$t_{q,q+1}z_{q+1} + \cdots + t_{q,r-1}z_{r-1} + t_{q,r} = e.$$

Hence, we solve

$$(T_{rr} - \alpha I)z = ey.$$

This does not affect the components already computed since $y_i = 0$, $(i > q)$.

For convenience we then take $z_q = 0$. We continue until reaching z_p. We now have

$$0z_p + t_{p,p+1}z_{p+1} + \cdots t_{p,r-1}z_{r-1} + t_{p,r} = ey_p,$$

i.e.,

$$t_{p,p+1}z_{p+1} + \cdots + t_{p,r-1}z_{r-1} + t_{p,r} = f.$$

If $f \neq 0$, we would get $z_p = 0$. To avoid this situation, we solve

$$(T_{rr} - \alpha I)z = ey + fx.$$

This does not affect previous components since $x_i = 0$ for $i > p$. The equation for z_p then becomes

$$0z_p = 0.$$

If we take $z_p = 0$ and determine $z_{p-1}, z_{p-2}, \cdots, z_1$, we then have

$$(T - \alpha I)x = 0, \quad (T - \alpha I)y = dx, \quad (T - \alpha I)z = ey + fx,$$

or

$$(11) \qquad (T - \alpha I)(x, y, z) = (x, y, z)\begin{pmatrix} \alpha & d & f \\ & \alpha & e \\ & & \alpha \end{pmatrix} = (x, y, z)T_\alpha.$$

$$x = (x_1, x_2, \cdots, x_{p-1}, 1, 0, 0, \cdots, 0, 0, 0, \cdots, 0)^T,$$

$$y = (y_1, y_2, \cdots, y_{p-1}, 0, y_{p+1}, y_{p+2}, \cdots, y_{q-1}, 1, 0, \cdots, 0)^T,$$

$$z = (z_1, z_2, \cdots, z_{p-1}, 0, z_{p+1}, z_{p+2}, \cdots, z_{q-1}, 0, z_{q+1}, \cdots, z_{r-1}, 1, 0, \cdots, 0)^T.$$

Clearly, x, y, z are linearly independent, and they span the three-dimensional invariant subspace associated with α. They are not orthogonal, in general, but we could develop an orthogonal basis from this. Specifically, if

$$(x, y, z) = (q_1, q_2, q_3)\begin{pmatrix} r_{11} & r_{12} & r_{13} \\ & r_{22} & r_{23} \\ & & r_{33} \end{pmatrix} \equiv Q_3 R_3,$$

then

$$(T - \alpha I)Q_3 R_3 = Q_3 R_3 T_\alpha$$

or

$$(T - \alpha I)Q_3 = Q_3[R_3 T_\alpha R_3^{-1}] = Q_3 M.$$

Q_3 is now an orthogonal basis, and M has α as a triple eigenvalue.

A derogatory matrix will be revealed by zero values among d, e, f. Thus if $d = e = f = 0$, we get three independent eigenvectors, and

$$T(x, y, z) = (x, y, z)\begin{pmatrix} \alpha & & \\ & \alpha & \\ & & \alpha \end{pmatrix}.$$

If $d = f = 0$ and $e \neq 0$, we have

$$T(x, y, z) = (x, y, z)\begin{pmatrix} \alpha & & \\ & \alpha & e \\ & & \alpha \end{pmatrix}.$$

Then we have a linear divisor $(\lambda - \alpha)$ and one quadratic, $(\lambda - \alpha)^2$.

If all computations are exact and T comes from exact computation, then we associate only the eigenvalues that are truly equal, and the vectors obtained in the way we have described are truly independent. In practice, however, T will rarely be an exact matrix. Usually it will have been obtained from a matrix A by, say, the QR algorithm. Even if A had defective eigenvalues, T will usually not have any repeated diagonal elements. A real problem is to decide which diagonal elements to associate together. We may need

to associate eigenvalues that are by no means pathologically close. If we have decided which eigenvalues we wish to associate, then we proceed exactly as described.

So far in this section we have tacitly assumed that T is exactly triangular, but the QR algorithm may give 2×2's on the diagonal. If a 2×2 corresponds to a pair of real eigenvalues, we can get rid of it by an orthogonal transformation. If it corresponds to a complex conjugate pair, we cannot. We assume then that all 2×2's correspond to complex conjugate eigenvalues.

We turn now to the case of 2×2 blocks. If we associate only real eigenvalues in an invariant subspace, there are no real new points. We merely need to know how to get the two components of any of our vectors in the position of a 2×2 block in the matrix. Clearly we solve a 2×2 system of equations for the two components. The technique for getting the generators and the M is unchanged.

Now, consider obtaining a pair of vectors spanning the two-space associated with complex conjugate pairs of eigenvalues, assuming for the moment that we are not associating it with any other eigenvalues. For T, illustrated by

$$T=\begin{pmatrix} * & * & * & * & * & * \\ & * & * & * & * & * \\ & & * & * & * & * \\ & & & * & * & * \\ & & & * & * & * \\ & & & & & * \end{pmatrix},$$

we merely solve the equations

$$T(x_p, x_{p+1}) = (x_p, x_{p+1}) \begin{pmatrix} t_{p,p} & t_{p,p+1} \\ t_{p+1,p} & t_{p+1,p+1} \end{pmatrix} \tag{12}$$

and take

$$(x_p, x_{p+1}) = \begin{pmatrix} * & * \\ * & * \\ * & * \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ \vdots & \vdots \end{pmatrix}$$

so that they are certainly independent. The two back substitutions for determining x_p and x_{p+1} are done as before. We determine $x_i^{(p)}$ and $x_i^{(p+1)}$ from the pair of equations obtained by equating row i on both sides of (12). This gives a well-separated pair of vectors even when the two eigenvectors are close, provided the earlier eigenvalues are well separated from them. Thus, for

$$\begin{pmatrix} 3 & 1 & 2 \\ 0 & 1 & 1 \\ 0 & -10^{-10} & 1 \end{pmatrix} \begin{pmatrix} x_1 & y_2 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} x_1 & y_1 \\ 1 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ -10^{-10} & 1 \end{pmatrix},$$

the eigenvalues are $1 \pm i10^{-5}$; they are close, but well separated from the other eigenvalue $\lambda_1 = 3$. The components x_1 and y_1 satisfy

$$3x_1 + 1 = x_1 - 10^{-10} y_1, \qquad 3y_1 + 2 = x_1 + y_1.$$

To eight decimals, $x_1 = -\frac{1}{2}$ and $y_1 = -\frac{5}{4}$. The vectors are extremely well separated and

$$T(x,y) - (x,y) \begin{pmatrix} 1 & 1 \\ -10^{-10} & 1 \end{pmatrix} = O(10^{-10}).$$

If, when computing the two vectors corresponding to a complex pair, we encounter another 2×2 block, say, in position i, $i+1$, then components i and $i+1$ of x and y are determined by solving a set of four linear equations derived by equating rows i and $i+1$ of (12). This will be a well-conditioned 4×4 system if λ_i, λ_{i+1} are well separated from λ_p, λ_{p+1}.

When we wish to associate $(\lambda_p, \lambda_{p+1})$ with some of the earlier eigenvalues (for which we have already done the back substitution), the solution is quite clear. When we encounter a real eigenvalue λ_i that is to be associated with them, we solve from that point on, namely,

$$T(x_p, x_{p+1}) = (x_p, x_{p+1}) \begin{pmatrix} t_{p,p} & t_{p,p+1} \\ t_{p+1,p} & t_{p+1,p+1} \end{pmatrix} + (x_i)(d_1, d_2)$$

and we chose d_1 and d_2 so that the ith component of x_p and x_{p+1} are zero. This gives us a pair of equations for d_1 and d_2. If λ_p, λ_{p+1}, and λ_i were the only three to be associated, we would have for the invariant three-space

$$T(x_i, x_p, x_{p+1}) = \begin{pmatrix} t_{i,i} & d_1 & d_2 \\ 0 & t_{p,p} & t_{p,p+1} \\ 0 & t_{p+1,p} & t_{p+1,p+1} \end{pmatrix}.$$

If during the back substitution for x_p, x_{p+1} we encounter a pair λ_i, λ_{i+1} that we wish to associate with them, we solve from that point on

$$T(x_p, x_{p+1}) = (x_p, x_{p+1}) \begin{pmatrix} t_{p,p} & t_{p,p+1} \\ t_{p+1,p} & t_{p+1,p+1} \end{pmatrix} + (x_i, x_{i+1}) \begin{pmatrix} d_{i,i} & d_{i,i+1} \\ d_{i+1,i} & d_{i+1,i+1} \end{pmatrix},$$

where the four d's are chosen so as to make components i and $i+1$ of x_p and x_{p+1} equal to zero.

For example, suppose we group (λ_9, λ_8), λ_6, (λ_4, λ_3) where (λ_9, λ_8) and (λ_4, λ_3) are complex pairs. We have

$$(x_3, x_4, x_6, x_8, x_9) = \left(\begin{array}{ccc|cc} * & * & * & * & * \\ * & * & * & * & * \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & * & * & * \\ 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 \end{array}\right)$$

and finally

$$T(x_3, x_4, x_6, x_8, x_9) = (x_3, x_4, x_6, x_8, x_9) \begin{pmatrix} t_{33} & t_{34} & d_{36} & d_{38} & d_{39} \\ t_{43} & t_{44} & d_{46} & d_{48} & d_{49} \\ & & t_{66} & d_{68} & d_{69} \\ & & & t_{88} & t_{89} \\ & & & t_{98} & t_{99} \end{pmatrix}.$$

The elements named d_{36} and d_{46} would have been determined when computing x_6 when we reached rows 3 and 4; the elements d_{68} and d_{69} would have been determined when computing x_8 and x_9 when we reached element 6; and the elements d_{48}, d_{49}, d_{38}, d_{39} would have been determined when we reached elements 4 and 3.

If we have made a good decision about our grouping, rows of the vectors will not be large, though this would not be sufficient to decide that the grouping is complete. First, there may be some λ_i which should also be associated with these five. Second, the vectors x_3, x_4, x_6, x_8, and x_9 might not be as linearly independent as we would like.

Other approaches have been suggested for computing the invariant subspace directly; see [4]–[6]. These are likely to be more stable but more expensive to compute.

5. Conclusions. The methods described in § 2 have been improved and generalized by Ng and Parlett [7] and implemented in LAPACK [1]. The LAPACK implementation includes tolerance checks and scaling to ensure numerical stability [2]. This is essentially achieved by not swapping blocks that are regarded as being too close.

We have discussed numerical issues concerned with the computation of invariant subspaces and proposed two methods related to their computation. The method discussed for swapping diagonal blocks can readily by extended to the generalized eigenvalue problem.

REFERENCES

[1] E. ANDERSON, Z. BAI, C. BISCHOF, J. DEMMEL, J. DONGARRA, J. DUCROZ, A. GREENBAUM, S. HAMMARLING, A. MCKENNEY, AND D. SORENSEN, LAPACK: *A portable linear algebra library for high-performance computers*, in Supercomputer 90, Institute of Electrical and Electronics Engineers Press, New York, 1990.

[2] Z. BAI, J. DEMMEL, AND A. MCKENNEY, *On the conditioning of the nonsymmetric eigenproblem: Theory and software*, Computer Science Department Tech. Report CS-89-86, University of Tennessee, Knoxville, TN, October 1990. (LAPACK Working Note 13.)

[3] C. BAVELY AND G. W. STEWART, *An algorithm for computing reducing subspaces by block diagonalization*, SIAM J. Numer. Anal., 16 (1979), pp. 359–367.

[4] J. DEMMEL, *Three methods for refining estimates of invariant subspaces*, Computing, 38 (1987), pp. 43–57.

[5] B. KÅGSTRÖM AND A. RUHE, *Algorithm* 560: *An algorithm for numerical computation of the Jordan normal form of a complex matrix*, ACM Trans. Math. Software, 6 (1980), pp. 398–419.

[6] ———, *An algorithm for numerical computation of the Jordan normal form of a complex matrix*, ACM Trans. Math. Software, 6 (1980), pp. 437–443.

[7] K. C. NG AND B. N. PARLETT, *Programs to swap diagonal blocks*, Center for Pure and Applied Mathematics Tech. Report 381, University of California, Berkeley, CA, 1988.

[8] A. RUHE, *Perturbation bounds for means of eigenvalues and invariant subspaces*, BIT, 10 (1970), pp. 343–354.

[9] G. W. STEWART, *Algorithm* 406: HQR3 *and* EXCRNG: *Fortran subroutines for calculating and ordering eigenvalues of a real upper Hessenberg matrix*, ACM Trans. Math. Software, 2 (1976), pp. 275–280.

[10] H. J. SYMM AND J. H. WILKINSON, *Realistic error bounds for a simple eigenvalue and its associated eigenvector*, Numer. Math., 35 (1980), pp. 113–126.

[11] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.

[12] ———, *Sensitivity of eigenvalues*, Utilitas Math., 25 (1984), pp. 5–76.

[13] ———, *Sensitivity of eigenvalues* II, Utilitas Math., 30 (1986), pp. 243–275.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 162–175, January 1992

013

BACKWARD ERROR AND CONDITION OF STRUCTURED LINEAR SYSTEMS*

DESMOND J. HIGHAM† AND NICHOLAS J. HIGHAM‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Existing definitions of backward error and condition number for linear systems do not cater to structure in the coefficient matrix, except possibly for sparsity. The definitions are extended so that when the coefficient matrix has structure the perturbed matrix has this structure too. It is shown that when the structure comprises linear dependence on a set of parameters, the structured componentwise backward error is given by the solution of minimal ∞-norm to an underdetermined linear system; an explicit expression for the condition number in this linear case is also obtained. Applications to symmetric matrices, Toeplitz matrices and the least squares problem are discussed and illustrated through numerical examples.

Key words. componentwise backward error, condition number, underdetermined system, symmetric matrix, Toeplitz matrix, least squares problem, augmented system

AMS(MOS) subject classification. 65F99

1. Introduction.

A backward error of an approximate solution y to a square linear system $Ax = b$ is a measure of the smallest perturbations ΔA and Δb such that

$$(A + \Delta A)y = b + \Delta b.$$

Backward error has two distinct uses. First, it can be compared with the size of any uncertainty in the data A and b to ascertain whether y solves a problem sufficiently close to the original one. Second, by invoking perturbation results a bound can be obtained on the forward error $y - x$ in terms of the backward error and an appropriate condition number.

Two classes of backward error definition are in current use, corresponding to different ways of measuring the size of the perturbations ΔA and Δb. The most familiar is the *normwise backward error*

$$\eta(y) = \min\{\epsilon : (A + \Delta A)y = b + \Delta b, \quad \|\Delta A\| \le \epsilon\|E\|, \quad \|\Delta b\| \le \epsilon\|f\|\}, \tag{1.1}$$

in which $\|\cdot\|$ denotes any vector norm and the corresponding subordinate matrix norm, and the matrix E and the vector f are arbitrary. Rigal and Gaches [19] derive the explicit expression

$$\eta(y) = \frac{\|r\|}{\|E\|\|y\| + \|f\|}, \tag{1.2}$$

where $r = b - Ay$; they also show that the minimum in (1.1) is achieved by the perturbations

$$\Delta A_{\min} = \frac{\|E\|\|y\|}{\|E\|\,\|y\| + \|f\|} r z^T, \qquad \Delta b_{\min} = -\frac{\|f\|}{\|E\|\,\|y\| + \|f\|} r, \tag{1.3}$$

* Received by the editors September 28, 1990; accepted for publication (in revised form) April 29, 1991.

† Department of Mathematics and Computer Science, University of Dundee, Dundee, DD1 4HN, Scotland (na.dhigham@na-net.ornl.gov). Part of this author's work was done while he was in the Department of Computer Science, University of Toronto, Toronto, Ontario, Canada. The work of this author was supported by the Natural Sciences and Engineering Research Council of Canada and the Information Technology Research Centre of Ontario.

‡ Department of Mathematics, University of Manchester, Manchester, M13 9PL, England (na.nhigham@na-net.ornl.gov). The work of this author was started during his visit to the Department of Computer Science, University of Toronto, Toronto, Ontario, Canada.

where z is a vector dual to y, that is,

$$z^T y = \|z\|_D \|y\| = 1, \quad \text{where } \|z\|_D = \max_{v \neq 0} \frac{|z^T v|}{\|v\|}.$$

For the particular choice $E = A$ and $f = b$, $\eta(y)$ is called the *normwise relative backward error*. The classic backward error analyses of Wilkinson [24], [25] for linear equation solvers provide bounds on the normwise relative backward error of a computed solution.

A more stringent measure of backward error results if the components of the perturbations ΔA and Δb are measured individually, rather than together in a norm. This way we obtain the *componentwise backward error*

$$\omega(y) = \min\{\epsilon : (A + \Delta A)y = b + \Delta b, \ |\Delta A| \leq \epsilon E, \ |\Delta b| \leq \epsilon f\}, \tag{1.4}$$

where $E \geq 0$ and $f \geq 0$ contain arbitrary tolerances, and inequalities between matrices hold componentwise. The current trend of using componentwise error analysis and perturbation theory began with the 1979 paper of Skeel [20]. However, componentwise backward error was introduced and studied much earlier in a 1964 paper of Oettli and Prager [18]. Oettli and Prager obtained the explicit formula

$$\omega(y) = \max_i \frac{|r_i|}{(E|y| + f)_i}, \tag{1.5}$$

in which $\xi/0$ is interpreted as zero if $\xi = 0$ and infinity otherwise. (A short proof of (1.5) is given in [15] and [20].) Perturbations that achieve the minimum in (1.4) are

$$\Delta A_{\min} = D_1 E D_2, \qquad \Delta b_{\min} = -D_1 f, \tag{1.6}$$

where $D_1 = \operatorname{diag}(r_i/(E|y| + f)_i)$ and $D_2 = \operatorname{diag}(\operatorname{sign}(y_i))$.

One reason for the current interest in componentwise backward error is that it provides a more meaningful measure of stability than the normwise version when the elements of A and b vary widely in magnitude. The most common choice of tolerances is $E = |A|$ and $f = |b|$, which yields the *componentwise relative backward error*. For this definition, zeros in A and b force zeros in the corresponding entries of ΔA and Δb in (1.4), and so if $\omega(y)$ is small, then y solves a problem that is relatively close to the original one and has the same sparsity pattern. Another attractive property of the componentwise relative backward error is that it is insensitive to the scaling of the system: if $Ax = b$ is scaled to $(S_1 A S_2)(S_2^{-1} x) = S_1 b$, where S_1 and S_2 are diagonal, and y is scaled to $S_2^{-1} y$, then ω remains unchanged. Recent work that makes use of componentwise backward error includes [1], [2], [13], [15], [16].

There are situations where even the componentwise backward error is not entirely appropriate, because it does not respect any structure (other than sparsity) in A or b. For example, if A is a Toeplitz matrix and $\eta(y)$ and $\omega(y)$ are small, it does not necessarily follow that y solves a nearby Toeplitz system, since ΔA in (1.1) or (1.4) is not required to be a Toeplitz matrix. Indeed, $\Delta A_{\min}$ in (1.3) or (1.6) is clearly not Toeplitz in general. Similar remarks can be made about condition numbers: the standard condition numbers are derived without requiring that perturbations preserve structure, hence they generally exceed the actual condition number for a linear system subject to structured perturbations. See Bunch [4] or Van Dooren [22] for a more detailed discussion of the desirability of preserving matrix structure in definitions of backward error; Van Dooren also discusses structured condition numbers and describes various structured linear algebra problems that arise in signal processing.

In this work, we extend the notions of componentwise backward error and condition number to allow for dependence of the data on a set of parameters. In §2 we define the *structured componentwise backward error* and show how to compute it when the dependence is linear. In §3 we define a *structured condition number* that measures the sensitivity of a linear system to structured perturbations measured componentwise. We derive an explicit expression for the condition number in the case where the parametrization of the data is linear.

In §4 we examine applications involving symmetric matrices, Toeplitz matrices, and the least squares (LS) problem. In particular, we explain why, when perturbations to a symmetric matrix are measured using the 2-norm, it makes little difference to the backward error or condition number whether the perturbations are required to preserve symmetry or not. For most algorithms for solving structured linear systems little is known about the size of the structured backward error of the computed solution. Some insight can be gained by computing the structured backward error in specific instances, as we illustrate with numerical examples in §5. We give some suggestions for further work in §6.

2. Structured componentwise backward error. Consider an approximate solution y to the linear system $Ax = b$, where $A \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$. Suppose A belongs to a set $S \subseteq \mathbb{R}^{n\times n}$ whose members depend on t real parameters ($t \leq n^2$); we write this dependence as $A = A[p]$ where $p \in \mathbb{R}^t$. We assume that b does not exhibit any such structure, although the analysis below could be modified to allow for structure in b. (An example of a problem where b has structure is the Yule–Walker Toeplitz system [9, p. 184] in which b depends on the same parameters as A.)

Given nonnegative vectors of tolerances $g \in \mathbb{R}^t$ and $f \in \mathbb{R}^n$, we define the *structured componentwise backward error*

$$\mu(y) = \min\{\epsilon : (A+\Delta A)y = b+\Delta b,\ \ A+\Delta A = A[p+\Delta p], \quad |\Delta p| \leq \epsilon g,\quad |\Delta b| \leq \epsilon f\}. \tag{2.1}$$

This definition differs from that of the componentwise relative backward error in two respects: we require $A + \Delta A \in S$, so that $A + \Delta A$ has the same structure as A, and we measure the size of the perturbation to A using Δp rather than ΔA. If $S = \mathbb{R}^{n\times n}$, then $\mu(y) \equiv \omega(y)$, assuming g and p comprise the elements of E in (1.4) and A, respectively.

The following transformation removes the absolute values from the constraints in the definition of $\mu(y)$ and replaces the inequalities by equalities. Let

$$\Delta p = D_1 v, \qquad \Delta b = D_2 w, \tag{2.2}$$

where $D_1 = \operatorname{diag}(g_i)$, $D_2 = \operatorname{diag}(f_i)$. Then the smallest ϵ satisfying $|\Delta p| \leq \epsilon g$ and $|\Delta b| \leq \epsilon f$ is $\epsilon = \max\{\|v\|_\infty, \|w\|_\infty\}$, and so

$$\mu(y) = \min\left\{ \left\| \begin{bmatrix} v \\ w \end{bmatrix} \right\|_\infty : (A+\Delta A)y = b+\Delta b,\ \ A+\Delta A = A[p+\Delta p], \right.$$
$$\left. \Delta p = D_1 v,\ \ \Delta b = D_2 w \right\}. \tag{2.3}$$

In general, this equality constrained nonlinear optimization problem has no closed form solution (and it will have no solution at all if the constraints cannot be satisfied). We therefore concentrate on the special case where S is a linear subspace

of $\mathbb{R}^{n\times n}$. Several classes of matrices of interest fall into this category, such as symmetric, Toeplitz, circulant, and Hankel matrices. Note that linearity implies $\Delta A = (A + \Delta A) - A \in S$. Furthermore, if each element of A is equal to a single element of p, that is, $a_{ij} = p_{k_{ij}}$, then we have the equivalence

$$|\Delta p| \le \epsilon g \iff |\Delta A| \le \epsilon E, \tag{2.4}$$

where $e_{ij} = g_{k_{ij}}$; we will use this equivalence below.

Defining $r = b - Ay$, the equation $(A+\Delta A)y = b+\Delta b$ may be written $\Delta A\, y - \Delta b = r$, or more usefully, $y^T \Delta A^T - \Delta b^T = r^T$, which places the variables ΔA to the right of the constant vector y. Applying the vec operator (which stacks the columns of a matrix into one long vector), we obtain

$$(I_n \otimes y^T)\mathrm{vec}(\Delta A^T) - \Delta b = r, \tag{2.5}$$

where $\otimes$ denotes the Kronecker product (see [17, Chap. 12] for properties of the vec operator and the Kronecker product).

By linearity we have

$$\mathrm{vec}(\Delta A^T) = B\Delta p \tag{2.6}$$

for some $B \in \mathbb{R}^{n^2\times t}$, which we assume to be of full rank.

Using (2.6) and (2.2) we can rewrite (2.5) as

$$(I_n \otimes y^T)BD_1 v - D_2 w = r,$$

or, with $Y = I_n \otimes y^T$,

$$[YBD_1,\ -D_2]\begin{bmatrix} v \\ w \end{bmatrix} = r. \tag{2.7}$$

This is an underdetermined system of the form $Cz = r$, with $C \in \mathbb{R}^{n\times(t+n)}$ and we seek the solution of minimal ∞-norm, the minimal value being $\mu(y)$.

Note that in the case where $t = n^2$ and $B = I$, the rows of C are "structurally independent," that is, there is at most one nonzero per column. Our minimization problem breaks into n independent problems of the form: minimize $\|x\|_\infty$ subject to $a^T x = \alpha$ (which has the solution $x = (\alpha/\|a\|_1)\mathrm{sign}(a)$). It is easy to see that we recover the Oettli–Prager formula (1.5).

If C is rank-deficient, then there may be no solution to $Cz = r$, in which case the structured componentwise backward error $\mu(y)$ may be regarded as being infinite. Assume, therefore, that C has full rank. If C^T has the QR factorization

$$C^T = Q\begin{bmatrix} R \\ 0 \end{bmatrix},$$

then $Cz = r$ may be written

$$r = \begin{bmatrix} R^T & 0^T \end{bmatrix} Q^T z \equiv \begin{bmatrix} R^T & 0^T \end{bmatrix}\begin{bmatrix} \overline{z}_1 \\ \overline{z}_2 \end{bmatrix} = R^T \overline{z}_1.$$

Thus $\overline{z}_1 = R^{-T} r$ is uniquely determined, and

$$z = Q\begin{bmatrix} \overline{z}_1 \\ \overline{z}_2 \end{bmatrix}.$$

Choosing $\overline{z}_2$ to minimize $\|z\|_\infty$ is equivalent to solving an overdetermined linear system in the ∞-norm sense, for which several methods are available [23, Chap. 2], [6].

We can obtain an approximation to the desired ∞-norm minimum by minimizing in the 2-norm, which amounts to setting $\overline{z}_2 = 0$ (and which yields $z = C^{+T} r$, where C^+ is the pseudo-inverse of C). In view of the fact that $n^{-1/2}\|x\|_2 \le \|x\|_\infty \le \|x\|_2$ for $x \in \mathbb{R}^n$, it follows that

$$\mu(y) \le \overline{\mu}(y) \equiv \left\| Q \begin{bmatrix} \overline{z}_1 \\ 0 \end{bmatrix} \right\|_\infty \le \sqrt{t+n}\,\mu(y). \tag{2.8}$$

How does the structured componentwise backward error $\mu(y)$ compare with the standard componentwise backward error $\omega(y)$? If we assume that (2.4) is valid and that E and f are the same for both backward errors then, clearly, $\mu(y) \ge \omega(y)$. More interestingly, if there are zeros in E and f, $\mu(y)$ can be infinite when $\omega(y)$ is finite. The reason is that there are more free parameters in the definition of $\omega(y)$ than in that of $\mu(y)$ and zeros in E or f reduce the number of free parameters in both definitions—potentially by enough for there to exist feasible perturbations ΔA and Δb for $\omega(y)$ but not for $\mu(y)$. Indeed, note that zeros in E and f introduce zero columns in C, making C more likely to be rank-deficient.

Two simple examples help to illustrate the points discussed above. Consider the system with

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \quad y = \begin{bmatrix} \epsilon \\ 1+\epsilon \end{bmatrix},$$

where $\epsilon > 0$, and let $E = |A|$, $f = 0$ in (1.1), (1.4), (2.1) and (2.4). It is easy to check that

$$\eta_\infty(y) = \frac{\epsilon}{1+\epsilon}, \qquad \omega(y) = 1,$$

and for symmetric structure, $\mu(y) = \infty$. If we alter A, b, and y to

$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ \epsilon \end{bmatrix}, \quad y = \begin{bmatrix} \epsilon \\ 1 \end{bmatrix},$$

we find that

$$\eta_\infty(y) = \frac{\epsilon}{2\max(1,\epsilon)}, \quad \omega(y) = \frac{\epsilon}{1+\epsilon}, \quad \mu(y) = 1,$$

which shows that even when the structured backward error is finite it can be arbitrarily larger than the normwise and componentwise backward errors.

It is of interest to characterize when C has full rank, as this guarantees that $\mu(y)$ is finite. C is certainly of full rank if f has no zero elements, because then D_2 is nonsingular, but little more can be said about the rank of C in general.

Finally, we note that if C has full rank, then

$$\begin{aligned} \mu(y) \le \overline{\mu}(y) &= \|C^{+T} r\|_\infty \\ &\le \|C^{+T} r\|_2 \le \|C^+\|_2 \|r\|_2 \\ &= \sigma_{\min}(C)^{-1} \|r\|_2, \end{aligned}$$

where $\sigma_{\min}$ denotes the smallest singular value. This inequality will often be a reasonable approximation and so it would be useful to determine the behavior of $\sigma_{\min}(C)$ as B and y vary. Unfortunately, the rectangularity of B and Y makes it difficult to obtain any results in this direction.

3. Structured condition number. With the same notation as in §2, we define the *structured condition number*

$$\text{cond}_\infty(A,x) = \lim_{\epsilon\to 0}\sup\left\{ \frac{\|\Delta x\|_\infty}{\epsilon\|x\|_\infty} : (A+\Delta A)(x+\Delta x) = b+\Delta b, \right.$$

$$\left. A+\Delta A = A[p+\Delta p],\ \ |\Delta p| \le \epsilon g, |\Delta b| \le \epsilon f \right\}. \tag{3.1}$$

This definition employs the same class of perturbations as in the definition of the structured componentwise backward error $\mu(y)$, and so we have the perturbation result, for any y,

$$\frac{\|y-x\|_\infty}{\|x\|_\infty} \le \text{cond}_\infty(A,x)\mu(y) + O(\mu(y)^2).$$

(Strictly, this result requires that $\|\Delta p\|_\infty = O(\epsilon) \Rightarrow \|\Delta A\|_\infty = O(\epsilon)$; otherwise the order term has to be weakened to $o(\mu(y))$.)

An explicit expression for $\text{cond}_\infty(A,x)$ can be derived in the case where A depends linearly on its parameters. For a given ϵ and perturbed system in the definition of $\text{cond}_\infty(A,x)$, we have

$$(A+\Delta A)\Delta x = \Delta b - \Delta A x,$$

which yields

$$\begin{aligned} \Delta x &= A^{-1}(\Delta b - \Delta A x - \Delta A \Delta x) \\ &= A^{-1}\Delta b - A^{-1}\Delta A x + O(\epsilon^2). \end{aligned} \tag{3.2}$$

First, we analyze the term $A^{-1}\Delta A x$. We have

$$\begin{aligned} \Delta A x &= \text{vec}((\Delta A x)^T) \\ &= \text{vec}(x^T \Delta A^T) \\ &= (I_n \otimes x^T)\text{vec}(\Delta A^T) \\ &= XB\Delta p, \end{aligned}$$

where we have used (2.6) and defined $X = I_n \otimes x^T$. Since $|\Delta p| \le \epsilon g$, it follows that

$$|A^{-1}\Delta A x| = |A^{-1}XB\Delta p| \le \epsilon |A^{-1}XB| g.$$

Similarly,

$$|A^{-1}\Delta b| \le \epsilon |A^{-1}| f.$$

Taking norms we obtain

$$\| - A^{-1}\Delta A x + A^{-1}\Delta b\|_\infty \le \epsilon \| \, |A^{-1}XB| g + |A^{-1}| f \, \|_\infty. \tag{3.3}$$

It is easy to see that equality is attainable in (3.3) for suitable choice of Δp and Δb. It follows from (3.2) and (3.3) that

$$\frac{\|\Delta x\|_\infty}{\epsilon\|x\|_\infty} \le \frac{\| \, |A^{-1}XB| g + |A^{-1}| f \, \|_\infty}{\|x\|_\infty} + O(\epsilon)$$

is a sharp bound, and hence

$$\text{cond}_\infty(A,x) = \frac{\| \, |A^{-1}XB|g + |A^{-1}|f \, \|_\infty}{\|x\|_\infty}. \tag{3.4}$$

In the special case where no structure is imposed ($B = I_{n^2}$), this expression can be written in the form

$$\text{cond}'_\infty(A,x) = \frac{\| \, |A^{-1}|E|x| + |A^{-1}|f \, \|_\infty}{\|x\|_\infty}, \tag{3.5}$$

where E is defined in (2.4); this is a generalization of a condition number of Skeel [20], as described in [1], [15]. It is possible for $\text{cond}'_\infty(A,x)$ to exceed $\text{cond}_\infty(A,x)$ by an arbitrary factor. However, if $\| \, |A^{-1}|f \, \|_\infty \approx \| \, |A^{-1}|E|x| \, \|_\infty$ then the two condition numbers will be of similar magnitude.

More convenient to work with than $\text{cond}_\infty(A,x)$ is the quantity

$$\theta_\infty(A,x) = \frac{\|A^{-1}XBD_1\|_\infty + \|A^{-1}D_2\|_\infty}{\|x\|_\infty},$$

where $D_1 = \text{diag}(g_i)$ and $D_2 = \text{diag}(f_i)$. It is easy to show that

$$\tfrac{1}{2}\theta_\infty(A,x) \le \text{cond}_\infty(A,x) \le \theta_\infty(A,x).$$

The quantity $\theta_\infty(A,x)$ can be estimated without explicitly forming the matrices $A^{-1}XBD_1 \in \mathbb{R}^{n\times t}$ and $A^{-1}D_2 \in \mathbb{R}^{n\times n}$ (assuming a factorization of A is available) by using the method of Hager [10] and Higham [12], [14]; this method estimates $\|C\|_\infty$ at the cost of forming a few matrix-vector products Cx and C^Ty.

We also mention two interesting nonlinear structures, those of Vandermonde matrices $V = (\alpha_j^{i-1})$ and Cauchy matrices $H = ((\alpha_i + \beta_j)^{-1})$. In [11], explicit expressions are derived for $\text{cond}_\infty(V,x)$ and $\text{cond}_\infty(V^T,x)$ in the case where $f = 0$ and $g = (1,1,\cdots,1)^T$. In [8] a structured condition number with respect to the inversion of H is derived.

4. Applications. In this section, we look in detail at the structured componentwise backward error and structured condition number for three applications—those involving symmetric matrices, Toeplitz matrices, and the augmented system for a LS problem.

4.1. Symmetric matrices. For the property of symmetry, there are $t = \frac{1}{2}n(n+1)$ parameters in the vector p. It is natural to take these parameters to be the elements in the upper triangle of A, in which case every row of B contains a single nonzero entry equal to one. To illustrate the form of the underdetermined system (2.7), we consider the case $n = 3$. It is easy to derive the system without using B. Also, it is convenient to work with the independent elements of ΔA rather than Δp, and a symmetric matrix of tolerances E rather than g (see (2.4)).

The constraint $\Delta Ay - \Delta b = r$, that is,

$$\begin{bmatrix} \Delta a_{11} & \Delta a_{12} & \Delta a_{13} \\ \Delta a_{12} & \Delta a_{22} & \Delta a_{23} \\ \Delta a_{13} & \Delta a_{23} & \Delta a_{33} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} - \begin{bmatrix} \Delta b_1 \\ \Delta b_2 \\ \Delta b_3 \end{bmatrix} = r,$$

is equivalent to the system

$$(4.1)\qquad \begin{bmatrix} y_1 & y_2 & y_3 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & y_1 & 0 & y_2 & y_3 & 0 & 0 & -1 & 0 \\ 0 & 0 & y_1 & 0 & y_2 & y_3 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \Delta a_{11} \\ \Delta a_{12} \\ \Delta a_{13} \\ \Delta a_{22} \\ \Delta a_{23} \\ \Delta a_{33} \\ \Delta b_1 \\ \Delta b_2 \\ \Delta b_3 \end{bmatrix} = r.$$

On using the transformation (2.2) (where $\Delta p = \mathrm{vec}(\Delta A)$), we obtain the underdetermined system (2.7),

$$(4.2)\qquad \begin{bmatrix} y_1 & y_2 & y_3 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & y_1 & 0 & y_2 & y_3 & 0 & 0 & -1 & 0 \\ 0 & 0 & y_1 & 0 & y_2 & y_3 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix} \begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \\ v_5 \\ v_6 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} = r,$$

where $D_1 = \mathrm{diag}(e_{11}, e_{12}, e_{13}, e_{22}, e_{23}, e_{33})$ and $D_2 = \mathrm{diag}(f_1, f_2, f_3)$. Note that the $n \times (n^2/2 + 3n/2)$ coefficient matrix C is upper trapezoidal. Solutions to (4.2) are easily obtained by inspection, but in general, none of these solutions will be of near minimal norm.

The special structure of the matrix C enables computation of the QR factorization of C^T in $O(n^3)$ operations, by careful use of Givens rotations.

A structured *normwise* backward error for symmetric matrices has been considered by Bunch, Demmel, and Van Loan [5]. They consider $\eta(y)$ (see (1.1)) with $f \equiv 0$ and show that enforcing symmetry of ΔA when A is symmetric does not increase $\eta(y)$ for the 2-norm, and it increases it by at most a factor $\sqrt{2}$ for the Frobenius norm. No such result holds for componentwise backward errors because, as explained in §2, it is possible for $\mu(y)$ to be infinite when $\omega(y)$ is finite. However, we note that in the special case where E is diagonal, $\mu(y) = \omega(y)$, because the inequality $|\Delta A| \leq \epsilon E$ in (1.4) automatically forces ΔA to be diagonal and hence symmetric.

The result of [5] can be loosely verified using (2.7). If we set all the elements of f and g to 1 (thus $D_1 = I_t$ and $D_2 = I_n$), then $\mu(y)$ differs from $\eta(y)$ for the 2-norm by at most a factor $\sqrt{n}$ when $B = I_{n^2}$. We will assume that $y = e_1$; this entails no loss of generality because an orthogonal transformation

$$(A + \Delta A)y = b + \Delta b \quad \rightarrow \quad Q(A + \Delta A)Q^T \cdot Qy = Q(b + \Delta b)$$

does not change the class of admissible perturbations or the 2-norms of the perturbations, although it does require g to be multiplied by a factor $\sqrt{n}$. Comparing $Y = I \otimes y^T$ with YB, where B corresponds to the symmetry constraint, we find that they differ only in that Y has extra zero columns. Thus imposing symmetry does not affect the norm of the minimum ∞-norm solution to the system (2.7) when B, D_1,

and D_2 are identity matrices, and this confirms the result of [5], to within a factor n. (The factor n is a consequence of switching from using components to 2-norms.)

A very similar argument shows that when D_1 and D_2 are identity matrices the condition number $\mathrm{cond}_2(A,x)$ is the same when symmetry is imposed as when there is no structural constraint ($\mathrm{cond}_2(A,x)$ is defined as in (3.4) but with the 2-norm replacing the ∞-norm). We note that a condition number that respects symmetry has been derived in a different context by Fletcher [7]. Making statistical assumptions about the perturbations to a linear system, Fletcher shows that the expected condition number of a system is changed little by the imposition of symmetry.

To summarize, when perturbations to a symmetric matrix are measured using the 2-norm it makes little difference to the backward error or to the condition number whether symmetry is enforced or not.

4.2. Toeplitz matrices. Recall that $A \in \mathbb{R}^{n\times n}$ is a Toeplitz matrix if there exist scalars $\{a_k\}_{k=1-n}^{n-1}$ such that $a_{ij} = a_{j-i}$, that is,

$$A = \begin{bmatrix} a_0 & a_1 & \cdots & a_{n-1} \\ a_{-1} & a_0 & \ddots & \vdots \\ \vdots & \ddots & \ddots & a_1 \\ a_{1-n} & \cdots & a_{-1} & a_0 \end{bmatrix} = \mathrm{Toeplitz}(a_{1-n}, \cdots, a_0, \cdots, a_{n-1}).$$

In computing the "Toeplitz componentwise backward error," we have to distinguish between unsymmetric and symmetric Toeplitz matrices, for which the number of parameters in A is $t = 2n-1$ and $t = n$, respectively. As in the previous section, it is easy to derive the relevant underdetermined system (2.7).

For illustration we again consider the case $n = 3$. It is straightforward to obtain the following analogues of (4.1), where $\Delta A = \mathrm{Toeplitz}(\Delta a_{1-n}, \cdots, \Delta a_0, \cdots, \Delta a_{n-1})$:

$$\text{(4.3)}\quad \text{unsymmetric:} \quad \begin{bmatrix} y_3 & y_2 & y_1 & 0 & 0 & -1 & 0 & 0 \\ 0 & y_3 & y_2 & y_1 & 0 & 0 & -1 & 0 \\ 0 & 0 & y_3 & y_2 & y_1 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \Delta a_2 \\ \Delta a_1 \\ \Delta a_0 \\ \Delta a_{-1} \\ \Delta a_{-2} \\ \Delta b_1 \\ \Delta b_2 \\ \Delta b_3 \end{bmatrix} = r,$$

$$\text{(4.4)}\quad \text{symmetric:} \quad \begin{bmatrix} y_3 & y_2 & y_1 & -1 & 0 & 0 \\ 0 & y_3 + y_1 & y_2 & 0 & -1 & 0 \\ y_1 & y_2 & y_3 & 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} \Delta a_2 \\ \Delta a_1 \\ \Delta a_0 \\ \Delta b_1 \\ \Delta b_2 \\ \Delta b_3 \end{bmatrix} = r.$$

Note that the coefficient matrix in (4.3) loses its Toeplitz structure when we carry out the column scaling necessary to reach (2.7) (cf. (4.2)). Since the number of columns of C is $t+n = O(n)$ in both cases, the cost of computing the QR factorization of C is no more than $O(n^3)$ operations.

Note that if we set $f = 0$, then in the symmetric case the system $Cz = r$ reduces to a square system, corresponding to the fact that the number of parameters in ΔA and Δb is the same as the number of equations.

4.3. The augmented system for the least squares problem. Let $A \in \mathbb{R}^{m\times n}$ have full rank n, let $b \in \mathbb{R}^m$, and let y be an approximate solution to the LS problem $\min_x \|Ax-b\|_2$. Suppose we wish to determine the backward error of y, that is, $\eta_{LS}(y)$ or $\omega_{LS}(y)$, defined as $\eta(y)$ in (1.1) or $\omega(y)$ in (1.4), respectively, but with the condition $(A+\Delta A)y = b$ replaced by the requirement that $\|(A+\Delta A)y-(b+\Delta b)\|_2$ is minimal. Obtaining an explicit formula for $\eta_{LS}(y)$ or $\omega_{LS}(y)$, or an effective way of computing these quantities, is an open problem, as discussed in [13] and [21]; here we make some progress on the problem.

Observe that the LS minimizer x satisfies the augmented system

$$\begin{bmatrix} I & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} r \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{4.5}$$

since this is simply a representation of the normal equations. Because this is a square system, the work in §2 can be exploited. The augmented system has a great deal of structure; to reflect this in the structured componentwise backward error $\mu(y)$, it is sufficient to impose symmetry and to take E (see (2.4)) and f of the form

$$E = \begin{bmatrix} 0 & E_A \\ E_A^T & 0 \end{bmatrix}, \qquad f = \begin{bmatrix} f_b \\ 0 \end{bmatrix}.$$

Let us denote this backward error by $\mu_{LS}(r,y)$. The main observation of this section is that $\mu_{LS}(r,y)$ respects the structure of the augmented system (unlike $\beta(r,y)$ below) and can be computed using standard methods (as described for $\mu(y)$ in §2).

A complicating factor is that r in (4.5) is effectively a vector of free parameters, so to obtain $\eta_{LS}(y)$ or $\omega_{LS}(y)$ we have to minimize $\mu_{LS}(r,y)$ over all r. Fortunately, in the applications of interest the naturally arising r is often a good approximation to the minimizer [16].

In [3] and [13] a "pseudo-componentwise backward error" $\beta(r,y)$ was defined for the augmented system in which different perturbations are allowed in the two occurrences of A. This quantity β is simply $\omega(y)$ of (1.4) applied to the augmented system with

$$E = \begin{bmatrix} 0 & |A| \\ |A^T| & 0 \end{bmatrix}, \qquad f = \begin{bmatrix} |b| \\ 0 \end{bmatrix},$$

and so an explicit formula is available for it from (1.5). In [16], β is proved to be small after one step of fixed precision iterative refinement, under suitable assumptions, when a QR factorization is used to solve the LS problem. Hence it is of interest to compare $\mu_{LS}(r,y)$ with $\beta(r,y)$ when $E_A = |A|$ and $f_b = |b|$. Clearly we have $\mu_{LS}(r,y) \geq \beta(r,y)$ because of the additional symmetry constraint in the definition of μ_{LS}. We report some numerical comparisons in the next section.

Finally, we note that it is possible to obtain a first-order approximation to the backward error $\omega_{LS}(y)$ by considering the perturbed normal equations

$$(A+\Delta A)^T(A+\Delta A)y = (A+\Delta A)^T(b+\Delta b).$$

Expanding and dropping the second-order terms $\Delta A^T\Delta A$ and $\Delta A^T\Delta b$, we have

$$A^T\Delta A y + \Delta A^T A y - A^T\Delta b - \Delta A^T b = A^T(b-Ay).$$

With manipulation similar to that in §2, this linearized problem can be reduced to the computation of a minimum ∞-norm solution to an underdetermined system.

5. Numerical experiments. In the three applications discussed in §3 it is more expensive to compute $\mu(y)$ or $\overline{\mu}(y)$ than to solve the original linear equation problem (but it is inexpensive to estimate $\mathrm{cond}_\infty(A,x)$). Thus, unlike the normwise and componentwise backward errors $\eta(y)$ and $\omega(y)$, $\mu(y)$ is not a quantity that we would compute routinely in the course of solving a problem. However, $\mu(y)$ is useful as a computational tool for studying the stability of a numerical algorithm for solving a structured linear equations problem. We describe some numerical experiments involving the applications of §3.

Our experiments were done using MATLAB, which has a unit roundoff $u \approx 2.2\times10^{-16}$. Computing μ involves finding the minimal ∞-norm solution to the underdetermined system (2.7). To do this we used the QR factorization transformation to an overdetermined system described in §2, and solved this system in the ∞-norm sense using the method of [6]. The cost of the method of [6] when applied to $\min \|Ax-b\|_\infty$ is approximately the cost of solving k weighted LS problems $\min \|B_i^{-1}Ax_i - b_i\|_2$, where k depends mildly on the problem dimensions (typically $k \le 20$), and where B_i is diagonal except for one full column.

In the first test we solved the system $Ax = b$ using Gaussian elimination with partial pivoting (GEPP), where A is the 10×10 Hilbert matrix and $b = (1,1,\cdots,1)^T/3$. For several E and f, we evaluated the backward errors η, ω, and μ for the computed vector $\widehat{x}$, and the condition numbers $\mathrm{cond}_\infty(A,x)$ of (3.4) and $\mathrm{cond}'_\infty(A,x)$ of (3.5); we imposed the constraint of symmetry for μ (symmetry is denoted by S in the first column of Table 5.1). For this matrix GEPP interchanges rows, and so symmetry is lost in the solution process. We know from standard error analysis that $\eta(\widehat{x})$ will be of order u for $E = A$ (assuming there is no undue element growth in the elimination), irrespective of f, and the result of Bunch, Demmel, and Van Loan referred to in §4.1 shows that imposing symmetry of ΔA in (1.1) cannot significantly increase η. Comparing $\mu(\widehat{x})$ and $\omega(\widehat{x})$, and cond_∞ and cond'_∞, in Table 5.1, we see that requiring symmetry also has little or no effect on the componentwise backward error or the componentwise condition number in this example.

The reason why the numbers in the first two rows of Table 5.1 are the same is that $|A||x|$ is large compared with $|b|$ and hence it makes relatively little difference to the formulas (1.2), (1.5), (3.4), (3.5), and the matrix C, whether we take $f = 0$ or $f = |b|$.

In Table 5.2, A is the symmetric part of a 10×10 matrix with elements from the random normal (0,1) distribution and b is the same vector as in the first example. We see that for the computed solution $\widehat{x}$ from GEPP, $\mu(\widehat{x}) = \omega(\widehat{x})$ in each case; this behavior is not uncommon. Our limited experience indicates that for well-conditioned, full symmetric matrices, $\mu(\widehat{x})$ is usually of similar size to $\omega(\widehat{x})$ for the $\widehat{x}$ from GEPP.

The next example involves the symmetric positive definite 10×10 Toeplitz matrix $A = (\rho^{|i-j|})$, with $\rho = 1-3\times10^{-5}$ and $b = (1,2,\cdots,10)^T/3$. We solved $Ax = b$ using GEPP and the $O(n^2)$ operations Levinson algorithm [9, p. 187]. [1] Tables 5.3 and 5.4 report the μ values obtained on imposing symmetry or Toeplitz structure alone (denoted by S or T in the first column), and both symmetry and Toeplitz structure. In the tables, $\|A\|_M$ denotes $\max_{i,j}|a_{ij}|$. Preserving the symmetric Toeplitz structure raises the backward errors three orders of magnitude. Note also that there is little difference in the backward errors between the two methods. This example shows that even when a method specific to Toeplitz systems is used, the computed solution is not guaranteed to be the solution to a nearby Toeplitz system.

[1] Note that the second plus should be a minus in the expression for α in [9, Algorithm 4.7.2].

TABLE 5.1
$A = Hilbert(10)$, $\kappa_2(A) = 1.60e13$, *GEPP*. ($E_1 = |\mathrm{diag}(A)|$).

	E	f	$\kappa_2(C)$	cond'_∞	cond_∞	$\eta_\infty(\widehat{x})$	$\omega(\widehat{x})$	$\mu(\widehat{x})$
S	$\|A\|$	$\|b\|$	2.40e0	3.05e12	3.05e12	1.99e-18	2.15e-17	2.18e-17
S	$\|A\|$	0	2.40e0	3.05e12	3.05e12	1.99e-18	2.15e-17	2.18e-17
S	0	$\|b\|$	1.00e0	1.72e6	1.72e6	4.08e-11	4.08e-11	4.08e-11
S	E_1	0	5.17e4	6.63e11	6.63e11	5.82e-18	3.70e-12	3.70e-12

TABLE 5.2
$A = \mathrm{symm}(\mathrm{rand}(10))$, $\kappa_2(A) = 6.24e1$, *GEPP*. ($E_1 = |\mathrm{diag}(A)|$).

	E	f	$\kappa_2(C)$	cond'_∞	cond_∞	$\eta_\infty(\widehat{x})$	$\omega(\widehat{x})$	$\mu(\widehat{x})$
S	$\|A\|$	$\|b\|$	3.63e0	5.06e1	4.79e1	3.81e-17	1.26e-16	1.26e-16
S	$\|A\|$	0	3.81e0	4.77e1	4.49e1	3.87e-17	1.42e-16	1.42e-16
S	0	$\|b\|$	1.00e0	2.97e0	2.97e0	2.33e-15	2.33e-15	2.33e-15
S	E_1	0	1.35e2	6.50e0	6.50e0	2.11e-16	4.77e-14	4.77e-14

It is perhaps surprising that the increase in the backward error $\mu(\widehat{x})$ between rows "S," "T," and "S,T" in Tables 5.3 and 5.4 is not matched by a decrease in $\mathrm{cond}_\infty(A, x)$. This means, for example, that a smaller forward error bound (equal to condition number times backward error) is obtained in this example if we do *not* utilize the full structure of the problem. Nevertheless, it is not difficult to find examples where $\mathrm{cond}'_\infty(A, x)/\mathrm{cond}_\infty(A, x)$ is large for symmetric Toeplitz structure if we set $f = 0$, which confines perturbations to the coefficient matrix.

We mention that in all the examples reported the approximation $\overline{\mu}$ in (2.8) satisfied $\overline{\mu} \leq 2\mu$.

In a further experiment we repeated some of the numerical tests from [16], which involve fixed precision iterative refinement of the LS problem using a QR factorization. We extended the testing of [16] by evaluating $\mu_{LS}(\widehat{r}, \widehat{y})$ in addition to $\beta(\widehat{r}, \widehat{y})$, where μ_{LS} and β are defined in §4.3 and $\widehat{r}$ and $\widehat{y}$ are the computed residual and LS solution (both after refinement), respectively. For [16, problem PR] (in which A is a 4×3 matrix with widely varying row norms) and problem set H (a parametrized set of problems involving a 6×5 submatrix of the inverse of the Hilbert matrix of order 6), we found that $\mu_{LS}(\widehat{r}, \widehat{y}) \approx \beta(\widehat{r}, \widehat{y}) \approx u$ in every case. Thus we can conclude that *in these examples* $\omega_{LS}(\widehat{y}) \equiv \min_r \mu_{LS}(r, \widehat{y}) \approx u$, that is, the computed solution obtained after iterative refinement is the exact solution to a small componentwise perturbation of the original LS problem.

6. Concluding remarks. The contribution of this work is to extend existing definitions of backward error and condition number in a way appropriate to structured linear systems and to show how these structure-respecting quantities can be computed in the important case of linear structure. Thus we have derived new theoretical and computational tools. Several questions merit further investigation:

(1) Are there any nonlinear structures for which $\mu(y)$ can be computed more efficiently than if it is treated as a general nonlinear optimization problem (for example, for Vandermonde matrices)?

(2) Is it possible to obtain further theoretical bounds on $\mu(y)$ that would help us to understand its behavior?

(3) Standard backward error analysis results for linear system solvers usually ignore structure. Are there problems and algorithms for which a structured backward

TABLE 5.3
A = *Toeplitz* (10), $\kappa_2(A) = 6.50e5$, *GEPP.* ($E_2 = \|A\|_M ee^T$, $f_2 = \|b\|_\infty e$.)

	E	f	$\kappa_2(C)$	cond'_∞	cond_∞	$\eta_\infty(\widehat{x})$	$\omega(\widehat{x})$	$\mu(\widehat{x})$
S	$\|A\|$	$\|b\|$	1.73e0	1.33e5	1.33e5	2.13e-17	1.07e-16	2.13e-16
S	$\|A\|$	0	1.73e0	1.33e5	1.33e5	2.13e-17	1.07e-16	2.13e-16
T	$\|A\|$	$\|b\|$	1.73e0	1.33e5	1.33e5	2.13e-17	1.07e-16	2.13e-16
T	$\|A\|$	0	1.73e0	1.33e5	1.33e5	2.13e-17	1.07e-16	2.13e-16
S,T	$\|A\|$	$\|b\|$	4.28e3	1.33e5	1.33e5	2.13e-17	1.07e-16	3.23e-13
S,T	$\|A\|$	0	6.06e3	1.33e5	1.33e5	2.13e-17	1.07e-16	6.46e-13
S,T	E_2	f_2	2.92e3	1.33e5	1.33e5	2.13e-17	1.07e-16	2.29e-13

TABLE 5.4
A = *Toeplitz* (10), *Levinson algorithm. Condition numbers as in Table* 5.3.

	E	f	$\eta_\infty(\widehat{x})$	$\omega(\widehat{x})$	$\mu(\widehat{x})$
S	$\|A\|$	$\|b\|$	4.07e-17	2.04e-16	2.32e-16
S	$\|A\|$	0	4.07e-17	2.04e-16	2.32e-16
T	$\|A\|$	$\|b\|$	4.07e-17	2.04e-16	2.32e-16
T	$\|A\|$	0	4.07e-17	2.04e-16	2.33e-16
S,T	$\|A\|$	$\|b\|$	4.07e-17	2.04e-16	4.22e-13
S,T	$\|A\|$	0	4.07e-17	2.04e-16	8.45e-13
S,T	$\|A\|_M ee^T$	$\|b\|_\infty e$	4.07e-17	2.03e-16	3.00e-13

error result can be developed? See [22] for further examples of structured problems.

(4) What can be said about the ratio $\text{cond}'_\infty(A,x)/\text{cond}_\infty(A,x)$ for particular structures and choices of tolerances, that is, how much can the imposition of structure change the condition number? We have answered this question in a particular case involving the property of symmetry.

Acknowledgments. We thank Yuying Li for providing us with MATLAB M-files that implement the method of [6]. The second author thanks the Numerical Analysis Group at the University of Toronto, for their hospitality.

REFERENCES

[1] M. ARIOLI, J. W. DEMMEL AND I. S. DUFF, *Solving sparse linear systems with sparse backward error*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 165–190.

[2] M. ARIOLI, I. S. DUFF AND P. P. M. DE RIJK, *On the augmented system approach to sparse least-squares problems*, Numer. Math., 55 (1989), pp. 667–684.

[3] A. BJÖRCK, *Component-wise perturbation analysis and error bounds for linear least squares solutions*, BIT, 31 (1991), pp. 238–244.

[4] J. R. BUNCH, *The weak and strong stability of algorithms in numerical linear algebra*, Linear Algebra Appl., 88/89 (1987), pp. 49–66.

[5] J. R. BUNCH, J. W. DEMMEL, AND C. F. VAN LOAN, *The strong stability of algorithms for solving symmetric linear systems*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 494–499.

[6] T. F. COLEMAN AND Y. LI, *A global and quadratically convergent method for linear L_∞ problems*, Tech. Report 90-1121, Department of Computer Science, Cornell University, Ithaca, NY, 1990.

[7] R. FLETCHER, *Expected conditioning*, IMA J. Numer. Anal., 5 (1985), pp. 247–273.

[8] I. GOHBERG AND I. KOLTRACHT, *On the inversion of Cauchy matrices*, in Signal Processing, Scattering and Operator Theory, and Numerical Methods, Proceedings of the International Symposium MTNS-89, Volume III, M.A. Kaashoek, J.H. van Schuppen and A.C.M. Ran, eds., 1990, pp. 381–392.

[9] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, Johns Hopkins University Press, Baltimore, MD, 1989.

[10] W. W. HAGER, *Condition estimates*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 311–316.
[11] N. J. HIGHAM, *Error analysis of the Björck–Pereyra algorithms for solving Vandermonde systems*, Numer. Math., 50 (1987), pp. 613–632.
[12] ———, FORTRAN *codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation (Algorithm* 674), ACM Trans. Math. Software, 14 (1988), pp. 381–396.
[13] ———, *Computing error bounds for regression problems*, in Statistical Analysis of Measurement Error Models and Applications, P. J. Brown and W. A. Fuller, eds., Contemporary Mathematics 112, American Mathematical Society, Providence, RI, 1990, pp. 195–208.
[14] ———, *Experience with a matrix norm estimator*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 804–809.
[15] ———, *How accurate is Gaussian elimination?*, in Numerical Analysis 1989, Proceedings of the 13th Dundee Conference, D. F. Griffiths and G. A. Watson, eds., Pitman Research Notes in Mathematics 228, Longman Scientific and Technical, Harlow, Essex, 1990, pp. 137–154.
[16] ———, *Iterative refinement enhances the stability of QR factorization methods for solving linear equations*, Numerical Analysis Report No. 182, University of Manchester, England, 1990; BIT, to appear.
[17] P. LANCASTER AND M. TISMENETSKY, *The Theory of Matrices*, Second Edition, Academic Press, New York, 1985.
[18] W. OETTLI AND W. PRAGER, *Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides*, Numer. Math., 6 (1964), pp. 405–409.
[19] J. L. RIGAL AND J. GACHES, *On the compatibility of a given solution with the data of a linear system*, J. Assoc. Comput. Mach., 14 (1967), pp. 543–548.
[20] R. D. SKEEL, *Scaling for numerical stability in Gaussian elimination*, J. Assoc. Comput. Mach., 26 (1979), pp. 494–526.
[21] G. W. STEWART, *Research, development, and* LINPACK, in Mathematical Software III, J.R. Rice, ed., Academic Press, New York, 1977, pp. 1–14.
[22] P. M. VAN DOOREN, *Structured linear algebra problems in digital signal processing*, Proceedings of North Atlantic Treaty Organization ASI, Leuven 1988, Series F, Springer-Verlag, Berlin, 1990.
[23] G. A. WATSON, *Approximation Theory and Numerical Methods*, John Wiley, Chichester, 1980.
[24] J. H. WILKINSON, *Rounding errors in algebraic processes*, Notes on Applied Science, No. 32, Her Majesty's Stationery Office, London, 1963.
[25] ———, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 176–190, January 1992

014

LOSS AND RECAPTURE OF ORTHOGONALITY IN THE MODIFIED GRAM–SCHMIDT ALGORITHM*

Å. BJÖRCK† AND C. C. PAIGE‡

To our close friend and mentor Gene Golub, on his 60th birthday. This is but one of the many topics on which Gene has generated so much interest, and shed so much light.

Abstract. This paper arose from a fascinating observation, apparently by Charles Sheffield, and relayed to us by Gene Golub, that the QR factorization of an $m \times n$ matrix A via the modified Gram-Schmidt algorithm (MGS) is *numerically* equivalent to that arising from Householder transformations applied to the matrix A augmented by an n by n zero matrix. This is explained in a clear and simple way, and then combined with a well-known rounding error result to show that the upper triangular matrix R from MGS is about as accurate as R from other QR factorizations. The special structure of the product of the Householder transformations is derived, and then used to explain and bound the loss of orthogonality in MGS. Finally this numerical equivalence is used to show how orthogonality in MGS can be regained in general. This is illustrated by deriving a numerically stable algorithm based on MGS for a class of problems which includes solution of nonsingular linear systems, a minimum 2-norm solution of underdetermined linear systems, and linear least squares problems. A brief discussion on the relative merits of such algorithms is included.

Key words. orthogonal matrices, QR factorization, Householder transformations, least squares, minimum norm solution, numerical stability, Gram–Schmidt, augmented systems

AMS(MOS) subject classifications. 65F25, 65G05, 65F05, 65F20

1. Introduction.

We consider a matrix $A \in \mathbf{R}^{m\times n}$ with rank $n \leq m$. The modified Gram–Schmidt algorithm (MGS) in theory produces Q_1 and R in the QR factorization

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix} = Q_1 R, \qquad Q = (\,Q_1 \quad Q_2\,) \tag{1.1}$$

where Q is orthogonal and R upper triangular. In practice, if the condition number $\kappa = \kappa(A) \equiv \sigma_1/\sigma_n$ is large ($\sigma_1 \geq \cdots \geq \sigma_n$ being the singular values of A), then the columns of Q_1 are not accurately orthogonal [3]. If orthogonality is crucial, then usually either rotations or Householder transformations have been used to compute the QR factorization. Here we show how MGS can be used just as stably for many problems requiring this orthogonality.

We derive some important properties of MGS in the presence of rounding errors. In particular, we show that the R obtained from MGS is numerically as good as that obtained from rotations or Householder transformations. We present new insights on the loss of orthogonality in Q_1 from MGS, and show how this can be effectively regained in computations that use Q_1, without altering the MGS algorithm or re-orthogonalizing the columns of Q_1. As a practical example of this, we indicate how Q_1 and R from MGS may be used to solve an important class of problems reliably,

* Received by the editors January 2, 1991; accepted for publication (in revised form) June 23, 1991. This research was partially supported by National Sciences and Engineering Research Council of Canada grant A9236.

† Department of Mathematics, Linköping University, S-581 83, Linköping, Sweden (akbjo@math.liu.se).

‡ Department of Computer Science, McGill University, Montreal, Quebec, Canada H3A 2A7 (chris@cs.mcgill.ca).

despite the loss of orthogonality in Q_1. This new approach seems applicable to most problems for which MGS is in theory relevant.

The class of problems we consider is that of solving the symmetric indefinite linear system involving $A \in \mathbf{R}^{m\times n}$ with rank n

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ c \end{pmatrix}. \tag{1.2}$$

In general we call (1.2) the augmented system formulation (ASF) of the following two problems, since it represents the conditions for their solution:

$$\min_x \|b - x\|_2, \qquad A^T x = c, \tag{1.3}$$

$$\min_y \{\|b - Ay\|_2^2 + 2c^T y\}. \tag{1.4}$$

We examine these problems more fully in [5]. The ASF can be obtained by differentiating the Lagrangian $\|b-x\|_2^2+2y^T(A^Tx-c)$ of (1.3), and equating to zero. Here y is the vector of Lagrange multipliers. The ASF can also be obtained by differentiating (1.4) to give $A^T(b - Ay) = c$, and setting x to be the "residual" $x = b - Ay$.

The ASF covers two important special cases. Setting $b = 0$ in (1.3), and so in (1.2), gives the problem of finding the minimum 2-norm solution of a linear underdetermined system (LUS). Setting $c = 0$ in (1.4) gives the much used linear least squares (LLS) problem. The ASF also occurs in its full form (1.2) in the iterative refinement of least squares solutions [2].

Using the QR factorization (1.1), we can transform (1.2) into

$$\begin{pmatrix} I & \begin{pmatrix} R \\ 0 \end{pmatrix} \\ \begin{pmatrix} R^T & 0 \end{pmatrix} & 0 \end{pmatrix} \begin{pmatrix} Q^T x \\ y \end{pmatrix} = \begin{pmatrix} Q^T b \\ c \end{pmatrix}.$$

This gives one method for solving (1.2):

$$z = R^{-T}c, \qquad \begin{pmatrix} d \\ f \end{pmatrix} = Q^T b, \qquad x = Q \begin{pmatrix} z \\ f \end{pmatrix}, \qquad y = R^{-1}(d - z). \tag{1.5}$$

Using $x = Q_1 z + Q_2 f = Q_1 z + Q_2 Q_2^T b = Q_1 z + (I - Q_1 Q_1^T)b$, we obtain an obvious variant:

$$z = R^{-T}c, \qquad d = Q_1^T b, \qquad x = b - Q_1(d - z), \qquad y = R^{-1}(d - z). \tag{1.6}$$

Björck [2] showed that (1.5) is backward stable for (1.2) using the Householder QR factorization. Since (1.5) uses Q, (1.6) seems preferable if x is required and only Q_1 is available. However, as we shall see, it cannot generally be recommended when Q_1 is obtained by MGS. We will show how to develop more reliable algorithms based on Q_1 from MGS.

In §2 we illustrate the important but not widely appreciated result that MGS is *numerically* equivalent to the Householder QR factorization applied to A augmented with a block of zeros. From this we show in §3 that the computed R from MGS is numerically as satisfactory as that obtained using Householder QR on A. The product P of the Householder transformations from the QR factorization of $\binom{O_n}{A}$ is crucial

for a full understanding of MGS. P has a simple and important structure, and this is derived in the theorem in §4. This structure shows exactly how the computed $\bar{Q}_1$ from MGS can lose orthogonality. In §5 this structure is used to bound the loss of orthogonality of $\bar{Q}_1$, while §6 shows how the lost orthogonality can be compensated for just by using $\bar{Q}_1$ differently without altering $\bar{Q}_1$ or MGS. We illustrate this by producing a new backward stable algorithm for (1.2) using the computed $\bar{Q}_1$ and $\bar{R}$ from MGS. In §7 we consider when we might use MGS in preference to the Householder QR factorization of A.

2. Modified Gram–Schmidt as a Householder method. The MGS algorithm computes a sequence of matrices $A = A^{(1)}, A^{(2)}, \cdots, A^{(n+1)} = Q_1 \in \mathbf{R}^{m\times n}$, where $A^{(k)} = (q_1, \cdots, q_{k-1}, a_k^{(k)}, \cdots, a_n^{(k)})$. Here the first $(k-1)$ columns are final columns in Q_1, and $a_k^{(k)}, \cdots, a_n^{(k)}$ have been made orthogonal to $q_1, \cdots, q_{k-1}$. In the kth step we take

$$q_k' = a_k^{(k)}, \quad \rho_{kk} = \|q_k'\|_2, \quad q_k = q_k'/\rho_{kk}, \tag{2.1}$$

and orthogonalize $a_{k+1}^{(k)}, \cdots, a_n^{(k)}$ against q_k using the orthogonal projector $I - q_k q_k^T$,

$$\begin{aligned} a_j^{(k+1)} &= (I - q_k q_k^T) a_j^{(k)} = a_j^{(k)} - q_k \rho_{kj}, \\ \rho_{kj} &= q_k^T a_j^{(k)}, \qquad j = k+1, \cdots, n. \end{aligned} \tag{2.2}$$

We see $A^{(k)} = A^{(k+1)} R_k$ where R_k has the same kth row as upper triangular $R \equiv (\rho_{ij})$, but is the unit matrix otherwise. After n steps we have obtained the factorization

$$A = A^{(1)} = A^{(2)} R_1 = A^{(3)} R_2 R_1 = A^{(n+1)} R_n \cdots R_1 = Q_1 R, \tag{2.3}$$

where in exact arithmetic the columns of Q_1 are orthonormal by construction. Note that in MGS, as opposed to the classical version, all the projections $q_k \rho_{kj}$ are subtracted from the $a_j^{(k)}$ sequentially as soon as q_k is computed. In practice, a square root free version is often used, where one computes Q_1', R', and $D = \operatorname{diag}(\gamma_1, \cdots, \gamma_n)$ in the scaled factorization, taking q_k' as above,

$$A = Q_1' R', \qquad Q_1' = (q_1', \cdots, q_n'), \qquad \gamma_k = (q_k')^T q_k', \qquad k = 1, \cdots, n, \tag{2.4}$$

with $R' = (\rho_{kj}')$ *unit* upper triangular, and $\rho_{kj}' = (q_k')^T a_j^{(k)}/\gamma_k$, $j > k$.

It was reported in [4] that MGS for the QR factorization can be interpreted as Householder's method applied to the matrix A *augmented with a square matrix of zero elements on top.* This is not only true in theory, but in the presence of rounding errors as well. This observation is originally due to Charles Sheffield, and was communicated to the authors by Gene Golub. Because it is such an important but unexpected result, we will discuss this relationship in some detail. First we look at the theoretical result.

Let $A \in \mathbf{R}^{m\times n}$ have rank n, and let $O_n \in \mathbf{R}^{n\times n}$ be a zero matrix. Consider the two QR factorizations (here we use Q for $m \times m$ and P for $(m+n) \times (m+n)$ orthogonal matrices),

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix},$$

$$\tilde{A} \equiv \begin{pmatrix} O_n \\ A \end{pmatrix} = P \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix} = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix} \begin{pmatrix} \tilde{R} \\ 0 \end{pmatrix}. \tag{2.5}$$

Since A has rank n, then P_{11} is zero, P_{21} is an $m \times n$ matrix of orthonormal columns, and $A = Q_1 R = P_{21}\tilde{R}$. If upper triangular R and $\tilde{R}$ are both chosen to have positive diagonal elements in $A^T A = R^T R = \tilde{R}^T \tilde{R}$, then $R = \tilde{R}$ by uniqueness, so $P_{21} = Q_1$ can be found from any QR factorization of the augmented matrix. The last m columns of P are then arbitrary up to an $m \times m$ orthogonal multiplier. The important result is that the *Householder* QR factorization of the augmented matrix is *numerically* equivalent to MGS applied to A.

To see this, remember that with e_k the kth column of the unit matrix, the Householder transformation $Pa = e_1 \rho$ uses $P = I - 2vv^T/\|v\|_2^2$, $v = a - e_1\rho$, $\rho = \pm\|a\|_2$. If (2.5) is obtained using Householder transformations, then

$$P^T = P_n \cdots P_2 P_1, \quad P_k = I - 2\hat{v}_k \hat{v}_k^T / \|\hat{v}_k\|_2^2, \quad k = 1, \cdots, n, \tag{2.6}$$

where the vectors $\hat{v}_k$ are described below. Now from MGS applied to $A^{(1)} = A$, $\rho_{11} = \|a_1^{(1)}\|_2$ and $a_1^{(1)} = q_1' = q_1 \rho_{11}$, so for the first Householder transformation applied to the augmented matrix

$$\tilde{A}^{(1)} \equiv \begin{pmatrix} O_n \\ A^{(1)} \end{pmatrix}, \qquad \tilde{a}_1^{(1)} = \begin{pmatrix} 0 \\ a_1^{(1)} \end{pmatrix},$$
$$\hat{v}_1^{(1)} \equiv \begin{pmatrix} -e_1 \rho_{11} \\ q_1' \end{pmatrix} = \rho_{11} v_1, \qquad v_1 = \begin{pmatrix} -e_1 \\ q_1 \end{pmatrix}$$

(since there can be no cancellation we take $\rho_{kk} \geq 0$). But $\|v_1\|_2^2 = 2$, giving

$$P_1 = I - 2\hat{v}_1 \hat{v}_1^T / \|\hat{v}_1\|_2^2 = I - 2v_1 v_1^T / \|v_1\|_2^2 = I - v_1 v_1^T,$$

and

$$P_1 \tilde{a}_j^{(1)} = \tilde{a}_j^{(1)} - v_1 v_1^T \tilde{a}_j^{(1)} = \begin{pmatrix} 0 \\ a_j^{(1)} \end{pmatrix} - \begin{pmatrix} -e_1 \\ q_1 \end{pmatrix} q_1^T a_j^{(1)} = \begin{pmatrix} e_1 \rho_{1j} \\ a_j^{(2)} \end{pmatrix},$$

so

$$P_1 \tilde{A}^{(1)} = \begin{pmatrix} \rho_{11} & \rho_{12} & \cdots & \rho_{1n} \\ 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & 0 \\ 0 & a_2^{(2)} & \cdots & a_n^{(2)} \end{pmatrix},$$

where these values are clearly *numerically* the same as in the first step of MGS on A. We see that the next Householder transformation produces the second row of R and $a_3^{(3)}, \cdots, a_n^{(3)}$, just as in MGS. Carrying on this way we see that this Householder QR is numerically equivalent to MGS applied to A, and that every P_k is effectively defined by Q_1, since

$$P_k = I - v_k v_k^T, \quad v_k = \begin{pmatrix} -e_k \\ q_k \end{pmatrix}, \quad k = 1, \cdots, n. \tag{2.7}$$

P gives us a key to understanding the *numerical* behavior of MGS. First note that *in theory* $v_i^T v_j = e_i^T e_j + q_i^T q_j = 0$ if $i \neq j$, so $P_i P_j = I - v_i v_i^T - v_j v_j^T$, and

$P^T = P_n \cdots P_1 = I - v_1 v_1^T - v_2 v_2^T - \cdots - v_n v_n^T$ is symmetric, so using Householder transformations in (2.5),

$$P_{11} = 0,$$
$$P_{12}^T = P_{21} = q_1 e_1^T + \cdots + q_n e_n^T = Q_1,$$
$$P_{22} = I - q_1 q_1^T - \cdots - q_n q_n^T = I - Q_1 Q_1^T = Q_2 Q_2^T.$$

This shows that such special orthogonal matrices are fully defined by their $(1,2)$ blocks,

$$P = \begin{pmatrix} O_n & Q_1^T \\ Q_1 & I - Q_1 Q_1^T \end{pmatrix}. \tag{2.8}$$

3. Accuracy of R from modified Gram–Schmidt. A rounding error analysis of MGS was given in [3]. There it was shown that the *computed* $\bar{Q}_1$ and $\bar{R}$ satisfy

$$\begin{aligned} A + \bar{E} = \bar{Q}_1 \bar{R}, \qquad & \|\bar{E}\|_2 \le \bar{c}_1 u \|A\|_2, \\ \|I - \bar{Q}_1^T \bar{Q}_1\|_2 \le \bar{c}_2 \kappa u, \qquad & \end{aligned} \tag{3.1}$$

where $\bar{c}_i$ are constants depending on m, n and the details of the arithmetic, and u is the unit roundoff. Hence $\bar{Q}_1 \bar{R}$ accurately represents A and the departure from orthogonality can be bounded in terms of the condition number $\kappa = \sigma_1/\sigma_n$.

From the numerical equivalence shown in the previous section, it follows that the backward error analysis for the Householder QR factorization of the augmented matrix in (2.5) can also be applied to the MGS on A. Here we will do this, and in this section and §5 we will rederive (3.1) as well as give some new results. This is a simple and unified approach, in that the one analysis of orthogonal transformations can be used to analyse the QR factorization via both Householder transformations and MGS. It also deepens our understanding of the MGS algorithm and its possible uses.

Let $\bar{Q}_1 = (\bar{q}_1, \cdots, \bar{q}_n)$ be the matrix of vectors computed by MGS, and for $k = 1, \cdots, n$ define

$$\bar{v}_k = \begin{pmatrix} -e_k \\ \bar{q}_k \end{pmatrix}, \qquad \bar{P}_k = I - \bar{v}_k \bar{v}_k^T, \qquad \bar{P} = \bar{P}_1 \bar{P}_2 \cdots \bar{P}_n, \tag{3.2}$$

$$\tilde{q}_k = \bar{q}_k / \|\bar{q}_k\|_2, \quad \tilde{v}_k = \begin{pmatrix} -e_k \\ \tilde{q}_k \end{pmatrix}, \quad \tilde{P}_k = I - \tilde{v}_k \tilde{v}_k^T, \quad \tilde{P} = \tilde{P}_1 \tilde{P}_2 \cdots \tilde{P}_n.$$

Then $\bar{P}_k$ is the computed version of the Householder matrix applied in the kth step of the Householder QR factorization of $\binom{O_n}{A}$, and $\tilde{P}_k$ is its orthonormal equivalent, so that $\tilde{P}_k^T \tilde{P}_k = I$. Wilkinson [11, pp. 153–162] has given a general error analysis of orthogonal transformations of this type. From this it follows that for $\bar{R}$ computed by MGS, the equivalent of (2.5) is

$$\begin{pmatrix} E_1 \\ A + E_2 \end{pmatrix} = \tilde{P} \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}, \qquad \bar{P} = \tilde{P} + E',$$

$$\|E_i\|_2 \le c_i u \|A\|_2, \quad i = 1, 2, \quad \|E'\|_2 \le c_3 u, \tag{3.3}$$

where again c_i are constants depending on m, n and the details of the arithmetic.

To show that this $\bar{R}$ from MGS, or the Householder QR factorization of the augmented matrix, is numerically about as good as that from the ordinary Householder QR factorization of A, we use the following general result.

LEMMA 3.1. *For* any *matrices satisfying*

$$\begin{pmatrix} E_1 \\ A+E_2 \end{pmatrix} = \begin{pmatrix} P_{11} \\ P_{21} \end{pmatrix} R, \qquad P_{11}^T P_{11} + P_{21}^T P_{21} = I,$$

there exist $\hat{Q}_1$ *and* E *such that*

$$A + E = \hat{Q}_1 R, \qquad \hat{Q}_1^T \hat{Q}_1 = I,$$

$$\|\hat{Q}_1 - P_{21}\|_2 \le \|P_{11}\|_2^2, \tag{3.4}$$

$$\|(\hat{Q}_1 - P_{21})R\|_2 \le \|P_{11}\|_2 \|E_1\|_2, \tag{3.5}$$

$$\|E\|_2 \le \|P_{11}\|_2 \|E_1\|_2 + \|E_2\|_2 \le \|E_1\|_2 + \|E_2\|_2. \tag{3.6}$$

Proof. Consider the CS decomposition (see, for example, [7, p. 77]) $P_{11} = U_1 C W^T$, $P_{21} = V_1 S W^T$, where $U = (U_1, U_2)$, $V = (V_1, V_2)$ are square orthonormal matrices and C and S are nonnegative diagonal matrices with $C^2 + S^2 = I$. Define $\hat{Q}_1 \equiv V_1 W^T$, the closest orthonormal matrix to P_{21} in any unitarily invariant norm; then since $(I+S)(I-S) = C^2$,

$$\begin{aligned} \hat{Q}_1 - P_{21} &= V_1(I-S)W^T = V_1(I+S)^{-1}W^T W C U_1^T U_1 C W^T \\ &= V_1(I+S)^{-1}W^T P_{11}^T P_{11}, \\ (\hat{Q}_1 - P_{21})R &= V_1(I+S)^{-1}W^T P_{11}^T E_1, \end{aligned}$$

from which the first two bounds follow. Next,

$$E = \hat{Q}_1 R - A = (\hat{Q}_1 - P_{21})R + E_2,$$

from which the third bound follows. □

Using these results we see when $\bar{R}$ is computed using MGS, so $\bar{R}$ satisfies (3.3), there exists *orthonormal* $\hat{Q}_1$ such that, writing $c = c_1 + c_2$,

$$A + E = \hat{Q}_1 \bar{R}, \qquad \hat{Q}_1^T \hat{Q}_1 = I, \qquad \|E\|_2 \le cu\|A\|_2. \tag{3.7}$$

This means if $\bar{\sigma}_1 \ge \cdots \ge \bar{\sigma}_n$ are the singular values of $\bar{R}$, and $\sigma_1 \ge \cdots \ge \sigma_n$ are those of A,

$$|\,\bar{\sigma}_i - \sigma_i\,| \le cu\sigma_1, \qquad i = 1, \cdots, n. \tag{3.8}$$

Thus $\bar{R}$ from MGS is not only the same as $\bar{R}$ from the Householder QR factorization applied to A augmented by a square block of zeros, but (3.7) shows it is comparable in accuracy to the upper triangular matrix from the Householder or Givens QR factorization applied to A alone. Also (3.8) shows that the singular values of $\bar{R}$ are very close to those of A. This means we could use MGS as a first step in finding the singular values of A, and justifies an algorithm by Longley in [9, Chap. 9]. Since we have not required A to be full rank as yet in this section, this fact also ensures that $\bar{R}$ from MGS can be used in any computation for finding the rank of A. Here we will just use this knowledge to simplify our bounds below.

In fact, $\bar{R}$ is usually even better than (3.7) suggests. We see $\bar{R}$ is nonsingular if $cu\sigma_1 < \sigma_n$, that is, if $cu\kappa < 1$, so we make the following assumption and definition,

$$cu\kappa < 1, \qquad \eta \equiv (1 - cu\kappa)^{-1}, \tag{3.9}$$

where usually $\eta \sim 1$. Then

$$\|A\|_2\|\bar{R}^{-1}\|_2 = \sigma_1/\bar{\sigma}_n \le \sigma_1/(\sigma_n - cu\sigma_1) = \eta\kappa, \tag{3.10}$$

and $E_1 = \tilde{P}_{11}\bar{R}$, so

$$\|\tilde{P}_{11}\|_2 = \|E_1\bar{R}^{-1}\|_2 \le c_1 u\eta\kappa, \qquad \|\bar{P}_{11}\|_2 \le (c_1\eta\kappa + c_3)u. \tag{3.11}$$

From (3.6),

$$\|E\|_2 \le \|\tilde{P}_{11}\|_2\|E_1\|_2 + \|E_2\|_2 \le c_1^2u^2\eta\kappa\|A\|_2 + \|E_2\|_2, \tag{3.12}$$

showing that the first term on the right will be negligible if $\eta c_1 u\kappa \ll 1$, which is usually true.

We will show how all of $\tilde{P}$ and $\bar{P}$ depend crucially on $\tilde{P}_{11}$ and $\bar{P}_{11}$, respectively, so the bounds in (3.11) are important in understanding the loss of orthogonality in MGS. Since $\bar{R}$ is numerically about as good as we can hope for, it is clear that the main drawback of MGS is this lack of orthogonality in $\bar{Q}_1 = (\bar{q}_1, \cdots, \bar{q}_n)$, so we examine this in the next two sections. (As is mentioned in §7, another less important drawback is that the operation count is slightly higher for MGS than for the Householder QR factorization.)

4. Structure of P, $\bar{P}$, and $\tilde{P}$ from the Householder QR factorization of the augmented matrix. It is well known that the orthogonality of the ideal Q_1 is lost in MGS because of cancellation in the subtractions in (2.2), and that this can give a severely nonorthogonal computed $\bar{Q}_1$. In order to understand this loss fully and later to bound it, the following theorem provides the detailed structures of $\bar{P}$ and $\tilde{P}$ in (3.2) as functions of the computed $\bar{Q}_1$ and the normalized $\tilde{Q}_1 \equiv (\tilde{q}_1, \cdots, \tilde{q}_n)$, respectively. Note that the theorem is for general $Q_1 = (q_1, \cdots, q_n)$, and so will apply to P, $\bar{P}$, and $\tilde{P}$. The idea is that *any* matrix $P = P_1P_2\cdots P_n$ with $P_k = I - v_kv_k^T$ and $v_k^T = (-e_k^T, q_k^T)$ has a very special structure, and the theorem reveals this. In this structure the whole matrix is seen to depend only on the leading $n \times n$ block P_{11} of P, and on Q_1. But we have bounds on *our* $\tilde{P}_{11}$ and $\bar{P}_{11}$ in (3.11), and so will be able to understand and bound the loss of orthogonality in $\tilde{Q}_1$ or $\bar{Q}_1$ from MGS. Furthermore, all such P_{11} have special structure too, being strictly upper triangular.

THEOREM 4.1. *Let $Q_1 = (q_1, \cdots, q_n) \in \mathbf{R}^{m\times n}$, and for $k = 1, \cdots, n$, define*

$$M_k = I - q_kq_k^T, \qquad v_k = \begin{pmatrix} -e_k \\ q_k \end{pmatrix} \in \mathbf{R}^{m+n}, \qquad P_k = I - v_kv_k^T.$$

Then with the partitioning we use throughout this theorem

$$P \equiv P_1P_2\cdots P_n \equiv \begin{array}{c} \\ n \\ m \end{array} \begin{array}{c} \begin{array}{cc} n & m \end{array} \\ \left(\begin{array}{c|c} P_{11} & P_{12} \\ \hline P_{21} & P_{22} \end{array}\right) \end{array} \tag{4.1}$$

$$= \left(\begin{array}{ccccc|c} 0 & q_1^Tq_2 & q_1^TM_2q_3 & \cdots & q_1^TM_2M_3\cdots M_{n-1}q_n & q_1^TM_2M_3\cdots M_n \\ 0 & 0 & q_2^Tq_3 & \cdots & q_2^TM_3M_4\cdots M_{n-1}q_n & q_2^TM_3M_4\cdots M_n \\ \vdots & \vdots & \vdots & \cdots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & q_{n-1}^Tq_n & q_{n-1}^TM_n \\ 0 & 0 & 0 & \cdots & 0 & q_n^T \\ \hline q_1 & M_1q_2 & M_1M_2q_3 & \cdots & M_1M_2\cdots M_{n-1}q_n & M_1M_2\cdots M_n \end{array}\right)$$

$$= \left(\begin{array}{c|c} P_{11} & (I-P_{11})Q_1^T \\ \hline Q_1(I-P_{11}) & I-Q_1(I-P_{11})Q_1^T \end{array}\right). \tag{4.2}$$

P is orthonormal if and only if $\|q_k\|_2 = 1$ for $k = 1, \cdots, n$; $P_{11} = 0$ if and only if $Q_1^T Q_1$ is diagonal.

There is a short proof that does not give (4.1), but since (4.1) reveals the detailed structure of P, we give a longer proof. Note that if q_k has length 1, then M_k is a projector, and from (4.1) the second column of P_{21} is that part of q_2 orthogonal to q_1; the third is q_3 orthogonalized against q_2 and the result orthogonalized against q_1, and so on. However, this is not the same as reorthogonalizing the q_k.

Proof. To determine the first n columns of $P = P_1 P_2 \cdots P_n$, note that

$$P_k = I - v_k v_k^T = I - \left(\frac{-e_k}{q_k}\right)\left(\begin{array}{c|c} -e_k^T & q_k^T \end{array}\right) = \left(\begin{array}{c|c} I - e_k e_k^T & e_k q_k^T \\ \hline q_k e_k^T & M_k \end{array}\right)$$

and let $1 \le j \le n$. If $j \ne k$ then $P_k e_j = e_j$, while

$$P_j e_j = \begin{pmatrix} 0 \\ I_m \end{pmatrix} q_j,$$

so

$$\begin{aligned} Pe_j = P_1 P_2 \cdots P_n e_j = P_1 P_2 \cdots P_j e_j &= P_1 P_2 \cdots P_{j-2} \left(\frac{e_{j-1} q_{j-1}^T}{M_{j-1}}\right) q_j \\ &= P_1 P_2 \cdots P_{j-3} \left(\frac{e_{j-1} q_{j-1}^T + e_{j-2} q_{j-2}^T M_{j-1}}{M_{j-2} M_{j-1}}\right) q_j \end{aligned}$$

$$= \left(\begin{array}{c} q_1^T M_2 \cdots M_{j-1} q_j \\ q_2^T M_3 \cdots M_{j-1} q_j \\ \vdots \\ q_{j-2}^T M_{j-1} q_j \\ q_{j-1}^T q_j \\ 0 \\ \vdots \\ 0 \\ \hline M_1 M_2 \cdots M_{j-1} q_j \end{array}\right) = \left(\frac{P_{11}}{P_{21}}\right) e_j = \left(\frac{p_{1j}}{p_{2j}}\right) \equiv \left(\begin{array}{c} \pi_{1j} \\ \pi_{2j} \\ \vdots \\ \pi_{j-2,j} \\ \pi_{j-1,j} \\ \pi_{jj} \\ \vdots \\ \pi_{nj} \\ \hline p_{2j} \end{array}\right), \tag{4.3}$$

say, which gives the $(1,1)$ and $(2,1)$ blocks of (4.1). For the last m columns we have

$$\left(\frac{P_{12}}{P_{22}}\right) = P\left(\frac{0}{I_m}\right) = P_1 P_2 \cdots P_{n-1} \left(\frac{e_n q_n^T}{M_n}\right)$$

$$= P_1 P_2 \cdots P_{n-2} \left(\frac{e_n q_n^T + e_{n-1} q_{n-1}^T M_n}{M_{n-1} M_n}\right) = \left(\begin{array}{c} q_1^T M_2 \cdots M_n \\ q_2^T M_3 \cdots M_n \\ \vdots \\ q_{n-1}^T M_n \\ q_n^T \\ \hline M_1 \cdots M_n \end{array}\right), \tag{4.4}$$

which completes the proof of (4.1). Next, from (4.3),

$$P_{21} e_j = (I - q_1 q_1^T) M_2 M_3 \cdots M_{j-1} q_j$$

$$
\begin{aligned}
&= M_2 M_3 \cdots M_{j-1} q_j - q_1 \pi_{1j} \\
&= M_3 M_4 \cdots M_{j-1} q_j - q_1 \pi_{1j} - q_2 \pi_{2j} \\
&= M_{j-1} q_j - q_1 \pi_{1j} - q_2 \pi_{2j} - \cdots - q_{j-2} \pi_{j-2,j} \\
&= q_j - q_1 \pi_{1j} - q_2 \pi_{2j} - \cdots - q_{j-1} \pi_{j-1,j} \\
&= Q_1 e_j - Q_1 P_{11} e_j,
\end{aligned}
$$

so $P_{21} = Q_1(I - P_{11})$, giving the $(2,1)$ block of (4.2). Next, from (4.4),

$$
\begin{aligned}
e_i^T P_{12} &= q_i^T M_{i+1} \cdots M_{n-1} M_n \\
&= q_i^T M_{i+1} \cdots M_{n-1} - q_i^T M_{i+1} \cdots M_{n-1} q_n q_n^T \\
&= q_i^T M_{i+1} \cdots M_{n-1} - \pi_{in} q_n^T \\
&= q_i^T M_{i+1} \cdots M_{n-2} - \pi_{i,n-1} q_{n-1}^T - \pi_{in} q_n^T \\
&= q_i^T M_{i+1} - \pi_{i,i+2} q_{i+2}^T - \cdots - \pi_{in} q_n^T \\
&= q_i^T - \pi_{i,i+1} q_{i+1}^T - \cdots - \pi_{in} q_n^T \\
&= e_i^T Q_i^T - e_i^T P_{11} Q_1^T,
\end{aligned}
$$

so $P_{12} = (I - P_{11}) Q_1^T$, giving the $(1,2)$ block of (4.2). We can now use the structure of P_{21} in (4.1) to give

$$
\begin{aligned}
P_{22} &= M_1 M_2 \cdots M_n \\
&= M_1 M_2 \cdots M_{n-1} - M_1 M_2 \cdots M_{n-1} q_n q_n^T \\
&= M_1 M_2 \cdots M_{n-1} - P_{21} e_n q_n^T \\
&= M_1 M_2 \cdots M_{n-2} - P_{21} e_{n-1} q_{n-1}^T - P_{21} e_n q_n^T \\
&= I - q_1 q_1^T - P_{21} e_2 q_2^T - P_{21} e_3 q_3^T - \cdots - P_{21} e_n q_n^T \\
&= I - P_{21}(e_1 q_1^T + e_2 q_2^T + \cdots + e_n q_n^T) \\
&= I - P_{21} Q_1^T = I - Q_1 (I - P_{11}) Q_1^T,
\end{aligned}
$$

completing the proof of (4.2).

Clearly P_k is orthonormal if $q_k^T q_k = 1$, so if $\|q_k\|_2 = 1$ for $k = 1, \cdots, n$, then P is orthonormal. Now suppose P is orthonormal; then $Pe_1 = P_1 e_1 = (0, q_1^T)^T$ must have length 1, so $\|q_1\|_2 = 1$ and P_1 and so $P_2 P_3 \cdots P_n$ is orthonormal. But then $P_2 P_3 \cdots P_n e_2 = P_2 e_2 = (0, q_2^T)^T$ must have length 1, and so on. Finally we see from (4.1) that the ith row of P_{11} is zero if and only if $q_i^T q_j = 0$ for $j = i+1, \cdots, n$, proving $P_{11} = 0$ if and only if $Q_1^T Q_1$ is diagonal. □

Since each of P (see (2.6) and (2.7)), $\bar{P}$ and $\tilde{P}$ (see (3.2)) has the structure of P, in the theorem, P has the form (2.8), and

$$
\tilde{P} = \begin{pmatrix} \tilde{P}_{11} & (I - \tilde{P}_{11}) \tilde{Q}_1^T \\ \tilde{Q}_1 (I - \tilde{P}_{11}) & I - \tilde{Q}_1 (I - \tilde{P}_{11}) \tilde{Q}_1^T \end{pmatrix} \tag{4.5}
$$

for some strictly upper triangular $\tilde{P}_{11}$, with $\bar{P}$ having a similar form. This shows how $\tilde{Q}_1$ loses orthogonality when $\tilde{P}_{11}$ is nonzero. Clearly, P and $\tilde{P}$ are orthogonal matrices, so their first n columns form orthonormal sets. Since P_{11} is zero, Q_1 is clearly an $m \times n$ matrix of orthonormal columns, but all we can say about the size of $\tilde{P}_{11}$ is $\|\tilde{P}_{11}\|_2 \leq c_1 u \eta \kappa$, from (3.11). If κ is not very much greater than 1, then $\tilde{P}_{11}$ is small, and from (4.5), $\tilde{Q}_1$ has nearly orthonormal columns. For larger κ, (4.5) shows how the columns of $\tilde{Q}_1$ can become less and less orthogonal, losing all

likelihood of orthogonality when $c_1 u\eta\kappa \simeq 1$. Clearly column pivoting would be useful in maintaining orthogonality as long as possible, and in revealing the rank of rank deficient A. Since $\tilde{Q}_1$ is just $\bar{Q}_1$ with normalized columns, the same comments on orthogonality apply to $\bar{Q}_1$. We will bound these losses of orthogonality in the next section, and show how to avoid them after that.

5. Loss of orthogonality in $\bar{Q}_1$ and $\tilde{Q}_1$ from MGS. Each column of $\tilde{Q}_1$ is just the correctly normalized column of the computed $\bar{Q}_1$ from MGS, whose columns already have norm almost 1, so what we prove for $\tilde{Q}_1$ effectively holds for $\bar{Q}_1$. We saw from Theorem 4.1 that the first n columns $\tilde{P}^{(n)}$ of $\tilde{P}$ are orthonormal and

$$\tilde{P}^{(n)} = \begin{pmatrix} \tilde{P}_{11} \\ \tilde{Q}_1 - \tilde{Q}_1\tilde{P}_{11} \end{pmatrix} = \begin{pmatrix} \tilde{P}_{11} \\ \tilde{P}_{21} \end{pmatrix} I, \qquad \tilde{P}^{(n)T}\tilde{P}^{(n)} = I,$$

so an easy result is obtained by applying Lemma 3.1 with $R = I$, $A = \tilde{Q}_1$, $E_1 = \tilde{P}_{11}$, and $E_2 = -\tilde{Q}_1\tilde{P}_{11}$, showing that there exist $\hat{Q}_1$ and E such that $\tilde{Q}_1 + E = \hat{Q}_1$ with $\hat{Q}_1^T\hat{Q}_1 = I$ and

$$\|E\|_2 = \|\tilde{Q}_1 - \hat{Q}_1\|_2 \le (\|\tilde{P}_{11}\|_2 + \|\tilde{Q}_1\|_2)\|\tilde{P}_{11}\|_2.$$

But then $\|\tilde{Q}_1\|_2 \le 1 + \|E\|_2$, giving

$$\|E\|_2 \le \|\tilde{P}_{11}\|_2(1 + \|\tilde{P}_{11}\|_2)/(1 - \|\tilde{P}_{11}\|_2),$$

and a bound on the distance of $\tilde{Q}_1$ from an orthogonal matrix when $c_1 u\eta\kappa < 1$,

$$\|\tilde{Q}_1 - \hat{Q}_1\|_2 \le c_1 u\eta\kappa \frac{1 + c_1 u\eta\kappa}{1 - c_1 u\eta\kappa}, \tag{5.1}$$

which for $c_1 u\eta\kappa \ll 1$ is effectively $c_1 u\eta\kappa$.

In order to bound the departure of $\tilde{Q}_1^T\tilde{Q}_1$ from the unit matrix, we could use (5.1) directly, but a more revealing result follows by noting in (3.3) that $E_1 = \tilde{P}_{11}\bar{R}$ is strictly upper triangular, since $\tilde{P}_{11}$ is so from Theorem 4.1. Thus

$$\tilde{P}^{(n)}\bar{R} = \begin{pmatrix} \tilde{P}_{11} \\ \tilde{Q}_1(I - \tilde{P}_{11}) \end{pmatrix} \bar{R} = \begin{pmatrix} E_1 \\ \tilde{Q}_1(\bar{R} - E_1) \end{pmatrix},$$

so that

$$\begin{aligned}(\bar{R} - E_1)^T\tilde{Q}_1^T\tilde{Q}_1(\bar{R} - E_1) &= \bar{R}^T\bar{R} - E_1^TE_1 \\ &= (\bar{R} - E_1)^T(\bar{R} - E_1) + (\bar{R} - E_1)^TE_1 + E_1^T(\bar{R} - E_1).\end{aligned}$$

Since $\bar{R}$ is nonsingular upper triangular, and E_1 is strictly upper triangular, $\bar{R} - E_1$ is nonsingular upper triangular, and

$$\tilde{Q}_1^T\tilde{Q}_1 = I + E_1(\bar{R} - E_1)^{-1} + (\bar{R} - E_1)^{-T}E_1^T, \tag{5.2}$$

with $E_1(\bar{R} - E_1)^{-1}$ the strictly upper triangular part of $\tilde{Q}_1^T\tilde{Q}_1$. This gives a clear picture of exactly how the loss of orthogonality depends on the computed $\bar{R}$. Thus from (3.3) and (3.8)–(3.10), if $(c + c_1)u\kappa < 1$, we obtain the bound

$$\|I - \tilde{Q}_1^T\tilde{Q}_1\|_2 \le \frac{2c_1 u\kappa}{1 - (c + c_1)u\kappa}, \tag{5.3}$$

and a loss of orthogonality of this magnitude can often be observed in practice.

The bound (5.3) is of similar form to the bound (3.1) given in [3], but here we also derived the relation of $\tilde{Q}_1$ to the orthonormal matrix $\tilde{P}$, and described the relation between the loss of orthogonality in $\tilde{Q}_1$ and the deviation of $\tilde{P}$ from the ideal form of P. We also note here that if the first k columns of A in (3.3) have a small κ, then the first k columns of $\tilde{P}_{11}$ will be small, and the first k columns of $\tilde{Q}_1$ will be nearly orthonormal.

Our main purpose is not to show how $\tilde{Q}_1$ or $\bar{Q}_1$ may be improved. Instead, the key point of this work is that although the computed $\bar{P}$ is very close to the exactly orthogonal $\tilde{P}$ in (3.3), the columns of $\bar{Q}_1$ need not be particularly orthonormal. Our thesis here is that as a result of this, it is usually inadvisable to use $\bar{Q}_1$ as our set of orthonormal vectors, but we *can* use $\bar{P}$ (as the theoretical product of the computed $\bar{P}_k = I - \bar{v}_k \bar{v}_k^T$, which is extremely close to $\tilde{P}$), to make use of the desired orthogonality, since we have all the necessary information in $\bar{Q}_1$, that is, $\bar{v}_k^T = (-e_k^T, \bar{q}_k^T)$. Thus we can solve problems as accurately using MGS as we can using Householder or Givens QR factorizations if, instead of using the computed $\bar{Q}_1$ directly, we formulate the problems in terms of (2.5) (see (3.3)) and use the $\bar{q}_k$ to define $\bar{P}$. Of course, in most cases no block of $\bar{P}$ need actually be formed. We illustrate an important use of this idea in the next section, and discuss the efficiency of such an approach in §7.

6. Backward stable solution of the ASF using MGS. Björck [2] showed that (1.5) is backward stable for the ASF (1.2) using the Householder QR factorization, but the same is not true when we use (1.6) with $\bar{R}$ and $\bar{Q}_1$ computed by MGS; see [5]. Here we use our new knowledge of MGS to produce a backward stable algorithm for the ASF based on $\bar{Q}_1$ and $\bar{R}$ from MGS. This new approach can be used to design good algorithms using MGS in general.

Our original ASF (1.2) is equivalent to the augmented system

$$\begin{pmatrix} I & 0 & 0 \\ 0 & I & A \\ 0 & A^T & 0 \end{pmatrix} \begin{pmatrix} w \\ x \\ y \end{pmatrix} = \begin{pmatrix} 0 \\ b \\ c \end{pmatrix}, \tag{6.1}$$

so applying Householder transformations as in (2.5) gives the augmented version of the method (1.5) as

$$z = R^{-T}c, \quad \begin{pmatrix} d \\ h \end{pmatrix} = P^T \begin{pmatrix} 0 \\ b \end{pmatrix}, \quad \begin{pmatrix} w \\ x \end{pmatrix} = P \begin{pmatrix} z \\ h \end{pmatrix}, \quad y = R^{-1}(d - z). \tag{6.2}$$

But as we saw in §2, we can use the q_k from MGS to produce $P_k = I - v_k v_k^T$, $v_k^T = (-e_k^T, q_k^T)$, and use $P^T = P_n \cdots P_2 P_1$ in (6.2). We show in [5] that this algorithm is *strongly* stable (see [6]) for (6.1), and also strongly stable for (1.2).

We now show how to take advantage of the structure of the P_k; then we will summarize this numerically stable use of MGS for the ASF. To compute d and h in (6.2) note that $P^T = P_n \cdots P_1$, and define

$$\begin{pmatrix} d^{(1)} \\ h^{(1)} \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}, \qquad \begin{pmatrix} d^{(k+1)} \\ h^{(k+1)} \end{pmatrix} = P_k \cdots P_1 \begin{pmatrix} 0 \\ b \end{pmatrix} = P_k \begin{pmatrix} d^{(k)} \\ h^{(k)} \end{pmatrix}.$$

Now using induction we see $d^{(k)}$ has all but its first $k-1$ elements zero, and

$$\begin{pmatrix} d^{(k+1)} \\ h^{k+1)} \end{pmatrix} = \begin{pmatrix} d^{(k)} \\ h^{(k)} \end{pmatrix} - \begin{pmatrix} -e_k \\ q_k \end{pmatrix} \begin{pmatrix} -e_k^T & q_k^T \end{pmatrix} \begin{pmatrix} d^{(k)} \\ h^{(k)} \end{pmatrix} = \begin{pmatrix} d^{(k)} + e_k(q_k^T h^{(k)}) \\ h^{(k)} - q_k(q_k^T h^{(k)}) \end{pmatrix},$$

giving the computation starting with $h^{(1)} := b$,

$$\textbf{for } k = 1, \cdots, n \quad \textbf{do}\{\delta_k := q_k^T h^{(k)};\ h^{(k+1)} := h^{(k)} - q_k \delta_k\},$$

so $h = h^{(n+1)}, d = d^{(n+1)} = (\delta_1, \cdots, \delta_n)^T$. This costs $2mn$ flops (1 flop = one multiplication and one addition in floating point arithmetic), compared with the mn flops required to form $d = Q_1^T b$ in (1.6). The computation for d and h is exactly the same as the one that would arise if the n MGS steps in (2.1)–(2.3) had been applied to (A, b) instead of just A, so that h is theoretically the component of b orthogonal to the columns of A. Note that now d has elements $q_k^T h^{(k)}$ instead of $q_k^T b$, as would be the case in (1.6).

To compute x in (6.2), define

$$\begin{pmatrix} w^{(n)} \\ x^{(n)} \end{pmatrix} = \begin{pmatrix} z \\ h \end{pmatrix}, \qquad \begin{pmatrix} w^{(k-1)} \\ x^{(k-1)} \end{pmatrix} = P_k \cdots P_n \begin{pmatrix} z \\ h \end{pmatrix} = P_k \begin{pmatrix} w^{(k)} \\ x^{(k)} \end{pmatrix},$$

so that

$$\begin{pmatrix} w^{(k-1)} \\ x^{(k-1)} \end{pmatrix} = \begin{pmatrix} w^{(k)} \\ x^{(k)} \end{pmatrix} - \begin{pmatrix} -e_k \\ q_k \end{pmatrix} \left(-e_k^T w^{(k)} + q_k^T x^{(k)} \right),$$

which shows that in this step only the kth element of $w^{(k)}$ is changed from $\zeta_k = e_k^T z$ to $\omega_k = q_k^T x^{(k)}$. This gives the computation starting with $x^{(n)} := h = h^{(n+1)}$,

$$\textbf{for } k = n, \cdots, 1 \ \ \textbf{do}\{\omega_k := q_k^T x^{(k)}; \quad x^{(k-1)} := x^{(k)} - q_k(\omega_k - \zeta_k)\},$$

so $x = x^{(0)}$, $w = (\omega_1, \cdots, \omega_n)^T$. This costs $2mn$ flops compared with mn flops for $x = b - Q_1(d - z)$ in (1.6). From (2.8) we see in theory (6.2) gives $x = Q_1 z + Q_2 Q_2^T h$ where $h = Q_2 Q_2^T b$, so $x = h + Q_1 z$. Note that $w = (\omega_1, \cdots, \omega_n)^T$ is ideally zero (see (6.1)), but can be significant when $\kappa(A)$ is large. The computation of x here can be seen to reorthogonalize each $x^{(k)}$ against the corresponding q_k before adding on $q_k \zeta_k$ to give $x^{(k-1)}$. The complete algorithm is then as follows.

ALGORITHM 6.1. Backward Stable Algorithm for the ASF based on MGS.

1. Carry out MGS on A to give $Q_1 = (q_1, \cdots, q_n)$ and R;
2. Solve $R^T z = c$ for $z = (\zeta_1, \cdots, \zeta_n)^T$;
3. **for** $k = 1, \cdots, n$ **do**$\{\delta_k := q_k^T b;\ b := b - q_k \delta_k\}$;
4. **for** $k = n, \cdots, 1$ **do**$\{\omega_k := q_k^T b;\ b := b - q_k(\omega_k - \zeta_k)\}$; $x := b$;
5. Solve $Ry = d - z$ for y, where $d = (\delta_1, \cdots, \delta_n)^T$.

A weakness in some other MGS-based algorithms is that the reorthogonalization in step 4 is not done. This is the case for the two algorithms denoted (3.4) and (3.6) in [1]. The first is equivalent to (1.6) and the second is the Huang algorithm [8] which, instead of steps 3 and 4, does (using our notation)

$$\textbf{for } k = 1, \cdots, n \quad \textbf{do}\{\delta_k := q_k^T b;\ b := b - q_k(\delta_k - \zeta_k)\}; \quad x := b.$$

The following implementation issues and specializations of the algorithm are fairly obvious. Steps 1, 2, and 3 can be combined, and there is a lot of parallelism inherent in these. When these are complete, steps 4 and 5 can be carried out independently.

For (1.3), step 5 can be omitted if the vector of Lagrange multipliers y is not needed, while for (1.4), step 4 can be omitted if the residual x is not needed.

If $b = 0$, corresponding to LUS, then $d = 0$ and step 3 will be omitted, as will step 5 if the Lagrange multipliers are not needed. If $c = 0$, corresponding to LLS, then $z = 0$ and step 2 will be omitted, and as will step 4 if the LLS residual x is not needed. Then the algorithm is equivalent to the following variant of MGS:

$$(A, b) = (Q_1, h)\begin{pmatrix} R & d \\ 0 & 1 \end{pmatrix}, \qquad y = R^{-1}d, \tag{6.3}$$

where d is computed as part of MGS. This is the approach recommended for LLS in [3]. The work here is another way of proving the backward stability of this approach, and adds insight into why it works. For LUS, however, the numerically stable algorithm made of steps 1, 2, and 4 constitutes a new algorithm which is superior to the usual approach that omits the ω_k in step 4.

If A is square and nonsingular, (1.3) becomes the solution of $A^T x = c$, and x is independent of b, so if y is not wanted, then b can be taken as zero in the algorithm, and steps 3 and 5 dropped. Similarly, if A is square and nonsingular and $c = 0$, then (1.4) becomes $Ay = b$ and steps 2 and 4 can be dropped. This gives *two different* backward stable algorithms for solving nonsingular systems using MGS. Note that the first algorithm applies MGS to the *rows* of the matrix (here A^T) and is numerically invariant under row scalings. The second algorithm applies MGS to the *columns* of A, and is invariant under column scalings. Hence the first algorithm is to be preferred if the matrix is badly row scaled, the second if A is badly column scaled.

A square root free version of Algorithm 6.1 is obtained if we instead use the factorization (2.4) $A = Q_1' R'$, where R' is *unit* upper triangular.

ALGORITHM 6.2.

1. Carry out MGS on A to give $Q_1' = (q_1', \cdots, q_n')$, R', and $D = \operatorname{diag}(\gamma_1, \cdots, \gamma_n)$, where $\gamma_i = \|q_i'\|_2^2$.
2. Solve $(R')^T D z' = c$ for $z' = (\zeta_1', \cdots, \zeta_n')^T$.
3. **for** $k = 1, \cdots, n$ **do**$\{\delta_k' := (q_k')^T b/\gamma_k;\ b := b - q_k'\delta_k'\}$;
4. **for** $k = n, \cdots, 1$ **do**$\{\omega_k' := (q_k')^T b/\gamma_k;\ b := b - q_k'(\omega_k' - \zeta_k')\}$; $x := b$;
5. Solve $R'y = d' - z'$ for y, where $d' = (\delta_1', \cdots, \delta_n')^T$.

This section has not only shown how MGS can be used in a numerically stable way to solve the very useful linear system (1.2), along with its many specializations, but it has hopefully shown how MGS can be used more effectively in general.

7. Comparison of MGS and Householder factorizations. There are four main approaches we need to compare:

(1) MGS on A producing computed $\bar{R}$ and $\bar{Q}_1$, and using these.
(2) MGS on A producing computed $\bar{R}$ and $\bar{Q}_1$, and using $\bar{R}$ and $\bar{P}_1, \cdots, \bar{P}_n$.
(3) Householder transformations on

$$\begin{pmatrix} O_n \\ A \end{pmatrix}$$

producing $\bar{R}$ and $\bar{P}_1, \cdots, \bar{P}_n$ and using these.

(4) Householder transformations on A producing $\hat{R}$ and $\hat{P}_1, \cdots, \hat{P}_n$, say, and using these.

We call these *approaches* rather than algorithms, since each includes a reduction algorithm, plus a choice of tools to use in problems that use the reduction. We only consider the case of a single processor computer, and a dense matrix A.

Approaches (2) and (3) are numerically equivalent, but it is clearly more efficient for computer storage to use approach (2) via (2.1) and (2.2) than to use (3), even though we may *think* in terms of (3) to design algorithms which use the $\bar{P}_1, \cdots, \bar{P}_n$ (these, of course, being "stored" as $\bar{q}_1, \cdots, \bar{q}_n$). Thus we would use the new approach (2) rather than (3) computationally, while being aware of both their properties theoretically.

The most usual case is where we wish to use the orthogonality computationally, but cannot rely on $\kappa(A)$ being small. Then the choice is between (2) and (4). For the initial QR factorization MGS requires mn^2 flops compared to $mn^2 - n^3/3$ for Householder. MGS also needs $n(n-1)/2$ more storage locations. Hence approach (4) has an advantage with respect to both storage and operation count for the initial factorization, although this is small when $m \gg n$.

If accurately orthogonal, Q or Q_1 in (1.1) is required as an entity in itself; then since such orthogonal matrices are not immediately produced by (2) when $\kappa(A)$ is large, the obvious choice is (4), where Q (or Q_1) is available as the product (or part of it) of the $\hat{P}_k$. To produce Q_1 doubles the cost using (4). To produce an accurately orthogonal Q_1 with MGS in general, we apparently need to reorthogonalize. This also approximately doubles the factorization cost, and again the operation count is higher than for Householder.

For both approaches (2) and (4) we have shown backward stability in the usual normwise sense. In agreement with this, both these approaches tend to give similar accuracy, although experience shows that MGS has a small edge here, in particular if the square root free version is used.

If the matrix A is not well row-scaled, then row interchanges may be needed in (4) to give accurate solutions for problem LLS; see [10]. In this context it is interesting to note that MGS is *numerically invariant* under row permutations of A as long as inner products are unaltered by the order of accumulation of terms. That is, if $\bar{Q}_1$ and $\bar{R}$ are the computed factors for A, then $\Pi\bar{Q}_1$ and $\bar{R}$ are the computed factors of ΠA. This shows that (2) is more stable than (4) without row interchanges. However, if row interchanges are included in (4), this approach is more accurate for problems where the row norms of A vary widely. In approach (2) a second-order error term $\sim O((wu)^2)$ appears, where w is the maximum ratio of row norms. This error term can be eliminated by reorthogonalization, which, however, increases the cost of MGS.

We finally mention that sometimes $\bar{R}$ is used alone to solve our problems, and then approaches (1) and (2) are identical. We will discuss this case in [5].

REFERENCES

[1] M. ARIOLI AND A. LARATTA, *Error analysis of algorithms for computing the projection of a point onto a linear manifold*, Linear Algebra Appl., 82 (1986), pp. 1–26.

[2] Å. BJÖRCK, *Iterative refinement of linear least squares solutions* I, BIT, 7 (1967), pp. 257–278.

[3] ———, *Solving linear least squares problems by Gram–Schmidt orthogonalization*, BIT, 7 (1967), pp. 1–21.

[4] ———, *Methods for sparse least squares problems*, in Sparse Matrix Computations, J. Bunch and D. J. Rose, eds., Academic Press, New York, 1976, pp. 177–199.

[5] Å. BJÖRCK AND C. PAIGE, *Solution of augmented linear systems using orthogonal factorizations.* In preparation.
[6] J. R. BUNCH, *The weak and strong stability of algorithms in numerical linear algebra*, Linear Algebra Appl., 88/89 (1987), pp. 49–66.
[7] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, Maryland, 1989.
[8] H. Y. HUANG, *A direct method for the general solution of a system of linear equations*, J. Optim. Theory Appl., 16 (1975), pp. 429–445.
[9] J. W. LONGLEY, *Least Squares Computations Using Orthogonalization Methods*, Marcel Dekker, Inc., New York, 1984.
[10] M. J. D. POWELL AND J. K. REID, *On applying Householder's method to linear least squares problems*, in Proc. Internat. Federation of Information Processing Societies Congress, Ljubljana, Yugoslavia, 1968, pp. 122–126.
[11] J. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 191–201, January 1992

A BLOCK-*LU* UPDATE FOR LARGE-SCALE LINEAR PROGRAMMING*

SAMUEL K. ELDERSVELD† AND MICHAEL A. SAUNDERS†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Stable and efficient updates to the basis matrix factors are vital to the simplex method. The "best" updating method depends on the machine in use and how the update is implemented. For example, the classical product-form update can take advantage of the vector hardware on current supercomputers, and this helps compensate for its well-known drawbacks. Conversely, the method of Bartels and Golub performs well on conventional machines, but is difficult to vectorize.

With vectorization in mind, we examine a method based on the block-*LU* factors of an expanding basis. The partitioned matrix involved was introduced by Bisschop and Meeraus [*Math. Programming*, 13 (1977), pp. 241–254], [*Math. Programming*, 18 (1980), pp. 7–15]. The update itself was proposed by Gill, Murray, Saunders, and Wright [*SIAM J. Sci. Statist. Comput.*, 5 (1984), pp. 562–589].

The main advantages of the block-*LU* update are that it is stable, it vectorizes well, and compared to the product-form update, the nonzeros increase at about two-thirds the rate. The update has been incorporated into MINOS and tested on 30 large, sparse linear programming problems. Results are given from runs on a Cray Y-MP.

Key words. matrix factorization, updating, simplex method, linear programming

AMS(MOS) subject classifications. 65F05, 65F50, 65K05, 90C05

1. Introduction.

We wish to use the simplex method [Dan63] to solve the standard linear programming (LP) problem,

$$\begin{aligned} \text{minimize} \quad & c^T x \\ \text{subject to} \quad & Ax = b \\ & l \leq x \leq u, \end{aligned}$$

where A is an m by n matrix and c, x, l, u, and b are of appropriate dimension.

The simplex method is an *active-set method* for optimization. At each iteration a rank-one modification (in the form of a column update) is made to a basis matrix B associated with constraints active at the current point. After k updates, the columns of A may be permuted to the form $(B_k \quad N_k)$. The next update replaces the pth column a_r of B_k by a column a_q from N_k. It can be written

$$B_{k+1} = B_k + (a_q - a_r)e_p^T, \tag{1}$$

where e_p is the pth column of the identity matrix. The basis is used to solve for the search direction y and the dual variables π in the following linear systems:

$$B_k y = a_q, \tag{2}$$

$$B_k^T \pi = c_k, \tag{3}$$

where c_k contains the objective coefficients corresponding to the columns of B_k.

* Received by the editors January 11, 1991; accepted for publication (in revised form) July 19, 1991. This research was supported in part by National Science Foundation grant ECS-8715153 and Office of Naval Research grant N00014-90-J-1242. The first author's research was also supported by an IBM Graduate Technical Fellowship.

† Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California 94305-4022 (na.eldersveld@na-net.ornl.gov and mike@sol-michael.stanford.edu).

Stable and efficient basis updates are vital to the computational success of the simplex method. The "best" updating method depends on the machine in use and how the update is implemented. For example, the classical *product-form* (PF) update,

$$B_k = B_0 T_1 T_2 \cdots T_k, \tag{4}$$

can take advantage of the vector hardware on current supercomputers such as the Cray X-MP and Y-MP. This helps compensate for its potential instability and for the typically high rate of growth of nonzeros in the "eta" vectors representing the elementary triangular factors T_k.

Conversely, the *Bartels–Golub* (BG) update [Bar71],

$$B_k = L_k U_k, \quad L_k = L_0 T_1 \cdots T_k, \tag{5}$$

performs well on conventional machines [Rei82], [GMSW87] but is difficult to vectorize fully because each T_k may be a product of triangular factors involving short vectors, and U_k is altered in an unpredictable manner. The *Forrest–Tomlin* (FT) update [FT72], also described by (5), makes simpler changes to L_k and U_k and is probably more amenable to vectorization.

With vector machines in mind, we examine two further updates in §§2 and 3. We then discuss implementation details for the second method and present computational results comparing a *block-LU* method to the BG update.

2. The Schur-complement update. As an alternative to (2), Bisschop and Meeraus [BM77], [BM80] drew attention to an augmented system $\tilde{B}_k \tilde{y} = \tilde{a}_q$ of the form

$$\begin{pmatrix} B_0 & V_k \\ U_k & \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} a_q \\ 0 \end{pmatrix}, \tag{6}$$

where

$$V_k = (a_{q_1} \ \cdots \ a_{q_k}), \qquad U_k = (e_{p_1} \ \cdots \ e_{p_k})^T. \tag{7}$$

Initially, B_0 is defined as a basis matrix at the start of the first iteration. After a number of iterations it may be necessary to factorize the current basis B_k and redefine it to be B_0. Each a_{q_j} $(j = 1, \cdots, k)$ corresponds to a basic column from A that has become basic since the last refactorization of B_0.

System (6) is equivalent to (2). To see this, note that the equation $U_k y_1 = 0$ sets k elements of y_1 to zero, so that the remaining elements of y_1 when combined with y_2 give the solution $y \in \Re^m$. Specifically, y may be formed by setting $y \leftarrow y_1$ and overwriting $y(j) \leftarrow y_2(i)$ for $i = 1, \cdots, k$, where j is defined as the unit-vector index of the ith row of U_k.

The solution to (6) can be found by solving in order

$$B_0 w = a_q, \tag{8}$$

$$C_k y_2 = U_k w, \tag{9}$$

$$B_0 y_1 = a_q - V_k y_2, \tag{10}$$

where $C_k = U_k B_0^{-1} V_k$ is the *Schur-complement* matrix. In general, this method requires two solves with B_0 as well as a single solve with the matrix C_k, which will have a maximal dimension of k. If a_q happens to be a column originally from B_0, we

have $a_q = B_0 e_s$ for some s. In this case $w = e_s$ and (9) reduces to $C_k y_2 = e_i$, where the ith row of U_k is e_s^T. In addition, (10) can be written as $B_0(y_1 - e_s) = -V_k y_2$, so that a_q itself need not be known.

Likewise, the solution to (3) can be found by solving the equivalent system

$$\begin{pmatrix} B_0^T & U_k^T \\ V_k^T & \end{pmatrix} \begin{pmatrix} \pi_1 \\ \pi_2 \end{pmatrix} = \begin{pmatrix} c_0 \\ d_k \end{pmatrix} \tag{11}$$

and taking $\pi = \pi_1$. That is, by solving in order

$$B_0^T z = c_0, \tag{12}$$

$$C_k^T \pi_2 = V_k^T z - d_k, \tag{13}$$

$$B_0^T \pi_1 = c_0 - U_k^T \pi_2. \tag{14}$$

During Phase 1 of the simplex method, c_0 in (11) and (12) may change for each k, but in Phase 2, system (12) need be solved only once each time the basis is refactorized. In addition, from (14) we see that $U_k^T \pi_2 = c_0 - B_0^T \pi_1$. This implies that π_2 corresponds to the set of reduced costs for columns of B_0 that are currently nonbasic.

Note that for most updates, a new column is added to V_k to obtain V_{k+1}. However, the updates occasionally involve replacing or deleting columns of V_k. From now on, k refers to the number of columns in V_k, not the simplex iteration number. The matrices involved then have the following dimensions: B_k is $m \times m$, N_k is $m \times (n-m)$, V_k is $m \times k$, U_k is $k \times m$, and C_k is $k \times k$.

2.1. Advantages. The Schur-complement (SC) update for linear programming was first described by Bisschop and Meeraus [BM77], [BM80], one of whose aims was to provide an updating technique with storage requirements that are *independent of the problem size m*. This is a unique feature.

The SC update shares an important advantage with the PF update, in that the factors L_0 and U_0 are used many times without modification. On a vector machine, the triangular solves with these factors can therefore be reorganized to take advantage of the vector hardware, as recently shown in [ER90]. The greater stability of the SC update allows the overhead associated with this reorganization to be spread over 100 iterations (say), whereas the PF update may fail a stability test at any stage (in the worst case after only one or two iterations).

A further advantage of the SC and PF updates is that it is only necessary to solve systems with B_0 and B_0^T; we do not need to access the columns of B_0 for pricing. This may be important for specially structured problems. See [GMSW84, pp. 578–580] for further discussion.

2.2. Stability. The matrix in (6) has the following block-triangular factorization:

$$\tilde{B}_k = \begin{pmatrix} B_0 & V_k \\ U_k & \end{pmatrix} = \begin{pmatrix} I & \\ U_k B_0^{-1} & I \end{pmatrix} \begin{pmatrix} B_0 & V_k \\ & -C_k \end{pmatrix}. \tag{15}$$

Recalling that U_k is composed of unit vectors, we see that if B_0 is "reasonably well conditioned," then the first triangular factor is also reasonably well conditioned. In such cases, the Schur complement C_k tends to reflect the condition of $\tilde{B}_k$, which is essentially the same as the condition of the true basis B_k.

This means that when C_k is updated, ill-conditioning need not persist (because certain rows and columns of C_k are explicitly added or deleted). For example, suppose

bases $B_0, B_1, \cdots, B_k$ are all well conditioned *except* for B_j. Then all of the Schur complements will be well conditioned except C_j, and hence all of the basis factorizations will be well conditioned except for the jth. This property, shared by the BG update, defines our meaning of stability.

In short, the SC update is essentially as stable as the BG update, *provided B_0 is well conditioned.* This cannot be said of the PF or FT updates. (Of course, the BG update remains superior in being stable regardless of the condition of B_0.)

2.3. Comments. A discussion of the Schur complement may be found in [Cot74]. Implementations of a Schur-complement method for general LP problems are described in [Pro85], and for specially structured linear programs in [Eld88].

The original descriptions of Bisschop and Meeraus [BM77], [BM80] involved updating C_k^{-1} explicitly (not a stable process). Proctor [Pro85] presented two implementations, one updating C_k^{-1} as a dense matrix, the other maintaining sparse LU factors of C_k. The latter is to be preferred for stability reasons. Since k can be limited to 100 (say), we believe it is more efficient to maintain dense factors of C_k; see §4.1.

Our original aim was to investigate the performance of the SC update on general LP problems. The method was implemented, but it soon became evident that the additional solves with B_0 and B_0^T were excessively expensive compared to the BG update. The following variation was therefore chosen as a means of trading workspace for time.

3. A block-LU update. Rather than using (15) we may factorize $\tilde{B}_k$ in the following manner:

$$\tilde{B}_k = \begin{pmatrix} B_0 & V_k \\ U_k & \end{pmatrix} = \begin{pmatrix} B_0 & \\ U_k & -C_k \end{pmatrix} \begin{pmatrix} I & Y_k \\ & I \end{pmatrix}, \tag{16}$$

where

$$B_0 Y_k = V_k, \qquad C_k = U_k Y_k. \tag{17}$$

We see that the solution to (6) and hence $B_k y = a_q$ may be obtained from

$$B_0 w = a_q, \tag{18}$$

$$C_k y_2 = U_k w, \tag{19}$$

$$y_1 = w - Y_k y_2. \tag{20}$$

Likewise, the solution to $B_k^T \pi = c_k$ may be obtained from

$$C_k^T \pi_2 = Y_k^T c_0 - d_k, \tag{21}$$

$$B_0^T \pi_1 = c_0 - U_k^T \pi_2. \tag{22}$$

The block-LU update was first discussed in [GMSW84].[1] All updating information is carried along via the Schur-complement matrix C_k and the matrix of transformed columns Y_k. The updates to these matrices will be discussed in the next section. Note that C_k is composed of some of the rows of Y_k. It may be described as "some of the rows and columns of the simplex tableau associated with the starting basis B_0."

[1] It was termed a *stabilized product-form* update because the columns of Y_k are handled similarly to the "eta" vectors in the classical product-form update, and because the factors of B_0 are not altered. Note, however, that (16) is an explicit block-triangular factorization. Nothing is held in product form.

3.1. Advantages. The block-*LU* update has most of the advantages of the SC update, in terms of using B_0 as a "black box." The storage for C_k remains independent of m. By storing Y_k we reduce the work per iteration of the simplex method by a solve with B_0 and (in Phase 1) a solve with B_0^T. For many iterations when a row of Y_k is needed to update C_k, we avoid a further solve with B_0^T.

Comparing the right-hand sides of (10) and (20), we see that the term $V_k y_2$ has become $Y_k y_2$, which is usually somewhat more expensive. The analogous term $Y_k^T c_0$ in (21) costs little because most of it does not require updating.

3.2. Stability. The block-*LU* update possesses the same stability properties as the SC update. The main requirement again is that B_0 be reasonably well conditioned.

In practice we can prevent excessive ill-conditioning in B_0 by replacing certain columns with the unit vectors associated with slack variables, according to the size of the diagonal elements in the initial *LU* factors. A rather lax tolerance is needed to prevent altering the basis after every factorization and thereby impeding convergence of the simplex method. In the computational tests reported here, provision was made to altered B_0 if its condition appeared to be greater than $\epsilon^{-2/3} \approx 10^{10}$ (where the machine precision was $\epsilon \approx 10^{-15}$). However, no such alterations occurred. Thus, after every 100 iterations the current B_k was always accepted as B_0, and no numerical difficulties were encountered.

4. Implementation issues. For the block-*LU* update to be efficient, we must be able to update C_k and Y_k efficiently at each iteration. The updates to these matrices consist of four cases:

1. Add a row and column to C_k, and add a column to Y_k.
2. Replace a column of C_k and Y_k.
3. Replace a row of C_k, leaving Y_k unchanged.
4. Delete a row and column of C_k, and delete a column from Y_k.

Each of these cases depends on the type of column entering or leaving the basis and whether or not the columns were in the initial B_0. A description of each case follows.

Case 1. The entering column is from N_0, and the leaving column is from B_0. A row and column are added to C_k:

$$U_{k+1} = \begin{pmatrix} U_k \\ e_p^T \end{pmatrix} \quad \text{and} \quad Y_{k+1} = \begin{pmatrix} Y_k & w \end{pmatrix}, \tag{23}$$

$$C_{k+1} = U_{k+1}\, B_0^{-1}\, V_{k+1} = \begin{pmatrix} C_k & U_k w \\ e_p^T Y_k & \delta \end{pmatrix}, \tag{24}$$

where $B_0 w = a_q$ and $\delta = e_p^T w$. Note that w is already available from (8) in the simplex algorithm. It becomes a new column of Y_k.

Case 2. The entering column is from N_0 and the leaving column is from V_k (not from B_0). A column of C_k is again replaced by $U_k w$, which is already available from the simplex algorithm. The dimension of C_k stays the same. A column in Y_k is replaced by the new transformed column w.

Case 3. The entering column is from B_0 and the leaving column is from B_0. A row of C_k is replaced with the pth row of Y_k. The dimension of C_k stays the same. Y_k is not altered.

Case 4. The entering column is from B_0 and the leaving column is from V_k (and not from B_0). We delete a row and column from C_k and we delete the corresponding column from Y_k.

4.1. Storage of C_k. The size of C_k will never be larger than the *refactorization frequency.* Since this is relatively small for most large-scale LP problems (we used 100), it is efficient to treat C_k as a dense matrix.

For maximum reliability, we maintain a dense orthogonal factorization $Q_k C_k = R_k$, where Q_k is orthogonal and R_k is upper triangular. The techniques for updating the QR factors of C_k involve sweeps of *plane rotations* as discussed in [GGMS74]. A set of routines called QRMOD were used for this purpose. For slightly greater efficiency, Q_k and R_k may be updated using sweeps of *stabilized elimination matrices;* see [Cli77].

4.2. Storage of Y_k. As Y_k has a row dimension of m, the method of dealing with this matrix is important. Y_k consists of transformed columns that have entered the basis since the last refactorization. We must be able to do matrix-vector multiplies with Y_k (20) and Y_k^T (21) as well as fetch rows of Y_k (24). The sparsity of each column of Y_k depends on the sparsity of the basis itself as well as the sparsity of each of the entering basic columns.

Since the use of indirect addressing reduces performance on most vector computers, indirect adressing should be avoided for all except very sparse vectors. On the other hand, performing computations with vectors containing a very large proportion of zero elements is also inefficient. With this in mind, each column of Y_k is stored in one of two ways depending on its density. We have used the following dynamic storage scheme for Y_k:

1. A column of Y_k that has a density of at least `NTHRSH` is considered to be *dense.* Such columns are stored "as is" and not packed. In the computational tests, a value of `NTHRSH` $= 0.40$ was used.
2. Columns with density less than `NTHRSH` are considered *sparse* and are packed in a conventional column list. For each column, the nonzero elements of these vectors are stored contiguously, along with a parallel array of row indices, the number of nonzeros, and a pointer to the first nonzero.

The average sparsity for Y_k's columns for each of the test problems is given in Table 4. A row of Y_k can be extracted trivially from columns in dense form. Packed columns require a search for the desired row index, which can usually be vectorized.

Dense columns of Y_k are stored separately from the sparse columns in order to make operations with the dense columns vectorizable. Thus, the storage array for Y_k consists of a dense part and a sparse part. The updates to Y_k consist of adding, deleting and replacing columns. Each case is described below.

1. When a new column is added to Y_k, the column is simply appended to the end of the "dense" or "sparse" arrays for Y_k. Dense columns are stored "as is" and sparse columns are packed in a conventional column list.
2. For simplicity, column deletion was implemented by moving all later columns one place to the left (thereby overwriting the deleted column and recovering its storage). The operations are essentially the same whether the deleted column is dense or sparse. Half of Y_k must be moved on average, but the copy operation is vectorizable and cheap. Also, less than half of the updates require deletion.
3. Column replacement occurs in one of two ways. When both new and old columns are dense, the new column simply overwrites the old column. In all other cases, the old column is deleted from Y_k (2 above) and the new column is appended to Y_k (1 above).

TABLE 1
Problem specifications.

No.	Problem	Rows	Cols	Elem	Objective value
1	80bau3b	2263	2266	29063	9.8722822814E+05
2	bp822	822	825	11127	5.5018458595E+03
3	cycle	1904	1907	21322	-5.2263930249E+00
4	czprob	930	933	14173	2.1851966988E+06
5	etamacro	401	404	2489	-7.5571519542E+02
6	fffff800	525	528	6235	5.5567961167E+05
7	ganges	1310	1313	7021	-1.0958627396E+05
8	greenbea	2393	2396	31499	-7.2462397960E+07
9	grow22	441	444	8318	-1.6083433648E+08
10	nesm	663	666	13988	1.4076079892E+07
11	perold	626	629	6026	-9.3807558690E+03
12	pilot.ja	941	944	14706	-6.1131579663E+03
13	pilot.we	723	726	9218	-2.7201045880E+06
14	pilot4	411	414	5145	-2.5811392641E+03
15	pilotnov	976	979	13129	-4.4972761882E+03
16	pilots	1442	3652	43220	-5.5760732709E+02
17	scfxm2	661	664	5229	3.6660261565E+04
18	scfxm3	991	994	7846	5.4901254550E+04
19	scrs8	491	494	4029	9.0429998619E+02
20	scsd6	148	151	5666	5.0500000078E+01
21	scsd8	398	401	11334	9.0499999993E+02
22	sctap3	1491	1494	17554	1.4240000000E+03
23	ship08l	779	782	17085	1.9090552114E+06
24	ship12l	1152	1155	21597	1.4701879193E+06
25	ship12s	1152	1155	10941	1.4892361344E+06
26	stair	357	360	3857	-2.5126695119E+02
27	stocfor2	2158	2161	9492	-3.9024408538E+04
28	tdesg1	3500	4050	18041	4.3560773922E+04
29	tdesg5	4215	22613	105002	4.3407357993E+04
30	woodw	1099	1102	37478	1.3044763331E+00

5. Computational results. In this section we compare numerical results obtained from an implementation of the algorithm described in §3. The standard basis update in MINOS 5.3 [MS87] is the Bartels–Golub update. For a complete discussion of LUSOL, the package of basis routines in MINOS 5.3, see [GMSW87].

The implementation of the block-*LU* update has been included as an option in a specially modified version of MINOS 5.3. The new version, MINOS/SC 5.3, includes other options including a special pricing routine designed especially for vector computers described in [FT88], and a vectorization algorithm for the solution of triangular systems of equations described in [ER90]. These options were disabled for the present computational tests.

The purpose of the tests is to demonstrate the efficiency of the new update and show that for vector machines the method is more efficient than the Bartels–Golub update on a representative set of large, sparse problems. The two algorithms are labeled BG for the Bartels–Golub update and BLU for the block-*LU* update. The tests consist of comparing timings of BG and BLU by solving 30 linear programming test problems. Many of these problems are available from the *netlib* collection [Gay85]. The test problem specifications are given in Table 1. The smallest *netlib* test problems were omitted from the results, as some timing categories for these problems were less than 1/100th of a second on the machine used.

TABLE 2
Update results.

	Method	BG:	$\bar{U}_{BG}$	BLU:	$\bar{U}_{BLU}$	$\bar{C}$	$\bar{U}_{BG}/\bar{U}_{BLU}$
No.	Problem name	Total update time (sec)	Mean update time (μsec)	Total update time (sec)	Mean update time (μsec)	Mean size C_k	Update speed-up
1	80bau3b	35.79	30137.88	3.87	3385.17	33.99	8.90
2	bp822	13.50	20079.65	3.83	5471.75	25.66	3.67
3	cycle	11.60	36633.00	2.91	9745.26	37.74	3.76
4	czprob	2.49	16289.44	0.67	4228.94	37.20	3.85
5	etamacro	0.39	7130.00	0.50	8427.22	34.73	0.85
6	fffff800	0.97	10281.49	0.79	7898.30	36.01	1.30
7	ganges	1.22	17342.51	0.44	6180.37	40.44	2.81
8	greenbea	118.99	46059.83	18.87	7391.08	31.06	6.23
9	grow22	1.28	18483.14	0.81	11508.06	40.50	1.61
10	nesm	3.64	12021.19	1.57	5621.24	30.99	2.14
11	perold	6.64	17083.67	1.82	4787.23	23.68	3.57
12	pilot.ja	14.98	23820.10	3.31	5134.65	24.27	4.64
13	pilot.we	7.55	15853.32	2.18	4737.04	24.65	3.35
14	pilot4	1.76	12291.56	0.66	4567.76	23.23	2.69
15	pilotnov	6.78	24923.93	1.57	5679.31	26.80	4.39
16	pilots	117.55	72976.57	8.89	5388.31	24.12	13.54
17	scfxm2	0.88	11478.61	0.64	8299.71	39.82	1.38
18	scfxm3	1.97	16756.42	0.98	8236.08	40.40	2.03
19	scrs8	0.64	9984.70	0.34	6497.42	29.36	1.54
20	scsd6	0.40	3611.14	0.85	7396.47	33.84	0.49
21	scsd8	2.83	8415.58	3.58	10152.39	40.66	0.83
22	sctap3	2.61	26277.24	0.64	5952.07	41.13	4.41
23	ship08l	0.71	13613.07	0.14	2673.06	42.42	5.09
24	ship12l	2.10	18806.31	0.31	2747.94	40.92	6.84
25	ship12s	0.93	17370.72	0.20	3632.95	42.79	4.78
26	stair	0.67	12352.67	0.29	5270.27	24.60	2.34
27	stocfor2	7.29	36568.73	2.19	9562.93	39.75	3.82
28	tdesg1	24.98	60392.59	2.31	5950.65	39.59	10.15
29	tdesg5	271.51	80266.15	27.61	7774.24	40.12	10.32
30	woodw	7.67	20271.52	2.63	6807.28	37.37	2.98
	MEAN	22.34	23919.09	3.18	6370.17	34.26	4.14

5.1. Test environment. The computational tests were performed on an 8-processor Cray Y-MP supercomputer. Only one processor was used. The operating system was UNICOS version 5.1, and the MINOS code was compiled using the CFT77 compiler with full optimization. Each run was made as a batch job.

For each test the number of iterations and total solution time are recorded in Table 4. The solution time was measured by timing the MINOS subroutine `M5SOLV`. The options used for MINOS were the standard MINOS/SC options, namely `PARTIAL PRICE 10, SCALE OPTION 2, FACTORIZATION FREQUENCY 100`. The set of problems was then run with (BLU) and without (BG) the `SCHUR-COMPLEMENT` option.

For purposes of evaluating the block-LU update, the following items were deemed to be of interest for each method:

1. Total and average time spent updating the basis.
2. Total time spent solving for dual variables π and the search direction y using the basis factors.
3. Average solve times with the basis factors.

TABLE 3
Solve results.

	Method	BG:		BLU:	
No.	Problem name	Mean solve π (μsec)	Mean solve y (μsec)	Mean solve π (μsec)	Mean solve y (μsec)
1	80bau3b	40568.12	32337.06	41853.63	27891.02
2	bp822	28050.47	27790.85	26730.54	22025.21
3	cycle	34750.42	44877.62	37200.98	39155.63
4	czprob	17566.18	12823.63	19565.78	10986.37
5	etamacro	8594.14	8356.12	10980.39	6787.33
6	fffff800	10509.23	12726.54	12594.22	11379.08
7	ganges	14002.71	17897.94	17465.92	16229.53
8	greenbea	67136.65	54699.46	70031.12	47843.61
9	grow22	21071.95	22581.79	19877.60	16715.92
10	nesm	14057.41	14433.41	14532.32	10878.63
11	perold	22935.79	23822.10	20340.22	16273.83
12	pilot.ja	30307.37	32987.48	26348.98	23197.10
13	pilot.we	24727.20	25234.13	22966.24	18282.15
14	pilot4	18750.67	18488.82	13718.79	10770.45
15	pilotnov	29968.91	32639.40	26490.74	23707.26
16	pilots	66454.81	65155.80	48638.38	44065.14
17	scfxm2	13580.72	13348.56	16225.99	11929.60
18	scfxm3	19622.53	19358.95	22415.36	17280.10
19	scrs8	12217.33	11776.33	12735.71	7971.36
20	scsd6	6011.64	4875.61	8348.29	4777.12
21	scsd8	16193.41	12426.66	18668.06	11411.83
22	sctap3	14512.53	20346.93	18591.30	19643.66
23	ship08l	14065.56	9978.80	17703.06	10714.05
24	ship12l	16048.90	12260.41	18865.25	12051.95
25	ship12s	16948.28	12683.65	20275.87	12718.61
26	stair	17300.74	17629.60	12552.57	9661.55
27	stocfor2	30451.35	40716.96	34382.52	36011.83
28	tdesg1	52146.37	50159.20	54627.08	46676.17
29	tdesg5	80847.35	69401.81	85958.77	67876.06
30	woodw	27375.91	24530.58	28535.34	20750.70
	MEAN	26225.82	25544.87	26640.70	21188.76

5.2. Updates. Time spent updating the basis was measured by timing the appropriate portion of the MINOS subroutine `M5SOLV`. The total and average updating times are recorded in Table 2. These results dramatize the efficiency of the block-*LU* update for the Cray Y-MP. In 27 of the 30 test problems the BLU method gave faster mean and total updating times than BG. The average update speedup was 4.14. A point of interest is that while update times grew for the larger problems using method BG, the average update time remained fairly constant for method BLU. The average BG update time ranged from 3611–80266 microseconds, while the range was 2673–11508 microseconds for the BLU update.

5.3. Solves. The average solve times for the two methods are quite similar, as exhibited in Table 3. It is important to note that although it was not performed here, the solves with L_0 *and* U_0 can be vectorized with method BLU. The solves with L_0 may be vectorized for method BG but as U_k is updated explicitly with this method,

TABLE 4
Overall Problem Results.

	Method	BG:		BLU:		
No.	Problem name	Itns	Soln. time (secs)	Itns	Soln. time (secs)	Mean dens. Y_k
1	80bau3b	11963	206.22	11425	166.94	.022
2	bp822	6792	71.70	6999	58.64	.621
3	cycle	3198	51.92	2987	39.63	NA
4	czprob	1544	11.97	1595	10.48	.016
5	etamacro	550	2.00	594	2.25	.159
6	fffff800	953	4.85	996	4.95	.277
7	ganges	708	5.78	718	5.34	.051
8	greenbea	26094	622.14	25527	493.56	.223
9	grow22	704	6.39	703	4.99	.691
10	nesm	3058	21.96	2792	17.66	.121
11	perold	3923	35.74	3801	25.71	NA
12	pilot.ja	6350	80.46	6445	58.94	.640
13	pilot.we	4805	48.17	4611	37.24	.719
14	pilot4	1446	9.98	1446	6.90	.577
15	pilotnov	2747	35.04	2773	26.65	.568
16	pilots	16267	577.24	16494	347.56	.736
17	scfxm2	772	4.33	772	4.24	.094
18	scfxm3	1184	9.61	1184	8.83	.104
19	scrs8	647	3.30	521	2.33	.155
20	scsd6	1127	3.12	1153	3.76	.343
21	scsd8	3400	22.17	3531	24.45	.338
22	sctap3	1003	9.72	1070	8.80	.024
23	ship08l	526	4.02	523	3.66	.011
24	ship12l	1125	12.10	1113	10.58	.007
25	ship12s	538	4.60	544	4.12	.006
26	stair	551	3.62	551	2.45	.655
27	stocfor2	2014	31.50	2292	30.28	.088
28	tdesg1	4177	95.50	3878	69.27	NA
29	tdesg5	34177	1334.49	35518	1144.78	.070
30	woodw	3822	65.17	3860	58.56	NA
	MEAN	4872.16	113.16	4880.53	89.45	.291

U_0 is not constant between refactorizations. This means that it is possible to decrease solution times with the factors of B_0 using method BLU even further.

5.4. Comparison with the product-form update. On average, the density of the columns of Y_k will be similar to that of the eta vectors in the classical product-form update. Note, however, that over a period of 100 iterations the average number of columns in Y_k is only 25 to 40, with a mean of 34. This means that the number of transformed vectors used in solving systems of equations is lower for the block-*LU* method than for the PF update, where the average would be 50 if stability requirements allow 100 updates. Since the size of the additional matrix C_k is small on average (25 to 40), this suggests that the block-*LU* update requires fewer floating-point operations per solve as well as lower storage requirments than the PF update on large problems. The ratio is $34/50 \approx 2/3$.

6. Conclusions. 1. A block-*LU* update technique is a viable alternative to a standard (Bartels–Golub) updating technique when vectorization is available.

2. Numerical experiments running a modified version of MINOS 5.3 on a Cray Y-MP showed the block-LU update to be superior to Bartels–Golub updating on 27 of 30 test problems.

3. Average solve times with basis factors using the block-LU update were comparable to the solve times using the standard method.

4. Use of the block-LU update reduced CPU times by approximately 21 percent on these test problems. Vectorization of all the solves with L_0, U_0, L_0^T, U_0^T as in [ER90] would give a further marked improvement.

Acknowledgments. The authors are indebted to Bill Kamp and John Gregory of Cray Research, Inc. for encouraging the research and providing computer time for the computational tests, and to the referee for some helpful comments.

REFERENCES

[Bar71] R. H. BARTELS, *A stabilization of the simplex method*, Numer. Math., 16 (1971), pp. 414–434.

[BM77] J. BISSCHOP AND A. MEERAUS, *Matrix augmentation and partitioning in the updating of the basis inverse*, Math. Programming, 13 (1977), pp. 241–254.

[BM80] ———, *Matrix augmentation and structure preservation in linearly constrained control problems*, Math. Programming, 18 (1980), pp. 7–15.

[Cli77] A. K. CLINE, *Two subroutine packages for the efficient updating of matrix factorizations*, Report TR-68, Department of Computer Sciences, The University of Texas, Austin, TX, 1977.

[Cot74] R. W. COTTLE, *Manifestations of the Schur-complement*, Linear Algebra Appl., 8 (1974), pp. 189–211.

[Dan63] G. B. DANTZIG, *Linear Programming and Extensions*, Princeton University Press, Princeton, NJ, 1963.

[Eld88] S. K. ELDERSVELD, *Stochastic linear programming with discrete recourse: An algorithm based on decomposition and the Schur-complement update*, Engineer Thesis, Department of Operations Research, Stanford University, Stanford, CA, 1988.

[Eld89] ———, MINOS/SC *User's guide supplement*, Internal Tech. Report, Department of Industry, Science and Technology, Cray Research, Inc., Mendota Heights, MN, 1989.

[ER90] S. K. ELDERSVELD AND M. C. RINARD, *A vectorization algorithm for the solution of large, sparse triangular systems of equations*, Report SOL 90-1, Department of Operations Research, Stanford University, Stanford, CA, 1990.

[FT72] J. J. H. FORREST AND J. A. TOMLIN, *Updating triangular factors of the basis to maintain sparsity in the product-form simplex method*, Math. Programming, 2 (1972), pp. 263–278.

[FT88] ———, *Vector processing in simplex and interior methods for linear programming*, IBM Res. Report No. RJ 6390 (62372), IBM, Yorktown Heights, NY, 1988.

[Gay85] D. M. GAY, *Electronic mail distribution of linear programming test problems*, Mathematical Programming Society COAL Newsletter, 13 (1985), pp. 10–12.

[GGMS74] P. E. GILL, G. H. GOLUB, W. MURRAY, AND M. A. SAUNDERS, *Methods for modifying matrix factorizations*, Math. Comput., 4 (1974), pp. 505–535.

[GMSW84] P. E. GILL, W. MURRAY, M. A. SAUNDERS, AND M. H. WRIGHT, *Sparse matrix methods in optimization*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 562–589.

[GMSW87] ———, *Maintaining LU factors of a general sparse matrix*, Linear Algebra Appl., 88/89 (1987), pp. 239–270.

[MS87] B. A. MURTAGH AND M. A. SAUNDERS, MINOS 5.1 *user's guide*, Report SOL 83-20R, Department of Operations Research, Stanford University, Stanford, CA, 1987.

[Pro85] P. E. PROCTOR, *Implementation of the double-basis simplex method for the general linear programming problem*, SIAM J. Algebraic Discrete Methods, 6 (1985), pp. 567–575.

[Rei82] J. K. REID, *A sparsity exploiting variant of the Bartels–Golub decomposition for linear programming bases*, Math. Programming, 24 (1982), pp. 55–69.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 202–211, January 1992

016

EXPLOITING STRUCTURAL SYMMETRY IN UNSYMMETRIC SPARSE SYMBOLIC FACTORIZATION*

STANLEY C. EISENSTAT† AND JOSEPH W. H. LIU‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. This paper shows how to exploit structural symmetry in determining the nonzero structures of the lower and upper triangular factors L and U of an unsymmetric sparse matrix A. Two symmetric reductions of the graphs of L and U are introduced and used to formulate symbolic factorization algorithms. Experimental results demonstrate the effectiveness of these algorithms versus other schemes in the literature.

Key words. sparse symbolic factorization, structural symmetry

AMS(MOS) subject classifications. 65F05, 65F50

1. Introduction.

Let A be a sparse $n \times n$ matrix that can be decomposed (without pivoting) into LU, where L is lower triangular and U is upper triangular. Symbolic factorization is the process of determining the nonzero structures of the factor matrices L and U from the nonzero structure of A.

When A is symmetric,[1] the *elimination tree* [8], [10] is an important structure in studying symbolic factorization. When A is unsymmetric, the *elimination dag* (*directed acyclic graph*) [6] was introduced as a natural generalization of the elimination tree.

Elimination dags are obtained from the underlying graphs of the factor matrices by a process called *transitive reduction* [1]. Edges are removed if they are redundant in preserving the set of paths. In other words, if there is a path of length greater than one from i to j in the graph, then the edge (i, j) is removed. But while transitive reduction produces the minimum representation of the set of paths required for use in symbolic factorization, computing it can be a significant overhead.

In this paper we consider *partial* transitive reductions of the graphs of L and U that exploit structural symmetry in the *filled matrix* $F = L + U$. We have therefore called them *symmetric reductions*. The goal is a more efficient method that prunes enough of the redundant edges to retain most of the practical advantages of elimination dags in determining the nonzero structures of the factors.

An outline of this paper follows. In §2 we review two characterizations of the structures of L and U, one based on the underlying graphs, the other on elimination dags. In §3 we define the symmetric reductions of the graphs of L and U. The set of edges removed is determined by the first symmetric nonzero in each row and column of the filled matrix F. In §4 we present an unsymmetric sparse symbolic factorization algorithm that is based on these symmetric reductions. In §5 we consider additional ways to exploit structural symmetry using paths in the factor graphs. The resulting

* Received by the editors November 14, 1990; accepted for publication (in revised form) June 26, 1991.

† Department of Computer Science and Research Center for Scientific Computation, Yale University, New Haven, Connecticut 06520. The research of this author was supported in part by Office of Naval Research contract N00014-86-K-0310 and National Science Foundation grant DCR-85-21451.

‡ Department of Computer Science, York University, North York, Ontario, Canada M3J 1P3. The research of this author was supported in part by Natural Sciences and Engineering Research Council of Canada grant A5509.

[1] Throughout the paper symmetry always means structural symmetry and not numeric symmetry.

partial path-symmetric reductions are the basis for a second symbolic factorization algorithm. In §6 we provide experimental results that compare these algorithms with the methods of Rose and Tarjan [9] and Gilbert and Liu [6]. In §7 we conclude with some remarks on the use of symmetric reduction in unsymmetric sparse numerical factorization with pivoting [7].

2. Structural characterizations of L and U. In this section we present characterizations of the row and column structures of L. Dual results hold for the column and row structures of U.

Let $G(L)$ denote the directed graph associated with the lower triangular matrix L. Here (i,j) is an edge of $G(L)$ if and only if ℓ_{ji} is nonzero, so that the orientation of the edge is from column to row in L as in [6]. The directed graph $G(U^t)$ is defined similarly.

Symbolic factorization can be implemented by simulating the numeric factorization using nonzero structures rather than numerical values. Since the structure of the jth column U_{*j} of U gives the set of all *numerical* column updates to A_{*j} from preceding columns of L needed to form L_{*j}, we can use the same set for *structural* column updates to form the structure of L_{*j}. We have therefore the following result.

OBSERVATION 2.1 ([9]). *The structure of L_{*j} is given by the union of the structure of the lower triangular portion of A_{*j} and the structures of those L_{*i} with (i,j) an edge in $G(U^t)$ (that is, $u_{ij} \neq 0$).*

Gilbert [5] observed that L_{j*} is the solution of a linear system with a sparse lower triangular coefficient matrix and a sparse right-hand side, and derived the following result.

OBSERVATION 2.2 ([5]). *The structure of L_{j*} is given by the subset of nodes in $\{1,\cdots,j\}$ reachable in the graph $G(U^t)$ from nodes associated with the structure of A_{j*}.*

The characterizations of the column and row structures of L in Observations 2.1 and 2.2 are based on the graph $G(U^t)$. But often not all of the edges in $G(U^t)$ are needed. *Elimination dags* were introduced in [6] to capture the minimum structural information required for unsymmetric sparse symbolic factorization.

The lower elimination dag $D(L)$ of the matrix A is the transitive reduction [1] of the directed graph $G(L)$. In other words, (i,j) is an edge in $D(L)$ if and only if it is an edge in $G(L)$ and there is no path of length greater than one from i to j in $G(L)$. The upper elimination dag $D(U^t)$ is defined similarly. The following results characterize the structure of L using the elimination dag $D(U^t)$.

OBSERVATION 2.3 ([6]). *The structure of L_{*j} is given by the union of the structure of the lower triangular portion of A_{*j} and the structures of those L_{*i} with (i,j) an edge in $D(U^t)$.*

OBSERVATION 2.4 ([6]). *The structure of L_{j*} is given by the subset of nodes in $\{1,\cdots,j\}$ reachable in the graph $D(U^t)$ from nodes associated with the structure of A_{j*}.*

3. Symmetric reductions. The directed graph $G(U^t)$ and the elimination dag $D(U^t)$ represent the maximum and minimum subgraphs of $G(U^t)$, respectively, for determining the row and column structures of L. Using $G(U^t)$ directly may result in redundant structural column updates (in Observation 2.1) or redundant graph traversals (in Observation 2.2). But while there is no such redundancy when using the elimination dag $D(U^t)$, there is an added cost in transitively reducing $G(U^t)$ to obtain $D(U^t)$. As a practical compromise we would like to determine a subgraph of

$G(U^t)$ that is easier to generate but that offers comparable speed-ups in symbolic factorization. We first establish the validity of this approach.

Let $\tilde{D}(U^t)$ be any subgraph of $G(U^t)$ that is also a supergraph of $D(U^t)$ (that is, it includes all of the edges in $D(U^t)$).

OBSERVATION 3.1. *The structure of L_{*j} is given by the union of the structure of the lower triangular portion of A_{*j} and the structures of those L_{*i} with (i,j) an edge in $\tilde{D}(U^t)$.*

OBSERVATION 3.2. *The structure of L_{j*} is given by the subset of nodes in $\{1,\cdots,j\}$ reachable in the graph $\tilde{D}(U^t)$ from nodes associated with the structure of A_{j*}.*

In order to exploit structural symmetry in $F = L + U$, we need the following result.

THEOREM 3.3. *Assume that $\ell_{sj} * u_{js} \neq 0$ for some $s > j$. Then, for all $k > s$, (j,k) cannot be an edge in the elimination dags $D(U^t)$ and $D(L)$.*

Proof. We first show that (j,k) cannot be an edge in $D(U^t)$. Let (j,k) be an edge in $G(U^t)$; that is, $u_{jk} \neq 0$. In Gaussian elimination, $\ell_{sj} \neq 0$ and $u_{jk} \neq 0$ implies that $u_{sk} \neq 0$. Therefore there is a path

$$j \longrightarrow s \longrightarrow k$$

in $G(U^t)$. Since $D(U^t)$ is the transitive reduction of $G(U^t)$, the edge (j,k) cannot be in $D(U^t)$. By a similar argument, the edge (j,k) cannot be in $D(L)$. □

For each symmetric nonzero $\ell_{sj} * u_{js} \neq 0$ in F, Theorem 3.3 says that all nonzero entries after location s in row j of U and column j of L are pruned in $D(U^t)$ and $D(L)$, respectively. This provides a simple condition to define $S(U^t)$, a subgraph of $G(U^t)$ and supergraph of $D(U^t)$, to which we can apply Observations 3.1 and 3.2.

For each j, define

$$\text{FSNZ}(j) = \min\{k > j : k = n+1 \text{ or } \ell_{kj} * u_{jk} \neq 0\},$$

where FSNZ is an acronym for First Symmetric NonZero. The *symmetric reduction* of the graph $G(U^t)$ is the subgraph $S(U^t)$ on n nodes such that (j,k) is an edge in $S(U^t)$ if and only if (j,k) is an edge in $G(U^t)$ and $k \leq \text{FSNZ}(j)$. The symmetric reduction $S(L)$ of $G(L)$ is defined similarly. Note that even though the two symmetrically reduced structures are generally different, they are determined using the same quantities FSNZ$(*)$. In effect, the pruning mechanisms are symmetric.

The next result follows directly from the definitions of $S(L)$ and $S(U^t)$ and Theorem 3.3. It implies that these symmetric reductions can be used to find the structures of L and U using Observations 3.1 and 3.2.

COROLLARY 3.4. *The transitive reduction $D(U^t)$ (respectively, $D(L)$) is a subgraph of the symmetric reduction $S(U^t)$ (respectively, $S(L)$).*

In Fig. 1 we present three filled matrix structures and their corresponding symmetrically reduced filled matrix structures. For the jth row and column of $F = L+U$, nonzero entries are set to zero except for those up to and including the nonzeros in location FSNZ(j). The lower and upper triangular portions of the resulting matrix give the structures of $S(L)$ and $S(U^t)$, respectively. For comparison we have also displayed the transitively reduced filled matrix structures.

We conclude this section with an interesting interpretation of the quantities FSNZ$(*)$ from the point of view of *elimination trees*. In general, the filled matrix F is unsymmetric. However, the matrix $\bar{F}$ defined by

$$\bar{f}_{ij} = \begin{cases} f_{ij} & \text{if } f_{ij} * f_{ji} \neq 0, \\ 0 & \text{otherwise} \end{cases}$$

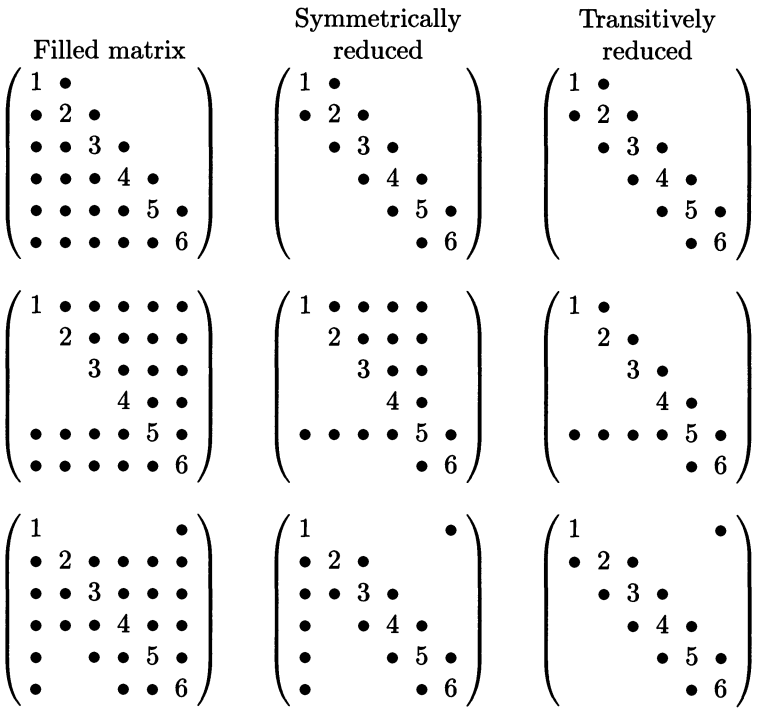


FIG. 1. *Filled matrices and their symmetrically reduced and transitively reduced structures .*

retains all of the structurally symmetric entries, and obviously has symmetric structure.

THEOREM 3.5. *$\bar{F}$ is a perfect elimination matrix.*

Proof. Assume that $\bar{f}_{ji} \neq 0$ and $\bar{f}_{ki} \neq 0$ for some $i < j < k$. We need to show that $\bar{f}_{kj}$ is also nonzero. By assumption, $\ell_{ki} = \bar{f}_{ki} \neq 0$. Since the structure of $\bar{F}$ is symmetric, $\bar{f}_{ji} \neq 0$ implies that $u_{ij} = \bar{f}_{ij} \neq 0$. It follows from Gaussian elimination that $\ell_{kj} \neq 0$. By a dual argument, we have $u_{jk} \neq 0$, so that $\bar{f}_{kj} = f_{kj} = \ell_{kj} \neq 0$. □

Since the nonzero structure of $\bar{F}$ is symmetric, it is meaningful to consider the elimination tree $T(\bar{F})$ associated with $\bar{F}$, and it is easy to verify that FSNZ(j) is the parent of node j in $T(\bar{F})$ if FSNZ(j) $< n$. Therefore, if A is symmetric to start with, then the symmetric reduction $S(U^t)$ will be the same as the elimination tree of A. In this sense we can view the symmetric reduction as another generalization of the elimination tree to the unsymmetric case.

4. Unsymmetric symbolic factorization using symmetric reductions. In this section we describe an unsymmetric symbolic factorization scheme based on symmetric reduction.[2] The algorithm computes the structures of both L and U by rows so as to be compatible with other algorithms described in the literature [9], [6]. However, *any* of the four $\{L/U\}$-{rows/columns} combinations can be formulated using symmetric reduction.

In Algorithm 1 (see Fig. 2), we use a working vector CUR_FSNZ($*$) of size n to record the current "first symmetric nonzero" information. It is easy to see that at the

[2] This algorithm first appeared (without explanation) in the Yale Sparse Matrix Package [4].

ALGORITHM 1. *Symbolic LU using symmetric reduction.*

```
for row j := 1 to n do
    CUR_FSNZ(j) = n + 1 ;
for row j := 1 to n do
    LROWJ := {i < j : a_ji ≠ 0} ;
    UROWJ := {k > j : a_jk ≠ 0} ;
    while there exists an unprocessed i ∈ LROWJ do
        for each h with u_ih ≠ 0 and h ≤ CUR_FSNZ(i) do
            if h < j then
                add h to LROWJ
            else
                add h to UROWJ
                if h = j then CUR_FSNZ(i) := j end if
            end if
        end for
    end while
    Structure of L_j* := LROWJ ;
    Structure of U_j* := UROWJ ;
end for
```

FIG. 2. *Symbolic LU using symmetric reduction .*

beginning of step j,

$$\text{CUR\_FSNZ}(i) = \begin{cases} \text{FSNZ}(i) & \text{if FSNZ}(i) < j \\ n+1 & \text{otherwise} \end{cases}$$

so that CUR_FSNZ(i) is never smaller than FSNZ(i). The correctness of the algorithm depends on this observation.

By adapting Observation 3.2 to the symmetric reduction $S(U^t)$, we note that the row structure of L_{j*} is given by the node subset of $\{1, \cdots, j\}$ reachable in the graph $S(U^t)$ from nodes associated with the structure of the lower triangular portion of A_{j*}. Since the symmetric reduction includes only those edges (i, h) with $h \leq \text{FSNZ}(i)$, by the property of CUR_FSNZ($*$) noted above, the **then**-clause in the inner **for**-loop computes the structure of L_{j*}.

On the other hand, to determine the row structure of U_{j*} we use the dual of Observation 3.1. That is, the row structure of U_{j*} is given by the union of the structure of the upper triangular portion of A_{j*} and the structure of those U_{i*} with $\ell_{ji} \neq 0$ and $j \leq \text{FSNZ}(i)$. Since CUR_FSNZ(i) $= n + 1$ for such i, the **else**-clause in the inner **for**-loop computes the structure of U_{j*}. It follows that the **for**-loop in Algorithm 1 is indeed computing the structures of L_{j*} and U_{j*} simultaneously.

We now address an important implementational detail. In the inner **for**-loop we need to access those entries in the structure of U_{i*} that are less than or equal to CUR_FSNZ(i). In order to avoid processing the remaining entries, we rearrange the entries of U_{i*} into two partitions: one with those subscripts less than or equal to CUR_FSNZ(i), the other with the remaining subscripts. In this way the innermost **for**-loop only needs to examine the subscripts in the first partition. The rearrangement takes place whenever the value of CUR_FSNZ(i) changes from $n + 1$ to j in Algorithm 1. This partitioning step is similar to that used by the *quicksort* algorithm [2, p. 264] with j the *pivot* value defining the partitions.

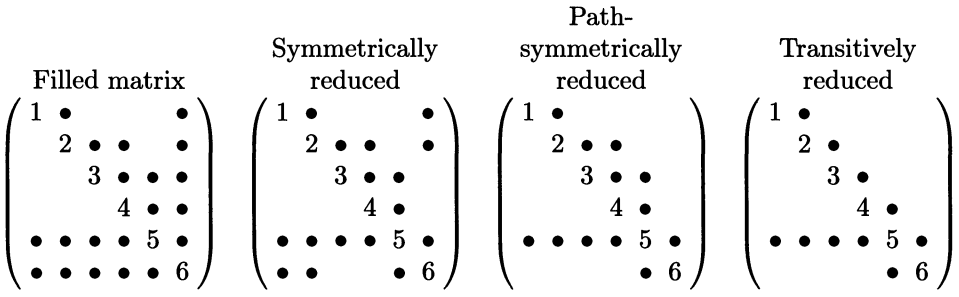


FIG. 3. *A filled matrix and its reduced structures .*

5. Path-symmetric reductions. The elimination dags $D(L)$ and $D(U^t)$ are minimum subgraphs of $G(L)$ and $G(U^t)$, respectively, that preserve the set of paths. They are minimum structures for the purpose of unsymmetric sparse symbolic factorization. The symmetric reductions $S(L)$ and $S(U^t)$ are one of many structures in the spectrum:

$$G(L)/G(U^t) \longrightarrow S(L)/S(U^t) \longrightarrow D(L)/D(U^t).$$

In this section we explore additional ways to exploit symmetry to obtain structures that lie between $S(L)/S(U^t)$ and $D(L)/D(U^t)$.

We first consider a generalization of Theorem 3.3.

THEOREM 5.1. *Assume that $j < s$. If there is a path from j to s in the graph $G(U^t)$ and a path from j to s in the graph $G(L)$, then for all $k > s$, (j,k) cannot be an edge in the elimination dags $D(U^t)$ and $D(L)$.*

Proof. We first show that (j,k) cannot be an edge in $D(U^t)$. Let (j,k) be an edge in $G(U^t)$. Since there is a path from j to s in $G(L)$, (s,k) must also be an edge in $G(U^t)$. Therefore there is a path

$$j \longrightarrow \cdots \longrightarrow s \longrightarrow k$$

from j to k in $G(U^t)$. By the definition of transitive reduction, the edge (j,k) cannot be in $D(U^t)$. Similarly, the edge (j,k) cannot be in $D(L)$. □

Theorem 5.1 implies an additional form of symmetric pruning: even though ℓ_{sj} and u_{js} are zero, the path condition allows us to prune nonzero entries symmetrically after location s in the jth row of U and the jth column of L. Thus, letting

$$\text{FPNZ}(j) = \min\{k > j : k = n+1 \text{ or } \exists \text{ paths in } G(U^t) \text{ and } G(L) \text{ from } j \text{ to } k\},$$

we define the *path-symmetric reduction* of $G(U^t)$ to be the subgraph $P(U^t)$ on n nodes such that (i,j) is an edge in $P(U^t)$ if and only if (i,j) is an edge in $G(U^t)$ and $i \leq \text{FPNZ}(j)$. The path-symmetric reduction $P(L)$ of $G(L)$ is defined similarly.

To illustrate these notions, we display the filled matrix and the symmetrically reduced, path-symmetrically reduced, and transitively reduced structures for a 6×6 matrix in Fig. 3. Note that there is a path $1 \to 2 \to 3 \to 4 \to 5$ in the graph $G(U^t)$ and paths $1 \to 5$ and $2 \to 5$ in the graph $G(L)$. Therefore we have

$$\text{FPNZ}(1) = \text{FPNZ}(2) = 5 < 6 = \text{FSNZ}(1) = \text{FSNZ}(2),$$

so that there is more pruning in path-symmetric reduction than in symmetric reduction.

ALGORITHM 2. *Symbolic LU using partial path-symmetric reduction.*

```
for row j := 1 to n do
    CUR_FPNZ(j) = n + 1 ;
    ON_PATH(j) = j ;
end for
for row j := 1 to n do
    AROWJ := {i < j : a_ji ≠ 0} ;
    LROWJ := φ ;
    UROWJ := {k > j : a_jk ≠ 0} ;
    for each i ∈ AROWJ do
        if i is not in LROWJ then
            perform a depth-first traversal from i of P̄(U^t) and
                update LROWJ, UROWJ, CUR_FPNZ(*), ON_PATH(*)
        end if
    end for
    Structure of L_j* := LROWJ ;
    Structure of U_j* := UROWJ
end for
```

FIG. 4. *Symbolic LU using partial path-symmetric reduction .*

Although the result of Theorem 5.1 implies that $P(U^t)$ and $P(L)$ could be used for symbolic factorization as in Observations 3.1 and 3.2, there does not appear to be any efficient way to compute them. But, by restricting one of the paths to be an edge, we obtain the following results, which do lead to an effective algorithm.

COROLLARY 5.2. *Assume that $\ell_{sj} \neq 0$ for some $j \leq s$. If there is a path from j to s in the graph $G(U^t)$, then for all $k > s$, (j,k) cannot be an edge in the elimination dags $D(U^t)$ and $D(L)$.*

COROLLARY 5.3. *Assume that $u_{js} \neq 0$ for some $j \leq s$. If there is a path from j to s in the graph $G(L)$, then for all $k > s$, (j,k) cannot be an edge in the elimination dags $D(U^t)$ and $D(L)$.*

For each j, define

$$\overline{\text{FPNZ}}(j) = \min\{k > j : k = n+1 \text{ or } \ell_{kj} \neq 0 \text{ and } \exists \text{ a path in } G(U^t) \text{ from } j \text{ to } k\}.$$

The *partial path-symmetric reduction* of $G(U^t)$ is the subgraph $\overline{P}(U^t)$ on n nodes such that (i,j) is an edge in $\overline{P}(U^t)$ if and only if (i,j) is an edge in $G(U^t)$ and $i \leq \overline{\text{FPNZ}}(j)$. The partial path-symmetric reduction $\overline{P}(L)$ of $G(L)$ is defined similarly. It is clear from the definition that

$$\text{FPNZ}(j) \leq \overline{\text{FPNZ}}(j) \leq \text{FSNZ}(j)$$

so that

$$D(L) \subseteq P(L) \subseteq \overline{P}(L) \subseteq S(L) \subseteq G(L),$$
$$D(U^t) \subseteq P(U^t) \subseteq \overline{P}(U^t) \subseteq S(U^t) \subseteq G(U^t).$$

In Algorithm 2 (see Fig. 4), we perform unsymmetric sparse symbolic factorization by rows using the quantities $\overline{\text{FPNZ}}(*)$. We use a working vector CUR_$\overline{\text{FPNZ}}(*)$ of size n to keep track of the current "first partial path-symmetric nonzero," and another vector ON_PATH$(*)$ of size n to implement the path condition in Corollary 5.2 (if $i < j$, then ON_PATH$(i) = j$ only if we have found a path in $G(U^t)$ from node i to node j).

```
for each h with u_ih ≠ 0 and h ≤ CUR_FPNZ(i) do
    if h has not been included in LROWJ or UROWJ then
        if h ≥ j then
            add h to UROWJ
        else
            add h to LROWJ ;
            perform recursively depth-first traversal from vertex h in P(U^t)
        end if
    end if
    if ON_PATH(h) = j then
        ON_PATH(i) := j ;
        if j < CUR_FPNZ(i) then CUR_FPNZ(i) = j end if
    end if
end for
```

FIG. 5. *Depth-first search from vertex i .*

Within the loop for each row j, the depth-first traversals determine the row structures of L_{j*} and U_{j*} and update the affected values of CUR_FPNZ(s) and ON_PATH(s) (for $s < j$). We can formulate the depth-first search starting at the node i recursively as in Fig. 5. As in Algorithm 1, if the value of CUR_FPNZ(i) is changed to j, then the entries of U_{i*} are rearranged into two partitions: one with those subscripts less than or equal to j, the other with the remaining subscripts. In this way subsequent depth-first searches will only process subscripts in the first partition.

6. Experimental results. In this section we provide experimental results comparing the performance of Algorithm 1 (using symmetric reductions) and Algorithm 2 (using partial path-symmetric reductions) with that of the FILL1 and FILL2 algorithms of Rose and Tarjan [9] and the EDAGS algorithm of Gilbert and Liu [6].

The FILL1 algorithm determines the row structures of L_{j*} and U_{j*} by simulating numerical row elimination using the row structure of A_{j*} and the computed row structures of U_{i*}, for $1 \leq i \leq j-1$. The FILL2 algorithm determines the row structures of L_{j*} and U_{j*} using path information from the row structures of the first j rows of A. The EDAGS algorithm computes the row structures using the elimination dags and the characterizations in Observations 2.3 and 2.4. See references [9] and [6] for details.

The programs were written in Fortran, and the experiments were performed on a Sun 3/80. The test problems are from the Harwell-Boeing Sparse Matrix Collection [3] and are described in Table 1.[3] The time in seconds for each of the five algorithms is presented in Table 2. Note that the factor structures are determined using the *original* ordering; no row/column preordering is used.

In Table 3 we give the sizes of the factor matrices and their corresponding reduced structures. The difference between $|G(L)|$ and $|D(L)|$ (or between $|G(U^t)|$ and $|D(U^t)|$) can be quite substantial; e.g., in problem "FS 541 1" there is a hundredfold reduction. By comparison, the difference between $|S(L)|$ and $|\overline{P}(L)|$ (or between $|S(U^t)|$ and $|\overline{P}(U^t)|$) is relatively insignificant.

[3] For each of these methods there exists a worst-case example on which it is not the fastest. We have chosen to present results only for "real" problems because we believe that they are more representative.

TABLE 1
Description of the test problems.

Problem	n	Description
ARC 130	130	Laser problem from Curtis
FS 541 1	541	Facsimile convergence matrix
BP 1600	822	Unsymmetric basis from LP problem BP
SHL 400	663	Unsymmetric basis from LP problem Shell
WEST 989	989	7 stage column section

TABLE 2
Time for symbolic factorization.

Problem	FILL1	FILL2	EDAGS	SYMMETRIC REDUCTION	PARTIAL PATH-SYMM REDUCTION
ARC 130	1.90	0.44	0.36	0.16	0.18
FS 541 1	27.28	4.04	2.72	1.12	1.42
BP 1600	27.96	5.42	3.72	1.70	2.12
SHL 400	1.28	1.18	0.94	0.20	0.28
WEST 989	11.14	6.42	3.60	1.46	1.86

TABLE 3
Structural statistics.

Problem	$\|G(L)\|$	$\|S(L)\|$	$\|\overline{P}(L)\|$	$\|P(L)\|$	$\|D(L)\|$	$\|G(U^t)\|$	$\|S(U^t)\|$	$\|\overline{P}(U^t)\|$	$\|P(U^t)\|$	$\|D(U^t)\|$
ARC 130	7525	232	232	232	124	7501	229	229	229	129
FS 541 1	59102	539	539	539	539	56672	540	540	540	540
BP 1600	66688	9522	7817	7747	1459	68078	14038	11785	11034	1431
SHL 400	9191	3272	2349	2160	892	8181	1938	1856	1800	1355
WEST 989	47567	12670	9324	9231	2814	55145	23430	22378	14696	2616

These differences partially explain the times in Table 2. FILL1 uses the graph structures $G(L)/G(U^t)$ and is by far the slowest. The scheme based on the symmetric reductions is the fastest, which can be attributed to the very modest sizes of $|S(L)|$ and $|S(U^t)|$ and to the efficient scheme for identifying structural symmetry. Although the sizes of the elimination dags and the partial path-symmetric reductions are often smaller, they are more expensive to generate and the savings in symbolic factorization cannot compensate for this extra overhead.

7. Concluding remarks. We have shown how to exploit structural symmetry in devising effective unsymmetric sparse symbolic factorization algorithms. The symmetric and partial path-symmetric reductions can be viewed as a practical compromise between the graphs of the factor matrices and their corresponding elimination dags. An important application will be in improving the symbolic phase of the sparse partial pivoting scheme of Gilbert and Peierls [7].

The experimental results in §6 show that the partial path-symmetric reductions can lead to smaller structures than the corresponding symmetric reductions, but that the overhead involved in performing depth-first search more than offsets the gain. Thus, in the context of symbolic factorization, the use of the simpler symmetric reductions is preferred. However, in the symbolic phase of the Gilbert and Peierls sparse partial pivoting scheme, depth-first traversals are used to topologically order the factor structures. In such a situation partial path-symmetric reductions may be of practical importance since no additional depth-first searches are required, yet potentially more

structural pruning can be achieved.

REFERENCES

[1] A. V. AHO, M. R. GAREY, AND J. D. ULLMAN, *The transitive reduction of a directed graph*, SIAM J. Comput., 1 (1972), pp. 131–137.

[2] A. V. AHO, J. E. HOPCROFT, AND J. D. ULLMAN, *Data Structures and Algorithms*, Addison-Wesley, Reading, MA, 1983.

[3] I. S. DUFF, R. G. GRIMES, AND J. G. LEWIS, *Sparse matrix test problems*, ACM Trans. Math. Software., 15 (1989), pp. 1–14.

[4] S. C. EISENSTAT, M. C. GURSKY, M. H. SCHULTZ, AND A. H. SHERMAN, *Yale sparse matrix package II: The nonsymmetric codes*, Research Report #114, Department of Computer Science, Yale University, New Haven, CT, July 1977.

[5] J. R. GILBERT, *Predicting structure in sparse matrix computations*, Tech. Report TR 86-750, Department of Computer Science, Cornell University, Ithaca, New York, 1986.

[6] J. R. GILBERT AND J. W. H. LIU, *Elimination structures for unsymmetric sparse LU factors*, Tech. Report CS-90-11, Department of Computer Science, York University, North York, Ontario, Canada, 1990.

[7] J. R. GILBERT AND T. PEIERLS, *Sparse partial pivoting in time proportional to arithmetic operations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 862–874.

[8] J. W. H. LIU, *The role of elimination trees in sparse factorization*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 134–172.

[9] D. J. ROSE AND R. E. TARJAN, *Algorithmic aspects of vertex elimination of directed graphs*, SIAM J. Appl. Math., 34 (1978), pp. 176–197.

[10] R. SCHREIBER, *A new implementation of sparse Gaussian elimination*, ACM Trans. Math. Software., 8 (1982), pp. 256–276.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 212–238, January 1992

017

THE INTERFACE PROBING TECHNIQUE IN DOMAIN DECOMPOSITION*

TONY F. C. CHAN† AND TAREK P. MATHEW†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. The interface probing technique, which was developed and used by Chan and Resasco and Keyes and Gropp, is an algebraic technique for constructing interface preconditioners in domain decomposition algorithms. The basic technique is to approximate interface matrices by matrices having a specified sparsity pattern. The construction involves only matrix-vector products, and thus the interface matrix need not be known explicitly. A special feature is that the approximations adapt to the variations in the coefficients of the equations and the aspect ratios of the subdomains. This preconditioner can then be used in conjunction with many standard iterative methods, such as conjugate gradient methods.

In this paper, some old results are summarized and new ones are presented, both algebraic and analytic, about the interface probing technique and its applications to interface operators. Comparisons are made with some optimal preconditioners.

Key words. interface probe, domain decomposition, elliptic equations, preconditioners

AMS(MOS) subject classifications. 65N20, 65F10

1. Introduction.

Many domain decomposition methods can be viewed as techniques for constructing preconditioners for solving linear systems arising from the discretization of elliptic partial differential equations. These preconditioners are based on the solution of smaller problems on subregions of the domain, together with a preconditioner for the reduced problem on the interface separating the subregions. Many algorithms have been developed having optimal or almost optimal rates of convergence with respect to mesh parameters, such as those of Bramble, Pasciak, and Schatz [4], [6], [7], [8]; Dryja and Widlund [20], [21]; and Smith [32].

An important component in these methods is the preconditioner for the reduced problem on the interface of a two-subdomain problem. For such problems, various interface preconditioners have been developed having rates of convergence independent of the mesh size h; see Bjørstad and Widlund [3]; Bramble, Pasciak, and Schatz [5]; Dryja [19]; and Golub and Mayers [24]. However, most of these interface preconditioners do not account for the coefficients of the problem or the geometry of the subdomains and hence, their performance can be sensitive to variations in these other parameters.

Versions of the interface probe were proposed by Chan and Resasco [15] and Eisenstat [22], and further extended and used by Keyes and Gropp [28], [29], as a technique for constructing preconditioners for interface problems. It differs from the other preconditioners mentioned, however, in that it is an algebraic technique, and as it turns out, one of its advantages is that the technique results in preconditioners which adapt to large variations in the coefficients and to most variations in the aspect ratios of the subdomains, though it does not adapt optimally to changes in mesh size. The basic idea behind this technique is to approximate the interface matrix by

* Received by the editors January 2, 1991; accepted for publication (in revised form) April 30, 1991. This work was supported in part by Department of Energy contract DE-FG03-87ER25037, by Army Research Office contract DAAL03-88-K-0085, National Science Foundation grant FDP NSF ASC 9003002, and Air Force Office for Scientific Research grant AFOSR-90-0271 (subcontract to UCLA UKRF-4-24384-90-87).

† Department of Mathematics, University of California, Los Angeles, California 90024 (mathew@math.ucla.edu).

a matrix having a specified sparsity pattern chosen to capture the strongest coupling of the interface operator, using a few matrix-vector products of the interface matrix (which is usually not known explicitly) with carefully chosen probe vectors.

Since the probing method is an algebraic method, it can easily be applied to any operator having decay properties, and is not necessarily restricted to second-order problems. It has been used successfully to construct preconditioners to fourth-order problems, to the Navier–Stokes equations (see Chan [10], Tsui [33]), to convection-diffusion problems (see Chan and Keyes [13]), and to problems where inexact solvers are used in the subdomain solves (see Chan and Goovaerts [11]). Preconditioners having other sparsity patterns (nonbanded) have been constructed in applications to domain decomposition algorithms involving many subdomains; see Chan and Mathew [14].

Our purpose in this paper is to survey various algebraic and analytic properties of the interface probing technique (including many new results). Most of our results deal with the specific case of tridiagonal approximations. In §2, we describe properties of the reduced interface matrix of an elliptic problem for a two-subdomain decomposition. A brief survey of standard interface preconditioners is presented in §3, and a version of the interface probing technique is described in §4. In §5, we discuss various purely algebraic properties of the probe approximation together with conditions under which the approximations preserve symmetry and nonsingularity. In addition, we discuss other versions of the probing technique. Section 6 concerns applications of the probing technique to the interface operator in domain decomposition. There we show for a model elliptic problem that the rate of convergence of a tridiagonal probe preconditioner is $O(h^{-1/2})$. The results thus indicate that the tridiagonal probe preconditioner performs as well asymptotically as the optimal *tridiagonal* preconditioner for the interface matrix, which has been conjectured by Greenbaum and Rodrigue [25] to have a rate of convergence bounded below by $O(h^{-1/2})$. We also present results concerning dependence of the probe preconditioned system on the aspect ratios of the subdomains, and on the scaling of the coefficients. Some numerical and theoretical comparisons are made with the Golub–Mayers preconditioner. Finally, in §7 we summarize the main properties of the probing technique.

2. A model elliptic problem and properties of the interface system. We consider the following second-order self-adjoint elliptic operator L on a polygonal domain Ω in R^2, with Dirichlet boundary conditions:

$$Lu = -\frac{\partial}{\partial x}\left(a(x,y)\frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(b(x,y)\frac{\partial u}{\partial y}\right) = f(x,y) \quad \text{in } \Omega, \tag{1}$$

with $u = 0$ on $\partial\Omega$, where $a(x,y)$ and $b(x,y)$ are assumed to be uniformly positive functions on the domain Ω.

We discretize (1) using the standard five-point difference approximation on a uniform mesh of width h, with nodes (x_i, y_j) lying in Ω, where $x_{i+1} = x_i + h$ and $y_{j+1} = y_j + h$; see Varga [34]:

$$\begin{aligned}(L_h u^h)_{ij} = {} & (a_{i+1/2,j} + a_{i-1/2,j} + b_{i,j+1/2} + b_{i,j-1/2})u^h_{ij} - a_{i+1/2,j}u^h_{i+1,j} \\ & -a_{i-1/2,j}u^h_{i-1,j} - b_{i,j+1/2}u^h_{i,j+1} - b_{i,j-1/2}u^h_{i,j-1} = h^2 f_{ij},\end{aligned} \tag{2}$$

where $u^h_{ij} \equiv u^h(x_i, y_j) \approx u(x_i, y_j)$ is the discrete solution and $a_{i\pm1/2,j} \equiv a(x_i \pm \frac{h}{2}, y_j)$, etc. This results in a symmetric positive definite linear system $L_h u^h = f$.

Let the domain Ω be partitioned into two nonoverlapping subregions Ω_1 and Ω_2 with interface Γ denoting the intersection of their boundaries:

$$\Gamma \equiv \partial\Omega_1 \cap \partial\Omega_2.$$

If we group the unknowns in the interior of Ω_1 in u_1, and those in the interior of Ω_2 in u_2, and finally those on the interface Γ in u_3, then in this new ordering of unknowns, L_h has the following block structure:

$$\begin{pmatrix} L_{11} & 0 & L_{13} \\ 0 & L_{22} & L_{23} \\ L_{13}^T & L_{23}^T & L_{33} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_3 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ f_3 \end{pmatrix}, \tag{3}$$

where L_{11} and L_{22} denote discretizations corresponding to local problems on Ω_1 and Ω_2, respectively, etc. If we eliminate the interior unknowns, we obtain the following reduced system for u_3:

$$\begin{aligned} Su_3 &= f_3 - L_{13}^T L_{11}^{-1} f_1 - L_{23}^T L_{22}^{-1} f_2, \quad \text{where} \\ S &\equiv L_{33} - L_{13}^T L_{11}^{-1} L_{13} - L_{23}^T L_{22}^{-1} L_{23}. \end{aligned} \tag{4}$$

The matrix S is the reduced interface operator, and is also referred to as a Schur complement or a capacitance matrix. It is expensive to compute, requiring n solves on each subdomain, where n is the number of unknowns on the interface Γ. Therefore most domain decomposition methods are based on the solution of the reduced interface system (4) by means of a preconditioned conjugate gradient method. This requires only matrix-vector products with S, which can be computed at a cost of one solve on each subdomain without computing S. Then, it is possible to solve problem (4) in less than n iterations, if the new system is well conditioned, and much research has gone into the construction of efficient preconditioners for S. Once u_3 is determined by solving system (4), the solution in the interior of the subdomains, u_1 and u_2 can be obtained at the cost of one solve in each subdomain, using the first two block rows of equation (3). Alternatively, once a preconditioner for S is available, it is easy to construct a preconditioner for the global matrix L_h, involving approximate solves on the subdomains; rather than exact solves on the subdomains; see, for instance, [5].

Much is known about the properties of the interface matrix S, which is easily seen to be symmetric and positive definite. S is a discrete approximation to a Steklov–Poincaré operator coupling the subdomain problems through a *transmission* boundary condition; see [1], [31]. This Steklov–Poincaré operator can be shown to be spectrally equivalent to the pseudodifferential operator given by the square root of the Laplacian on the interface. Hence, it can be shown that S is spectrally equivalent to the square root of the discrete Laplacian on the interface Γ, and its condition number can be shown to grow at a rate $O(h^{-1})$, as $h \to 0$; see [3]. Some details of this will be given in §3 on preconditioners. S is a dense matrix, however, its entries can be shown to decay away from the diagonal at a rate $|S_{ij}| = O(|i-j|^{-2})$; see [24]. Other properties of S will be described following the definition of *discrete harmonic functions* below, and in Theorem 2.2.

DEFINITION. A grid function w^h satisfying $(L^h w^h)_{ij} = 0$, for nodes ij lying in the interior of Ω_1 and Ω_2, will be referred to as a *piecewise discrete harmonic function.*

Piecewise discrete harmonic functions w^h for discretization (2) can easily be shown to satisfy the following discrete strong maximum principle.

LEMMA 2.1. *If w^h is a nonconstant, piecewise discrete harmonic function, then its nodal values on the interior of the subdomains is strictly bounded above and below*

by the maximum and minimum, respectively, of the nodal values of w^h *on the boundary of the subdomains, i.e., if* $W_{\min} \le w^h_{ij} \le W_{\max}$, *for all* $ij \in \partial\Omega_1 \cup \partial\Omega_2$, *and if* w^h *is piecewise discrete harmonic, then, either* $w^h_{ij} \equiv W_{\max} = W_{\min}$, *or* $W_{\min} < w^h_{ij} < W_{\max}$, *for all* $ij \in$ *interior of* Ω.

Proof. See [26]. □

Remark. The maximum principle is valid only on the *computational* domain, i.e., on the nodes that are connected to other nodes by the stencil. For instance, if the five-point discretization is used on a rectangular domain, the four corner nodes are not coupled to the other nodes, and they are not considered to be part of the computational domain.

The Schur complement is related to *piecewise discrete harmonic functions* as follows. Given a grid function w defined on the interface Γ, if we let Ew denote the *piecewise discrete harmonic extension* of w into the two subdomains, i.e.,

$$Ew \equiv (-L_{11}^{-1}L_{13}w, -L_{22}^{-1}L_{23}w, w)^T,$$

then Sw is obtained from (3) by applying the stencil L_h to Ew on the nodes lying on Γ:

$$L_hEw = (0, 0, (L_{33} - L_{13}^T L_{11}^{-1} L_{13} - L_{23}^T L_{22}^{-1} L_{23})w)^T = (0, 0, Sw). \tag{5}$$

The above properties can be used to prove strict diagonal dominance and other properties of S, stated in Theorem 2.2. We recall that a matrix C is said to be diagonally dominant if

$$|C_{ii}| \ge \sum_{j \ne i} |C_{ij}| \quad \text{for } i = 1, \cdots, n. \tag{6}$$

C is said to be strictly diagonally dominant, if a strict inequality holds in (6) for all rows i. We also recall that a matrix C is said to be an M-matrix, if $C_{ij} \le 0$, for $i \ne j$, and if $C_{ij}^{-1} \ge 0$, for all i, j.

THEOREM 2.2. *The Schur complement* S, *defined in equation* (4) *for discretization* (2), *is an M-matrix and satisfies:*

1. $S_{km} < 0$ *for* $k \ne m$,
2. $S_{kk} > 0$ *for all* k,
3. $S_{kk} - \sum_{m \ne k} |S_{km}| > 0$ *for* $k = 1, \cdots, n$ *(i.e.,* S *is strictly diagonally dominant).*

Proof. All of the above statements are easily proved using the strong maximum principle and the relation between the Schur complement and discrete harmonic functions. Let (i_k, j_k) be the index of the kth node on Γ and let δ_k denote the Kronecker delta function on Γ, which is 1 on the kth node on Γ, and zero on all other nodes of Γ. Then, applying the stencil of L_h to $E\delta_k$ at the mth node on Γ, we obtain:

$$\begin{aligned} S_{mk} = (L_hE\delta_k)_{i_m,j_m} = {} & (a_{i_k+1/2,j_k} + a_{i_k-1/2,j_k} + b_{i_k,j_k+1/2} + b_{i_k,j_k-1/2})(E\delta_k)_{i_m,j_m} \\ & - a_{i_k+1/2,j_k}(E\delta_k)_{i_m+1,j_m} - a_{i-1/2,j}(E\delta_k)_{i_m-1,j_m} \\ & - b_{i,j+1/2}(E\delta_k)_{i_m,j_m+1} - b_{i,j-1/2}(E\delta_k)_{i_m,j_m-1}. \end{aligned} \tag{7}$$

Since $E\delta_k = \delta_k$ on Γ we have that $(E\delta_k)_{i_m,j_m} = 0$, for nodes $m \ne k$ on Γ. And so (7) becomes: for $m \ne k$,

$$S_{mk} = (a_{i_k+1/2,j_k} + a_{i_k-1/2,j_k} + b_{i_k,j_k+1/2} + b_{i_k,j_k-1/2})0$$

$$-a_{i_k+1/2,j_k}(E\delta_k)_{i_m+1,j_m} - a_{i-1/2,j}(E\delta_k)_{i_m-1,j_m}$$
$$-b_{i,j+1/2}(E\delta_k)_{i_m,j_m+1} - b_{i,j-1/2}(E\delta_k)_{i_m,j_m-1},$$

which is negative since the coefficients $a_{i_k+1/2,j_k}, a_{i_k-1/2,j_k}, b_{i_k,j_k+1/2}, b_{i_k,j_k-1/2}$ are positive, and $0 \leq (E\delta_k)_{ij} \leq 1$ by the maximum principle, with strict inequality holding for the nodes (i,j) lying in the interior of the subdomains (because $E\delta_k$ is nonconstant). Thus, we have proved (1).

To prove (2), we apply the stencil to $E\delta_k$ at the kth node on Γ:

$$\begin{aligned} S_{kk} = &(a_{i_k+1/2,j_k} + a_{i_k-1/2,j_k} + b_{i_k,j_k+1/2} + b_{i_k,j_k-1/2})1 \\ &-a_{i_k+1/2,j_k}(E\delta_k)_{i_k+1,j_k} - a_{i-1/2,j}(E\delta_k)_{i_k-1,j_k} \\ &-b_{i,j+1/2}(E\delta_k)_{i_k,j_k+1} - b_{i,j-1/2}(E\delta_k)_{i_k,j_k-1} \\ &> 0, \end{aligned}$$

since by the strong version of the maximum principle $0 \leq (E\delta_k)_{ij} \leq 1$ at all nodes, with strict inequality at the nodes (i,j) lying in the interior of the subdomains.

We now show that S is strictly diagonally dominant. Let $\mathbf{1} \equiv (1,1,\cdots,1)^T$ on Γ. Then, since $S_{km} \leq 0$ for $k \neq m$, by (5):

$$S_{kk} - \sum_{m \neq k} |S_{km}| = S_{kk} + \sum_{m \neq k} S_{km} = (S\mathbf{1})_k = (L_h E\mathbf{1})_{i_k j_k}.$$

By the discrete maximum principle, $0 < (E\mathbf{1})_{ij} < 1$, for all nodes ij lying in the interior of the subdomains, so we obtain:

$$\begin{aligned} (L_h E\mathbf{1})_{i_k,j_k} = &(a_{i_k+1/2,j_k} + a_{i_k-1/2,j_k} + b_{i_k,j_k+1/2} + b_{i_k,j_k-1/2})1 \\ &-a_{i_k+1/2,j_k}(E\mathbf{1})_{i_k+1,j_k} - a_{i-1/2,j}(E\mathbf{1})_{i_k-1,j_k} \\ &-b_{i,j+1/2}(E\mathbf{1})_{i_k,j_k+1} - b_{i,j-1/2}(E\mathbf{1})_{i_k,j_k-1} \\ &> 0. \end{aligned}$$

This proves (3).

Since S is symmetric and positive definite, and since $S_{ij} \leq 0$ for $i \neq j$, it follows that S is an M-matrix; see Varga [34]. □

3. Some well-known preconditioners for S. The rate of convergence and the efficiency of most domain decomposition algorithms depend on the choice of preconditioners M for S, both in the case of two subdomains and in the case of many subdomains. (Though our studies in this paper are restricted to the case of two subdomains, we mention here that in the case of many subdomains, a preconditioner for S can be built in terms of preconditioners for the reduced interface operator of two subdomain problems; see [4], [20].) In this section, we mention some of the preconditioners that have been proposed for S in the case of two subdomains. They include preconditioners by Axelsson and Polman [2]; Bjørstad and Widlund [3]; Bramble, Pasciak, and Schatz [5]; Chan [9]; Chan and Hou [12]; Chan and Keyes [13]; Dryja [19]; Golub and Mayers [24]; Keyes and Gropp [28], [29]; and Funaro, Quarteroni, and Zanolli [17].

The rate of convergence of these preconditioned systems are determined by the quotient of the maximum and minimum eigenvalues of $M^{-1}S$. We use the term condition number and use $\kappa(M^{-1}S)$ to denote this ratio. Many of the above-mentioned

preconditioners M are known to have *optimal* convergence rates with respect to mesh size variation, i.e.,

$$\kappa(M^{-1}S) = O(1) \quad \text{independent of } h.$$

Of these, several are based on the property that the Steklov–Poincaré operator coupling the subproblems is spectrally equivalent to the pseudodifferential operator given by the square root of the Laplacian on the interface, i.e., the operator obtained when in the eigenfunction expansion of the Laplace operator, the eigenvalues are replaced by its square roots. In the discrete case, this can be implemented efficiently using discrete sine transforms, as they form the eigenfunctions of the discrete one-dimensional Laplacian:

$$-\Delta_h = \text{tridiag}(-1, 2, -1) = W\Lambda_1 W,$$

where $W = W^{-1} = W^T$ is the $n \times n$ sine transform matrix with entries

$$W_{ij} = \sqrt{\frac{2}{n+1}} \sin\left(\frac{ij\pi}{n+1}\right)$$

where $\Lambda_1 = \text{diag}(4\sin^2(i\pi/2(n+1)))$. Such preconditioners for S have the form

$$M = W\Lambda W^{-1},$$

where W is the same as above, but where Λ is a diagonal matrix which approximates $\Lambda_1^{1/2}$. Such preconditioners can be inverted in $O(n\log(n))$ operations, if the fast sine transform is used.

We list two such commonly used preconditioners, which differ in their choice of eigenvalues:

1. The Dryja preconditioner [19] is $M_D \equiv W\Lambda_1^{1/2}W^{-1}$. Equivalently, $M_D = (-\Delta_h)^{1/2}$.

2. The Golub–Mayers preconditioner [24], $M_{GM} \equiv W(\Lambda_1 + (\Lambda_1^2/4))^{1/2}W^{-1}$, for the same Λ_1 used in M_D. We note that there is a close connection between the Golub–Mayers preconditioner and "Dirichlet–Neumann" preconditioners; see [3], [9].

A similar idea using properties of the boundary trace operator to construct effective preconditioners has been used by Glowinski and Pironneau [23] to solve the biharmonic problem.

Both M_D and M_{GM} have been shown to be *optimal* with respect to mesh refinement; see [3]. However, their performance could be sensitive to variations in the aspect ratios of the subdomains Ω_1 and Ω_2, and variations in the coefficients of L. Numerical results are presented in Tables 1, 2, and 3 of §6, which illustrate the dependence on the mesh size h, aspect ratios of the subdomains, and coefficients, respectively. Some theory for model problems is also presented in §6.

4. The interface probing preconditioner. The probing technique was introduced in Chan and Resasco [15], Keyes and Gropp [28], and Eisenstat [22], as an algebraic technique for constructing sparse approximation to the interface operator S in the two-subdomain case. One of its advantages is to account for the deterioration in the performance of some of the previously mentioned preconditioners, with respect to coefficients and aspect ratios. The main idea is to approximate S by a matrix having a specified sparsity pattern using matrix-vector products of S with a few carefully chosen probe vectors. The sparsity pattern is chosen to capture the strongest coupling

of the interface operator S, and is usually banded. The motivation for using a sparse approximation to the interface operator S is the observation that it has weak global coupling, i.e., the entries of S decay rapidly away from the diagonal. For instance, for model problems and geometries, it has been shown that

$$|S_{ij}| = O\left(\frac{1}{|i-j|^2}\right),$$

for i, j away from the diagonal; see Golub and Mayers [24]. Thus, if M is a banded approximation to S, we would hope or expect that M be an effective preconditioner. However, since S is seldom known explicitly, it is not possible to choose the exact band of S, and so we construct a banded approximation M to S using matrix-vector products of S (which is computable even if S is not explicitly known) with a few carefully chosen *probe vectors.* It turns out that the banded approximation to S obtained by probing often leads to a better preconditioner than using the exact band of S, even if these were known, because the row sums of the probed approximation tend to approximate the row sums of S, unlike the row sums of the bands of S. Note that the name interface probing originates from the fact that the approximation to S is constructed using matrix-vector products of S with a few test vectors defined on the interface Γ, to *probe* the "large" entries of S. Each such matrix-vector product of S involves the inversion of L_h on each Ω_i, with a few *probing* boundary conditions on Γ, given by the *probe vectors.* Once the matrix-vector products are found, the construction of the preconditioner M from the vector outputs is done at very little expense, with the number of operations being linearly proportional to the number of nonzero elements in M.

Consider then the problem of constructing a banded approximation M_d of upper and lower bandwidth d, to an arbitrary square matrix $C \in R^{n\times n}$ (C may be nonsymmetric in general, and may not be known explicitly). We introduce the notation

$$M_d = \text{PROBE}(C, d),$$

to denote that M_d is constructed from C using the PROBE procedure. We make the following two observations. First, as noted by Curtiss, Powell, and Reid [16], if C were a banded matrix having upper and lower bandwidth d, and C were not explicitly known, then the entries of C can be reconstructed from its action on $2d+1$ carefully chosen test vectors (instead of the standard choice of the n unit vectors e_i, forming the columns of the identity matrix). Second, if the matrix C is close to banded, i.e., its entries are small away from a band, we can compute its action on the same test vectors mentioned above and use these just as in the case where C is exactly banded, to construct a banded *approximation* to C (which we hope is good).

We illustrate the PROBE procedure for the case $d = 1$, in which case M_1 is a tridiagonal matrix, and the following three probe vectors are commonly used: $v_1 = (1,0,0,1,0,0,\cdots)^T$, $v_2 = (0,1,0,0,1,0,\cdots)^T$, and $v_3 = (0,0,1,0,0,1,\cdots)^T$. Since M_1 is tridiagonal, it can easily be checked that all its nonzero entries appear in the vectors $M_1 v_i$, $i = 1,2,3$, as illustrated below:

(8)

$$\begin{pmatrix} m_{11} & m_{12} & & & & \\ m_{21} & m_{22} & m_{23} & & & \\ & m_{32} & m_{33} & m_{34} & & \\ & & m_{43} & m_{44} & m_{45} & \\ & & & m_{54} & m_{55} & \ddots \\ & & & & \ddots & \ddots \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ \vdots & \vdots & \vdots \end{pmatrix} = \begin{pmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & m_{23} \\ m_{34} & m_{32} & m_{33} \\ m_{44} & m_{45} & m_{43} \\ m_{54} & m_{55} & m_{56} \\ \vdots & \vdots & \vdots \end{pmatrix}.$$

The probe algorithm reconstructs the nonzero entries of M_1 by equating the right-hand side of (8) to the corresponding entries in the vectors $[Cv_1, Cv_2, Cv_3]$.

By construction of M_1, we would obtain $M_1 = C$, if C is indeed tridiagonal. Note that the product Cv_1 is the sum of columns 1, 4, 7, etc. of C, while Cv_2 is the sum of columns 2, 5, 8, etc., of C, and Cv_3 is the sum of columns 3, 6, 9, etc., of C. Therefore, if C were not tridiagonal but close to being tridiagonal, then the sum of every third column of C would contain contributions from other nonzero entries in other columns. If these off-band entries of C are small, then their effects may be ignored if we only wish to construct an *approximation* to the tridiagonal band of C.

Also note that the procedure for constructing M_1 =PROBE$(C, 1)$ can be viewed as an attempt to determine a tridiagonal matrix M_1, which has the same action on the set of vectors v_i as C, i.e.,

$$M_1 v_i = C v_i, \qquad i = 1, 2, 3. \tag{9}$$

Though (9) is satisfied if C is tridiagonal, it may not be satisfied for arbitrary C, since a simple count gives $3n$ equations (one equation from each component of the three test vectors) for the $3n - 2$ unknowns on the tridiagonal band of $M_1 \in R^{n \times n}$. However, as illustrated in (8), a unique tridiagonal matrix M_1 can always be determined by choosing $3n - 2$ appropriate equations in (9), which pick out the nonzero entries of M_1. The remaining two equations in (9) may not be satisfied for arbitrary C. In addition, the tridiagonal matrix M_1 may not be symmetric, even if C is symmetric. We present an example of a 4×4 symmetric matrix C for which (9) is not satisfied, and for which M_1 is nonsymmetric.

$$\begin{pmatrix} m_{11} & m_{12} & 0 & 0 \\ m_{21} & m_{22} & m_{23} & 0 \\ 0 & m_{32} & m_{33} & m_{34} \\ 0 & 0 & m_{43} & m_{44} \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix} = \begin{pmatrix} 10 & 0 & 0 & 5 \\ 0 & 10 & 0 & 2 \\ 0 & 0 & 10 & 0 \\ 5 & 2 & 0 & 10 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

$$\Rightarrow \begin{pmatrix} m_{11} & m_{12} & 0 \\ m_{21} & m_{22} & m_{23} \\ m_{34} & m_{32} & m_{33} \\ m_{44} & 0 & m_{43} \end{pmatrix} = \begin{pmatrix} 15 & 0 & 0 \\ 2 & 10 & 0 \\ 0 & 0 & 10 \\ 15 & 2 & 0 \end{pmatrix}. \tag{10}$$

Therefore,

$$C = \begin{pmatrix} 10 & 0 & 0 & 5 \\ 0 & 10 & 0 & 2 \\ 0 & 0 & 10 & 0 \\ 5 & 2 & 0 & 10 \end{pmatrix} \Longrightarrow \text{PROBE}(C, 1) = \begin{pmatrix} 15 & 0 & 0 & 0 \\ 2 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 15 \end{pmatrix}.$$

From (10) we obtain that $m_{12} = 0$ and that $m_{21} = 2$, and thus M_1 is not symmetric. Moreover, equating the entries in row 4 column 2, we obtain the inconsistent equation that $0 = 2$, etc. However, if the matrix C is close to being tridiagonal, then M_1 and C will have almost the same action on v_i.

The procedure in equation (8) for constructing the tridiagonal PROBE$(C, 1)$ can easily be generalized to the case of banded matrices of upper and lower bandwidth d. We denote the general banded approximation by PROBE(C, d) (which can be nonsymmetric in general; see §5.3). As mentioned earlier, this requires $2d + 1$ probe vectors. Rather than describe this procedure, which is a straightforward generalization, we provide a Matlab code for constructing PROBE(A, d). Given a matrix A, the following procedure returns the probed preconditioner M =PROBE(A, d) of upper and lower bandwidth d.

MATLAB CODE FOR PROBE(A, d).

```
function M=probe(A,d)
n=length(A); k=min([2*d+1,n]);
if k == 1
  M=diag(A*ones(n,1));
else
  v= rem([1:n]'*ones(1,k),k) == ones(n,1)*[1:(k-1), 0];
  av=A*v;
  M=zeros(A);
  for c=1:k,
     for i=c:k:n,
        M(max([i-d 1]):min([i+d n]),i) = av(max([i-d 1]):min([i+d n]),c);
     end
  end
end
```

It is also possible to extend the probing technique to construct approximations having a specified structure, which is not necessarily sparse. For instance, probe approximations which are Toeplitz or circulant matrices have been constructed; see Keyes [27] and Li [30]. Probing has also been used to compute the eigenvalues of a matrix M_S approximating the interface operator S; see Chan and Keyes [13]. For instance, in [13], M_S is assumed to have the following form: $M_S = WDW^{-1}$, where W is the discrete sine transform matrix, and D is a diagonal matrix consisting of the eigenvalues of M_S. There are several ways to choose D. In [13], they let

$$D = \text{PROBE}(W^{-1}SW, 0). \tag{11}$$

If D is chosen suitably, we obtain preconditioners M_S, which are spectrally equivalent to S [13], [14]. This is called the *spectral probe* method.

5. Algebraic properties of banded probes. Although there have been many experimental studies and successful applications of the probing technique, there has not been much focus on the algebraic and analytical properties of these methods. In this section, we summarize some new as well as old results on the algebraic properties of the tridiagonal ($d = 1$) probe preconditioners.

5.1. Linearity. We note that by construction, the probe preconditioner is linearly dependent on the matrix C, i.e.,

$$\text{PROBE}(\alpha C_1 + C_2, d) = \alpha\text{PROBE}(C_1, d) + \text{PROBE}(C_2, d).$$

This property implies that probing S in (4) or the two terms $L_{13}^T L_{11}^{-1} L_{13}$ and $L_{23}^T L_{22}^{-1} L_{23}$ separately produce the same results (since L_{33} can be obtained from L).

5.2. Nonsingularity. The PROBE approximation can sometimes be singular even if the original matrix is nonsingular. However, under certain conditions, nonsingularity of the preconditioner can be proved. The following result concerns the preservation of diagonal dominance.

THEOREM 5.1. *If the ith row of C is (strictly) diagonally dominant, then the ith row of the* PROBE *approximation* $M = \text{PROBE}(C, d)$ *is also (strictly) diagonally dominant. From this it follows that* $\text{PROBE}(C, d)$ *will be nonsingular if either C is strictly diagonally dominant or if C is irreducibly diagonally dominant and* $\text{PROBE}(C, d)$ *is irreducible.*

Proof. We consider only the case $d = 1$. (The proof for general d is similar.) Recall that the entries M_{ij}, for $|i - j| \leq 1$, are defined by:

$$M_{ij} = \sum_{k:(k-j) \bmod 3 = 0} C_{ik}.$$

Using the definition of M_{ij} we obtain that

$$\begin{aligned} &|M_{ii}| - |M_{ii-1}| - |M_{ii+1}| \\ &= \left| C_{ii} - \sum_{\substack{k \neq i: \\ (k-i) \bmod 3 = 0}} C_{ik} \right| - \left| \sum_{\substack{k: \\ (k-i) \bmod 3 = 1}} C_{ik} \right| - \left| \sum_{\substack{k: \\ (k-i) \bmod 3 = 2}} C_{ik} \right|. \end{aligned}$$

Applying the triangle inequality to all three terms, we obtain that

$$|M_{ii}| - |M_{ii-1}| - |M_{ii+1}| \geq |C_{ii}| - \sum_{k \neq i} |C_{ik}|,$$

which is nonnegative due to the diagonal dominance of C. Since M is tridiagonal, this is the same as

$$|M_{ii}| \geq \sum_{j \neq i} |M_{ij}|, \tag{12}$$

which proves that M is diagonally dominant. Note that the inequalities can be replaced by strict inequalities, if S is strictly diagonally dominant. □

Unfortunately, $\text{PROBE}(C, d)$ can result in singular approximations if the given matrix is not *strictly* diagonally dominant, as the following example illustrates.

$$C = \begin{pmatrix} 1 & 0 & 0 & -1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \Longrightarrow \text{PROBE}(C, 1) = \begin{pmatrix} 0 & 0 & & \\ 0 & 1 & 0 & \\ & 0 & 1 & 0 \\ & & 0 & 1 \end{pmatrix}.$$

The next example illustrates that even if C is symmetric positive definite, $\text{PROBE}(C, 0)$ need not be.

$$C = \begin{pmatrix} 1 & -2 \\ -2 & 10 \end{pmatrix} \Longrightarrow \text{PROBE}(C, 0) = \begin{pmatrix} -1 & 0 \\ 0 & 8 \end{pmatrix}.$$

However, in our applications to the interface matrices S in elliptic problems, we are able to prove the following result.

THEOREM 5.2. *If S is the interface operator (Schur complement) corresponding to the discrete elliptic operator L_h defined in equation* (2)*, then*

1. PROBE$(S, 0)$ *is a diagonal matrix with positive diagonal entries;*
2. PROBE(S, d) *is nonsingular and strictly diagonally dominant. Hence, it will be symmetric positive definite if* PROBE(S, d) *is symmetric.*

Proof. The proof follows trivially from Theorem 2.2 and Theorem 5.1. □

5.3. Symmetry. Recall from example (10) that PROBE$(C, 1)$ can be nonsymmetric even if C is symmetric. In some preconditioned conjugate gradient methods, it is desirable to have preconditioners which preserve the symmetry of the coefficient matrix. One possible remedy which preserves the bandwidth of PROBE is to take the symmetric part of the resulting PROBE(.,.) matrix, i.e., define:

$$\text{symmetrized-PROBE}(C, 1) \equiv \left(\text{PROBE}(C, 1) + \text{PROBE}(C, 1)^T\right)/2.$$

Unfortunately, symmetrized-PROBE(.,.) does not preserve diagonal dominance of even strictly diagonally dominant matrices, as the following example illustrates:

$$C = \begin{pmatrix} 100 & 0 & 0 & 0 & 50 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 50 & 0 & 0 & 0 & 100 \end{pmatrix} \Rightarrow \text{PROBE}(C, 1) = \begin{pmatrix} 100 & 50 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 50 & 100 \end{pmatrix}$$

(13)

$$\Rightarrow \text{symmetrized-PROBE}(C, 1) = \begin{pmatrix} 100 & 25 & 0 & 0 & 0 \\ 25 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 25 \\ 0 & 0 & 0 & 25 & 100 \end{pmatrix}.$$

However, such problems occur very rarely in applications to interface matrices, where there is a decay in the entries of the matrix we probe. As an alternative to the symmetrized-PROBE, the following minmodsym-PROBE can also be used to obtain a symmetric banded approximation, preserving diagonal dominance. The off-diagonal entries (i, j) and (j, i) of minmodsym-PROBE(C, d) are chosen to be the (i, j) or (j, i) entry of PROBE(C, d) having smaller modulus, i.e., if we let M =PROBE(C, d), then:

$$\text{minmodsym-PROBE}(C, d)_{ij} \equiv \begin{cases} M_{ij} & \text{if } |M_{ij}| = \min\{|M_{ij}|, |M_{ji}|\}, \\ M_{ji} & \text{if } |M_{ji}| = \min\{|M_{ij}|, |M_{ji}|\}. \end{cases}$$

However, the minmodsym procedure is no longer linear. As the next theorem indicates, this procedure preserves symmetry and strict diagonal dominance.

THEOREM 5.3. *If C is symmetric and strictly diagonally dominant, and $C_{ii} > 0$ then*

$$M = \text{minmodsym-PROBE}(C, d)$$

is symmetric positive definite and strictly diagonally dominant.

Proof. Symmetry follows by construction. Diagonal dominance is preserved since PROBE(C, d) preserves diagonal dominance, and since the off-diagonal entries of minmodsym-PROBE(C, d) are chosen to decrease the modulus of the off-diagonal terms of PROBE(C, d). □

There is an alternative procedure to compute symmetric approximations, due to Keyes and Gropp [28], [29], in which a banded, symmetric approximation having upper and lower bandwidth d is constructed based on using $d + 1$ probe vectors. We will

refer to this as the symmetric-PROBE and denote it by symmetric-PROBE(.,.). The procedure is linear, and it is computationally less expensive than PROBE(.,d). We illustrate the procedure by an example for the case $d = 1$, i.e., to construct a symmetric tridiagonal approximation. In this case the probe vectors are $v_1 = (1,0,1,0,\cdots)^T$ and $v_2 = (0,1,0,1,\cdots)^T$, and we have:

$$\text{(14)}\qquad \begin{pmatrix} a_1 & b_2 & & & \\ b_2 & a_2 & b_3 & & \\ & b_3 & a_3 & b_4 & \\ & & b_4 & a_4 & \ddots \\ & & & \ddots & \ddots \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ \vdots & \vdots \end{pmatrix} = \begin{pmatrix} a_1 & b_2 \\ b_2 + b_3 & a_2 \\ a_3 & b_3 + b_4 \\ b_4 + b_5 & a_4 \\ \vdots & \vdots \end{pmatrix}$$
$$= [Cv_1, Cv_2].$$

The symmetric tridiagonal approximation symmetric-PROBE$(C,1)$ = tridiag(b_i, a_i, b_{i+1}) is obtained from the probed output vectors Cv_1, Cv_2, as indicated in the following algorithm.

SYMMETRIC-PROBE ALGORITHM.

For $i = 1,\cdots,n$

$$a_i = \begin{cases} (Cv_1)_i & \text{if } i \text{ is odd,} \\ (Cv_2)_i & \text{if } i \text{ is even,} \end{cases}$$

$b_2 = (Cv_2)_1$.

For $i = 3,\cdots,n$,

$$b_i = \begin{cases} (Cv_1)_{i-1} - b_{i-1} & \text{if } i \text{ is odd,} \\ (Cv_2)_{i-1} - b_{i-1} & \text{if } i \text{ is even.} \end{cases}$$

System (14) consists of $2n$ equations for $2n-1$ unknowns, and is therefore an overdetermined system. However, if the matrix C we probe is symmetric, it can be shown that the resulting system is consistent (see [28], [29]) and from this it then follows that Cv_i = symmetric-PROBE$(C,1)v_i$ for $i = 1,2$. The general banded symmetric-PROBE follows easily by using $d+1$ probe vectors, with 1s every $(d+1)$th column. Note that symmetric-PROBE(.,d) requires d less probe vectors than PROBE(.,d) and hence less subdomain solves.

Unfortunately, the symmetric-PROBE(.,.) does not preserve diagonal dominance or positive definiteness in general, as the following example illustrates:

$$\text{If } C = \begin{pmatrix} 3 & -1 & 0 & -2 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ -2 & 0 & -1 & 4 \end{pmatrix} \quad \text{then PROBE}(C,1) = \begin{pmatrix} 1 & -1 & & \\ -1 & 2 & -1 & \\ & 1 & -2 & 1 \\ & & -1 & 2 \end{pmatrix},$$

which preserves diagonal dominance, but

$$\text{symmetric-PROBE}(C,1) = \begin{pmatrix} 3 & -3 & & \\ -3 & 2 & 1 & \\ & 1 & 2 & -2 \\ & & -2 & 4 \end{pmatrix},$$

which is not diagonally dominant. However, such difficulties are encountered only rarely in applications for matrices having decay properties.

An idea similar to probing has been used by Axelsson and Polman [2] to construct a tridiagonal symmetric approximation M_{AP} to a symmetric matrix C, based

on the test vectors $v_1 = (1, \cdots, 1)^T$ and $v_2 = (1, 2, 3, \cdots, n)^T$. They have shown that their resulting approximation M_{AP} satisfies $M_{AP} v_i = C v_i$, for $i = 1, 2$. The Axelsson–Polman PROBE preserves diagonal dominance. In addition, under certain assumptions, they have been able to obtain lower bounds for the spectrum of M_{AP} in terms of C.

6. Probe preconditioners in domain decomposition. The preceding section dealt with purely algebraic properties of the probing technique, valid for arbitrary matrices. In this section we study convergence properties of the tridiagonal probing technique applied to the interface matrix S in (4) for a model elliptic problem known to have a condition number growing at a rate h^{-1}, as $h \to 0$. Our studies focus on how the mesh size, aspect ratios of the subdomains, and variations in the coefficients affect the rate of convergence of the probe preconditioned system for the interface matrix S.

In particular, we show that an application of a version of the tridiagonal probe preconditioner results in a condition number that grows at a rate $h^{-1/2}$ as $h \to 0$. We also show that this condition number is generally insensitive to variations in the aspect ratios of the subdomains. Finally, we consider how this condition number depends on the coefficients of the elliptic problem. There we present theoretical bounds for the condition number of the preconditioned system when the coefficients are scaled by positive scalar constants on each subdomain. In all these cases, we present both theoretical and numerical comparisons of the convergence rates of the Golub–Mayers preconditioner with the probed preconditioner.

6.1. Eigendecomposition of the Schur complement S and the probe preconditioner for a model elliptic problem. The model problem we consider is the Laplacian on the rectangle $\Omega = [0, l_1 + l_2] \times [0, 1]$, as illustrated in Fig. 1, with Dirichlet boundary conditions on the vertical boundaries and *periodic* boundary conditions on the horizontal boundaries:

$$\left\{ \begin{array}{rcll} -\Delta u & = & f & \text{in } \Omega, \\ u(x,y) & = & 0 & \text{for } y \in [0,1] \text{ and } x = 0, \\ u(x,y) & = & 0 & \text{for } y \in [0,1] \text{ and } x = l_1 + l_2, \\ u(x,0) & = & u(x,1) & \text{for } x \in (0, l_1 + l_2). \end{array} \right. \tag{15}$$

Ω is partitioned into two subdomains $\Omega_1 = [0, l_1] \times [0, 1]$ and $\Omega_2 = [l_1, l_1 + l_2] \times [0, 1]$, with the interface $\Gamma = \{l_1\} \times [0, 1]$. Problem (15) is discretized by the five-point Laplacian on an $(m_1 + m_2 + 3) \times (n + 2)$ grid, which includes the boundary nodes, with mesh size $h = 1/(n + 1)$, $l_1 = (m_1 + 1)h$, and $l_2 = (m_2 + 1)h$. Note that there are $n + 1$ distinct unknowns on each vertical line, due to periodicity, and there are $m_1 + m_2 + 1$ unknowns in the interior of each horizontal line.

For this model problem, the eigendecomposition of the Schur complement S, as well as the eigendecomposition of a suitable tridiagonal PROBE of the Schur complement, can be computed exactly. The eigendecomposition of S can be found using the discrete eigenfunctions of the five-point Laplacian (see Chan [9] and Donato [18]) and is given below:

$$S = F \text{diag}(\lambda_0, \cdots, \lambda_n) F^{-1},$$

where

$$\lambda_0 = \left(\frac{1}{m_1 + 1} + \frac{1}{m_2 + 1} \right)$$

and

$$\lambda_j = \left(\frac{1+\gamma_j^{m_1+1}}{1-\gamma_j^{m_1+1}} + \frac{1+\gamma_j^{m_2+1}}{1-\gamma_j^{m_2+1}} \right) \sqrt{\sigma_j + \frac{\sigma_j^2}{4}}$$

for $j = 1, \cdots, n$ with σ_j and γ_j defined by

$$\sigma_j = 4 \sin^2(j\pi h),$$

$$\gamma_j \equiv \frac{1 + (\sigma_j/2) - \sqrt{\sigma_j + (\sigma_j^2/4)}}{1 + (\sigma_j/2) + \sqrt{\sigma_j + (\sigma_j^2/4)}}.$$

$F = [f_0, \cdots, f_n]$ is a unitary matrix ($F^{-1} = F^H$), with $f_j = \sqrt{h}(1, e^{2\pi jh}, \cdots, e^{2\pi njh})^T$.

Note that if l_1, $l_2 \to \infty$, then $\lambda_0 \to 0$, and the interface matrix becomes singular. Therefore, we restrict to the case where either l_1 or l_2 is $O(1)$ independent of h.

For this model problem, $S \in R^{(n+1)\times(n+1)}$ is symmetric and circulant (due to periodicity), and so rather than defining M =symmetrized-PROBE(S, 1) discussed in §4 (which would result in a tridiagonal but not circulant approximation), we construct a tridiagonal, circulant approximation $M_{CP} \in R^{(n+1)\times(n+1)}$ by using a variant of the tridiagonal-PROBE which we denote M_{CP} = circulant-PROBE(S, 1). We describe it for the case $n+1$ being even, in which case we need just one probe vector $\equiv (1,0,1,0,\cdots,1,0)^T$:

(16)

$$M_{CP} = \begin{pmatrix} \alpha & -\beta & & -\beta \\ \ddots & \ddots & \ddots & \\ & \ddots & \ddots & \ddots \\ -\beta & & -\beta & \alpha \end{pmatrix} \quad \text{with } M_{CP} \begin{pmatrix} 1 \\ 0 \\ 1 \\ \vdots \end{pmatrix} = \begin{pmatrix} \alpha \\ -2\beta \\ \alpha \\ \vdots \end{pmatrix} := S \begin{pmatrix} 1 \\ 0 \\ 1 \\ \vdots \end{pmatrix}.$$

The values α and β are easily found. The following lemma contains estimates for $\alpha - 2\beta$ and β.

LEMMA 6.1. *The row sum $\alpha - 2\beta$ of M_{CP} = circulant-*PROBE *(S, 1) satisfies:*

$$\alpha - 2\beta = \sum_{i=0}^{n} S_{ji} = \lambda_0 = \left(\frac{h}{l_1} + \frac{h}{l_2} \right) \qquad \text{for } j = 0, \cdots, n,$$

and $1 \le \beta \le 2$.

Proof. Our proof will be based on the fact that S and M_{CP} have the same row sums. We use two probe vectors $v_1 = (1,0,1,0,\cdots,1,0)^T$ and $v_2 = (0,1,0,1,\cdots, 0,1)^T$. First, it can be easily verified that if $Sv_1 = M_{CP}v_1$ (which holds by construction of M_{CP}), then $Sv_2 = M_{CP}v_2$. Since $v_1 + v_2 = (1,1,\cdots,1)^T$, we obtain $M_{CP}(1,\cdots,1)^T = S(1,\cdots,1)^T$, i.e., the row sums must be equal. Using the fact that the row sum of M_{CP} is $\alpha - 2\beta$, we obtain that:

$$\alpha - 2\beta = \sum_{i=0}^{n} S_{ji} = (S\mathbf{1})_j = \lambda_0 = \left(\frac{h}{l_1} + \frac{h}{l_2} \right) > 0 \quad \text{for } j = 0, \cdots, n,$$

since $(1,\cdots,1)^T$ is an eigenvector of S corresponding to eigenvalue λ_0.

To prove that $1 \leq \beta \leq 2$, we use the expression for -2β given in (16), and use the alternative expression for the entries of S_{ij} in terms of *discrete harmonic extensions*, as described in (5) of §2, to obtain that:

$$-2\beta = (L_h E v_1)_{ij},$$

where (i,j) is any node on Γ with 0 nodal value for the probe vector v_1, and E denotes the discrete harmonic extension and L_h denotes the discretization of the elliptic operator. Applying the five-point Laplacian at node (i,j) on Γ results in:

$$-2\beta = -2 - (Ev_1)_{i,j+1} - (Ev_1)_{i,j-1}.$$

By the maximum principle the entries of Ev_1 lie between 0 and 1 at all other nodes; thus it follows that $-2 \geq -2\beta \geq -4$ and the result follows. □

The eigendecomposition of M_{CP} can be explicitly found for the model problem.

LEMMA 6.2. *The circulant-*PROBE *matrix M_{CP} is diagonalized by the discrete Fourier transform F and has eigendecomposition:*

$$M_{CP} = F \ \mathrm{diag}\,(\lambda_0 + \beta\sigma_j) \ F^{-1}. \tag{17}$$

Proof. Since M_{CP} is circulant, it is diagonalized by the discrete Fourier transform F. Its eigenvalues can be determined by applying M_{CP} to each column of F, for $j = 0, \cdots, n$:

$$\begin{aligned}(M_{CP} f_j)_i &= -\beta e^{(i-1)j2\pi h} + \alpha e^{ij2\pi h} - \beta e^{(i+1)j2\pi h} \\ &= \left(\alpha - 2\beta + \beta 4\sin^2(j\pi h)\right) e^{ij2\pi h} \\ &= (\lambda_0 + \beta\sigma_j)\, e^{ij2\pi h} = (\lambda_0 + \beta\sigma_j)\,(f_j)_i \quad \text{for } j = 0, \cdots, n,\end{aligned}$$

where $\sigma_j = 4\sin^2(j\pi h)$. Thus, $M_{CP} = F \ \mathrm{diag}(\lambda_0 + \beta\sigma_j) \ F^{-1}$. □

Since M_{CP} and S are diagonalized by F, we obtain that:

$$M_{CP}^{-1} S = F \ \mathrm{diag}\left(\frac{\lambda_j}{\lambda_0 + \beta\sigma_j}\right) \ F^{-1}.$$

For convenience, we define $\Phi_j \equiv \lambda_j/(\lambda_0 + \beta\sigma_j)$ for $j = 0, \cdots, n$. Then the condition number of $M_{CP}^{-1}S$ is determined by the quotient of the maximum and minimum of Φ_j for $j = 0, \cdots, n$. This quotient is 1 when $j = 0$, since $\sigma_0 = 0$. To determine bounds for the extrema when $j = 1, \cdots, n$, we replace the discrete optimization problem by the optimization problem for its natural continuous extension, $\Phi(\sigma)$, where

$$\Phi(\sigma) = \left(\frac{1 + \gamma(\sigma)^{l_1/h}}{1 - \gamma(\sigma)^{l_1/h}} + \frac{1 + \gamma(\sigma)^{l_2/h}}{1 - \gamma(\sigma)^{l_2/h}}\right) \frac{\sqrt{\sigma + \frac{\sigma^2}{4}}}{\left(\frac{h}{l_1} + \frac{h}{l_2}\right) + \beta\sigma},$$

with $\gamma(\sigma)$ defined by

$$\gamma(\sigma) \equiv \frac{1 + \frac{\sigma}{2} - \sqrt{\sigma + \frac{\sigma^2}{4}}}{1 + \frac{\sigma}{2} + \sqrt{\sigma + \frac{\sigma^2}{4}}}, \tag{18}$$

and where the discrete values of σ_j for j varying from 1 to n were replaced by the continuous variable $\sigma \in [4\sin^2(\pi h), 4]$, and the eigenvalues $\Phi_j = \lambda_j/(\lambda_0 + \beta\sigma_j)$ were replaced by its continuous counterparts $\Phi(\sigma) = \lambda(\sigma)/(\lambda_0 + \beta\sigma)$.

An upper bound for the condition number of the preconditioned system can thus be expressed in terms of $\Phi(\sigma)$ as follows:

$$\kappa(M_{CP}^{-1}S) \leq \frac{\max\{\max_\sigma \Phi(\sigma), 1\}}{\min\{\min_\sigma \Phi(\sigma), 1\}} \quad \text{for } \sigma \in [4\sin^2(\pi h), 4], \tag{19}$$

and so the bounds for $\Phi(\sigma)$ determine the rate of convergence of the preconditioned system.

6.2. Dependence on mesh size h. We now consider bounds for the eigenvalues $\Phi(\sigma)$, for $\sigma \in [4\sin^2(\pi h), 4]$. Because there should be at least one line of unknowns in the interior of each subdomain, $2h \leq l_1$ and $2h \leq l_2$. The eigenvalues $\Phi(\sigma)$ can be rewritten as: $\Phi(\sigma) = \mu(\sigma)H(\sigma)$, where

$$\mu(\sigma) \equiv \left(\frac{1+\gamma(\sigma)^{l_1/h}}{1-\gamma(\sigma)^{l_1/h}} + \frac{1+\gamma(\sigma)^{l_2/h}}{1-\gamma(\sigma)^{l_2/h}}\right) \quad \text{and} \quad H(\sigma) \equiv \frac{\sqrt{\sigma + \frac{\sigma^2}{4}}}{\left(\frac{h}{l_1} + \frac{h}{l_2}\right) + \beta\sigma}. \tag{20}$$

In the following lemma, we list some properties of $\Phi(\sigma)$, $\mu(\sigma)$, $H(\sigma)$, and $\gamma(\sigma)^{l_i/h}$.

LEMMA 6.3. *The following hold:*

1. *There exist positive constants c_1 and c_2 independent of h, l_1, and l_2 such that $\gamma(\sigma)^{l_i/h}$ satisfies:*

$$0 \leq e^{-(l_i/h)c_1\sqrt{\sigma}} \leq \gamma(\sigma)^{l_i/h} \leq e^{-(l_i/h)c_2\sqrt{\sigma}} \leq 1 \quad \text{for } \sigma \in [4\sin^2(\pi h), 4].$$

2. *For the constants c_1 and c_2 given in* (1) *above, the function $\mu(\sigma)$ satisfies:*

$$2 + \sum_{i=1}^{2} \frac{2e^{-(l_i/h)c_1\sqrt{\sigma}}}{1 - e^{-(l_i/h)c_1\sqrt{\sigma}}} \leq \mu(\sigma) \leq 2 + \sum_{i=1}^{2} \frac{2e^{-(l_i/h)c_2\sqrt{\sigma}}}{1 - e^{-(l_i/h)c_2\sqrt{\sigma}}}.$$

3. *$H(\sigma)$ has a unique critical point at*

$$\sigma = \sigma^\star \equiv \frac{\frac{h}{l_1} + \frac{h}{l_2}}{\beta - \left(\frac{h}{2l_1} + \frac{h}{2l_2}\right)},$$

where the maximum of the function is attained. $H(\sigma)$ is a monotonically decreasing function for $\sigma > \sigma^\star$ and is monotonically increasing for $\sigma < \sigma^\star$.

4. *$\mu'(\sigma) \leq 0$ for $\sigma \in [4\sin^2(\pi h), 4]$, and is thus a decreasing function.*
5. *$\Phi'(\sigma) \leq 0$ for $\sigma \geq \sigma^\star$.*

Proof. To prove (1), we first write

$$\gamma(\sigma)^{l_i/h} = e^{(l_i/h)\log(\gamma(\sigma))},$$

and expand $\log(\gamma(\sigma))$ in a Taylor series in $\sqrt{\sigma}$, with remainder term and evaluate the remainder term to obtain uniform error bounds. We outline the steps. Substituting the expression for $\gamma(\sigma)$ into $\log(\gamma(\sigma))$, we obtain:

$$\log(\gamma(\sigma)) = \log(1 - z(\sigma)), \qquad \text{where } z(\sigma) \equiv \frac{2\sqrt{\sigma + \frac{\sigma^2}{4}}}{1 + \frac{\sigma}{2} + 2\sqrt{\sigma + \frac{\sigma^2}{4}}},$$

where, for $\sigma \in [0,4]$, the function $z(\sigma) \in [0, 4\sqrt{2}/3 + 4\sqrt{2}]$. Expanding $\log(1-z)$ in a Taylor series with remainder, about $z = 0$, it can be easily shown that:

$$-z\left(\frac{59+24\sqrt{2}}{18}\right) \leq \log(1-z) \leq -z \quad \text{for } z \in \left[0, \frac{4\sqrt{2}}{3+4\sqrt{2}}\right].$$

Expanding $z(\sigma)$ in a series in $\sqrt{\sigma}$, we obtain:

$$\frac{2\sqrt{\sigma}}{3+4\sqrt{2}} \leq z(\sigma) \leq 2\sqrt{2}\sqrt{\sigma} \quad \text{for } \sigma \in [0,4].$$

Combining the preceding results, we obtain that $\log(\gamma(\sigma))$ is uniformly equivalent to $\sqrt{\sigma}$ for $\sigma \in I_1$. Substituting this as argument for the exponential function, which is monotone increasing, we obtain the uniform upper and lower bounds given in (1), with specific values for the constants c_1 and c_2. We omit the details.

The proof of (2) follows easily from (1) by using the definition of $\mu(\sigma)$.

Here we prove (3). The derivative of $H(\sigma)$ is easily verified to be:

$$H'(\sigma) = \frac{\left(\beta - \frac{h}{2l_1} - \frac{h}{2l_2}\right)\left(-\sigma + \frac{(h/l_1)+(h/l_2)}{\beta-(h/2l_1)-(h/2l_2)}\right)}{2\sqrt{\sigma+\frac{\sigma^2}{4}}\left(\frac{h}{l_1}+\frac{h}{l_2}+\beta\sigma\right)^2}.$$

From this, we see that the only critical point of $H(\sigma)$ occurs at

$$\sigma^\star = \frac{\frac{h}{l_1}+\frac{h}{l_2}}{\beta - \left(\frac{h}{2l_1}+\frac{h}{2l_2}\right)}.$$

That this critical point corresponds to a maximum is easily shown by observing that $H'(\sigma)$ is positive to the left of the critical point and that it is negative to the right of the critical point.

Part (4) is easily proved using the expression for the derivative of $\mu(\sigma)$:

$$\mu'(\sigma) = \sum_{i=1}^{2} \frac{2\left(\frac{l_i}{h}\right)\gamma'(\sigma)\gamma(\sigma)^{(l_i/h)-1}}{\left(1-\gamma(\sigma)^{l_i/h}\right)^2},$$

which is nonpositive since

$$\gamma'(\sigma) = \frac{-1}{\sqrt{\sigma+\frac{\sigma^2}{4}}\left(1+\frac{\sigma}{2}+\sqrt{\sigma+\frac{\sigma^2}{4}}\right)^2} \leq 0.$$

To prove (5), we consider the expression for $\Phi'(\sigma) = \mu'(\sigma)H(\sigma) + \mu(\sigma)H'(\sigma)$. We note that $\mu(\sigma)$ and $H(\sigma)$ are nonnegative, and that $\mu'(\sigma) \leq 0$ from part (4) of the proof. Using the expression for $H'(\sigma)$ given in part (3), we see that $H'(\sigma) \leq 0$ for $\sigma \geq \sigma^\star$. Combining these results, we obtain that $\Phi'(\sigma) \leq 0$ for $\sigma \geq \sigma^\star$. □

The following theorem contains the main result of this section.

THEOREM 6.4. *If* $\min\{l_1, l_2\}$ *is* $O(1)$ *independent of* h, *then for small enough* h, *the eigenvalues* $\Phi(\sigma)$ *of* $M_{CP}^{-1}S$ *satisfy:*

$$C_1 \leq \Phi(\sigma) \leq C_2\sqrt{\frac{l_1 l_2}{(l_1+l_2)}}h^{-1/2},$$

for positive constants C_1, C_2 independent of h, l_1, and l_2.

Proof. Essentially, the bounds for $\Phi(\sigma)$ will be shown to be determined by bounds for the maximum and minimum of $H(\sigma)$, though there are some difficulties due to the presence of the $\mu(\sigma)$ term, which can become large for small values of l_1 and l_2. $H(\sigma)$ is the quotient of a square root function representing the eigenvalues of S and a linear function representing the eigenvalues of the PROBE approximation, and its maximum and minimum can be computed explicitly. The details are now described.

We consider two cases separately. In Case 1, we assume that the aspect ratios l_1 and l_2 are both strictly greater than 1. In this case, the function $\mu(\sigma)$ will be shown to be uniformly bounded and the bounds for $\Phi(\sigma)$ are obtained by finding bounds for the maximum and minimum of $H(\sigma)$. This can be done using results in Lemma 6.3. In Case 2, we assume that at least one of the aspect ratios l_1 or l_2 is smaller than or equal to 1. In this case, the function $\mu(\sigma)$ is not uniformly bounded, and the proof used in Case 1 has to be modified. We prove the bounds for $\Phi(\sigma)$ by considering two subintervals separately. The details are outlined below. For convenience, throughout the proof we let C_1 and C_2 denote some generic positive constants independent of h, l_1 and l_2.

Case 1. In this case, l_1 and l_2 are both assumed greater than 1. Then, since

$$\frac{\sqrt{\sigma}}{h} \geq \frac{2\sin(\pi h)}{h} \geq C_1,$$

it follows that $l_i\sqrt{\sigma}/h \geq C_1$. Substituting this into the expression for the bounds for $\mu(\sigma)$, given in part 2 of Lemma 6.3, we obtain uniform upper and lower bounds:

$$C_1 \leq \mu(\sigma) \leq C_2 \quad \text{for } \sigma \in [4\sin^2(\pi h), 4].$$

Since $\Phi(\sigma) = \mu(\sigma)H(\sigma)$, we can obtain bounds for $\Phi(\sigma)$ by considering bounds for $H(\sigma)$. Since by assumption $\min\{l_1, l_2\}$ is $O(1)$ independent of h, it follows that for small enough h, we have $\sigma^\star = O(h)$, and therefore $\sigma^\star > 4\sin^2(\pi h)$. Thus the maximum of $H(\sigma)$ occurs in the interior of the interval $[4\sin^2(\pi h), 4]$, by part 3 of Lemma 6.3. In this case, $H(\sigma)$ is monotone increasing to the left of $\sigma^\star$, and monotone decreasing to the right of $\sigma^\star$. Thus, we obtain that:

$$\min\{H(4\sin^2(\pi h)), H(4)\} \leq H(\sigma) \leq H(\sigma^\star) \quad \text{for } \sigma \in [4\sin^2(\pi h), 4].$$

Substituting the expression for $\sigma^\star$ into $H(\sigma)$, it can be easily shown that:

$$H(\sigma^\star) \leq C_2\sqrt{\frac{l_1 l_2}{l_1 + l_2}}h^{1/2}.$$

At $\sigma = 4\sin^2(\pi h)$, it can be shown that:

$$H(4\sin^2(\pi h)) \geq \frac{1}{((l_1 + l_2)/l_1 l_2) + \beta h},$$

which becomes large if both l_1 and l_2 become large. At $\sigma = 4$, it can easily be shown that:

$$\frac{\sqrt{8}}{9} \leq H(4).$$

Thus the minimum of $H(\sigma)$ is always $O(1)$. Combining these bounds with the uniform bounds for $\mu(\sigma)$, we obtain that:

$$C_1 \leq \Phi(\sigma) \leq C_2 \sqrt{\frac{l_1 l_2}{l_1 + l_2}} h^{-1/2}.$$

Case 2. Here we assume that either l_1 or l_2 is smaller than or equal to 1. For definiteness, let us suppose that $l_1 \leq 1$ and $l_1 \leq l_2$. In this case, $\mu(\sigma)$ may no longer be uniformly bounded. Indeed, $\mu(4\sin^2(\pi h))$ can be of size $O(1/l_1)$. Consequently, we do not consider bounds for $H(\sigma)$ and $\mu(\sigma)$ separately, as they lead to bounds which are larger than is the case. Instead, we find uniform bounds for $\Phi(\sigma)$ on the two subintervals $I_1 \equiv [4\sin^2(\pi h), \sigma^\star]$ and $I_2 \equiv [\sigma^\star, 4]$, separately (as mentioned before, when either l_1 or l_2 is $O(1)$, for small enough h we have $\sigma^\star > 4\sin^2(\pi h)$). In Case 2a, we obtain bounds for $\Phi(\sigma)$ on the interval I_1 using uniformly valid expansions for $\Phi(\sigma)$. In Case 2b, we obtain bounds for $\Phi(\sigma)$ on the interval I_2, using the fact that $\Phi(\sigma)$ is monotone decreasing on I_2, by Lemma 6.3. The details of both subcases are given below.

Case 2a. On interval I_1, we will first show that $\Phi(\sigma)$ satisfies:

$$C_1 \frac{l_1\sqrt{\sigma}}{h}\left(1 + \frac{e^{-(c_1 l_1/h)\sqrt{\sigma}}}{1 - e^{-(c_1 l_1/h)\sqrt{\sigma}}}\right) \leq \Phi(\sigma) \leq C_2 \frac{\sqrt{\sigma} l_1}{h}\left(1 + \frac{e^{-(c_2 l_1/h)\sqrt{\sigma}}}{1 - e^{-(c_2 l_1/h)\sqrt{\sigma}}}\right). \tag{21}$$

To show this, we note that:

$$\frac{h}{2l_1} \leq \sigma^\star = \frac{h/l_1 + h/l_2}{\beta - \left(\frac{h}{2l_1} + \frac{h}{2l_2}\right)} \leq \frac{4h}{l_1},$$

since $2h \leq l_1 \leq l_2$, and $1 \leq \beta \leq 2$. From this it follows that:

$$\beta\sigma \leq \beta\sigma^\star \leq 4\beta\frac{h}{l_1} \leq 8\frac{h}{l_1} \quad \text{for } \sigma \in I_1.$$

This result can be used to obtain bounds for the denominator of $H(\sigma)$:

$$\frac{h}{l_1} \leq \frac{h}{l_1} + \frac{h}{l_2} + \beta\sigma \leq 10\frac{h}{l_1} \quad \text{for } \sigma \in I_1, \tag{22}$$

which in turn gives bounds for $H(\sigma)$:

$$\frac{C_1 l_1\sqrt{\sigma}}{h} \leq H(\sigma) \leq \frac{C_2 l_1\sqrt{\sigma}}{h} \quad \text{for } \sigma \in I_1. \tag{23}$$

Next, we obtain bounds for $\mu(\sigma)$ by modifying part 2 of Lemma 6.3, in which the sums are replaced with bounds for each term. We easily obtain:

$$1 + \frac{e^{-(c_1 l_1/h)\sqrt{\sigma}}}{1 - e^{-(c_1 l_1/h)\sqrt{\sigma}}} \leq \mu(\sigma) \leq 4 + \frac{4e^{-(c_2 l_1/h)\sqrt{\sigma}}}{1 - e^{-(c_2 l_1/h)\sqrt{\sigma}}} \quad \text{for } \sigma \in [4\sin^2(\pi h), 4]. \tag{24}$$

Combining (23) and (24), we obtain the bounds for $\Phi(\sigma) = \mu(\sigma)H(\sigma)$ given in (21).

Note that the upper and lower bounds for $\Phi(\sigma)$ in (21) can be expressed in terms of a single function $T(z)$:

$$\frac{C_1}{c_1} T\left(\frac{c_1 l_1\sqrt{\sigma}}{h}\right) \leq \Phi(\sigma) \leq \frac{C_2}{c_2} T\left(\frac{c_2 l_1\sqrt{\sigma}}{h}\right), \quad \text{where } T(z) \equiv z\left(1 + \frac{e^{-z}}{1 - e^{-z}}\right),$$

for $z \equiv c_i l_1 \sqrt{\sigma}/h$ varying in the interval

$$\left[\frac{2l_1 c_i}{h} \sin(\pi h), \frac{l_1 c_i}{h}\sqrt{\sigma^\star}\right],$$

which is a subset of the interval

$$\left[0, \frac{l_1 c_i}{h}\sqrt{\sigma^\star}\right].$$

It is easily verified that

$$T'(z) = \frac{e^z\,(e^z - 1 - z)}{(e^z - 1)^2} > 0 \quad \text{for } z > 0.$$

Thus, $T(z)$ is a monotone increasing function satisfying:

$$T(0) \le T(z) \le T\left(\frac{c_2 l_1 \sqrt{\sigma^\star}}{h}\right) \quad \text{for } z \in \left[0, \frac{c_2 l_1 \sqrt{\sigma^\star}}{h}\right].$$

We note that the lower bound is $T(0) = 1$. It is easily shown for sufficiently large z, say $z \ge 1$, that $T(z) \le C_2 z$, for some positive constant C_2. Since

$$\frac{h}{2l_1} \le \sigma^\star \le 4\frac{h}{l_1},$$

we obtain that

$$c_2\sqrt{\sigma^\star}\frac{l_1}{h} \ge c_2\sqrt{\frac{l_1}{2h}} \ge 1$$

for sufficiently small h, 8 and that

$$T\left(\frac{c_2 l_1 \sqrt{\sigma^\star}}{h}\right) \le C_1 \frac{c_i l_1 \sqrt{\sigma^\star}}{h} \le C_2 \sqrt{l_1} h^{-1/2}.$$

Substituting this in the expression for $\Phi(\sigma)$, we obtain that

$$C_1 \le \Phi(\sigma) \le C_2 \sqrt{l_1} h^{-1/2} \quad \text{for } \sigma \in I_1.$$

Since

$$\frac{l_1}{2} \le \frac{l_1 l_2}{l_1 + l_2},$$

it follows that this is our desired result.

Case 2b. Finally, we consider bounds for $\Phi(\sigma)$, when $\sigma \in I_2$. Since $\sigma \ge \sigma^\star$, we obtain by part 5 of Lemma 6.3 that $\Phi'(\sigma) \le 0$ and thus

$$\Phi(\sigma^\star) \ge \Phi(\sigma) \ge \Phi(4) \quad \text{for } \sigma \in I_2.$$

Since $\Phi(\sigma^\star) \le C_2 \sqrt{l_1} h^{-1/2}$ and $\Phi(4) \ge C_2$, we obtain the same bounds for $\Phi(\sigma)$ on the interval I_2 as on the interval I_1. Since

$$\frac{l_1}{2} \le \frac{l_1 l_2}{l_1 + l_2},$$

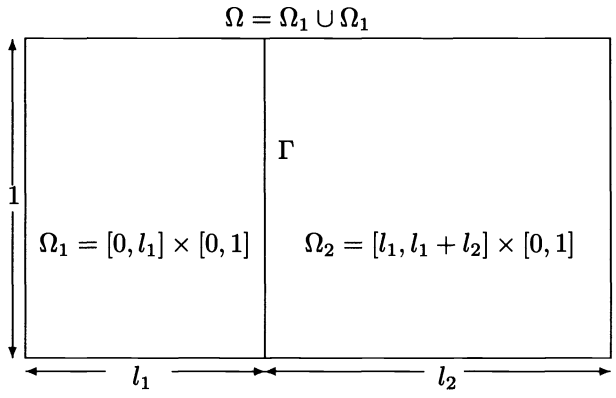


FIG. 1. *A model domain.*

TABLE 1
h dependence for $\theta_1 = 2$, $\theta_2 = -2$, *and* $l_1 = l_2 = \frac{1}{2}$.

n	$\kappa(M_1^{-1}S)$	Iters		$\kappa(M_{GM}^{-1}S)$	Iters		$\kappa(M_{SGM}^{-1}S)$	Iters
10	1.22	6		1.80	7		-	-
20	1.62	8		1.85	7		2.29	9
30	1.97	9		1.87	7		2.34	10
40	2.28	10		1.88	7		2.38	9

it follows that this is our desired result. □

Next we present some sample numerical results that compare the symmetrized tridiagonal probe preconditioner M_1 = symmetrized-PROBE$(S, 1)$ with the Golub–Mayers preconditioner M_{GM} for the following elliptic problem on the domain Ω of Fig. 1:

$$(25) \quad Lu = -\frac{\partial}{\partial x}\left(e^{\theta_1 xy}\frac{\partial u}{\partial x}\right) - \frac{\partial}{\partial y}\left(e^{\theta_2 xy}\frac{\partial u}{\partial y}\right) = f \quad \text{in } \Omega, \quad u = 0 \quad \text{on } \partial\Omega.$$

The five-point centered scheme (2) was used on an $n \times n$ grid, with subdomains of size $m_1 \times n$ and $m_2 \times n$, where $m_1 + m_2 = n$. The condition number and the iterations required to reduce the residual by a factor of 10^{-7} (in the Euclidean norm) are listed for varying choices of h, l_1, l_2, θ_1 and θ_2.

Table 1 lists the condition numbers and the number of iterations required for both preconditioners as the mesh width h is varied, with the values of l_1, l_2, θ_1 and θ_2 fixed as indicated. Since the Golub–Mayers preconditioner is independent of possibly highly varying coefficients, we also used a scaled Golub–Mayers preconditioner $M_{SGM} \equiv D^{1/2} M_{GM} D^{1/2}$, where D is the diagonal of the matrix L_{33}. As expected, $\kappa(M_{GM}^{-1}S)$ is uniformly bounded for varying h, whereas $\kappa(M_1^{-1}S)$ depends mildly on h (about $O(h^{-1/2})$, consistent with Theorem 6.4), even though the boundary conditions are different. The cross-over is about $n = 20$ or 30. For this case, M_{SGM} performs slightly worse than M_{GM}.

Based on studies of optimal preconditioners of a given sparsity pattern, Greenbaum and Rodrigue [25] had conjectured that the optimal symmetric positive definite tridiagonal preconditioner for the interface matrix S has condition number bounded

TABLE 2
Aspect ratio dependence for $h = 1/40$, $m_1 = 10$, m_2 given, $\theta_1 = 2$, and $\theta_2 = -2$.

m_2	$\kappa(M_1^{-1}S)$	Iters		$\kappa(M_{GM}^{-1}S)$	Iters		$\kappa(M_{SGM}^{-1}S)$	Iters
8	1.87	9		1.79	8		2.91	12
6	1.76	9		1.97	9		3.57	14
4	1.60	8		2.37	10		4.62	16
2	1.37	7		3.81	12		6.47	18

below by $O(h^{-1/2})$. Our numerical results indicate that the tridiagonal probe preconditioner performs as well asymptotically as the optimal tridiagonal preconditioner (which of course cannot be computed easily, in general) for S.

6.3. Dependence on aspect ratios. We shall next discuss the dependence of $\kappa(M_{CP}^{-1}S)$ on the aspect ratios l_1 and l_2. The result from Theorem 6.4 for the model problem indicates that the condition number of the probe preconditioned system does depend on the aspect ratios of the two subdomains:

$$\kappa(M_{CP}^{-1}S) \leq C\sqrt{\frac{l_1 l_2}{l_1 + l_2}}h^{-1/2} \leq C\sqrt{\min\{l_1, l_2\}}h^{-1/2}. \tag{26}$$

For instance, if $l_1 \leq l_2$, then $\kappa \leq C\sqrt{l_1}h^{-1/2}$, and this can grow if l_1 becomes large, and thus the performance of the probe preconditioner can deteriorate if both aspect ratios become large. However, we note that in this case S itself becomes close to singular. On the other hand, if $l_1 = O(1)$, and l_2 is allowed to vary independent of l_1, then $\kappa(M_{CP}^{-1}S)$ can be bounded independent of l_2.

In comparison, for the Dirichlet case, the Golub–Mayers preconditioned system has been shown (see Bjørstad and Widlund [3] and Chan [9]) to have a condition number

$$\kappa(M_{GM}^{-1}S) \leq C\left(1 + \frac{1}{l_1} + \frac{1}{l_2}\right).$$

This bound can become large if either of the aspect ratios l_1 or l_2 becomes small. However, the performance is good when both the aspect ratios are $O(1)$ or larger.

Table 2 illustrates the varying performance of M_{GM}, M_{SGM}, and M_1 with respect to aspect ratios of the subdomains, for test problem (25) on a unit square with a $(10+m_2)\times 40$ grid, partitioned into two subdomains of size 10×40 and $m_2 \times 40$, with $\theta_1 = 2$ and $\theta_2 = -2$ and using Dirichlet boundary conditions. Note that $\kappa(M_1^{-1}S)$ appears to decrease like $O(\sqrt{m_2})$ as $m_2 \to 0$, as predicted by (26), whereas $\kappa(M_{GM}^{-1}S)$ deteriorates mildly. Again, M_{SGM} performed slightly worse than M_{GM} for this problem.

Unlike the theoretical bounds presented in Theorem 6.4 for the model periodic case, which showed that the condition number of the probe preconditioned system grows as $\sqrt{\min\{l_1, l_2\}}$ for fixed h, the numerical results for the *Dirichlet* case indicate that the condition number of $M_1^{-1}S$ are bounded independent of l_1 or l_2, and for small l behaves like $O(\sqrt{l})$. This will be illustrated in Fig. 2, in §6.4.

6.4. Dependence on scalings of the coefficients. In this subsection, we focus on the performance of the preconditioners for various scalings of the coefficients. Here, some of the results are not restricted to the model problem. First, we consider the operator L of problem (1) with coefficients $a(x, y)$ and $b(x, y)$. As before, Ω is

partitioned into two subdomains Ω_1 and Ω_2. We note that the Schur complement S can be written as

$$S = S^{(1)} + S^{(2)} = (L_{33}^{(1)} - L_{13}^T L_{11}^{-1} L_{13}) + (L_{33}^{(2)} - L_{23}^T L_{22}^{-1} L_{23}),$$

where $L_{33}^{(i)}$ denotes the contribution to L_{33} from subdomain Ω_i, and $L_{33} = L_{33}^{(1)} + L_{33}^{(2)}$.

The coefficients of the original operator L are modified to obtain a scaled operator $L(\rho)$ as follows: $a(x,y)$ and $b(x,y)$ are multiplied by a positive constant scaling ρ in subdomain Ω_1, thereby making the coefficients possibly discontinuous across the interface Γ. The Schur complement for the scaled operator $L(\rho)$ will be denoted by $S(\rho)$, and it is easily seen that: $S(\rho) = \rho S^{(1)} + S^{(2)}$. If M denotes any preconditioner for $S = S(1)$, then the following theorem gives an upper bound for the condition number of the preconditioned system $M^{-1}S(\rho)$.

THEOREM 6.5. *The condition number of the preconditioned system* $M^{-1}S(\rho)$ *is bounded by* $\max\{\kappa(M^{-1}S^{(1)}), \kappa(M^{-1}S^{(2)})\}$.

Proof. Let $\lambda_i^{\min}$ and $\lambda_i^{\max}$ denote the lower and upper bounds for the following Rayleigh quotients:

$$\lambda_i^{\min} \leq \frac{x^T S^{(i)} x}{x^T M x} \leq \lambda_i^{\max}.$$

From this it follows that

$$\rho\lambda_1^{\min} + \lambda_2^{\min} \leq \frac{x^T S(\rho) x}{x^T M x} \leq \rho\lambda_1^{\max} + \lambda_2^{\max}.$$

Thus the condition number of the preconditioned system is bounded by

$$(\rho\lambda_1^{\max} + \lambda_2^{\max})/(\rho\lambda_1^{\min} + \lambda_2^{\min}),$$

which is easily shown to be a monotone function of ρ with the asymptotes given by $\lambda_1^{\max}/\lambda_1^{\min}$ and $\lambda_2^{\max}/\lambda_2^{\min}$. □

This theorem indicates that the scaling ρ could possibly affect the conditioning of $M^{-1}S(\rho)$ adversely only if $\kappa(M^{-1}S^{(1)})$ and $\kappa(M^{-1}S^{(2)})$ are significantly different. The upper bounds on the worst possible conditioning, however, depends only on M, $S^{(1)}$, and $S^{(2)}$ and these are independent of ρ. Though in the case of two subdomains, a simple scaling of the interface preconditioner will not affect the preconditioning in the case of more than two subdomains, however, proper scaling of the preconditioner on each of the edges constituting the interface is required for efficient preconditioning (see for instance [4]). This can also be done by suitable use of probing; see [14].

Applying Theorem 6.5 to the case of the scaled version of a model Dirichlet problem preconditioned by the Golub–Mayers preconditioner, we obtain the following corollary.

COROLLARY 6.6. *If* $M = M_{GM}$ *is used to precondition* $S(\rho)$, *then* $\kappa(M_{GM}^{-1}S(\rho))$ *can vary between* $O(1 + \frac{1}{l_1})$ *and* $O(1 + \frac{1}{l_2})$, *depending on* ρ.

The preceding case corresponded to preconditioners that did not adapt to the scale of each term in the Schur complement $S(\rho)$. In case the preconditioner M adapts to the scaling ρ, as in the case of probe preconditioners that are linearly dependent on the matrices they approximate, then we obtain different upper bounds for the preconditioned system, as given in the following theorem.

THEOREM 6.7. *If the preconditioner for* $S(\rho) \equiv \rho S^{(1)} + S^{(2)}$ *is of the form:* $M(\rho) = \rho M^{(1)} + M^{(2)}$, *and if*

$$\lambda_{\min}^i \leq \frac{x^T S^{(i)} x}{x^T M^{(i)} x} \leq \lambda_{\max}^i \quad \textit{for } i = 1, 2,$$

then,

$$\kappa(M(\rho)^{-1}S(\rho)) \leq \frac{\max\{\lambda^1_{\max}, \lambda^2_{\max}\}}{\min\{\lambda^1_{\min}, \lambda^2_{\min}\}}.$$

Proof. The proof follows trivially from the assumptions. □

Thus, if the bounds for the subdomain problems are independent of the aspect ratios, then the scaled version will also be independent of the aspect ratios. For the scaled version of the model operator L of (15), with $S(\rho) = \rho S^{(1)} + S^{(2)}$, we easily obtain that $M_{CP}(\rho) = \rho M^{(1)} + M^{(2)}$, where $S^{(i)} = F\ \mathrm{diag}(\lambda^{(i)}_j)\ F^{-1}$ and

$$\lambda^{(i)}_j = \left(\frac{1+\gamma(\sigma_j)^{l_i/h}}{1-\gamma(\sigma_j)^{l_i/h}}\right)\sqrt{\sigma_j + \frac{\sigma_j^2}{4}} \quad \text{and} \quad \gamma(\sigma_j) \equiv \frac{1 + \frac{\sigma_j}{2} - \sqrt{\sigma_j + \frac{\sigma_j^2}{4}}}{1 + \frac{\sigma_j}{2} + \sqrt{\sigma_j + \frac{\sigma_j^2}{4}}},$$

with $\lambda^i_0 = h/l_i$, and where

$$M^{(i)} = \mathrm{F}\ \mathrm{diag}\left(\frac{h}{l_i} + \beta_i\sigma_j\right)F^{-1}, \quad \text{for } j = 0, \cdots, n,$$

where $1 \geq \beta_i \geq \frac{1}{2}$. In this case, we obtain the following bounds for the condition number of the preconditioned system $\kappa(M_{CP}(\rho)^{-1}S(\rho))$.

COROLLARY 6.8. *If $M_{CP}(\rho) =$ circulant-*PROBE *$(S(\rho), 1)$ is used to precondition $S(\rho)$ for the scaled version of the model problem, then*

$$\kappa(M^{-1}_{CP}(\rho)S(\rho)) \leq C\sqrt{\max\{l_1, l_2\}}h^{-1/2},$$

for a constant C independent of ρ, h, and the aspect ratios l_1 and l_2.

Proof. By using linearity of the probing procedure, it is easy to verify that $(M^{(i)}_{CP})^{-1}S^{(i)}$ is the same as the preconditioned system $M^{-1}_{CP}S$ when both subdomains have the same aspect ratio l_i. Thus, it follows that:

$$C_1 \leq \lambda((M^{(i)})^{-1}S^{(i)}) \leq C\sqrt{l_i}h^{-1/2},$$

for $i = 1, 2$, where $\lambda(.)$ denotes the eigenvalues of the matrix argument. Applying Theorem 6.7, the desired result follows. □

This theoretical result indicates that the probe preconditioner in the model problem can be sensitive to scalings of the coefficients only if the aspect ratios of the two subdomains are significantly different, in which case the condition number can vary from $O(\sqrt{\min\{l_1, l_2\}}h^{-1/2})$ to $O(\sqrt{\max\{l_1, l_2\}}h^{-1/2})$. However, in the case that $\max\{l_1, l_2\}$ is large, then $\kappa(S(\rho)) = O(\max\{l_1, l_2\})$, which is also large. If both l_1 and l_2 are $O(1)$, then the scalings do not affect the convergence of the preconditioner.

In the Dirichlet case, however, numerical results seem to indicate that the probe preconditioned system performs better than as suggested in Corollary 6.8. In Fig. 2, we illustrate the results for the model scaled Poisson problem with Dirichlet boundary conditions with varying l_1, l_2, and ρ. The results indicate that the condition number of the probe preconditioned system can be bounded independently of l_1, l_2, and ρ. Based on Fig. 2, we conjecture that the condition number of the probe preconditioned system satisfies:

$$\kappa(M_1^{-1}S) \leq C\sqrt{\min\{l_1, l_2, 1\}}h^{-1/2},$$

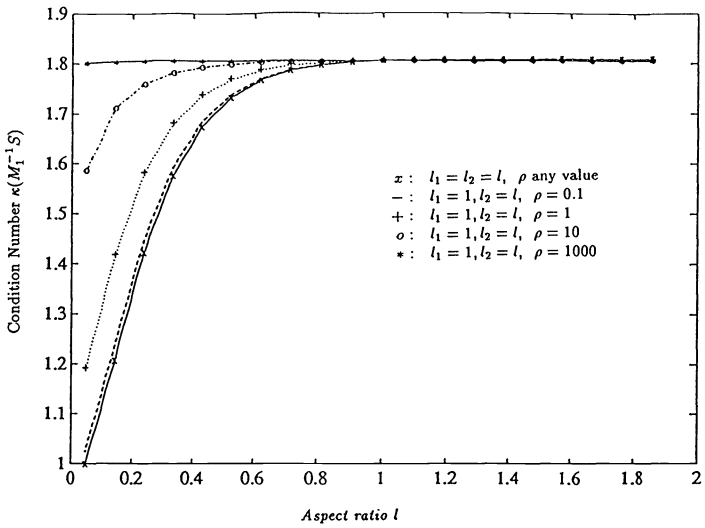


FIG. 2. *Probing for model Dirichlet problem with varying l_1, l_2, and ρ, and $n = 20$.*

TABLE 3
Dependence on coefficients for $\theta_1 = \theta_2$, $n = 20$, and $l_1 = l_2 = 1/2$.

θ_1	$\kappa(M_1^{-1}S)$	Iters		$\kappa(M_{GM}^{-1}S)$	Iters		$\kappa(M_{SGM}^{-1}S)$	Iters
0	1.68	7		1.09	3		1.09	3
2	1.67	8		2.48	12		1.11	4
4	1.66	8		6.17	17		1.18	4
6	1.63	8		15.37	21		1.28	5

for the Dirichlet case. This is similar to the bound obtained for the model periodic case in Theorem 6.4, except that $\min\{l_1, l_2\}$ is replaced by $\min\{l_1, l_2, 1\}$, which is uniformly bounded for large l_1 and l_2.

We now present numerical results on the rate of convergence of the probe preconditioned system and both the regular and scaled version of the Golub–Mayers preconditioned system, in the case of continuous, but highly varying coefficients. The tests were carried out on problem (25) with *Dirichlet* boundary conditions, with the parameters as shown. The results are presented in Table 3. The probing preconditioners seem to adapt well to such variations in the coefficients of L, while the performance of other preconditioners that are independent of the coefficients, like M_{GM}, deteriorate. However, unlike the results in Tables 1 and 2, the scaled version M_{SGM} improves over the performance of M_{GM} significantly. This may be due to the isotropy of the coefficients in Table 3. More tests seem to be needed to study the effect of scaling on optimal preconditioners such as M_{GM}.

7. Summary. Unlike various interface preconditioners in domain decomposition, the probing preconditioners are constructed as algebraic approximations to the interface operator. They have the disadvantage of being nonspectrally equivalent with respect to mesh size variation. However, since the techniques are algebraic in

nature, they can and have been applied to construct preconditioners to more general differential operators for which optimal preconditioners are not known.

We have shown that under certain conditions that are often valid in applications, the probing technique leads to nonsingular approximations. In addition, the preconditioners are linearly dependent on the matrices they approximate and preserve diagonal dominance. However, not all the probing techniques preserve symmetry of the matrices they approximate, and symmetric positive definiteness is generally not preserved.

For a model elliptic problem we have shown that the probing technique has some desirable properties: it reduces the condition number of the interface operator from $O(h^{-1})$ to $O(h^{-1/2})$. Moreover, the probing technique is also fairly robust with respect to aspect ratios and coefficient variations, though there could be some mild dependence for large aspect ratios. However, for the Dirichlet problem, our numerical results indicate that the rates are bounded independent of the aspect ratios l_1, l_2 and the scaling ρ, but retains the $O(h^{-1/2})$ dependence.

In summary, if h is not very small, and the aspect ratios and coefficients are highly varying, then probing can provide a competitive alternative to other available interface preconditioners.

Acknowledgments. The authors would like to thank Professor Olof Widlund for his helpful discussions on various topics in this paper. They would also like to thank the referees for their helpful suggestions.

REFERENCES

[1] V. I. AGOSHKOV, *Poincaré–Steklov operators and domain decomposition methods in finite dimensional spaces*, in First International Symposium on Domain Decomposition Methods for Partial Differential Equations, R. Glowinski, G. H. Golub, G. A. Meurant, and J. Périaux, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1988, pp. 73–112.

[2] O. AXELSSON AND B. POLMAN, *Block preconditioning and domain decomposition methods*, I, Tech. Report 8735, Catholic University, Nijmegen, the Netherlands, December 1987.

[3] P. E. BJØRSTAD AND O. B. WIDLUND, *Iterative methods for the solution of elliptic problems on regions partitioned into substructures*, SIAM J. Numer. Anal., 23 (1986), pp. 1093–1120.

[4] J. H. BRAMBLE, J. E. PASCIAK, AND A. H. SCHATZ, *The construction of preconditioners for elliptic problems by substructuring*, I, Math. Comp., 47 (1986), pp. 103–134.

[5] ———, *An iterative method for elliptic problems on regions partitioned into substructures*, Math. Comp., 46 (1986), pp. 361–369.

[6] ———, *The construction of preconditioners for elliptic problems by substructuring*, II, Math. Comp., 49 (1987), pp. 1–16.

[7] ———, *The construction of preconditioners for elliptic problems by substructuring*, III, Math. Comp., 51 (1988), pp. 415–430.

[8] ———, *The construction of preconditioners for elliptic problems by substructuring*, IV, Math. Comp., 53 (1989), pp. 1–24.

[9] T. F. CHAN, *Analysis of preconditioners for domain decomposition*, SIAM J. Numer. Anal., 24 (1987), pp. 382–390.

[10] ———, *Boundary probe preconditioners for fourth order elliptic problems*, in Domain Decomposition Methods, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1989, pp. 160–167.

[11] T. F. CHAN AND D. GOOVAERTS, *A note on the efficiency of domain decomposed incomplete factorizations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 794–803.

[12] T. F. CHAN AND T. Y. HOU, *Domain decomposition interface preconditioners for general 2nd order elliptic problems*, Tech. Report CAM 88-16, Department of Mathematics, University of California, Los Angeles, 1990.

[13] T. F. CHAN AND D. F. KEYES, *Interface preconditioning for domain-decomposed convection-diffusion operators*, Tech. Report CAM 89-28, Department of Mathematics, University

of California, Los Angeles, 1989, pp. 245–262; Proceedings of the Third International Symposium on Domain Decomposition Methods, Houston, Texas, April 1989.

[14] T. F. CHAN AND T. P. MATHEW, *An application of the probing technique to the vertex space method in domain decomposition*, Tech. Report CAM 90-22, Department of Mathematics, University of California, Los Angeles, 1990; To appear in Proceedings of 5th Domain Decomposition Conference, Moscow, April 1990.

[15] T. F. CHAN AND D. C. RESASCO, *A survey of preconditioners for domain decomposition*, Tech. Report /DCS/RR-414, Department of Computer Science, Yale University, New Haven, CT, 1985.

[16] A. R. CURTIS, M. J. POWELL, AND J. K. REID, *On the estimation of sparse Jacobian matrices*, J. Inst. Math. Appl., 13 (1974), pp. 117–120.

[17] A. Q. D. FUNARO AND P. ZANOLLI, *An iterative procedure with interface relaxation for domain decomposition methods*, SIAM J. Numer. Anal., 25 (1988), pp. 1213 –1236.

[18] J. DONATO, *Eigendecompositions of the capacitance matrix for Poisson's equation on a strip*, Course M285J Project, Department of Mathematics, University of California, Los Angeles, 1988.

[19] M. DRYJA, *A capacitance matrix method for Dirichlet problem on polygon region*, Numer. Math., 39 (1982), pp. 51–64.

[20] M. DRYJA AND O. B. WIDLUND, *An additive variant of the Schwarz alternating method for the case of many subregions*, Tech. Report 339, Ultracomputer Note 131, Department of Computer Science, Courant Institute, New York, 1987.

[21] ———, *Some domain decomposition algorithms for elliptic problems*, in Proceedings of the Conference on Iterative Methods for Large Linear Systems held in Austin, Texas, October 1988, to celebrate the Sixty-fifth Birthday of David M. Young, Jr., Academic Press, Orlando, Florida, 1989.

[22] S. C. EISENSTAT, Personal communication, 1985.

[23] R. GLOWINSKI AND O. PIRONNEAU, *Numerical methods for the first biharmonic equation and for the two-dimensional stokes problem*, SIAM Review, 21 (1979), pp. 167–212.

[24] G. GOLUB AND D. MAYERS, *The use of preconditioning over irregular regions*, in Computing Methods in Applied Sciences and Engineering, VI, R. Glowinski and J. L. Lions, eds., North-Holland, Amsterdam, New York, Oxford, 1984, pp. 3–14.

[25] A. GREENBAUM AND G. RODRIGUE, *Optimal preconditioners of a given sparsity pattern*, BIT, 29 (1989), pp. 610–634.

[26] E. ISAACSON AND H. B. KELLER, *Analysis of Numerical Methods*, John Wiley and Sons, New York, 1966.

[27] D. E. KEYES, *Domain decomposition methods for the parallel computation of reacting flows*, Comput. Phys. Comm., 53 (1989), pp. s181–s200.

[28] D. E. KEYES AND W. D. GROPP, *A comparison of domain decomposition techniques for elliptic partial differential equations and their parallel implementation*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. s166 – s202.

[29] ———, *Domain decomposition techniques for the parallel solution of nonsymmetric systems of elliptic bvps*, in Domain Decomposition Methods, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1989.

[30] H. LI, *Toeplitz preconditioner in domain decomposition methods*, M285J Project Report, University of California, Los Angeles, 1988.

[31] A. QUARTERONI AND A. VALLI, *Theory and applications of Steklov–Poincaré operators for boundary-value problems: The heterogeneous operator case*, in Proceedings of 4th International Conference on Domain Decomposition Methods, Moscow, T. Chan, R. Glowinski, J. Périaux, and O. Widlund, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1990.

[32] B. F. SMITH, *An optimal domain decomposition preconditioner for the finite element solution of linear elasticity problems*, Tech. Report 482, Department of Computer Science, Courant Institute, New York, 1989. SIAM J. Sci. Statist. Comput., 13 (1992), to appear.

[33] W. TSUI, *Domain Decompostion of Biharmonic and Navier–Stokes Equations*, Ph.D. thesis, Department of Mathematics, University of California, Los Angeles, 1991.

[34] R. S. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 239–247, January 1992

018

REFINED INTERLACING PROPERTIES*

R. O. HILL, JR.† AND B. N. PARLETT‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Interlacing results for eigenvalues due to Cauchy, Golub, and Kahan are extended and related to the last component of eigenvectors.

Key words. eigenvalues, eigenvectors, interlacing

AMS(MOS) subject classifications. primary, 15A18; secondary, 15A42

1. Background and definitions. As far back as 1821, in the Cours d'Analyze of the École Polytechnique, Cauchy published a proof of the following remarkable result, which we now call Cauchy's Interlace Theorem. If any row, together with its matching column, is deleted from a real symmetric matrix, then the eigenvalues of the new matrix interlace the eigenvalues of the old one. In the presence of more information, much more can be said about the interlacing of eigenvalues and the relationship between the spacing and the corresponding eigenvectors.

An example of such results can be found in a 1972 paper by Golub [2], which discusses aspects of the Lanczos algorithm. Golub, seeking bounds for eigenvalues, constructs a special rank-one update H to an $n \times n$ symmetric, tridiagonal matrix T_n. Letting $\tilde{T}_k$ be the leading principal $(k+1) \times (k+1)$ submatrix of H, Golub shows that each interval determined by the eigenvalues of $\tilde{T}_k$ contains an eigenvalue of T_n. Our Theorem 1 extends this by replacing H with T_n itself.

In order to pursue these lines of investigation, some notational conventions will prove helpful. Let A_k denote the leading principal $k \times k$ submatrix of a real symmetric matrix $A_n = A$. The ordered eigenvalues of A_k are denoted by

$$\lambda_1^{(k)} \leq \lambda_2^{(k)} \leq \cdots \leq \lambda_k^{(k)}.$$

For each k we assume we have k orthonormal eigenvectors $\mathbf{z}_i^{(k)}$ associated with the $\lambda_i^{(k)}$ (where, of course, we are using the usual inner product on $\mathbb{R}^n$). When necessary, the jth entry of $\mathbf{z}_i^{(k)}$ is denoted by $\mathbf{z}_i^{(k)}(j)$. Frequently, we shall be concerned only with the magnitude of the last entry $\mathbf{z}_i^{(k)}(k)$ and, for simplicity, we shall denote this by the last letter of the Greek alphabet

$$\omega_{ik} = |\mathbf{z}_i^{(k)}(k)|.$$

Denote by $p_k(x) = \det(xI - A_k)$ the characteristic polynomial of A_k and let $p(x) = p_n(x)$.

* Received by the editors January 14, 1991; accepted for publication (in revised form) July 31, 1991.

† Department of Mathematics, Michigan State University, East Lansing, Michigan 48824 (16705roh@msu. bitnet).

‡ Mathematics Department and Computer Science Division, University of California, Berkeley, California 94720 (parlett@math.berkeley.edu). This author gratefully acknowledges partial support from Office of Naval Research contract N00014-90-J-1372.

2. We now consider properties of the whole triangular set $\{\lambda_i^{(k)}\}$

$$\begin{array}{ccccccccc} & & & & \lambda_1^{(1)} & & & & \\ & & & \lambda_1^{(2)} & & \lambda_2^{(2)} & & & \\ & & \lambda_1^{(3)} & & \lambda_2^{(3)} & & \lambda_3^{(3)} & & \\ & \lambda_1^{(4)} & & \lambda_2^{(4)} & & \lambda_3^{(4)} & & \lambda_4^{(4)} & \\ \cdot & & \cdot & & \cdot & & \cdot & & \cdot \end{array}$$

By Cauchy's Interlace Theorem, we know $\lambda_i^{(k)} \leq \lambda_i^{(k-1)} \leq \lambda_{i+1}^{(k)}$, as indicated by the spacing in the table. The first question likely to arise after looking at this table is:

$$\text{Does } [\lambda_1^{(2)}, \lambda_2^{(2)}] \text{ contain an eigenvalue of } A_4?$$

An affirmative answer does not follow from Cauchy's theorem, and, in fact, the answer is *no* in general. However, the answer is *yes* when A is tridiagonal. Furthermore, the open interval $(\lambda_1^{(2)}, \lambda_2^{(2)})$ must contain an eigenvalue of A_4 if A_4 is an unreduced tridiagonal matrix. (Recall, a tridiagonal matrix

$$T_k = \begin{bmatrix} a_1 & b_1 & & & \\ b_1 & a_2 & & & \\ & & \ddots & & \\ & & & a_{k-1} & b_{k-1} \\ & & & b_{k-1} & a_k \end{bmatrix}$$

is called *unreduced* if all the b_i's are nonzero.) Furthermore, this little-known result can be proved by elementary means eminently suitable for introductory courses in linear algebra. We give brief, introductory proofs now, because we shall establish a more general result by other means later. Here is the general statement, which is simplified by letting $\lambda_0^{(k)} = -\infty$ and $\lambda_{k+1}^{(k)} = \infty$.

THEOREM 1. *Let T be an $n \times n$ real symmetric unreduced tridiagonal matrix with T_k its leading principal $k \times k$ submatrix. Then, for $i = 0, 1, \cdots, k$, each open interval $(\lambda_i^{(k)}, \lambda_{i+1}^{(k)})$ contains a distinct eigenvalue of T.*

Note. The word "distinct," although unnecessary since the intervals are disjoint, is included for emphasis.

Note. Theorem 1 is true even if T is reduced, provided that we replace the open intervals by closed intervals. This follows by using just a little care after breaking T up into tridiagonal blocks at each zero b_i. When $n = k + 1$, this is just Cauchy's theorem; we shall need later the fact that Cauchy's theorem yields strict inequalities for unreduced tridiagonal matrices.

Note. Suppose A is an $n \times n$ symmetric matrix and T_k is a partial tridiagonalization of A (obtained, perhaps, by the Lanczos algorithm). Then each interval determined by the eigenvalues of T_k contains an eigenvalue of A, since if we complete the tridiagonalization we obtain a T_n similar to A.

Before proving Theorem 1 we consider the special case $n = k+2$ given in Theorem 2, below. For this case, the proof is very apparent, it illustrates the general case, and more detailed information is obtained that not only has other interesting implications, but also hints at the results to follow in the next section. In the statement of Theorem 2 we use the convention $(a, b) = \{a\}$ if $a = b$. Also, from before, a_{k+2} is the lowest

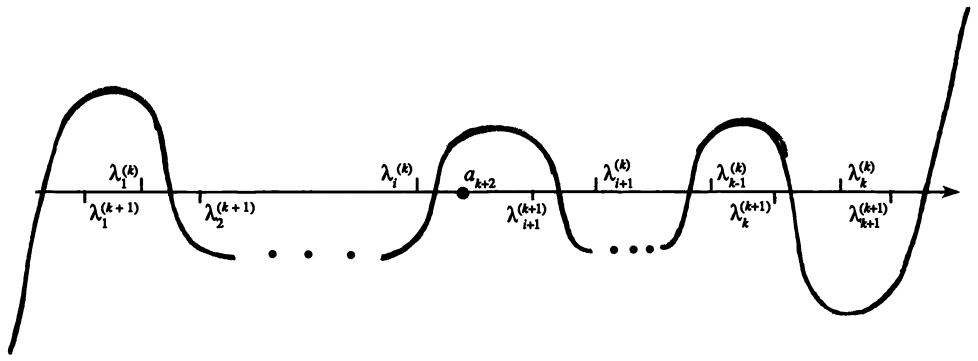


FIG. 1. *A possible graph of* $y = p_{k+2}(x)$.

diagonal entry of T_{k+2}. Because of our convention that $\lambda_0^{(k)} = -\infty$, $\lambda_{k+1}^{(k)} = +\infty$, a_{k+2} must lie in some subinterval $[\lambda_i^{(k)}, \lambda_{i+1}^{(k)}]$.

THEOREM 2. *Let* T_{k+2} *be unreduced and assume* a_{k+2} *lies in the subinterval* $[\lambda_i^{(k)}, \lambda_{i+1}^{(k)}]$.

(a) *If* $\lambda_i^{(k)} \le a_{k+2} \le \lambda_{i+1}^{(k+1)}$, *then each of the* $k+2$ *intervals*

$$(-\infty, \lambda_1^{(k+1)}), (\lambda_1^{(k)}, \lambda_2^{(k+1)}), \cdots, (\lambda_i^{(k)}, a_{k+2}), (\lambda_{i+1}^{(k+1)}, \lambda_{i+1}^{(k)}), \cdots, (\lambda_k^{(k+1)}, \lambda_k^{(k)}), (\lambda_{k+1}^{(k+1)}, \infty)$$

contains one of the $k+2$ *eigenvalues of* T_{k+2}, *and* $(a_{k+2}, \lambda_{i+1}^{(k+1)})$ *contains none.*

(b) *If* $\lambda_{i+1}^{(k+1)} \le a_{k+2} \le \lambda_{i+1}^{(k)}$, *then* (a) *holds when the middle two intervals are replaced by* $(\lambda_i^{(k)}, \lambda_{i+1}^{(k+1)}), (a_{k+2}, \lambda_{i+1}^{(k)})$, *and* $(\lambda_{i+1}^{(k+1)}, a_{k+2})$ *contains none.*

These relationships are illustrated in Fig. 1.

Proof of Theorem 2. Assume that $a_{k+2} \le \lambda_{i+1}^{(k+1)}$ (and the proof of the case $a_{k+2} \ge \lambda_{i+1}^{(k+1)}$ is similar). By Cauchy's interlace theorem, we know p_{k+2} has zeros in the intervals $(-\infty, \lambda_1^{(k+1)})$ and $(\lambda_{k+1}^{(k+1)}, \infty)$. We shall prove one of the three remaining cases and leave the others as an exercise.

Pick an arbitrary j, $1 \le j < i$ and consider $(\lambda_j^{(k)}, \lambda_{j+1}^{(k+1)})$. The usual expansion of $\det(xI - T_{k+2})$ along its bottom row gives the well-known recurrence relationship

$$p_{k+2}(x) = (x - a_{k+2})p_{k+1}(x) - b_{k+1}^2 p_k(x). \tag{1}$$

We know $p_k(\lambda_j^{(k)}) = 0$ and $p_{k+1}(\lambda_{j+1}^{(k+1)}) = 0$. Using interlacing and the factorizations

$$\begin{aligned} p_k(x) &= (x - \lambda_1^{(k)}) \cdots (x - \lambda_j^{(k)})(x - \lambda_{j+1}^{(k)}) \cdots (x - \lambda_k^{(k)}), \\ p_{k+1}(x) &= (x - \lambda_1^{(k+1)}) \cdots (x - \lambda_j^{(k+1)})(x - \lambda_{j+1}^{(k+1)}) \cdots (x - \lambda_{k+1}^{(k+1)}), \end{aligned}$$

we see $\operatorname{sgn} p_k(\lambda_{j+1}^{(k+1)}) = (-1)^{k-j}$ and $\operatorname{sgn} p_{k+1}(\lambda_j^{(k)}) = (-1)^{k-j+1}$. Since $\lambda_j^{(k)} < \lambda_i^{(k)} \le a_{k+2}$, $\lambda_j^{(k)} - a_{k+2} < 0$. Putting this all together we have

$$\begin{aligned} \operatorname{sgn} p_{k+2}(\lambda_{j+1}^{(k+1)}) &= \operatorname{sgn}[0 - b_{k+1}^2 p_k(\lambda_{j+1}^{(k+1)})] = -(-1)^{k-j} = (-1)^{k-j+1}, \\ \operatorname{sgn} p_{k+2}(\lambda_j^{(k)}) &= \operatorname{sgn}[(\lambda_j^{(k)} - a_{k+2})p_{k+1}(\lambda_j^{(k)}) - 0] = -(-1)^{k-j+1} = (-1)^{k-j}. \end{aligned}$$

Thus, the polynomial p_{k+2} has a zero in $(\lambda_j^{(k)}, \lambda_{j+1}^{(k+1)})$. □

Before proving Theorem 1, there are two comments to make. First, Theorem 2 shows that not only is $\lambda_{j+1}^{(k+2)}$ in the interval $(\lambda_j^{(k+1)}, \lambda_{j+1}^{(k+1)})$, it is in whichever of the subintervals $(\lambda_j^{(k+1)}, \lambda_j^{(k)})$ or $(\lambda_j^{(k)}, \lambda_{j+1}^{(k+1)})$ is further from a_{k+2}. Hence, $\lambda_{j+1}^{(k+2)}$ will tend to be closer to whichever endpoint of $(\lambda_j^{(k+1)}, \lambda_{j+1}^{(k+1)})$ is further from a_{k+2}; how close will be discussed further in §3. Second, Theorem 2 shows that each interval $(\lambda_j^{(k)}, \lambda_{j+1}^{(k)})$ contains a unique eigenvalue of T_{k+2} except for the interval that contains a_{k+2}. From this, if we knew suitable bounds for all the a_j, $j \geq k+2$, then we would know that intervals $(\lambda_j^{(k)}, \lambda_{j+1}^{(k)})$ outside those bounds still contained unique eigenvalues of T.

Proof of Theorem 1. Let $n \geq k+3$. Let $T_m^{\#}$ be the trailing principal $(n-m+1) \times (n-m+1)$ submatrix of T, which is given by

$$T_m^{\#} = \begin{bmatrix} a_m & b_m & & & & \\ b_m & a_{m+1} & & & & \\ & & \ddots & & & \\ & & & a_{n-1} & b_{n-1} \\ & & & b_{n-1} & a_n \end{bmatrix}$$

and let $p_m^{\#}$ be its characteristic polynomial. Then

$$T = \begin{bmatrix} T_k & \vdots & & & & & \\ \cdots\cdots\cdots & & b_k & & & & \\ & b_k & a_{k+1} & b_{k+1} & & & \\ & & b_{k+1} & a_{k+2} & b_{k+2} & & \\ & & & b_{k+2} & a_{k+3} & b_{k+3} & \cdots\cdots\cdots \\ & & & & b_{k+3} & & \\ & & & & & \vdots & T_{k+4}^{\#} \end{bmatrix}.$$

Now expand $p(x) = \det(xI - T)$ along the $(k+2)$nd row to obtain

$$\begin{aligned} p(x) &= -(-b_{k+1}) \det \begin{bmatrix} xI - T_k & \vdots & & & \\ \cdots\cdots\cdots & & 0 & & \\ & -b_k & -b_{k+1} & 0 & \\ & & -b_{k+2} & & \cdots\cdots\cdots \\ & & & \vdots & xI - T_{k+3}^{\#} \end{bmatrix} \\ &\quad + (x - a_{k+2}) p_{k+1}(x) p_{k+3}^{\#}(x) \\ &\quad - (-b_{k+2}) \det \begin{bmatrix} xI - T_{k+1} & \vdots & & & \\ \cdots\cdots\cdots & & -b_{k+1} & & \\ & 0 & -b_{k+2} & -b_{k+3} & \\ & & 0 & & \cdots\cdots\cdots \\ & & & \vdots & xI - T_{k+4}^{\#} \end{bmatrix} \\ &= -b_{k+1}^2 p_k(x) p_{k+3}^{\#}(x) + (x - a_{k+2}) p_{k+1}(x) p_{k+3}^{\#}(x) - b_{k+2}^2 p_{k+1}(x) p_{k+4}^{\#}(x) \\ &= -b_{k+1}^2 p_k(x) p_{k+3}^{\#}(x) + [(x - a_{k+2}) p_{k+3}^{\#}(x) - b_{k+2}^2 p_{k+4}^{\#}(x)] p_{k+1}(x) \\ &= -b_{k+1}^2 p_k(x) p_{k+3}^{\#}(x) + p_{k+2}^{\#}(x) p_{k+1}(x). \end{aligned}$$

The last step follows by expanding $p^{\#}_{k+2}(x) = \det(xI - T^{\#}_{k+2})$ along the top row. Thus, the analogue of (1) is

$$p(x) = p^{\#}_{k+2}(x)p_{k+1}(x) - b^2_{k+1}p^{\#}_{k+3}(x)p_k(x). \tag{2}$$

The proof now proceeds in a manner very similar to the proof of Theorem 2. In each interval $(\lambda^{(k)}_j, \lambda^{(k)}_{j+1})$ there is at least one zero of p in either $(\lambda^{(k)}_j, \lambda^{(k+1)}_{j+1})$ or $(\lambda^{(k+1)}_{j+1}, \lambda^{(k)}_{j+1})$, depending on the relative position of $(\lambda^{(k)}_j, \lambda^{(k)}_{j+1})$ to the zeros of $p^{\#}_{k+2}$ and $p^{\#}_{k+3}$. In this, you must use the fact that the zeros of $p^{\#}_{k+2}$ and $p^{\#}_{k+3}$ interlace each other also (again by Cauchy's Interlace Theorem). The details are left as an exercise. □

3. If all that we wanted to show was that each closed interval $[\lambda^{(k)}_j, \lambda^{(k)}_{j+1}]$ contains a distinct eigenvalue of A, then the tridiagonal assumption in Theorem 1 is not necessary, as we shall show in §4. However, it is necessary for our next result, which concerns what we call *crowded interlacing*, i.e., when the next eigenvalue along is very close to one of the eigenvalues between which it is interlaced,

$$\min\{|\lambda^{(k)}_i - \lambda^{(k-1)}_i|, |\lambda^{(k)}_{i+1} - \lambda^{(k-1)}_i|\} << |\lambda^{(k)}_{i+1} - \lambda^{(k)}_i|.$$

In the context of the Lanczos algorithm it is important to understand when an eigenvalue of T_{k+1} is almost on top of an eigenvalue of T_k. This phenomenon is controlled by the numbers ω_{ik} introduced earlier. This is surprising but not without precedent. In studies of the inverse eigenvalue problem it was discovered independently by most researchers that an unreduced tridiagonal T_k is completely determined (to within ± signs) by the eigenvalues $\{\lambda^{(n)}_i\}^n_1$ and the "weights" $\{\omega^2_{in}\}^n_1$. There are only $2n-1$ free parameters here, since $\Sigma\, \omega^2_{in} = 1$. Among physicists the parameters $\{\lambda^{(n)}_i, \omega_{in}\}$ are called *action-angle variables.*

The following result implies, among other things, that

$$\omega^2_{1n}(\lambda^{(n)}_2 - \lambda^{(n)}_1) < \lambda^{(n-1)}_1 - \lambda^{(n)}_1 < \omega^2_{1n}(\lambda^{(n)}_n - \lambda^{(n)}_1).$$

Consequently, the smaller ω_{1n} is, the closer $\lambda^{(n-1)}_1$ and $\lambda^{(n)}_1$ are.

THEOREM 3. *If T_n is an unreduced $n \times n$ tridiagonal matrix and $1 \le k \le n$, then*

$$\left.\begin{array}{c} \dfrac{\lambda^{(k-1)}_1 - \lambda^{(k)}_1}{\lambda^{(k)}_k - \lambda^{(k)}_1} \\ \dfrac{\lambda^{(k)}_i - \lambda^{(k-1)}_{i-1}}{\lambda^{(k)}_i - \lambda^{(k)}_1} \dfrac{\lambda^{(k-1)}_i - \lambda^{(k)}_i}{\lambda^{(k)}_k - \lambda^{(k)}_i} \\ \dfrac{\lambda^{(k)}_k - \lambda^{(k-1)}_{k-1}}{\lambda^{(k)}_k - \lambda^{(k)}_1} \end{array}\right\} < \omega^2_{ik} < \begin{cases} \dfrac{\lambda^{(k-1)}_1 - \lambda^{(k)}_1}{\lambda^{(k)}_2 - \lambda^{(k)}_1}, & i = 1 \\ \dfrac{\lambda^{(k)}_i - \lambda^{(k-1)}_{i-1}}{\lambda^{(k)}_i - \lambda^{(k)}_{i-1}} \dfrac{\lambda^{(k-1)}_i - \lambda^{(k)}_i}{\lambda^{(k)}_{i+1} - \lambda^{(k)}_i}, & i \neq 1,\ k \\ \dfrac{\lambda^{(k)}_k - \lambda^{(k-1)}_{k-1}}{\lambda^{(k)}_k - \lambda^{(k)}_{k-1}}, & i = k. \end{cases}$$

Proof. It is known that the tridiagonal T_k can be reconstructed from the $2k-1$ values $\{\lambda^{(k)}_i\}^k_1$, $\{\lambda^{(k-1)}_i\}^{k-1}_1$. The expression for ω_{ik} is remarkably simple. As before, let $p_k(x) = \det(xI - T_k)$ be the characteristic polynomial of T_k. Then

$$\omega^2_{ik} = -p_{k-1}(\lambda^{(k)}_i)/p'_k(\lambda^{(k)}_i).$$

A proof of this may be found in [4, p. 129]. It remains only to reorganize the expression above as a product of quotients:

$$\omega_{ik}^2 = \begin{cases} \displaystyle\prod_{j=1}^{k-1} \frac{\lambda_j^{(k-1)} - \lambda_1^{(k)}}{\lambda_{j+1}^{(k)} - \lambda_1^{(k)}}, & i = 1, \\ \displaystyle\prod_{j=1}^{i-1} \frac{\lambda_i^{(k)} - \lambda_j^{(k-1)}}{\lambda_i^{(k)} - \lambda_j^{(k)}} \prod_{j=i}^{k-1} \frac{\lambda_j^{(k-1)} - \lambda_i^{(k)}}{\lambda_{j+1}^{(k)} - \lambda_i^{(k)}}, & i \neq 1,\ k, \\ \displaystyle\prod_{j=1}^{k-1} \frac{\lambda_k^{(k)} - \lambda_j^{(k-1)}}{\lambda_k^{(k)} - \lambda_j^{(k)}}, & i = k. \end{cases}$$

Since T_k is unreduced, Cauchy's theorem yields $\lambda_j^{(k)} < \lambda_j^{(k-1)} < \lambda_{j+1}^{(k)}$, $j = 1, \cdots, k-1$, so each factor in the above products is positive and less than one. To obtain the second inequality in the theorem, simply discard all factors in each $\prod$ term except for the smallest one.

In order to obtain the first inequality in the theorem, the products given above must be rearranged as follows:

$$\prod_{j=1}^{k-1} \frac{\lambda_j^{(k-1)} - \lambda_1^{(k)}}{\lambda_{j+1}^{(k)} - \lambda_1^{(k)}} = \frac{\lambda_1^{(k-1)} - \lambda_1^{(k)}}{1} \left(\prod_{j=2}^{k-1} \frac{\lambda_j^{(k-1)} - \lambda_1^{(k)}}{\lambda_j^{(k)} - \lambda_1^{(k)}} \right) \frac{1}{\lambda_k^{(k)} - \lambda_1^{(k)}},$$

$$\prod_{j=1}^{i-1} \frac{\lambda_i^{(k)} - \lambda_j^{(k-1)}}{\lambda_i^{(k)} - \lambda_j^{(k)}} = \frac{1}{\lambda_i^{(k)} - \lambda_1^{(k)}} \left(\prod_{j=1}^{i-2} \frac{\lambda_i^{(k)} - \lambda_j^{(k-1)}}{\lambda_i^{(k)} - \lambda_{j+1}^{(k)}} \right) \frac{\lambda_i^{(k)} - \lambda_{i-1}^{(k)}}{1},$$

$$\prod_{j=1}^{k-1} \frac{\lambda_j^{(k-1)} - \lambda_i^{(k)}}{\lambda_{j+1}^{(k)} - \lambda_i^{(k)}} = \frac{\lambda_i^{(k-1)} - \lambda_i^{(k)}}{1} \left(\prod_{j=i+1}^{k-1} \frac{\lambda_j^{(k-1)} - \lambda_1^{(k)}}{\lambda_j^{(k)} - \lambda_i^{(k)}} \right) \frac{1}{\lambda_k^{(k)} - \lambda_i^{(k)}},$$

$$\prod_{j=1}^{k-1} \frac{\lambda_k^{(k)} - \lambda_j^{(k-1)}}{\lambda_k^{(k)} - \lambda_j^{(k)}} = \frac{1}{\lambda_k^{(k)} - \lambda_1^{(k)}} \left(\prod_{j=1}^{k-2} \frac{\lambda_k^{(k)} - \lambda_j^{(k-1)}}{\lambda_k^{(k)} - \lambda_{j+1}^{(k)}} \right) \frac{\lambda_k^{(k)} - \lambda_{k-1}^{(k+1)}}{1}.$$

Cauchy's theorem shows that every quotient in the four products in parentheses shown above exceeds one. Deleting them yields the first inequality, as claimed. □

Next we return to full symmetric matrices.

4. We now extend the results of Theorem 1 beyond the tridiagonal case. The eigenvalues of the $k \times k$ matrix A_k define $k+1$ intervals $(-\infty, \lambda_1^{(k)}], [\lambda_1^{(k)}, \lambda_2^{(k)}], \cdots, [\lambda_k^{(k)}, \infty)$, and we want to know when every principal supermatrix of A_k has at least one eigenvalue in each of these intervals. We do not demand that these intervals be distinct. (Note that A_k need not be tridiagonal, so it may have multiple eigenvalues.) Recall our earlier convention that $\lambda_0^{(k)} = -\infty$ and $\lambda_{k+1}^{(k)} = \infty$.

THEOREM 4. *Let*

$$A_n = \begin{bmatrix} A_k & C^t \\ C & U \end{bmatrix}$$

be a symmetric partition of A_n. *If* $\operatorname{rank}(C) = 1$, *then each interval* $[\lambda_i^{(k)}, \lambda_{i+1}^{(k)}]$, $i = 0, 1, \cdots, k$, *contains at least one eigenvalue of* A_n.

Proof. Let $C = \mathbf{w}\mathbf{v}^t$, $\mathbf{w} \in \mathbb{R}^{n-k}$, $\mathbf{v} \in \mathbb{R}^k$. The case of the external intervals follows from Cauchy's Interlace Theorem, so pick an $i = 1, \cdots, k-1$.

If $\lambda_i^{(k)} = \lambda_{i+1}^{(k)}$, find a unit eigenvector $\mathbf{z}$ of A_k associated with $\lambda_i^{(k)}$ and satisfying $\mathbf{v}^t\mathbf{z} = 0$. Now let $\mathbf{x} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \end{bmatrix}$ and it is trivial to verify that $\mathbf{x}$ is an eigenvector of A_n corresponding to $\lambda_i^{(k)}$.

Suppose $\lambda_i^{(k)} \neq \lambda_{i+1}^{(k)}$. Let $\mathbf{z} = \mathbf{z}_i^{(k)} \cos\psi + \mathbf{z}_{i+1}^{(k)} \sin\psi$, where ψ will be determined later, let $\mathbf{x} = \begin{bmatrix} \mathbf{z} \\ \mathbf{0} \end{bmatrix}$, and let $\gamma = \left(\lambda_i^{(k)} + \lambda_{i+1}^{(k)}\right)/2$. Then

$$(A - \gamma I)\mathbf{x} = \begin{bmatrix} (\lambda_i^{(k)} - \gamma)\mathbf{z}_i^{(k)} \cos\psi + (\lambda_{i+1}^{(k)} - \gamma)\mathbf{z}_{i+1}^{(k)} \sin\psi \\ \mathbf{w}\mathbf{v}^t(\mathbf{z}_i^{(k)} \cos\psi + \mathbf{z}_{i+1}^{(k)} \sin\psi) \end{bmatrix}.$$

Now take any ψ such that $\mathbf{v}^t(\mathbf{z}_i^{(k)} \cos\psi + \mathbf{z}_{i+1}^{(k)} \sin\psi) = 0$. For such a ψ,

$$\|(A - \gamma I)\mathbf{x}\|^2 = (\lambda_i^{(k)} - \gamma)^2 \cos^2\psi + (\lambda_{i+1}^{(k)} - \gamma)^2 \sin^2\psi = \left(\frac{\lambda_{i+1}^{(k)} - \lambda_i^{(k)}}{2}\right)^2.$$

It is a standard result (see [3, Thm. 4-5-1]) that if $\|\mathbf{x}\| = 1$ and $\|A\mathbf{x} - \gamma\mathbf{x}\| = \delta$, then there is an eigenvalue of A in $[\gamma - \delta, \gamma + \delta]$. If we apply this to the above, where $\delta = \left(\lambda_{i+1}^{(k)} - \lambda_i^{(k)}\right)/2$, we easily see that there is an eigenvalue of A in $[\gamma - \delta, \gamma + \delta] = [\lambda_i^{(k)}, \lambda_{i+1}^{(k)}]$, and we are done. □

In general it can be shown that if $\operatorname{rank}(C) = r$, then each union of r abutting subintervals defined by the $\lambda_i^{(k)}$'s holds at least one eigenvalue of A_n. As soon as r exceeds k, the result becomes vacuous.

It is worth pointing out that Theorems 1 and 3 can both be obtained as special cases of Lehmann's optimal intervals. Lehmann's results [3] were published in the 1960s in German and are complicated by the use of an additional parameter. He assumes that A_k and C are known, but U is not, and then finds, for each $\xi \in \mathbb{R}$, the smallest interval centered at ξ that contains exactly j eigenvalues of A_n. The answer turns out to be that the radius δ_j of the smallest interval is the jth smallest singular value of

$$\begin{bmatrix} A_k - \xi I_k \\ C \end{bmatrix}.$$

It is not hard to see that if C has rank one and $\xi = \left(\lambda_i^{(k)} + \lambda_{i+1}^{(k)}\right)/2$, then

$$\sigma_{\min}\begin{bmatrix} A_k - \xi I_k \\ C \end{bmatrix} = \left(\lambda_{i+1}^{(k)} - \lambda_i^{(k)}\right)/2,$$

as expected.

5. An alternative approach to Lehmann's work was taken by Kahan. He proved the following refined interlace theorem. See [4, pp. 194–197].

Assume $A = A_n$ has the form

$$A = \begin{bmatrix} H & C^t & O^t \\ C & V & Z^t \\ O & Z & W \end{bmatrix},$$

where H is $m \times m$ and V is $k \times k$ (and O is the zero matrix). In applications, H and C are known but V, Z, and W probably are not. Being ignorant of V, we replace it with a $k \times k$ X, which we are free to choose, and define an auxiliary matrix

$$M = M(X) = \begin{bmatrix} H & C^t \\ C & X \end{bmatrix}.$$

Denote the eigenvalues of M by $\mu_i = \mu_i(X)$ and assume $\mu_1 \leq \cdots \leq \mu_{m+k}$.

THEOREM 5 (Kahan). *Assume A, M, and X are given as above, where X is any $k \times k$ matrix satisfying*

$$V - X \text{ is invertible.}$$

Then for each index $j = 1, \cdots, m$, the interval $[\mu_j, \mu_{j+k}]$ contains a different eigenvalue λ_J of A. In addition, for each index $i = 1, \cdots, k$, there is a different eigenvalue γ_I of A outside of the open interval (μ_i, μ_{i+m}).

The only blemish in this result is the unverifiable assumption that $V - X$ is invertible. Our final contribution is to remove this hypothesis by looking carefully at the general case when $V - X$ is singular.

THEOREM 6. *Kahan's interlacing theorem (Theorem 5) remains true if the hypothesis "$V - X$ is invertible" is removed.*

Proof. Let A, X, and M be as above, except assume that $V - X$ is singular. Let $N = \text{Null Space } (V - X)$, so that $N \oplus N^{\perp} = \mathbb{R}^k$. Picking orthonormal bases for N and $N^{\perp}$, we can change A and M so that $V - X = \left[\begin{smallmatrix} O & O \\ O & Y \end{smallmatrix}\right]$ where Y is invertible. Thus,

$$V = \begin{bmatrix} V_{11} & V_{21}^t \\ V_{21} & V_{22} \end{bmatrix} = X + \begin{bmatrix} O & O \\ O & Y \end{bmatrix}, \quad \text{so } X = \begin{bmatrix} V_{11} & V_{21}^t \\ V_{21} & X_{22} \end{bmatrix}, \quad V_{22} = X_{22} + Y.$$

Break up $Z = [Z_1 \vdots Z_2]$, compatibly. Then for each $\varepsilon > 0$, let $W_\varepsilon^{\#} = Z_2 Y^{-1} Z_2^t + \frac{1}{\varepsilon} Z_1 Z_1^t$ and obtain

$$\begin{aligned} A &= \begin{bmatrix} H & & C^t & & O \\ & V_{11} & & V_{21}^t & Z_1^t \\ C & & & & \\ & V_{21} & & V_{22} & Z_2^t \\ O & Z_1 & & Z_2 & W \end{bmatrix} \\ &= \begin{bmatrix} H & & C^t & & O \\ & V_{11} - \varepsilon I & & V_{21}^t & \\ C & & & & O \\ & V_{21} & & X_{22} & \\ O & & O & & W - W_\varepsilon^{\#} \end{bmatrix} + \begin{bmatrix} O & & O & & O \\ & \varepsilon I & & O & Z_1^t \\ O & & & & \\ & O & & Y & Z_2^t \\ O & Z_1 & & Z_2 & W_\varepsilon^{\#} \end{bmatrix} \\ &= T + U, \; T = \begin{bmatrix} M_\varepsilon & \\ & W_\varepsilon \end{bmatrix}. \end{aligned}$$

Hence if we let

$$X_\varepsilon = \begin{bmatrix} V_{11} - \varepsilon I & V_{21}^t \\ V_{21} & X_{22} \end{bmatrix},$$

we can now easily see that $V - X_\varepsilon$ is invertible, so that Kahan's Interlacing Theorem applies to A, M_ε, and X_ε. If we carefully let $\varepsilon \to 0$, then $M_\varepsilon \to M$ (although $W_\varepsilon \not\to W$), so that the eigenvalues of M_ε go to the eigenvalues of M. Hence the conclusion follows, since A has only a finite number of eigenvalues. □

Note added in proof. In [5], Z. Strakoš reports that he and A. Greenbaum, in an unpublished work, have an alternative proof of Theorem 1 using orthogonal polynomials.

REFERENCES

[1] G. H. GOLUB, *Some Uses of the Lanczos Algorithm in Numerical Linear Algebra*, in Topics in Numerical Analysis, J. J. H. Miller, ed., Academic Press, New York, 1973, pp. 173–184.
[2] Y. IKEBE, T. INAGAKE, AND S. MIYAMOTO, *The monotonicity theorem, Cauchy's interlace theorem, and the Courant–Fischer theorem*, The Amer. Math. Monthly, 94 (1987), pp. 352–354.
[3] N. J. LEHMANN, *Optimale Eigenwerteinschiessung*, Numer. Math., 5 (1963) pp. 246–272.
[4] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice–Hall, Englewood Cliffs, NJ, 1980.
[5] Z. STRAKOŠ, Private communication, Sept., 1991.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 248–273, January 1992

019

A PARALLEL ITERATION METHOD AND THE CONVECTION-DIFFUSION EQUATION*

JOHN DE PILLIS†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. A pair of parallel sequences $\{u_k\} \to u$, $\{v_k\} \to 0$ (1.3) is generated to solve the $n \times n$ linear system $Au = (I_n - B)u = f$. Convergence depends only on the geometry or shape of $\sigma(B)$, the set of eigenvalues of B. The parallel method is applied to the singularly perturbed convection-diffusion equation (6.1), when the Reynolds number in the direction of flow is large. Numerical comparisons with known results are given.

Our theory also applies to the class of possibly nonsymmetric A with real spectrum (cf. Theorem 5.1) and to several other classes of systems as well. Computations to generate the sequences are relatively straightforward, as is indicated in our main result, Theorem 4.1. In fact, the parameters of the embracing ellipse for $\sigma(B)^2$ (4.6) completely determine (i) the coefficients for the parallel sequences $\{u_k\} \to u$ and $\{v_k\} \to 0$ and (ii) the spectral radius (4.4), which characterizes their asymptotic convergence rate (2.4).

Figure 5.1 illustrates some geometries for $\sigma(B)$ that are accommodated by our theory and Figure 7.1 shows the eigenvalue bowtie region arising from the convection-diffusion equation with large Reynolds number.

Key words. convection-diffusion, Reynolds cell number, SOR iteration, parallel algorithm

AMS(MOS) subject classifications. 35J99, 65M06, 15A18

1. Introduction. Notation. The set of eigenvalues (or the spectrum) of matrix A will be denoted by $\sigma(A)$ and the spectral radius, the largest modulus of all the eigenvalues, will be written as $\rho(A)$.

The problem. Although our techniques extend to a wide range of linear operators, in §6, we will focus on the solution to the finite difference representation of the singularly perturbed convection-diffusion equation

$$-\epsilon \Delta u + a(x,y)\, u_x + b(x,y)\, u_y + c(x,y)\, u = f, \tag{1.1}$$

which was studied recently by Chin and Manteuffel [3]. Equation (1.1) is defined on a bounded domain Ω with Dirichlet boundary conditions where functions a, b, c, and f are in C^2, $c \geq 0$, and constant $\epsilon > 0$.

In [3], a block (BSOR) method is developed that involves extensive case analysis for computing best-fitting ellipses for bowtie-shaped eigenvalue sets. In this paper, we are able to avoid these cases.

The parallel iterative method. To solve the invertible $n \times n$ linear system

$$Au = (I_n - B)u = f, \tag{1.2}$$

we construct the two-step parallel stationary iterative process $\{v_k\}$, $\{u_k\}$

$$\begin{aligned} v_k &= (a_0 I_n + a_1 B + a_2 B^2 - I_n)v_{k-1} - q(I_n - B)u_{k-1} + qf, \\ u_k &= \frac{1}{q}(a_1 I_n + a_2(I_n + B))v_{k-1} + u_{k-1}, \qquad k = 1, 2, 3, \cdots, \end{aligned} \tag{1.3}$$

* Received by the editors March 7, 1991; accepted for publication July 18, 1991.
† Department of Mathemtics, University of California, Riverside, CA 92521 (jdp@ucrmath.ucr.edu).

where fixed scalar $q \neq 0$. (We may therefore take $q = 1$.) Now if the vector iterates (1.3) converge for any and all initial vectors v_0, u_0, then necessarily [7, Thm. 10.1]

$$v_k \to 0,$$
$$u_k \to u.$$

As we shall see in (4.6), scalars a_2, a_1, a_0 of (1.3) are well defined by the geometry of the eigenvalue set of matrix B.

Related work. The objective is to replace a given linear system $(I_n - B)u = f$ with an equivalent system $(I_n - \tilde{B})u = \tilde{f}$, where iteration matrix $\tilde{B}$ has a more favorable (i.e., smaller) spectrum $\sigma(\tilde{B})$ than does B.

In [13], Niethammer obtained an SOR result without assuming Property A. In [9], Eiermann and Niethammer studied systems (2.1) with arbitrary compact spectrum, which they transformed into equivalent systems with circular or disk-like spectra. Generalizations of these Euler methods were further developed with k-step methods and semi-iterative methods (SIMS) by Niethammer and Varga [15] and by Niethammer, Eiermann, and Varga in [10].

One difficulty is in the estimation of bounds for the eigenvalues $\sigma(B)$. But Manteuffel in [12] got around this problem with an adaptive technique that computes parameters for the best-fitting ellipse during the computations.

In 1989, plus-shaped spectra were considered in [8] by Eiermann, Li, and Varga.

A recent work by Kellog [11] focuses on the convection-diffusion equation but no numerical results are given.

This paper proceeds in the spirit of Chin and Manteuffel [3], whose block SOR method was applied to obtain the solution of the singularly perturbed convection-diffusion equation (1.1). We will refer to this work often throughout the remainder of this paper.

Outline of this paper.

Section 2. Iterative strategies. This section describes standard stationary iterative acceleration and splitting strategies. This is a prelude to presenting the SOR and BSOR method in exactly the form used by Chin and Manteuffel [3], an understanding of which is of prime importance in this paper.

Section 3. Parallel sequences. We present some basics related to the parallel sequences (1.3) that solve $(I - B)u = f$. In particular, we need to know how the geometry of $\sigma(B)$ determines both the coefficients of (1.3) and the asymptotic convergence rate (2.4). This is the content of Theorem 3.1.

Section 4. The main theorem. An acceleration of a stationary method often takes the original spectrum $\sigma(B)$ to a circular spectrum (*cf. the remark following* (3.8)). Consistent with this idea, Theorem 4.1, our main theorem, transforms $\sigma(B)$ to a "generalized circle," i.e., an *annulus* with outside (maximum) radius given by (4.4). Also, the geometry of $\sigma(B)$ provides both (i) the coefficients (4.6) for sequences $\{u_k\}$ and $\{v_k\}$ of (1.3), along with (ii) their asymptotic convergence rate (4.8).

Section 5. Some applications. Theorem 4.1 is applied to systems $(I-B)u = f$ when the spectrum of B is (i) real (Theorem 5.1), (ii) plus-shaped (Theorem 5.2), and (iii) positive (Theorem 5.3). Figure 5.1 illustrates some other spectral geometries that can be handled by this theory.

Section 6. Discretization of the convection-diffusion equation. We present details for the finite difference discretization of the singularly perturbed convection-diffusion equation (6.1). Splitting strategies (6.3) and eigenvalue estimates for relevant band matrices (6.17) are also provided.

Section 7. A special case. We apply Theorem 4.1 to show that for a large Reynolds number in the direction of flow, the parallel method is about twice as fast as BSOR; cf. Theorem 7.1 and Fig. 7.2. Other cases of this equation (for various Reynolds numbers), which are studied in [3], we leave to a later paper.

Section 8. Ellipses and cardioids. Our main theorem, Theorem 4.1, depends on knowing the best embracing ellipse for $\sigma(B)^2$. At the same time, $\sigma(B)$ of Chin and Manteuffel is often a bowtie whose square is a cardioid. Thus, we need to know the best embracing ellipse for a cardioid and this section establishes these facts in Theorem 8.1.

2. Iterative strategies. The basic stationary iterative strategy. Given the invertible $n \times n$ linear system $Au = b$, let A_0 be any matrix for which A_0^{-1} is easy to invert, i.e., the system $A_0 y = b$ is "easy" to solve. Then A_0 induces the splitting $A = (A_0 - A_1)$, which defines the sequence $\{y_k\}$ by the following two-step procedure:
(1) Find matrix B and right-hand side f by writing

$$\begin{aligned} Au = (A_0 - A_1)u = b \quad &\text{iff } A_0(I_n - A_0^{-1}A_1)u = b \\ &\text{iff } (I_n - \underbrace{A_0^{-1}A_1}_{B})u = \underbrace{A_0^{-1}b}_{f}. \end{aligned} \tag{2.1}$$

(2) Use B and f above to define the iterative sequence $\{y_k\}$:

$$y_k = By_{k-1} + f, \qquad k = 1, 2, 3, \cdots \tag{2.2a}$$

for any initial vector y_0. Then

$$y_k \to u \quad \text{iff} \quad \rho(B) < 1. \tag{2.2b}$$

Acceleration/preconditioning. Suppose we are given a pair of easy-to-invert A_0 and $\tilde{A}_0$ that induce the splittings

$$Au = (A_0 - A_1)u = (\tilde{A}_0 - \tilde{A}_1)u = b, \tag{2.3a}$$

which, from (2.1), implies

$$\begin{aligned} B &= A_0^{-1}A_1, \qquad f = A_0^{-1}b, \\ \tilde{B} &= \tilde{A}_0^{-1}\tilde{A}_1, \qquad \tilde{f} = \tilde{A}_0^{-1}b. \end{aligned} \tag{2.3b}$$

If, moreover, we have

$$\rho(\tilde{B}) < 1 \quad \text{and} \quad \rho(\tilde{B}) < \rho(B),$$

then splitting $(A_0 - A_1)$ is called a **preconditioning for** $(\tilde{A}_0 - \tilde{A}_1)$ and the induced sequence $\tilde{y}_k = \tilde{B}y_{k-1} + \tilde{f}$ is called an **acceleration** of sequence $y_k = By_{k-1} + f$ (see (2.2a)).

Spectral radius as a measure of convergence. For iteration matrix $B = A_0^{-1}A_1$ of (2.1), the spectral radius $\rho(B)$ provides an asymptotic measure of how fast

sequence y_k of (2.2a) converges to the solution vector x of (2.1). In fact, $\log_{10}(\rho(B))$ gives us the measures

$$\text{step\_count} = \frac{-1}{\log_{10}(\rho(B))}, \qquad R = \text{Rate of Convergence} = -\log_{10}(\rho(B)), \tag{2.4}$$

where step_count represents (asymptotically) the number of iterations that suffice to produce one decimal place of accuracy: the reciprocal, R, is called the **asymptotic convergence rate**. Hence, convergence occurs for any initial y_0 (2.2a) if and only if $\rho(B) < 1$. Moreover, we see from (2.4) that smaller $\rho(B)$ implies faster convergence (cf. [17]).

The SOR acceleration. Chin and Manteuffel [3] employ block successive over-relaxation (BSOR) in their solution of the convection-diffusion equation (6.1). For this reason, we now present the salient points of this theory.

Suppose $A = D-(L+U)$ where D is block diagonal and matrices $-L$ and $-U$ are, respectively, the lower and upper block triangular parts of matrix A. (Of course, if the blocks are 1×1, then D is a diagonal of scalars.) The Jacobi splitting (2.1) is defined by taking $A_0 = D$, which induces the Jacobi iteration matrix $B_J = D^{-1}(L+U)$.

The SOR acceleration, a second class of splittings (2.1), is defined for each $\omega \in (0,2)$ by setting $A_0(\omega) = (\frac{1}{\omega})D - L$. The SOR iteration matrices take the form $\mathcal{L}_\omega = A_0(\omega)^{-1}A_1(\omega)$. We summarize these observations in (2.5).

(2.5)

$(A_0 - A_1)u = (D - (L+U))u = f$			
Splitting	A_0	Iteration Matrix $A_0^{-1}A_1$	RHS
Jacobi	D	$B_J = D^{-1}(L+U)$	$D^{-1}f$
SOR	$\left(\frac{1}{\omega}\right)D - L$	$\mathcal{L}_\omega = (D-\omega L)^{-1}\left([1-\omega]D + \omega U\right)$	$\omega(D-\omega L)^{-1}f$

If B_J has **Property A** (*or is* **block 2-cyclic**, *or is* **consistently ordered**, cf. [17], [18]) then there exists a linking (2.6) between each eigenvalue $\mu \in \sigma(B_J)$ with (a pair of) $\lambda \in \sigma(\mathcal{L}_\omega)$, according to the formula

$$\sigma(\mathcal{L}_\omega) = \left\{\lambda \,:\, (\lambda + \omega - 1)^2 - \lambda\mu^2\omega^2 = 0,\ \mu \in \sigma(B_J)\right\}. \tag{2.6}$$

If, moreover, all the eigenvalues μ of B_J are contained in the real straight line

$$\sigma(B_J) \subset [-\rho_{B_J}, \rho_{B_J}], \qquad 0 \le \rho_{B_J} < 1,$$

then, from (2.6), it follows that optimal $\omega = \omega_b$, which minimizes the "new" spectral radius $\rho(\mathcal{L}_\omega)$, has value

$$w_b = \frac{2}{(1+\sqrt{1-\rho_{B_J}^2})} > 1 \qquad \text{in which case} \qquad \rho(\mathcal{L}_{\omega_b}) = \omega_b - 1. \tag{2.7}$$

3. Parallel sequences. This section sketches some of the essentials introduced in [6] and [7].

In matrix language, the sequence solution (1.3) to system (1.2) is rephrased as follows (see [7]): Solution vector u for the $n \times n$ invertible linear system

$$Au = (I_n - B)u = f \tag{3.1}$$

is obtained by

(i) computing the scalars a_2, a_1 and a_0 from the geometry of $\sigma(B)$ as specified in [7]; (*the algorithm is generalized in* (4.6));

(ii) constructing the vector iterates $\tilde{y}_k = \mathrm{col}[v_k, u_k]$ as described by

$$\underbrace{\begin{bmatrix} v_k \\ u_k \end{bmatrix}}_{\tilde{y}_k} = \underbrace{\begin{bmatrix} (a_0 - 1)I_n + a_1 B + a_2 B^2 & -q(I_n - B) \\ (1/q)((a_1 + a_2)I_n + a_2 B) & I_n \end{bmatrix}}_{\tilde{B}} \cdot \underbrace{\begin{bmatrix} v_{k-1} \\ u_{k-1} \end{bmatrix}}_{\tilde{y}_{k-1}} + \underbrace{\begin{bmatrix} qf \\ 0 \end{bmatrix}}_{\tilde{f}} \tag{3.2}$$

where $q \neq 0$ is fixed. Then for arbitrary initial vectors v_0, u_0, we always have convergence:

$$\left.\begin{array}{c} v_k \to 0 \\ \text{and} \\ u_k \to u \end{array}\right\} \quad \textit{equivalently,} \quad \left\{ \tilde{y}_k = \begin{bmatrix} v_k \\ u_k \end{bmatrix} \to \tilde{y} = \begin{bmatrix} 0 \\ u \end{bmatrix} \right. \tag{3.3}$$

with asymptotic convergence rate (2.4):

$$R = -\log_{10} \rho(\tilde{B})$$

(see (2.2a)). It is easy to check that u satisfies the $n \times n$ system $(I_n - B)u = f$ of (3.1) if and only if $\tilde{y}$ of (3.3) satisfies the $2n \times 2n$ system (3.2), i.e.,

$$(I_{2n} - \tilde{B}) \underbrace{\begin{bmatrix} 0 \\ u \end{bmatrix}}_{\tilde{y}} = \begin{bmatrix} qf \\ 0 \end{bmatrix}. \tag{3.4}$$

Spectral radius of $\tilde{B}$. How do we compute the spectral radius $\rho(\tilde{B})$ of (3.2), (3.4)? In [7] we answered the more general question: What is the linking between each "old" eigenvalue μ of matrix B (3.1) and "new" eigenvalues $\lambda \in \sigma(\tilde{B})$ (3.2), (3.4).

THEOREM 3.1 (de Pillis [7]). *Given the second- and third-degree polynomials*

$$\begin{aligned} p(\mu) &= a_2\mu^2 + a_1\mu + a_0, \\ q(\mu) &= b_3\mu^3 + b_2\mu^2 + b_1\mu + b_0 \end{aligned} \tag{3.5}$$

subject to the single constraint that $p(1)$ and $q(1)$ are real and

$$1 - p(1) + q(1) = 0. \tag{3.6}$$

Then from the seven coefficients a_i, b_j of (3.5), *we construct* (*by a constructive algorithm in* [7]) *the $2n \times 2n$ iteration matrix $\tilde{B}$* (3.2), (3.4) *whose eigenvalues λ are characterized by*

$$\sigma(\tilde{B}) = \left\{ \lambda \ : \ \lambda^2 - p(\mu)\lambda + q(\mu), \ \mu \in \sigma(B) \right\}. \tag{3.7}$$

Remark. (Scalars a_i of (1.3) supplied by (3.5).) It is shown in Theorem 7.1 of [7] that if polynomial $q(\mu)$ in (3.5) is constant, then the coefficients a_i, which define $p(\mu)$ in (3.5), are exactly those that appear in (1.3). Throughout this paper, we always assume that polynomial $q(\mu)$ is constant.

Example. The SOR linkage (2.6) can be obtained as a special case of (3.7) once we write p and q of (3.5), (3.6) as follows:

$$p(\mu) = \mu^2\omega^2 + 2(1-\omega),$$
$$q(\mu) = (1-\omega)^2 \quad \textit{constant.} \tag{3.8}$$

Remark. Although (3.7), a generalization of (2.6), links "old" eigenvalues μ with "new" eigenvalues λ, the difficulty in practice lies in the proper choice of polynomials $p(\mu)$ and $q(\mu)$ ((3.5), (3.6))—SOR makes the correct choice in (3.8). In general, the polynomials are dependent on the geometry of the given spectrum $\sigma(B)$ and that of the "new" eigenvalues $\sigma(\tilde{B})$. For example, the SOR theory assumes a straight-line spectrum $\sigma(B)$ to begin with and produces a circular spectrum $\sigma(\mathcal{L}_{\omega_b})$.

4. The main theorem. Notation. The symbol $\mathcal{E}(X,Y)$ denotes the zero-centered ellipse in $\mathbf{C}$ whose horizontal and vertical semi-axes have dimensions X and Y, respectively. The symbol $\overline{\mathcal{E}(X,Y)}$ represents the set-theoretic union of the region enclosed by $\mathcal{E}(X,Y)$ and its boundary (see also (8.1)).

Preview of the main theorem. Theorem 4.1 below uses the dimensions X and Y of the best-fitting ellipse $\mathcal{E}(X,Y)$ that embraces $\sigma(B)^2$; cf. (4.2) and (4.3). Using these dimensions, we automatically produce the coefficients (4.6) necessary in the construction of the parallel sequences $\{u_k\} \to u$ and $\{v_k\} \to 0$ of (1.3). The asymptotic convergence rate is also obtained in (4.8). We present our theorem now.

THEOREM 4.1. *Given the invertible $n \times n$ linear system*

$$Au = (I_n - B)u = f \tag{4.1}$$

where, for complex (shift) constants $\mathbf{c}$, $\mathbf{d}$ *and ellipse* $\mathcal{E}(X,Y)$, *we have for every* $\mu \in \sigma(B)$, *that* $(\mu + \mathbf{d})^2 \in \mathbf{c} + \overline{\mathcal{E}(X,Y)}$. *Symbolically,*

$$\left\{\sigma(B) + \mathbf{d}\right\}^2 \subset \left\{\mathbf{c} + \overline{\mathcal{E}(X,Y)}\right\}. \tag{4.2}$$

For the complex shift scalars $\mathbf{c}$, $\mathbf{d}$ *and semi-axis lengths* X *and* Y, *we have the inequality*

$$\mathbf{Q} = \frac{(\mathbf{d}+1)^2 - \mathbf{c}}{\sqrt{|X^2 - Y^2|}} \quad \textit{is real and} \quad |\mathbf{Q}| > 1. \tag{4.3}$$

Then all eigenvalues λ *of induced iteration matrix* $\tilde{B}$ (3.2), (3.4) *lie within an annulus which has outside (maximum) circle of radius*

$$\rho(\tilde{B}) = \sqrt{\frac{X+Y}{|X-Y|}} \cdot |\tilde{\rho}| \tag{4.4}$$

where real $\tilde{\rho}$ *of* (4.4) *is given by*

$$\tilde{\rho} = \underbrace{\operatorname{sign}\mathbf{Q}}_{\pm 1} \left(|\mathbf{Q}| - \sqrt{\mathbf{Q}^2 - 1} \right), \qquad |\tilde{\rho}| < 1. \tag{4.5}$$

The scalars $\{a_2, a_1, a_0\}$, *which define the parallel sequences* $\{v_k\}$, $\{u_k\}$ *of* (1.3), *are given by*

$$a_2 = \frac{2\tilde{\rho}}{\sqrt{|X^2 - Y^2|}}, \qquad a_1 = 2\mathbf{d}\, a_2, \qquad a_0 = (\mathbf{d}^2 - \mathbf{c})\, a_2. \tag{4.6}$$

Convergence of parallel sequences $\{v_k\}$, $\{u_k\}$ *of* (1.3) *obtains if and only if*

$$\sqrt{\frac{X+Y}{|X-Y|}} < |\mathbf{Q}| + \sqrt{\mathbf{Q}^2 - 1}, \tag{4.7}$$

in which case $\rho(\tilde{B}) < 1$ *is given by* (4.4) *and*

$$(4.8)\qquad \left.\begin{matrix} u_k \to 0 \\ v_k \to u \end{matrix}\right\} \text{ with asymptotic convergence rate } R = -\log_{10}(\rho(\tilde{B}))$$

where u *is the desired solution vector for* (4.1).

Proof of (4.4). We appeal to (3.5) of Theorem 3.1 to define polynomials $p(\mu)$ and $q(\mu)$ as follows:

$$(4.9)\qquad p(\mu) = a_2\mu^2 + a_1\mu + a_0, \qquad a_2,\ a_1,\ a_0 \text{ real},$$

$$(4.10)\qquad q(\mu) = (\pm\tilde{\rho}^2), \qquad \text{constant } \tilde{\rho}^2 > 0.$$

From Theorem 3.1, polynomials $p(\mu)$ and $q(\mu)$ do two things. They

(a) provide the coefficients $\{a_i\}$ for construction of the iteration matrix $\tilde{B}$ (3.2), (3.4) and

(b) characterize the eigenvalues $\sigma(\tilde{B})$ as follows (see (3.7)):

$$(4.11)\qquad \sigma(\tilde{B}) = \{\lambda : \lambda^2 - \underbrace{p(\mu)}_{\lambda_\mu + \lambda'_\mu}\lambda + \underbrace{(\pm\tilde{\rho}^2)}_{\lambda_\mu\lambda'_\mu} = 0, \text{ where } \mu \in \sigma(B),\ \lambda = \lambda_\mu, \lambda'_\mu\}.$$

As (4.11) indicates, every eigenvalue μ of B induces a pair of complex eigenvalues $\lambda_\mu, \lambda'_\mu$ of $\tilde{B}$. Now the constant real term $\pm(\tilde{\rho}^2)$ of (4.11) is necessarily the product of these complex roots λ_μ and λ'_μ, so that

$$(4.12)\qquad \begin{matrix} \lambda_\mu \text{ and } \lambda'_\mu \in \sigma(\tilde{B}) \\ (\lambda_\mu\lambda'_\mu) = \pm(\tilde{\rho}^2) \text{ real} \end{matrix} \qquad \text{implies} \qquad \begin{cases} \lambda_\mu = (\ \alpha_\mu\ )\ \tilde{\rho}\, e^{i\theta_\mu} \\ \lambda'_\mu = \left(\pm\dfrac{1}{\alpha_\mu}\right)\ \tilde{\rho}\, e^{-i\theta_\mu} \end{cases}$$

for some angle $\theta \in \mathbf{R}$, $\alpha_\mu \geq 1$, where $\tilde{\rho} > 0$ is fixed. Define

$$(4.13)\qquad \alpha_M = \max_{\mu\in\sigma(B)} \alpha_\mu \geq 1.$$

Then, from (4.12) and (4.13), it follows that the spectral radius of $\tilde{B}$ is

$$(4.14)\qquad \rho(\tilde{B}) = \max_{\mu\in\sigma(B)} |\ \underbrace{\alpha_\mu\,\tilde{\rho}\,e^{i\theta}}_{\lambda\,\in\,\sigma(\tilde{B})}\ | = \alpha_M\,\tilde{\rho}.$$

Similarly, $p(\mu)$, the λ-coefficient in (4.11), is necessarily the *sum* of the complex roots λ_μ and λ'_μ. Using the form for $\lambda_\mu, \lambda'_\mu$ given in (4.12), we obtain

$$(4.15)\qquad p(\mu) = \lambda_\mu + \lambda'_\mu = \underbrace{\tilde{\rho}\left(\alpha_\mu e^{i\theta_\mu} \pm \frac{1}{\alpha_\mu}e^{-i\theta_\mu}\right)}_{\text{a point on } \mathcal{E}\{\tilde{\rho}, \alpha_\mu\}\ (8.2)} \qquad \text{for all } \mu \in \sigma(B),$$

which says that $p(\mu)$ lies on the ellipse $\mathcal{E}(X_\mu, Y_\mu) = \mathcal{E}\{\tilde{\rho}, \alpha_\mu\}$, within the family of confocal ellipses (8.2).

Finally, equate $p(\mu)$ in (4.10) and (4.15) to obtain the statement: $\mu \in \sigma(B)$ implies

$$a_2\mu^2 + a_1\mu + a_0 = \tilde{\rho}\left(\alpha_\mu e^{i\theta_\mu} \pm \frac{1}{\alpha_\mu} e^{-i\theta_\mu}\right),$$

which, in factored form, gives us the equivalent statement: $\mu \in \sigma(B)$ implies

$$\left(\mu + \underbrace{\frac{a_1}{2a_2}}_{\mathbf{d}}\right)^2 + \underbrace{\frac{a_0}{a_2} - \underbrace{\left(\frac{a_1}{2a_2}\right)^2}_{\mathbf{d}^2}}_{-\mathbf{c}} = \underbrace{\left(\frac{\tilde{\rho}}{a_2}\right)}_{\mathbf{r}} \left(\alpha_\mu e^{i\theta_\mu} \pm \frac{1}{\alpha_\mu} e^{-i\theta_\mu}\right). \tag{4.16}$$

Now equation (4.16) defines the three scalars $\mathbf{r} = \tilde{\rho}/a_2$, $\mathbf{c} = \mathbf{d}^2 - a_0/a_2$ and $\mathbf{d} = a_1/2a_2$. In other words, we obtain another equivalent statement: $\mu \in \sigma(B)$ implies

$$(\mu + \mathbf{d})^2 - \mathbf{c} = \underbrace{\mathbf{r}\left(\alpha_\mu e^{i\theta_\mu} \pm \frac{1}{\alpha_\mu} e^{-i\theta_\mu}\right)}_{\text{a point in } \mathcal{E}\{\mathbf{r},\, \alpha_\mu\}\ (8.2)} \subset \overline{\mathcal{E}\{\mathbf{r}, \alpha_M\}}, \tag{4.17}$$

where

$$\begin{aligned} \mathbf{r}\, a_2 &= \tilde{\rho}, \\ (2\mathbf{d})\, a_2 - a_1 &= 0, \\ (\mathbf{d}^2 - \mathbf{c})\, a_2 - a_0 &= 0. \end{aligned} \tag{4.18}$$

(In (4.17), we use the fact that $\mathcal{E}(\mathbf{r}, \alpha_\mu) \subset \mathcal{E}(\mathbf{r}, \alpha_M)$ since $\alpha_\mu \leq \alpha_M$ (see (8.1), (4.13)).) Another way to state (4.17) is as follows:

$$\mu \in \sigma(B) \quad \text{implies } (\mu + \mathbf{d})^2 \in \begin{cases} \mathbf{c} + \overline{\mathcal{E}(X, Y)}, \\ \text{equivalently,} \\ \mathbf{c} + \overline{\mathcal{E}\{\mathbf{r}, \alpha_M\}} \end{cases}$$

where (8.3c, d) provide the following relations among semi-axes X and Y and constants $\mathbf{r}$, α_M of the maximal ellipse:

$$\mathbf{r} = \sqrt{|X^2 - Y^2|}/2, \tag{4.19a}$$

$$\alpha_M = \sqrt{|(X+Y)/(X-Y)|} \geq 1. \tag{4.19b}$$

Substituting α_M of (4.19b) into (4.14) proves (4.4).

Proof of (4.5). Before we validate (4.5), we must first compute polynomial coefficients $\{a_2, a_1, a_0\}$ (3.5) with the given constants $\mathbf{c}$, $\mathbf{d}$, X, and Y in hand (4.2). Combining earlier observations, we obtain the following 4×3 linear system in the

three unknowns $\{a_2, a_1, a_0\}$.

$$(4.20)\qquad \begin{matrix}\text{from (4.16), (4.18)} \longrightarrow \\ \text{from (3.6), (4.10)} \longrightarrow\end{matrix} \left\{ \underbrace{\begin{bmatrix} \mathbf{r} & 0 & 0 \\ 2\mathbf{d} & -1 & 0 \\ \mathbf{d}^2-\mathbf{c} & 0 & -1 \\ 1 & 1 & 1 \end{bmatrix}}_{A} \cdot \underbrace{\begin{bmatrix} a_2 \\ a_1 \\ a_0 \end{bmatrix}}_{\vec{a}} = \underbrace{\begin{bmatrix} \tilde{\rho} \\ 0 \\ 0 \\ 1+\tilde{\rho}^2 \end{bmatrix}}_{f} \right.$$

Consistency of (4.20) is easily established as follows: Multiply both sides of (4.20) by the 4×4 nonsingular diagonal matrix $\mathbf{D}$=diag $[[(\mathbf{c}-(\mathbf{d}+1))^2]/\mathbf{r}, 1, 1, 1]$. Since the left-hand vector $\mathbf{D}(A\vec{a}) = \mathbf{0}$, consistency of (4.20) demands that the right-hand vector $\mathbf{D}f = 0$ as well. But to require $\mathbf{D}f = 0$ is to require

$$\begin{aligned} 0 &= \tilde{\rho}^2 - \frac{(\mathbf{d}+1)^2-\mathbf{c}}{\mathbf{r}}\tilde{\rho} + 1 \\ &= \tilde{\rho}^2 - 2\left(\frac{(\mathbf{d}+1)^2-\mathbf{c}}{\sqrt{|X^2-Y^2|}}\right)\tilde{\rho} + 1 \qquad \text{from (4.19a)} \\ &= \tilde{\rho}^2 - 2\mathbf{Q}\tilde{\rho} + 1 \qquad \text{from (4.3),} \end{aligned}$$

whose real solutions $\tilde{\rho}$ and $(\tilde{\rho})'$ are both positive or both negative. The constant term of the last equation equals one, which says that exactly one of the roots, $\tilde{\rho}$, say, has absolute value strictly less than one, if and only if

$$(4.21a)\qquad \mathbf{Q} \text{ is real and } |\mathbf{Q}| > 1$$

$$(4.21b)\qquad \tilde{\rho} = \underbrace{\text{sign}\mathbf{Q}}_{\pm 1}\left(|\mathbf{Q}| - \sqrt{\mathbf{Q}^2-1}\right), \qquad |\tilde{\rho}| < 1,$$

$$(4.21c)\qquad \tilde{\rho}' = \underbrace{\text{sign}\mathbf{Q}}_{\pm 1}\left(|\mathbf{Q}| + \sqrt{\mathbf{Q}^2-1}\right), \qquad |\tilde{\rho}'| > 1.$$

The inequalities in (4.21) are guaranteed since $|\mathbf{Q}| > 1$ in (4.3). It is (4.21b) that proves (4.5).

Proof of (4.6). Substitute the value of $\mathbf{r}$ from (4.19a) into (4.20). Since the equations (4.20) are consistent (thanks to (4.21)), we may select any three of the four equations to find the solution scalars $\{a_2, a_1, a_0\}$. Choosing the first three, then, the proof of (4.6) is done.

Proof of (4.7) *and* (4.8). Finally, we note

$$\begin{aligned} \tilde{\rho}(B) &= \alpha_M \cdot \tilde{\rho} \quad \text{from (4.14),} \\ &= \sqrt{\frac{X+Y}{|X-Y|}} \cdot \tilde{\rho} \quad \text{from (4.19b),} \end{aligned}$$

which reconfirms (4.4). Also, the last term above (the product of the square root and $\tilde{\rho}$), has absolute value less than one if and only if

$$\begin{aligned} \sqrt{\frac{X+Y}{|X-Y|}} &< \frac{1}{\tilde{\rho}} && \text{from (4.19b),} \\ &= (\tilde{\rho})' && \text{from (4.21c),} \\ &= \mathbf{Q} + \sqrt{\mathbf{Q}^2-1} && \text{from (4.21c).} \end{aligned}$$

This proves (4.7). The fact that $\rho(\tilde{B}) < 1$, coupled with (2.4), validates (4.8). This ends the proof of the main theorem. □

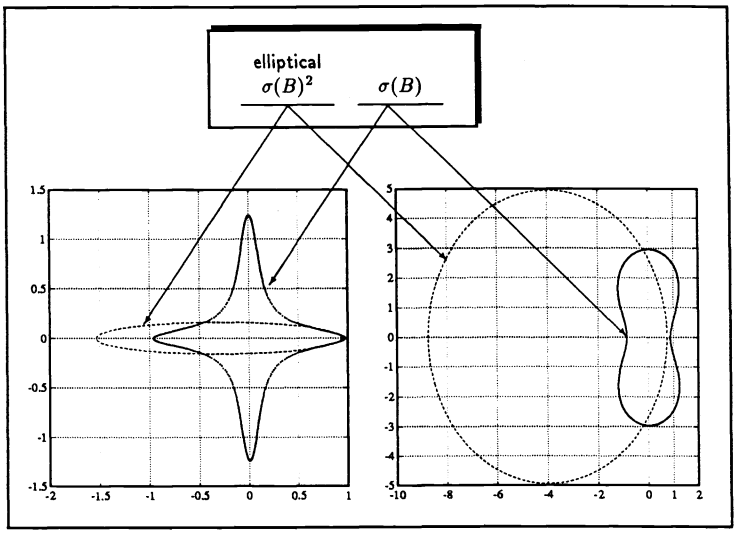


FIG. 5.1.

5. Some applications. It is relatively easy to apply our main theorem, Theorem 4.1, to any of several geometric configurations for $\sigma(B)$ where $A = I_n - B$. Basically, we need only substitute parameters into the hypotheses (4.2) and (4.3). For example, in this section, we consider the system

$$Au = (I_n - B)u = f \tag{5.1}$$

and we characterize the coefficients and convergence rates for the sequences $v_k \to 0$ and $u_k \to u$ (1.3) when

- A has real eigenvalues (Theorem 5.1). This class includes all invertible symmetric matrices $A = A^*$.
- A has a plus-shaped set of eigenvalues (Theorem 5.2). This class includes all matrices with horizontal and/or vertical straight-line spectra.
- A has positive eigenvalues (Theorem 5.3). This class includes all positive definite matrices.

Remark. (Fig. 5.1: Other spectral configurations.) Our theory accommodates other interesting $\sigma(A)$ or $\sigma(B)$ geometries that could occur and be handled by our theory. Figure 5.1 illustrates a "top-shaped" and a "peanut shell" region that might cover our $\sigma(B)$—in both cases, the squares, $\sigma(B)^2$, are ellipses $\mathbf{c} + \mathcal{E}(X, Y)$, as required by Theorem 4.1.

An interesting geometry for $\sigma(A)$ (or $\sigma(B)$) that has already gained some attention in the literature is the "bowtie" introduced by Chin and Manteuffel [3] in their application of the block SOR method for solving the perturbed convection-diffusion equation. (More about this in §§6 and 7 (see also Fig. 7.1).)

THEOREM 5.1. (*A has real eigenvalues.*) *Given the invertible system* (5.1) *where* (*by scaling A if necessary*), *we may assume the spectrum of A lies within the real*

interval $[-1, 1]$. *That is,*

$$\begin{aligned} &\sigma(A) \subset [-1, -\epsilon] \cup [\epsilon, 1], \qquad \textit{or} \\ &\sigma(B) \subset [0, 1-\epsilon] \cup [1+\epsilon, 2] \quad \textit{for some } \epsilon > 0. \end{aligned} \tag{5.2}$$

Then for spectral radius

$$\tilde{\rho}_{real} = \left(\frac{1-\epsilon}{1+\epsilon}\right),$$

we have convergence for the sequence pair u_k, v_k *of* (1.3),

$$\left.\begin{matrix} u_k \to u \\ v_k \to 0 \end{matrix}\right\} \textit{ with rate } R_{real} = -\log_{10}(\tilde{\rho}_{real}), \tag{5.3}$$

where u *is the solution vector of* (5.1). *The coefficients* a_2, a_1, *and* a_0, *which generate the discretization sequences* u_k, v_k (1.3), *are given by*

$$a_2 = \frac{4}{1-\epsilon^2}\,\tilde{\rho}_{real}, \qquad a_1 = \frac{-8}{1-\epsilon^2}\,\tilde{\rho}_{real}, \qquad a_0 = -\frac{2(1+\epsilon)}{1-\epsilon^2}\,\tilde{\rho}_{real}. \tag{5.4}$$

Proof. In (4.2), choose $\mathbf{d} = -1$ so that we obtain from (5.2) the set-theoretic inclusion

$$\sigma(B + \mathbf{d}) \subset [-1, -\epsilon] \cup [\epsilon, 1],$$

which, in turn, implies

$$\begin{aligned} \sigma(B+\mathbf{d})^2 &\subset [\epsilon^2, 1] \\ &= \mathbf{c} + \overline{\mathcal{E}(X, 0)} \quad \text{a degenerate ellipse.} \end{aligned}$$

The (degenerate) ellipse, $[\epsilon^2, 1] = \mathbf{c} + \mathcal{E}(X, 0)$ above, allows us to specify all the parameters $\mathbf{c}$, $\mathbf{d}$ and semiaxes X and $Y = 0$ that are required for the computational test of hypotheses (4.2) of Theorem 4.1. In fact,

$$\mathbf{d} = -1, \quad \mathbf{c} = \frac{1+\epsilon^2}{2}, \quad X = \frac{1-\epsilon^2}{2}, \quad Y = 0, \tag{5.5}$$

so that substitution into (4.3) yields

$$\mathbf{Q} = \frac{(\mathbf{d}+1)^2 - \mathbf{c}}{\sqrt{|X^2 - Y^2|}} = -\frac{1+\epsilon^2}{1-\epsilon^2}.$$

We see that $\mathbf{Q}$ is real and $|\mathbf{Q}| > 1$, which means that the computational test of (4.3) is satisfied. A straightforward substitution of values (5.5) into (4.5) produces intermediate value $\tilde{\rho}$, which, when substituted into (4.4), gives us convergence rate (4.8), which reduces to (5.3). Finally, substitution of intermediate $\tilde{\rho}$ (4.5) into (4.6) produces the form (5.4) for the sequence coefficients and the theorem is proved. □

Remark. The following theorem recaptures Theorem 10.1 of [7] which solves $Au = f$ when $\sigma(A)$ is plus-shaped ("plus-shaped" is described by (5.6)). This case was also studied by Eiermann, Li, and Varga in [8], where they achieved a convergence rate equal to one half that of R_+ given by (5.7) and (5.8).

THEOREM 5.2. (*A has plus-shaped spectrum.*) *Given the invertible system* (5.1) *where matrix B has top-shaped spectrum centered at* $z = 0$, *i.e.,*

$$\sigma(B) \subset \{\mu : \mu \in [-r, r] \cup [-it, it]\} \begin{cases} 0 \le r < 1, \\ 0 \le t, \\ i^2 = -1. \end{cases} \tag{5.6}$$

Then for spectral radius

$$\tilde{\rho}_+ = \frac{2\left[1 - \sqrt{(1+t^2)(1-r^2)}\right] + t^2 - r^2}{t^2 + r^2}, \tag{5.7}$$

we have convergence for the sequence pair u_k, v_k *of* (1.3),

$$\left.\begin{matrix} u_k \to u \\ v_k \to 0 \end{matrix}\right\} \text{ with rate } R_+ = -\log_{10}(\tilde{\rho}_+), \tag{5.8}$$

where u *is the solution vector of* (5.1). *The coefficients* a_2, a_1, *and* a_0, *which generate the sequences* u_k, v_k (1.3), *are given by*

$$a_2 = \frac{4}{t^2 + r^2}\tilde{\rho}_+, \qquad a_1 = 0, \qquad a_0 = -\frac{2(t^2 - r^2)}{t^2 + r^2}\tilde{\rho}_+. \tag{5.9}$$

Proof. The proof proceeds by substituting values derived from hypothesis (5.6) and substituting into (4.4) and (4.6). In fact, from (5.6), we have

$$\sigma(B)^2 \subset [-t^2, r^2],$$

where the interval $[-t^2, r^2]$ above is the degenerate ellipse with center $\mathbf{c} = (r^2 - t^2)/2$, horizontal semiaxis $X = (t^2 + r^2)/2$, and vertical semiaxis $Y = 0$. Once we observe that $\mathbf{d} = 0$ in (4.2), we establish all the necessary parameters for substitution into (4.3). Thus,

$$\mathbf{Q} = \frac{2 + t^2 - r^2}{t^2 + r^2} > 1 \quad \text{for all } 0 \le r < 1 \qquad \text{and } 0 \le t.$$

Further substitution of $\mathbf{Q}$ into (4.5) yields $\tilde{\rho}$, which, when substituted into (4.4), gives us the effective spectral radius $\tilde{\rho}_+$ of (5.7) which, from (4.8), proves (5.8). Finally, substitution of intermediate $\tilde{\rho}$ (4.4) into (4.6) gives us the coefficients (5.9) for the parallel sequence (1.3) and the theorem is done. □

If $\sigma(A)$ is positive, then, as our next theorem shows, our method produces spectral radius $\tilde{\rho}_{pos}$ (5.10), which agrees with the SOR spectral radius $\rho(\mathcal{L}_{\omega_b})$ of (2.7).

THEOREM 5.3. (*A has positive eigenvalues.*) *Given the invertible system* (5.1) *where (by scaling A if necessary), we may assume that for some positive* $\rho < 1$, *the spectrum of A lies within the positive interval* $[1 - \rho, 1 + \rho]$. *That is,*

$$\begin{aligned} &\sigma(A) \subset [1 - \rho, 1 + \rho], \quad 0 \le \rho < 1, \quad \textit{or} \\ &\sigma(B) \subset [-\rho, \rho] \quad \textit{for some } \epsilon > 0. \end{aligned}$$

Then for spectral radius

$$\tilde{\rho}_{pos} = \underbrace{\frac{2}{1 + \sqrt{1 - \rho^2}}}_{\omega_b \textit{ of } (2.7)} - 1, \tag{5.10}$$

we have convergence for the sequence pair u_k, v_k *of* (1.3),

$$\left.\begin{matrix} u_k \to u \\ v_k \to 0 \end{matrix}\right\} \text{ with rate } R_{pos} = -\log_{10}(\tilde{\rho}_{pos}), \tag{5.11}$$

where u *is the solution vector of* (5.1). *The coefficients* a_2, a_1, *and* a_0, *which generate the sequences* u_k, v_k (1.3), *are given by*

$$a_2 = \frac{4\tilde{\rho}_{pos}}{\rho^2}, \qquad a_1 = 0, \qquad a_0 = -2\tilde{\rho}_{pos}. \tag{5.12}$$

Proof. In Theorem 5.2, take $r = \rho$ and $t = 0$ so that (5.7), (5.8), and (5.9) reduce to (5.10), (5.11), and (5.12), respectively. □

Remark. In §6, we apply our methods to a discretization model for the singularly perturbed convection-diffusion equation.

6. Discretization of the convection-diffusion equation. Writing the convection-diffusion equation (1.1) over a rectangular grid, in the form

$$-\epsilon\,(u_{xx} + u_{yy}) + u_x + c\,u = 0, \qquad c = \text{ constant}, \tag{6.1}$$

we will establish the following:

- If h_x is the grid mesh size in the x direction (see also (6.11)), then $-\epsilon u_{xx} + u_x + cu$ is represented in discrete form by the matrix equation (6.12).
- If h_y is the mesh size in the y direction, then (6.13) is the matrix discretization for the term u_{yy}.
- Therefore, (6.15) describes the discretization matrices H, V and (6.14) characterizes the right-hand n-vector $\tilde{f}$ for the matrix discretization

$$A[u] = (H + V)[u] = \tilde{f} \tag{6.2}$$

  where $[u]$ is the approximation to the solution u of (6.1).
- Eigenvalue estimates (6.17) are given for appropriate matrices.

Remark. Equation (6.1) is only slightly more general than the example studied in [3].

Assume that the flow is modeled over a rectangular grid where the grid points are ordered lexicographically. We use the usual five-point finite difference discretization for approximating the second-order derivatives—central differences will represent the first-order derivative(s). The resulting discretizing matrix $A=H+V$ has the familiar pentadiagonal (or block tridiagonal) form diagrammed in Fig. 6.1. (*The same form would result from the discretization of virtually any second-order partial differential equation in two variables over a rectangular grid.*)

As is seen in Fig. 6.1, there are two choices for H in the splitting $A = H+V$. That is, we may assign H to be the "inside" tridiagonal matrix diag$\{ L_0, U_0, D \}$ (which includes half the main diagonal), and then set V equal to the "outside" tridiagonal matrix, diag$\{ L_1, U_1, D \}$. Or, we may reverse the rôles of H and V. That is, either

$$A = \underbrace{D + L_0 + U_0}_{H} + \underbrace{D + L_1 + U_1}_{V} \qquad \text{flow} \parallel x \tag{6.3a}$$

or

$$A = \underbrace{D + L_1 + U_1}_{H} + \underbrace{D + L_0 + U_0}_{V} \qquad \text{flow} \perp x, \tag{6.3b}$$

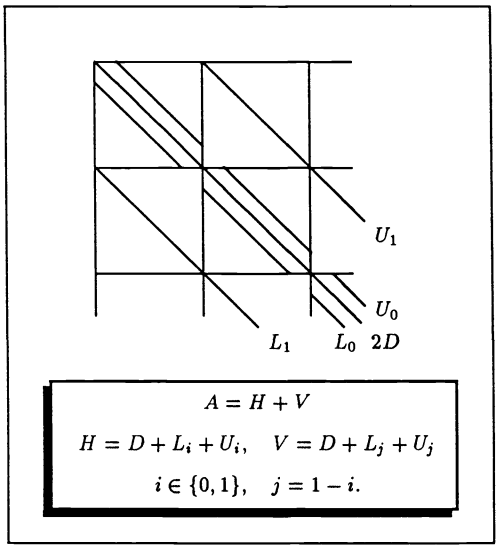


FIG. 6.1.

which is to say,

$$A = \underbrace{D + L_i + U_i}_{H} + \underbrace{D + L_{1-i} + U_{1-i}}_{V}, \qquad i \in \{0,1\}. \tag{6.3c}$$

Remark. (Common main diagonal.) As (6.3) indicates, $A = H + V$ is split so that H and V have a common diagonal D. This is useful since, if necessary, we can exploit the fact that both $H - D$ and $V - D$ are block two-cyclic (or consistently ordered).

In point of fact, Chin and Manteuffel [3] split $A = H + V$ (6.3a, b) as per (2.3a) so that *all* the diagonal part of A, namely $2D$, is assigned to $A_0 = 2D - L_i - U_i$ leaving A_1 with a zero block diagonal. That is,

$$A = \underbrace{\underbrace{2D + L_i + U_i}_{H+D}}_{A_0} + \underbrace{\underbrace{L_{1-i} + U_{1-i}}_{V-D}}_{-A_1}, \qquad i \in \{0,1\}, \tag{6.4}$$

so that A_1, with zero block diagonal, is block two-cyclic (or consistently ordered). Moreover, the matrix product

$$B_J = \underbrace{(H + D)^{-1}}_{A_0^{-1}} \underbrace{(V - D)}_{A_1} \tag{6.5}$$

remains block two-cyclic (and consistently ordered); cf. [17] where A_0 and A_1 are given by (6.4).

Eigenvalue structure when diagonals are constant. Tridiagonal matrices H and V (Fig. 6.1) with *constant* diagonals have eigenvalue bounds that are especially

easy to estimate. In fact, the tridiagonal matrix

(6.6)
$$M = \begin{bmatrix} d & b & 0 & 0 & \cdots \\ a & d & b & 0 & \cdots \\ 0 & a & \ddots & \ddots & \\ \vdots & & \ddots & \ddots & \\ & & & & \end{bmatrix} \text{ implies } \sigma(M) \subset d + \underbrace{[-2\sqrt{ab},\, 2\sqrt{ab}]}_{\substack{\text{horizontal/vertical} \\ \text{line-interval in } \mathbf{C}}}.$$

Since ab can be either positive or negative, $\sqrt{ab}$ in (6.6) is either real or imaginary. Also, the interval end-points, $\pm 2\sqrt{ab}$, become tighter for $\sigma(M)$ as matrix dimension n increases. (*This is easily seen by choosing diagonal D so that $D^{-1}MD$ is either Hermitian or skew-Hermitian according to whether ab is positive or negative.*)

Applying the eigenvalue estimates (6.6) to matrices H and V of (6.3) where the non-main diagonals a, b take on the values U_i, L_i, $i = 0, 1$, we obtain

(6.7)

Spectral Bounds for H and V.
$\sigma(H) \subset d + [-2\sqrt{L_iU_i},\ \underbrace{2\sqrt{L_iU_i}}_{\alpha}]$
$\sigma(V) \subset d + [-2\sqrt{L_{1-i}U_{1-i}},\ \underbrace{2\sqrt{L_{1-i}U_{1-i}}}_{\beta}]$.

Remark. (H, V have straight-line spectra.) As (6.6) indicates, eigenvalues $\sigma(M)$ lie on a horizontal line centered at d when $ab > 0$ and $\sigma(M)$ lies on the vertical line centered at d when $ab < 0$. This carries over to the shifted matrices $(H + D)$ and $(V - D)$ (6.7), which, likewise, have straight-line (horizontal or vertical) spectra.

Remark. (H and V commute.) Constant diagonals imply that H and V of Fig. 6.1 and displays (6.4), (6.5) will commute. Commutivity of H and V is somewhat special and holds only on rectangular regions, as was shown by Birkhoff and Varga [1]. In Birkhoff, Varga, and Young [2], we find characterizations for commutivity for H and V. See also [17] and [18].

Commutivity of H and V means that

(6.8)
$$\left.\begin{matrix} \mu_V \in \sigma(V) \\ \mu_H \in \sigma(H) \end{matrix}\right\} \quad \text{implies } \frac{\mu_V - d}{\mu_H + d} \in \sigma\,(\underbrace{(H + dI_n)^{-1}\,(V - dI_n)}_{B_J}).$$

Remark. (B_J has straight-line or bowtie spectrum.) As Chin and Manteuffel observed in [3], $\sigma(B_J)$ either lies on a straight line or is contained in a "bowtie" region (see (6.8)). These observations on the geometry of $\sigma(H)$, $\sigma(V)$, and $\sigma(B_J)$ are

summarized in (6.9) (cf. Fig. 7.1 for an illustration of the third case of (6.9)).

(6.9)

(see Fig. 6.1)

$A = H + V, \quad \mathbf{diag}(A) = 2D = 2dI_n,$

$\alpha = 2\sqrt{L_i U_i}, \quad \beta = 2\sqrt{L_{1-i}U_{1-i}}, \quad i = 0, 1$

EMBRACING SET for

α	β	$\sigma(H)$	$\sigma(V)$	$\sigma(B_J) = \sigma\frac{(V - dI_n)}{(H + dI_n)}$	
real	*real*	$d + \alpha[-1, 1]$	$d + \beta[-1, 1]$	$\frac{\beta}{2d - \alpha}[-1, 1]$	horiz line
real	*imag*	$d + \alpha[-1, 1]$	$d + \beta[-1, 1]$	$\frac{\beta}{2d - \alpha}[-1, 1]$	vert line
imag	*real*	$d + \alpha[-1, 1]$	$d + \beta[-1, 1]$	$\pm\frac{\beta}{4d}\{1 + \overline{\mathcal{U}}\}$	horiz bowtie
imag	*imag*	$d + \alpha[-1, 1]$	$d + \beta[-1, 1]$	$\pm\frac{\beta}{4d}\{1 + \overline{\mathcal{U}}\}$	vert bowtie

where $\overline{\mathcal{U}}$ denotes the closed unit disk in $\mathbf{C}$.

Note that (6.1) describes flow in the x direction only (*there is no u_y term*) and therefore, decomposition (6.3a) is indicated for the approximating matrices. In other words matrices H and V are defined in terms of their common diagonal D and the (non-main) diagonal matrices L_0, U_0, L_1, U_1, by

$$H = D + U_0 + L_0, \qquad V = D + U_1 + L_1. \tag{6.10}$$

The perturbation constant $\epsilon \ll 1$ of (6.1) and the mesh-dependent dimensions h_x and h_y define the half-grid Reynolds numbers R_x and R_y as follows:

$$R_x = \frac{h_x}{2\epsilon}, \qquad R_y = \frac{h_y}{2\epsilon}. \tag{6.11}$$

To determine H and V of (6.10), first represent the x derivatives of (6.1), with matrix H':

(6.12)

$$\frac{1}{2h_x R_x}\underbrace{\begin{bmatrix} 2(1 + c\,R_x h_x) & (R_x - 1) & 0 \\ -(R_x + 1) & 2(1 + c\,R_x h_x) & \ddots \\ 0 & \ddots & \ddots \end{bmatrix}}_{H'}[u] \approx -\epsilon\, u_{xx} + u_x + c\,u$$

and represent the y derivatives with V' given by

$$\frac{1}{2h_xR_x} \cdot \underbrace{\left(\frac{h_x}{h_y}\right)^2 \begin{bmatrix} 2 & 0 & \cdots & -1 & \cdots \\ 0 & 2 & \ddots & & \ddots \\ \vdots & \ddots & \ddots & \ddots & \\ -1 & & & & \\ \vdots & \ddots & & & \end{bmatrix}}_{V'} [u] \approx u_{yy}. \tag{6.13}$$

Adding (6.12) and (6.13) and multiplying through by $2h_xR_x$, we obtain the matrix discretization of (6.1)

$$A[u] = (H' + V')[u] = 2h_xR_xf = \tilde{f}. \tag{6.14}$$

We finally construct matrices H and V (6.2) by shifting H' and V' of (6.12), (6.13) in a way that guarantees a shared or common diagonal. That is, choose scalar s such that $H = H' + sI_n$ and $V = V' - sI_n$ so that

$$\begin{aligned} \textbf{diag}(H) &= \textbf{diag}(V) &&= D = dI_n, \\ \textbf{off-diagonal}(H) &= \textbf{off-diagonal}\ (H') &&= L_0,\ U_0, \\ \textbf{off-diagonal}(V) &= \textbf{off-diagonal}\ (V') &&= L_1,\ U_1, \end{aligned} \tag{6.15}$$

where, in the convention of Fig. 6.1, the exact values for the five constant diagonals D, L_0, U_0, L_1, U_1 of H and V described in (6.15) are given by

(6.16)

$D = d \cdot I_n = \left((h_x/h_y)^2 + 1 + c\,R_xh_x\right) \cdot I_n$	
$L_0 = -(R_x + 1)$	$L_1 = -(h_x/h_y)^2$
$U_0 = (R_x - 1)$	$U_1 = -(h_x/h_y)^2$

Eigenvalues structure of H and V. From (6.16), we see that the signs of the non-main diagonals U_0, L_0 of Fig. 6.1 will be positive or negative according as the Reynolds number R_x is greater than or less than one. Substitution of the values U_0, L_0, U_1, L_1 (6.16) into (6.7) or (6.9) yields

(6.17)

Spectral Bounds for H and V.
$\sigma(H) \subset d + [-2\sqrt{1 - R_x^2},\ +\underbrace{2\sqrt{1 - R_x^2}}_{\alpha}]$,
$\sigma(V) \subset d + [-2(h_x/h_y)^2,\ +\underbrace{2(h_x/h_y)^2}_{\beta}]$.

From (6.16), we see that R_y (6.11) plays no rôle in the *sign* of the non-main diagonals L_1, U_1 of V.

7. A special case. We analyze the third case of (6.9), that is, we find the solution to (6.1) when $R_x > 1$.

THEOREM 7.1. *Given the convection-diffusion equation* (6.1) *defined over a rectangular region where* $\epsilon \ll 1$ *and* c *is constant, suppose mesh sizes* h_x *and* h_y *are such that*

$$1 < \frac{h_x}{2\epsilon} = R_x, \tag{7.1a}$$

$$0 < \frac{2\epsilon}{h_x^2} + c, \tag{7.1b}$$

$$-\frac{\sqrt{2}+1}{4\epsilon} < \frac{2\epsilon}{h_y^2} + c < \frac{\sqrt{2}-1}{4\epsilon}. \tag{7.1c}$$

Then the scalar r *defined by*

$$r = \frac{(h_x/h_y)^2}{(h_x/h_y)^2 + (ch_x^2/(2\epsilon)) + 1} \tag{7.2}$$

induces the spectral radius $\tilde{\rho}_0(r)$ *and intermediate parameter* $\tilde{\rho}$ *defined by*

(7.3)

$$\tilde{\rho}_0(r) = \underbrace{\sqrt{\frac{3\sqrt{3}+5}{3\sqrt{3}-5}}}_{\approx 7.2098} \cdot \left(\frac{8 - 3r^2 - \sqrt{7r^4 - 48r^2 + 64}}{\sqrt{2}r^2} \right) < 1 \quad \textit{for all } r < 0.988\cdots,$$

which, for the parallel sequence u_k, v_k *of* (1.3), *implies*

$$\left.\begin{matrix} u_k \to u \\ v_k \to 0 \end{matrix}\right\} \textit{ with rate } R_0(r) = -\log_{10}(\tilde{\rho}_0(r)), \tag{7.4}$$

where u *is the solution vector of* (6.2). *The coefficients* a_2, a_1, *and* a_0, *which generate the parallel sequences* u_k, v_k (1.3), *are given by*

$$a_2 = 8\sqrt{2}\,\tilde{\rho}/r^2, \qquad a_1 = 0, \qquad a_0 = 3\sqrt{2}\,\tilde{\rho} \tag{7.5}$$

where $\tilde{\rho}$ *is defined in* (7.3).

Proof. We proceed in four steps, the first three of which establish geometric equivalents to the three hypotheses (7.1a, b, c), which is diagrammed in Fig. 7.1. The fourth step of the proof applies the main theorem, Theorem 4.1, to this geoemtry.

Step I. (Equivalent to hypothesis (7.1a).) The spectral bounds α and β are given in (6.17), which implies that hypothesis (7.1a) is equivalent to

$$\alpha = 2\sqrt{1 - R_x^2} \qquad \text{is imaginary,}$$

$$\beta = 2\left(\frac{h_x}{h_y}\right)^2 \qquad \text{is real.}$$

But if α is imaginary and β is real, then we are placed in the third case of (6.9): thus, $\sigma(B_J)$ is embraced by the horizontal bowtie $\pm\beta/(4d)\{1 + \overline{U}\}$ illustrated in Fig. 7.1.

Step II. (Equivalent to hypothesis (7.1b).) For β and d given by (6.17), we confirm that r of (7.2) has the form

$$r = \beta/(2d).$$

Geometrically, the values $r = \pm\beta/(2d)$ mark the outermost intersections on the real axis of the bowtie containing $\sigma(B_J)$ (see Fig. 7.1). In other words, hypothesis (7.1b) says that the embracing bowtie (Fig. 7.1) lies within the vertical strip $-1 < \text{real}(z) < 1$.

STEP III. (Equivalent to hypothesis (7.1c).) As Fig. 7.1 indicates, the angle of the bowtie depends on α/d. In fact, we claim that hypotheses (7.1c) says that the arc of Fig. 7.1 is a semicircle or larger, and this is equivalent to the inequality

$$\frac{|\alpha|}{2d} > 1, \tag{7.6}$$

where α and d are given in (6.17).

To see this, use (6.9) to characterize $\mu_H \in \sigma(H)$ and $\mu_V \in \sigma(V)$ as follows:

$$\mu_H = d + i|\alpha|s, \quad \mu_V = d + \beta t \quad \text{for some } -1 \le s, t \le 1. \tag{7.7}$$

Then for any s, t of (7.7), eigenvalue $\lambda_{s,t} \in \sigma(B_J)$ has the form

$$\begin{aligned}
\lambda_{s,t} &= \frac{\mu_V - d}{\mu_H + d} && \text{from (6.8), (6.9)} \\
&= \frac{\beta t}{d}\left(\frac{1}{2 + i\,(|\alpha|/d)\,s}\right) && \text{from (7.7)} \\
&= \frac{\beta t}{d}\left(\frac{2 - i\,(|\alpha|/d)\,s}{4 + ((|\alpha|/d))^2\,s^2}\right) && [*] \\
&\in \pm\frac{\beta}{d}\left(\frac{1}{4}\cdot\{1 + \overline{U}\}\right) && \text{since } -1 \le t \le 1.
\end{aligned} \tag{7.8}$$

See (6.9), where $\overline{U}$ is the closed unit disc.

Note. In (7.8)[*], the quotient inside parentheses describes an arc of a circle centered at $z = \frac{1}{4}$ with radius $\frac{1}{4}$. As the real parameter $|\alpha|/d$ increases, the arc, which begins at the right-hand end-point of the circle, increases as well. In the fourth line of (7.8), we note that the circle in (7.8)[*] is multiplied through or scaled by $\pm\beta/(4d)$. This accounts for the bowtie shape indicated in Fig. 7.1.

We conclude that each $\lambda_{s,t} \in \sigma(B_J)$ is captured by a region that is enclosed by a pair of discs with radius $\beta/(4d)$ and centered at $z = \pm\beta/(4d)$, respectively. (Hence, they are each tangent to the imaginary axis.) Since the real and imaginary parts of (7.8)[*] represent cosine and sine of $\lambda_{s,t}$, we see that the "bowtie arc" (Fig. 7.1) is at least 180° when the line from the origin is at least 45°. This says the angle described by the Cartesian form of (7.8)[*] is at least 45°, which means the modulus of the real part of (7.8)[*] is less than or equal to that of the imaginary part. That is, $|\alpha|/(2d) \ge 1$ and (7.6) is thus confirmed.

Step IV. (Applying Theorem 4.1.) Application of our main theorem, Theorem 4.1, depends on parameters of $\sigma(B_J)^2$. From (7.2), the bowtie that embraces $\sigma(B_J)$

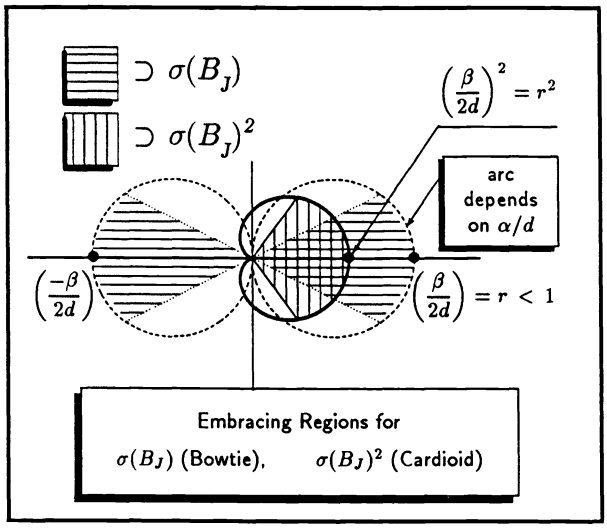


FIG. 7.1.

has right and left end-points $\pm r = \pm\beta/(2d)$, so that its square, $\sigma(B_J)^2$, must be covered by the *cardioid* with respective left and right end-points

$$z = 0 \quad \text{and} \quad r^2 = (\beta/(2d))^2.$$

What is the best-covering ellipse for the cardioid $\supset \boldsymbol{\sigma(B_J)^2}$? Theorem 8.1 gives the dimensions of the best-fitting ellipse for the cardioid that has left and right real-axis intersections at $z = 0$ and $z = 8$ (cf. Fig. 8.1). Our cardioid (Fig. 7.1) has corresponding real-axis intersections $z = 0$ and $z = r^2$. Multiplying or scaling the cardioid of Fig. 8.1 by $r^2/8$ produces the cardioid of Fig. 7.1, which embraces $\sigma(B_J)^2$. Thus, the best-fitting ellipse for Fig. 7.1 is described by the following parameters, as required by (4.3):

(7.9a) Real Intersections $z = 0, \quad z = r^2$

(7.9b) Semi Axes $X = 5r^2/8, \quad Y = 3\sqrt{3}r^2/8$

(7.9c) Shift Scalar $\mathbf{c} = 3r^2/8, \quad \mathbf{d} = 0,$

which, when substituted into (4.3), says that for r of (7.2),

$$\mathbf{Q}_r = \frac{8 - 3r^2}{r^2\sqrt{2}} > 1 \quad \text{for } 0 < r < 1.346\cdots.$$

Now from (4.5), we have $\tilde{\rho} = \mathbf{Q}_r - \sqrt{\mathbf{Q}_r^2 - 1}$, or

$$\tilde{\rho} = \left(\frac{8 - 3r^2 - \sqrt{7r^4 - 48r^2 + 64}}{\sqrt{2}r^2}\right) < 1 \quad \text{for } 0 < r < 1.346\cdots.$$

Substituting this intermediate variable $\tilde{\rho}$ with values of X and Y from (7.9) into (4.4)

and (4.8), respectively, we obtain (i) spectral radius

$$\tilde{\rho}_0(r) = \underbrace{\sqrt{\frac{3\sqrt{3}+5}{3\sqrt{3}-5}}}_{\approx 7.2098} \cdot \left(\frac{8 - 3r^2 - \sqrt{7r^4 - 48r^2 + 64}}{\sqrt{2}r^2} \right) < 1 \quad \text{for } 0 < r < 0.988\cdots,$$

proving (7.3) and (ii) convergence rate (7.4), which is a restatement of (4.8). Finally, the coefficients (7.5) are obtained by substituting $\tilde{\rho}$ above and X and Y values of (7.9) into (4.6). This ends the proof of Theorem 4.1. □

Remark. (Convergence fails for r near 1.) Testing (4.7), we see that $\tilde{\rho}_0(r) > 1$ of (7.3) when r of (7.2) has the property $r > 0.98796$. Thus, when the bowtie of Fig. 7.1 strays too close to $z = 1$, (i.e., when $r = \beta/(2d) \to 1$), then we lose convergence altogether. This explains why the BSOR method of Chin and Manteuffel is superior to the parallel method when $r > 0.967$. See Fig. 7.2 for a comparison of the parallel method and BSOR.

Chin and Manteuffel's application of BSOR. Here is an overview of how Chin and Manteuffel apply the SOR theory toward the solution of (6.1). From the SOR theory (2.5), two-cyclic matrices (6.5) induce an ω-dependent family of iteration matrices

$$\mathcal{L}_\omega = \underbrace{(H + D - \omega L_{1-i})^{-1}}_{(A_0 - \omega L_{1-i})^{-1}} \underbrace{([1-\omega](H+D) + \omega U_{1-i})}_{([1-\omega]A_0 + \omega U_{1-i})} \tag{7.10}$$

with a smaller or more favorable spectrum.

Matrix $\mathcal{L}_{\omega_b}$ of (7.10) is a special case of (2.5) wherein D is replaced with $H + D$ from (6.4) and L of (2.5) is replaced with L_{1-i} of (6.4).

Eigenvalues λ of $\mathcal{L}_\omega$ (7.10) are linked with the eigenvalues μ of Jacobi iteration matrix B_J of (6.5), as described by (2.6).

Note. Matrix $\mathcal{L}_\omega$ of (7.10) is denoted by G in [3].

The Chin–Manteuffel spectral radius $\tilde{\rho}_{BSOR}$ of (2.4) is computed from $\mathcal{E}_{CM}$, the best-fitting ellipse for the bowtie (*NOT for the cardioid!*). In particular [3, (3.17c)], if $X > Y$ represent the semi-axis dimensions of the $\mathcal{E}_{CM}$ (so that $F = \sqrt{X^2 - Y^2}$ is the focal dimension), then the optimal spectral radius ρ_{BSOR} is given by

$$\rho_{BSOR} = \frac{Y + \sqrt{X}}{1 + \sqrt{1 - F^2}}.$$

In our theory, the spectral radius $\tilde{\rho}_0(r)$ of (7.3) (and (4.4)) depends on the best-fitting ellipse for the *cardioid* shown in Fig. 7.1. Chin and Manteuffel consider solution of (6.1) when $h_x = h_y$ and $c = 0$.

Comparison of parallel and BSOR methods (Fig. 7.2). We diagram a comparison of the spectral radii produced by the two methods in Fig. 7.2 which shows that our method produces a little more than twice the BSOR convergence rate until $r = \beta/(2d) \approx 0.967$.

In case the mesh sizes h_x and h_y are equal, and if constant $c = 0$ in (6.1), then $r = 0.5$. With these parameters, we see from Fig. 7.2 that the parallel method exhibits a bit more than twice the convergence rate of BSOR.

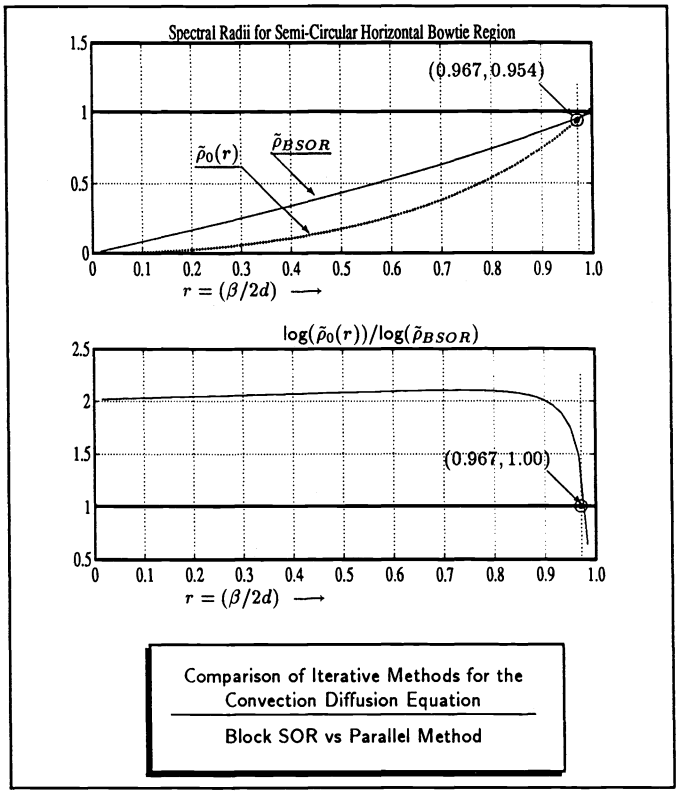


FIG. 7.2.

Remark. In a personal communication, Bruce Kellog observed the following: When modelling very high cell Reynolds numbers with central differences, we inherently induce unreliable results. This is not a matter of physics so much as mathematics, or even matrix theory. Consider, for example, the central difference approximation to

$$-\epsilon u'' + u' = f, \qquad u(0) = u(1) = 0$$

on a uniform mesh, which induces the linear system as $A(\epsilon)U = F$. For an odd number of interior mesh points, matrix $A(0)$ is singular. One would hope that $A(0)$ would give an approximation to the reduced problem

$$u' = f, \qquad u(0) = 0.$$

We would also expect that $A(0)$ is at least nonsingular, and that for some constant C independent of ϵ and n, we have

$$||A(\epsilon)^{-1}||_2 < C. \tag{7.11}$$

This is the "stability problem." There are other issues, relating to accuracy, mesh refinements near the boundary, etc., but these other issues are common to *all* discretizations. The stability criterion, as exemplified by (7.11), is not satisfied by the central difference approximation.

These comments pertain to very large cell Reynolds numbers. If ϵ is moderate, the central difference approximation is a good one.

8. Ellipses and cardioids. Notation for Cartesian form of the ellipse. For x, y real, we define the *Cartesian* form for the ellipse and its closure (in the complex plane) by

$$\text{ellipse } \mathcal{E}(X,Y) = \left\{ x+iy \,:\, \frac{x^2}{X^2} + \frac{y^2}{Y^2} = 1 \right\},$$

$$\text{closure } \overline{\mathcal{E}(X,Y)} = \left\{ x+iy \,:\, \frac{x^2}{X^2} + \frac{y^2}{Y^2} \le 1 \right\}. \tag{8.1}$$

Notation for exponential form of the ellipse. For real $\rho \neq 0$ and fixed $\alpha \ge 1$, we define the *exponential form* of the ellipse

$$\text{ellipse } \mathcal{E}\{\rho,\alpha\} = \left\{ \rho \left(\alpha e^{i\theta} \pm \frac{1}{\alpha} e^{-i\theta} \right) \right\}, \qquad \theta \in \mathbf{R},$$

$$\text{closure } \overline{\mathcal{E}\{\rho,\alpha\}} = \left\{ \rho \left(\beta e^{i\theta} \pm \frac{1}{\beta} e^{-i\theta} \right) \right\}, \qquad \theta \in \mathbf{R}, \tag{8.2}$$

$$\text{for all } \beta \text{ where } \frac{1}{\alpha} \le \beta,\ \frac{1}{\beta} \le \alpha.$$

Relating the Cartesian and exponential forms. There is a direct connection between the semi-axes X, Y of (8.1) and the parameters ρ, α of (8.2), namely,

$$X = \rho \left(\alpha \pm \left(\frac{1}{\alpha} \right) \right) \ge 0, \tag{8.3a}$$

$$Y = \rho \left(\alpha \mp \left(\frac{1}{\alpha} \right) \right) \ge 0, \tag{8.3b}$$

$$(\alpha)^2 = |(X+Y)/(X-Y)| \ge 1, \tag{8.3c}$$

$$4\rho^2 = |(X+Y)\cdot(X-Y)|. \tag{8.3d}$$

Note that $+(1/\alpha)$ (respectively, $-(1/\alpha)$) is chosen in (8.3a) if and only if horizontal semi-axis length X is greater than (respectively, less than) vertical semi-axis length Y.

A cardioid and its covering ellipse. We use the symbols $\mathcal{E}$ and $\mathcal{C}$ as a shorthand to denote the particular cardioid and shifted ellipse

$$\begin{aligned} \mathcal{E} &\equiv (3,0) + \mathcal{E}(5, 3\sqrt{3}) && \text{Cartesian form (8.1),} \\ \mathcal{C} &\equiv 4\,(1+\cos(\theta)) = \rho_\theta,\ 0 \le \theta \le \pi && \text{polar form (8.2).} \end{aligned} \tag{8.4}$$

We use barred notation to denote the closures of the indicated curve, e.g., $\overline{\mathcal{E}}$ is the set-theoretic union of the curve $\mathcal{E}$ and all its interior points.

THEOREM 8.1. *For the ellipse $\mathcal{E}$ and cardiod $\mathcal{C}$ of* (8.4), *we have the inclusion* $\overline{\mathcal{E}} \supset \overline{\mathcal{C}}$. (*See Fig.* 8.1.)

Proof. We show that if ellipse $\mathcal{E}$ is to cover cardioid $\mathcal{C}$(if $\overline{\mathcal{E}} \supset \overline{\mathcal{C}}$), then it is enough to show that $\mathcal{E}$ and its derivatives agree with $\mathcal{C}$ at the two "obvious" axes points A = A$(3, 3\sqrt{3})$ and B = B$(8,0)$. (We use the term "obvious" since the tangent lines

at A and B are, respectively, horizontal and vertical.) The horizontal cardioid tangent at A and vertical tangent at B then define the semi-axes [B-C] and [A-C] of ellipse $\mathcal{E}$ (indicated by double lines in Fig. 8.1) with lengths

$$|\,\mathrm{B}-\mathrm{C}\,| = 5, \qquad |\,\mathrm{A}-\mathrm{C}\,| = 3\sqrt{3}.$$

Using $\overline{\mathcal{E}}$ to denote the closure of the shifted ellipse (8.1), (8.4), we have

$$x + iy \in \overline{\mathcal{E}} \quad \text{iff} \quad \frac{(x-3)^2}{25} + \frac{(y)^2}{27} \leq 1. \tag{8.5}$$

Of course, equality obtains in (8.5) if and only if $x + iy$ lies on the boundary curve $\mathcal{E}$. From Fig. 8.1, we see that for all angles θ, point $P(\theta) = x + iy$ lies on the cardioid $\mathcal{C}$ if and only if $P(\theta)$ has Cartesian coordinate form

$$\begin{aligned} P(\theta) &= \underbrace{4[1+\cos(\theta)]}_{\rho_\theta} \cdot (\cos(\theta) + i\sin(\theta)) \\ &= \underbrace{4\,(1+\cos(\theta))\cos(\theta)}_{x} + i\,\underbrace{4\,(1+\cos(\theta))\sin(\theta)}_{y}, \end{aligned}$$

defining the values of x and y, which, when substituted into (8.5), tells us that for all angles θ,

$$\begin{aligned} P(\theta) \in \overline{\mathcal{E}} \quad \text{iff} \quad & 27[(4(1+\cos(\theta)))^2 - 24(1+\cos(\theta)) + 9] \\ & +25[(4(1+\cos(\theta)))^2\sin(\theta)^2] \leq 25 \cdot 27. \end{aligned}$$

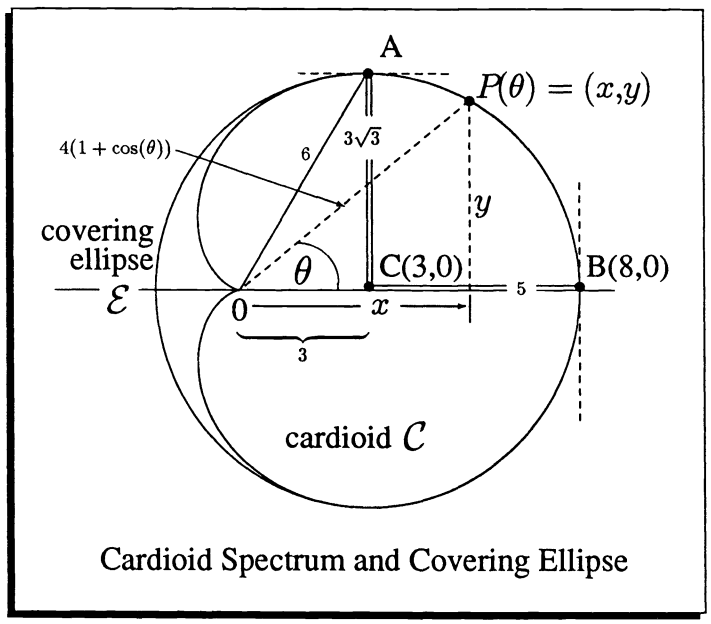


FIG. 8.1.

Now substitute u for $\cos(\theta)$ in the last equation. After a bit of algebraic manipulation, we finally obtain the simplified inequality

$$P(\theta) \in \overline{\mathcal{E}} \quad \text{iff} \quad \phi(u) = 4u^4 + 8u^3 - 27u^2 + 19u - 4 \leq 0,$$

for all real u where $-1 \leq u \leq 1$. It is now an exercise in calculus to show that ϕ has only one local maximum for $u \in (-1,1)$ (i.e., $\phi'(u) = 0$ and $\phi''(u) < 0$ for $|u| < 1$) and this occurs only at the point $u = \frac{1}{2}$. Since this maximum $\phi(\frac{1}{2}) = 0$, we are assured that $\phi \leq 0$ over the open interval $(-1,1)$. Calculation at the end-points $\{-1\}, \{+1\}$ shows that $\phi(-1) = -54$ and $\phi(1) = 0$, which guarantees $\phi \leq 0$ over the *closed* interval $[-1,1]$. The theorem is proved. □

REFERENCES

[1] G. BIRKHOFF AND R. VARGA, *Implicit alternating direction methods*, Trans. Amer. Math. Soc., 92 (1959), pp. 13–24.

[2] G. BIRKHOFF, R. VARGA, AND D. YOUNG, *Alternating direction implicit methods*, Adv. Comput., 3 (1962), pp. 189–273.

[3] R. CHIN AND T. MANTEUFFEL, *An analysis of block successive overrelaxation for a class of matrices with complex spectra*, SIAM J. Numer. Anal., 25 (1988), pp. 564–585.

[4] R. CHIN, T. MANTEUFFEL, AND J. DE PILLIS, ADI *as a preconditioning for solving the convection-diffusion equation*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 281–299.

[5] J. DE PILLIS, *How to embrace your spectrum for faster iterative results*, Linear Algebra Appl., 34 (1980), pp. 125–143.

[6] ———, *Tensor equivalents for solution of linear systems*, SIAM J. Algebraic. Discrete Meth., 8 (1987), pp. 291–312.

[7] ———, *A parallelizable* SOR*-like algorithm for plus-shaped spectra*, Linear Algebra Appl., 154/156 (1991), pp. 551–582.

[8] M. EIERMANN, X. LI, AND R. S. VARGA, *On hybrid semi-iterative methods*, SIAM J. Numer. Anal., 26 (1989), pp. 152–168.

[9] M. EIERMANN AND W. NIETHAMMER, *On the construction of semi-iterative methods*, SIAM J. Numer. Anal., 20 (1983), pp. 1153–1160.

[10] M. EIERMANN, W. NIETHAMMER, AND R. S. VARGA, *A study of semi-iterative methods for nonsymmetric systems of linear equations*, Numer. Math., 47 (1985), pp. 505–533.

[11] R. B. KELLOG, *Spectral bounds and iterative methods in convection-dominated flow*, in Advanced Computational Methods for Boundary and Interior Layers, J. J. H. Miller, ed., Boole Press, Dublin, 1991.

[12] T. A. MANTEUFFEL, *Adaptive procedures for estimating parameters for a nonsymmetric Tchebychev iteration*, Numer. Math., 31 (1978), pp. 183–208.

[13] W. NIETHAMMER, *On different splittings and the associated iteration methods*, SIAM J. Numer. Anal., 16 (1979), pp. 186–200.

[14] W. NIETHAMMER, J. DE PILLIS, AND R. S. VARGA, *Convergence of block iterative methods applied to sparse least-squares problems*, Linear Algebra Appl., 58 (1984), pp. 327–341.

[15] W. NIETHAMMER AND R. VARGA, *On the analysis of k-step iterative methods for linear systems from summability theory*, Numer. Math., 41 (1983), pp. 17–206.

[16] D. C. SMOLARSKI, *Optimum semi-iterative methods for the solution of any linear algebraic system with a square matrix*, Department of Computer Science, University of Illinois at Urbana-Champaign, December 1981.

[17] R. S. VARGA, *Matrix Iterative Analysis*, Prentice–Hall, Englewood Cliffs, NJ, 1962.

[18] D. M. YOUNG, *Iterative Solution of Large Linear Systems*, Academic Press, New York, London, 1971.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 274–291, January 1992

020

FAST ADAPTIVE CONDITION ESTIMATION*

DANIEL J. PIERCE† AND ROBERT J. PLEMMONS‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Recursive condition number estimates of matrices are useful in many areas of scientific computing, including recursive least squares computations, optimization, eigenanalysis, and general nonlinear problems solved by linearization methods where matrix modification techniques are used. The purpose of this paper is to propose a fast adaptive condition estimator, called *ACE*, for tracking the condition number of a modified matrix over time, in terms of its triangular factors. Symmetric rank-one modifications are considered, and it is noted how the schemes generalize to higher rank modifications and thus to nonsymmetric rank-one updates. *ACE* is fast in the sense that only $O(n)$ operations are required for n parameter problems, and is adaptive over time, i.e., estimates at time t are used to produce estimates at time $t+1$. Traditional condition estimators for triangular factors, such as the LINPACK and LAPACK type schemes, generally require $O(n^2)$ operations and are not adaptive. The only situation where *ACE* can break down is characterized, and a remedy is provided. *ACE* is based upon min-max principles where a small generalized eigenvalue problem is solved at each recursive step. Numerical experiments are reported here and elsewhere, indicating that the method yields an accurate and robust, yet inexpensive, adaptive condition estimator for recursive matrix modifications.

Key words. adaptive methods, condition estimation, downdating, eigenvalues, matrix modifications, optimization, recursive least squares, singular values, updating

AMS(MOS) subject classifications. 15A12, 15A18, 65F10, 65F20, 65F35

1. Introduction.

Repeated estimates for the condition number of a matrix are required in many application areas of scientific computing, including: optimization, least squares computations, eigenanalysis, and general nonlinear problems solved iteratively by linearization techniques, e.g., [1], [3], [4], [13], [28]. For instance, it is often desired to solve a sequence of n by n systems of linear equations

$$A_t x_t = b_t, \qquad t = 1, 2, \cdots, \tag{1}$$

where

$$A_{t+1} = A_t + U_t, \qquad t = 1, 2, \cdots, \tag{2}$$

with the update matrices U_t having low rank.

Our interest is primarily in the case where the matrices A_t are symmetric positive definite, and where the symmetric updates (2) are performed by updating the Cholesky factor R_t (or its inverse R_t^{-1}) of A_t, and where the rank of U_t is small relative to the dimension of A_t. (In particular, $\text{rank}(U_t) \leq 2$ is quite common, e.g., [4], [13], [16], [18].)

We wish to monitor the spectral condition number

$$\kappa_2(A_t) = \frac{\lambda_{\max}(A_t)}{\lambda_{\min}(A_t)} = \left[\frac{\sigma_{\max}(R_t)}{\sigma_{\min}(R_t)}\right]^2$$

* Received by the editors September 19, 1990; accepted for publication (in revised form) July 29, 1991.

† Boeing Computer Services, P.O. Box 24346, MS 7L-21, Seattle, Washington 98124-0346 (dpierce@atc.boeing.com).

‡ Department of Mathematics and Computer Science, Wake Forest University, P.O. Box 7388, Winston-Salem, North Carolina 27109 (plemmons@mthcsc.wfu.edu). This research was supported by United States Air Force grants AFOSR-88-0285 and AFOSR-91-0163.

over time t, where $\lambda(A_t)$ denotes an eigenvalue of A_t and $\sigma(R_t)$ a singular value of R_t. Since the updates can be performed in $O(n^2)$ operations, one would like to estimate the condition number in fewer than $O(n^2)$ operations.

Our purpose in this paper is to develop an $O(n)$ adaptive condition estimator, in the spirit of the incremental condition estimator, *ICE*, developed by Bischof [6]. As in [6], the key computation is the estimation of extreme singular values for a triangular factor by optimization techniques. Higham [21] has surveyed condition estimation techniques for triangular factors, and all the methods he surveys are $O(n^2)$, where n is the dimension of the triangular factor. The condition estimator *ICE* is applied incrementally as the Cholesky factor is constructed by rows or columns and, in total, would also require $O(n^2)$ operations for each update in our application.

The scheme *ACE* (for *a*daptive *c*ondition *e*stimator), that we are proposing here is fast in the sense that only $O(n)$ operations are required for each update of R_t, or R_t^{-1}. It is helpful to clarify the use of the words *incremental* versus *adaptive*. Incremental *ICE* obtains condition estimates of a triangular factor that grows, whereas adaptive *ACE* obtains condition estimates when information is added to or extracted from an already existing factorization. Both *ICE* and *ACE* use the estimates generated at the previous step in constructing new estimates. A preliminary version of *ACE*, in the context of exponential weighting for recursive least squares methods for signal processing, has been given in [26]. An alternative adaptive condition estimation scheme, which is $O(n^2)$ and is based on Lanczos methods, is considered in [15]. The scheme in [15] is called *ALE* for *a*daptive *L*anczos *e*stimator.

We point out that *ICE* has also been applied to updating computations in signal processing by Bischof and Shroff [7]. Their scheme is applied in a different matrix modification context inappropriate for *ACE*. The application of *ICE* in [7] is to maintain a rank-revealing triangular factorization through condition estimates of intermediate leading triangular factors, which then allows for easy computation of an approximate basis for the nullspace. In our context *ICE* would not be adaptive between updates. Shroff and Bischof [29] have also proposed a sheme, *GRACE*, which can be used to track the condition number for rank-one updates of QR factorizations, where both Q and R are updated. Their scheme uses *ACE* as part of the computations. We will have additional comments concerning some heuristic methods associated with *GRACE* later in this paper.

In §2 we examine the method of updating the Cholesky factor or its inverse, after a symmetric rank-one modification of the matrix A_t. We describe the *ACE* algorithm in detail in §3. The only situation where *ACE* can break down is characterized, and a remedy is provided in §4. Section 5 reports experimental results on the effectiveness of the condition number estimator on a variety and volume of problems. Similar results for *ACE* on different condition estimation problems have been reported elsewhere [14], [15], [26], [29]. Some conclusions, comparisons, and observations on our work are contained in §6.

2. Symmetric rank-one modifications. In this section we describe the typical methods used for the symmetric update problems under consideration. For the sake of exposition we will consider here only the rank-one update problem. (We will return to the low rank updates in §6 and discuss the applicability of *ACE*.) For now let us assume that A is a symmetric positive definite matrix to which we apply the symmetric rank-one modification determined by a vector y. That is,

$$\tilde{A} = A + \rho y y^T, \qquad \rho = \pm 1. \tag{3}$$

Note that we take $\rho = \pm 1$ without loss of generality. If $\rho = 1$, the process (3) is called *updating*; while if $\rho = -1$, the process is called *downdating*. For simplicity we will use the general term *update* to refer to (3).

As was mentioned, the update is accomplished by appropriately modifying the Cholesky factor R of A to produce the Cholesky factor $\widetilde{R}$ of $\widetilde{A}$; or as is also common, modifying R^{-1} to produce $\widetilde{R}^{-1}$. Such modifications arise, for example, in recursive least squares methods in control and signal processing [1], [3], [5], [19], [26], [27], [28], and in optimization methods [4], [13], [16]. Below we consider the modification of R, and later we show that the modification of R^{-1} is similar. We first rewrite (3) in the form

$$\widetilde{R}^T\widetilde{R} = R^TR + \rho yy^T \tag{4}$$

$$= \begin{bmatrix} R^T & y \end{bmatrix} H \begin{bmatrix} R \\ y^T \end{bmatrix}, \tag{5}$$

where

$$H = \begin{bmatrix} I & 0 \\ 0 & \rho \end{bmatrix}. \tag{6}$$

Then, typically, a series of transformations, Q_ρ^i, are computed, which simultaneously preserve the property

$$Q_\rho^{i\,T} H Q_\rho^i = H$$

and for which

$$Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 \begin{bmatrix} R \\ y^T \end{bmatrix} = \begin{bmatrix} \widetilde{R} \\ 0^T \end{bmatrix}. \tag{7}$$

If $\rho = 1$ then the Q_ρ^i's are simply Givens rotations. If $\rho = -1$ then the Q_ρ^i's correspond to hyperbolic rotations (see, e.g., [18, §12.6.4], for a discussion of hyperbolic rotations in the context of matrix reduction). Both Givens and hyperbolic rotations can be written in the common form

$$Q_\rho^i = \begin{bmatrix} I_{i-1} & & & \\ & \cos(\rho^{1/2}\xi_i) & & -\rho^{1/2}\sin(\rho^{1/2}\xi_i) \\ & & I_{n-i-1} & \\ & \rho^{-\frac{1}{2}}\sin(\rho^{1/2}\xi_i) & & \cos(\rho^{1/2}\xi_i) \end{bmatrix},$$

for scalars ξ_i. Our interest is in recursively updating ($\rho = 1$) and/or downdating ($\rho = -1$) the Cholesky factor of the modified problem over time.

The modification of the inverse proceeds in a similar manner as the modification of R, in that a sequence of transformations is applied to a matrix. The modification of R^{-1} is accomplished by computing transformations Q_ρ^i so that

$$Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 \begin{bmatrix} -\rho a \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \delta \end{bmatrix}, \tag{8}$$

where

$$a = R^{-T}y \quad \text{and} \quad \delta^2 = 1 + \rho a^T a$$

and

$$(Q_\rho^i)^T H Q_\rho^i = H.$$

Then one computes

$$Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 \begin{bmatrix} R^{-T} \\ 0 \end{bmatrix} = \begin{bmatrix} \widetilde{R}^{-T} \\ v^T \end{bmatrix},$$

where

$$v = \frac{-R^{-T}a}{\delta}.$$

It has been shown by Pan and Plemmons [25] that the transformations Q_ρ^i in (8) are identical to those in (7). Moreover,

$$Q_\rho \equiv Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix} \equiv \begin{bmatrix} C^{-T} & \rho Ca/\delta^2 \\ -a^T/\delta & 1/\delta \end{bmatrix} \tag{9}$$

where

$$a = R^{-T}y, \qquad \delta = \sqrt{1 + \rho a^T a},$$

and C is the upper triangular Cholesky factor of the matrix $I + \rho aa^T$, that is,

$$C^T C = I + \rho aa^T.$$

These results from [25] are exploited in the application of ACE in the matrix updating problem.

In the next section we show how one can adaptively monitor the condition of the factor R (or R^{-1}) in only $O(n)$ operations for the updating procedures that we have described in this section.

3. ACE. The Cholesky factor modification methods described in the previous section require $O(n^2)$ operations. The ability to monitor the condition of the computed factors provides insight into the accuracy of the computed solution as well as providing information on the behavior of the underlying iterative process. However, if the cost of monitoring the condition is of the same order of magnitude as that required to update the factor, then this may reduce the applicability of the estimator, or simply make it infeasible. In this section we describe a condition estimator, ACE, which requires only $O(n)$ operations per modification. The estimator has proven quite accurate on a wide range of problems. There are rare occurrences of numerical difficulties, but a heuristic "fix" has already been proposed by Shroff and Bischof in [29].

The fundamental relationship that ACE exploits is, for $A = R^T R$,

$$R^T x = b \Rightarrow \sigma_{\min}(R) \leq \frac{\|b\|_2}{\|x\|_2} = \left[\frac{x^T R R^T x}{x^t x}\right]^{1/2} \leq \sigma_{\max}(R).$$

Here

$$\kappa_2(A) = \kappa_2^2(R) = \frac{\sigma_{\max}^2(R)}{\sigma_{\min}^2(R)}.$$

ACE, following the example of *ICE*, will seek to maximize (minimize) the ratio

$$\frac{\|b\|_2}{\|x\|_2}$$

so as to estimate $\sigma_{\max}$ ($\sigma_{\min}$) with a constraint on how we construct x and b so as to only require $O(n)$ operations.

In the next section we show in detail how to apply *ACE* to modifications of R. In the following section we comment on how to apply *ACE* to the modification of R^{-1}. It is important to note that we are restricting ourselves to those methods of modification described in §2.

3.1. ACE and the modification of R. We begin by deriving the adaptive condition estimator *ACE* for the particular situation of monitoring the condition of the modified factor R.

Suppose that at time t we have available the pairs of vectors $x_{\max}, b_{\max}$ and $x_{\min}, b_{\min}$ such that

$$R^T x_{\max} = b_{\max}, \quad \|x_{\max}\|_2 = 1, \qquad R^T x_{\min} = b_{\min}, \quad \|x_{\min}\|_2 = 1. \tag{10}$$

Here $\|b_{\max}\|_2$ is a lower bound for $\sigma_{\max}(R)$ and $\|b_{\min}\|_2$ is an upper bound for $\sigma_{\min}(R)$. When the factor R is updated to $\widetilde{R}$, we wish to update the pairs $x_{\max}, b_{\max}$ and $x_{\min}, b_{\min}$ to the pairs $\widetilde{x}_{\max}, \widetilde{b}_{\max}$ and $\widetilde{x}_{\min}, \widetilde{b}_{\min}$ such that

$$\widetilde{R}^T \widetilde{x}_{\max} = \widetilde{b}_{\max}, \quad \|\widetilde{x}_{\max}\|_2 = 1, \qquad \widetilde{R}^T \widetilde{x}_{\min} = \widetilde{b}_{\min}, \quad \|\widetilde{x}_{\min}\|_2 = 1, \tag{11}$$

and so that the values $\|\widetilde{b}_{\max}\|_2$, $\|\widetilde{b}_{\min}\|_2$ now estimate $\sigma_{\max}(\widetilde{R})$ and $\sigma_{\min}(\widetilde{R})$, respectively.

In theory, we choose the pairs $\widetilde{x}_{\max}, \widetilde{b}_{\max}$ and $\widetilde{x}_{\min}, \widetilde{b}_{\min}$ in (11) so that

$$\frac{\|\widetilde{R}^T \widetilde{x}_{\max}\|_2}{\|\widetilde{x}_{\max}\|_2}$$

is maximized and

$$\frac{\|\widetilde{R}^T \widetilde{x}_{\min}\|_2}{\|\widetilde{x}_{\min}\|_2}$$

is minimized, and such that the construction of the $\widetilde{x}$'s and $\widetilde{b}$'s use the previously constructed x's and b's. The approach, given Q_ρ in (9) and an x, b pair, is to compute scalars α and β such that, if

$$\begin{bmatrix} \widetilde{x} \\ \theta \end{bmatrix} = Q_\rho \begin{bmatrix} \alpha x \\ \beta \end{bmatrix},$$

where θ is a scalar, then $\|\widetilde{x}\|_2 = 1$ and the corresponding $\|\widetilde{b}\|_2$ will be large (small) so as to approximate $\sigma_{\max}(\widetilde{R})$ ($\sigma_{\min}(\widetilde{R})$).

More specifically, for the maximization case, we seek α and β, which solve the problem

$$\max_{\|\widetilde{x}\|_2=1} \left\| [R^T, y]\, H \begin{bmatrix} \alpha x_{\max} \\ \beta \end{bmatrix} \right\|_2^2 = \max_{\|\widetilde{x}\|_2=1} \left\| [R^T, y]\, Q_\rho^T H Q_\rho \begin{bmatrix} \alpha x_{\max} \\ \beta \end{bmatrix} \right\|_2^2 \tag{12}$$

$$= \max_{\|\widetilde{x}\|_2=1} \left\| \left[\widetilde{R}^T, 0\right] H \begin{bmatrix} \widetilde{x} \\ \theta \end{bmatrix} \right\|_2^2$$
$$= \max_{\|\widetilde{x}\|_2=1} \left\| \widetilde{R}^T \widetilde{x} \right\|_2^2,$$

for computing the pair $\widetilde{x}_{\max}, \widetilde{b}_{\max}$ (where $\widetilde{b}_{\max}$ is defined as $\widetilde{R}^T \widetilde{x}_{\max}$). Recall that the matrix H from (6) is, along with Q_ρ, a function of ρ. The pair $\widetilde{x}_{\min}, \widetilde{b}_{\min}$ is computed by solving the above optimization problem with max replaced by min throughout.

Note that once the values α and β have been computed, which solve (12), then

$$\widetilde{b}_{\max} = \widetilde{R}^T \widetilde{x}_{\max} = \alpha R^T x_{\max} + \rho\beta y = \alpha b_{\max} + \rho\beta y. \tag{13}$$

Thus we see from (13) that the computation of $\widetilde{b}_{\max}$ ($\widetilde{b}_{\min}$) requires only $O(n)$ operations. Moreover, since the matrix Q_ρ given in (9) is made of a product of Givens or hyperbolic rotations, the computation

$$Q_\rho \begin{bmatrix} \alpha x \\ \beta \end{bmatrix} = \begin{bmatrix} \widetilde{x} \\ \theta \end{bmatrix}$$

is only an $O(n)$ operation. Thus given α and β, $\widetilde{x}_{\max}$ ($\widetilde{x}_{\min}$) can be constructed in $O(n)$ operations. We show next that the computation of α and β only requires $O(n)$ computations, thus establishing that ACE requires only $O(n)$ operations.

We proceed now to solving the optimization problem in (12). We will write $\widetilde{x}$ and $\widetilde{b}$ without the subscripts max and min for ease of exposition.

Observe that

$$\begin{aligned} \|\widetilde{R}^T \widetilde{x}\|_2^2 &= \|\alpha b + \rho\beta y\|_2^2 \\ &= \begin{bmatrix} \alpha & \beta \end{bmatrix} \begin{bmatrix} b^T b & \rho b^T y \\ \rho b^T y & y^T y \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \\ &= \begin{bmatrix} \alpha & \beta \end{bmatrix} K \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \end{aligned} \tag{14}$$

Moreover, note that

$$\|\widetilde{x}\|_2^2 = \alpha^2 + \rho\beta^2 - \rho\theta^2$$

where

$$\theta = \alpha Q_{21} x + Q_{22}\beta.$$

Q_{21} and Q_{22} are, respectively, the row vector and scalar given in (9). Through algebraic simplifications, $\|\widetilde{x}\|_2^2$ can be rewritten as

$$\begin{aligned} \|\widetilde{x}\|_2^2 &= \alpha^2 + \rho\beta^2 - \rho(\alpha Q_{21} x + Q_{22}\beta)^2 \\ &= \begin{bmatrix} \alpha & \beta \end{bmatrix} \begin{bmatrix} 1 - \rho(Q_{21}x)^2 & -\rho Q_{21} x Q_{22} \\ -\rho Q_{21} x Q_{22} & \rho - \rho Q_{22}^2 \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \\ &= \begin{bmatrix} \alpha & \beta \end{bmatrix} M \begin{bmatrix} \alpha \\ \beta \end{bmatrix}. \end{aligned} \tag{15}$$

Thus we can rewrite the optimization problem in (12) as

$$\max_{\|\widetilde{x}\|_2=1} \|\widetilde{R}^T\widetilde{x}\|_2^2 = \max_{\alpha,\beta} \frac{\begin{bmatrix} \alpha & \beta \end{bmatrix} K \begin{bmatrix} \alpha \\ \beta \end{bmatrix}}{\begin{bmatrix} \alpha & \beta \end{bmatrix} M \begin{bmatrix} \alpha \\ \beta \end{bmatrix}},$$

the solution of which can be found by solving the generalized eigenvalue problem

$$K\Phi = \lambda M\Phi \tag{16}$$

where

$$M = \begin{bmatrix} 1-\rho(Q_{21}x)^2 & -\rho Q_{21}xQ_{22} \\ -\rho Q_{21}xQ_{22} & \rho - \rho Q_{22}^2 \end{bmatrix}, \quad K = \begin{bmatrix} b^Tb & \rho b^Ty \\ \rho b^Ty & y^Ty \end{bmatrix}, \quad \text{and} \quad \Phi = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}.$$

Hence, we solve two generalized eigenvalue problems of the form (16). We use the $x_{\max}, b_{\max}$ ($x_{\min}, b_{\min}$) pair and seek the largest (smallest) generalized eigenvalue $\lambda_{\max}$ ($\lambda_{\min}$). The corresponding α's and β's are simply the components of the corresponding generalized eigenvectors.

Note that the solution of a generalized eigenvalue problem of size two is only of $O(1)$ operations, and that the matrices K and M require only inner products for their formation. Moreover, the last row of Q_ρ, given in (9) and consisting of the row vector Q_{21} and the scalar Q_{22}, can be generated in $O(n)$ operations by computing

$$Q_\rho^{1T}Q_\rho^{2T}\cdots Q_\rho^{nT}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} Q_{21}^T \\ Q_{22}^T \end{bmatrix}.$$

Thus the computation of the scalars α and β is an $O(n)$ operation. The condition number of $\widetilde{R}$ is thus estimated by

$$\kappa_2(\widetilde{A}) = \kappa_2^2(\widetilde{R}) \approx \frac{\|\widetilde{b}_{\max}\|_2^2}{\|\widetilde{b}_{\min}\|_2^2} = \frac{\lambda_{\max}}{\lambda_{\min}}.$$

If the matrices M and K in (16) are both positive definite then the generalized eigenvalues are positive. For now we assume that this is the case. We will consider the other cases after presenting the *ACE* algorithms for the modification of R and R^{-1}.

Algorithm *ACE* for modifying R with condition estimation. Given the current factor R, update vector y, $\rho = \pm 1$, the $(x_{\max}, b_{\max})$ and $(x_{\min}, b_{\min})$ pairs with $\|b_{\max}\|_2 \approx \sigma_{\max}(R)$ and $\|b_{\min}\|_2 \approx \sigma_{\min}(R)$, the algorithm computes the updated factor $\widetilde{R}$, the pairs $(\widetilde{x}_{\max}, \widetilde{b}_{\max})$ and $(\widetilde{x}_{\min}, \widetilde{b}_{\min})$, and the estimates $\|\widetilde{b}_{\max}\|_2$, $\|\widetilde{b}_{\min}\|_2$ for $\sigma_{\max}(\widetilde{R})$ and $\sigma_{\min}(\widetilde{R})$.

1. Choose orthogonal ($\rho = 1$) or hyperbolic ($\rho = -1$) plane rotations Q_ρ^i, rotating the ith row into the $(n+1)$st row, to form

$$Q_\rho^n Q_\rho^{n-1}\cdots Q_\rho^1\begin{bmatrix} R \\ y^T \end{bmatrix} = \begin{bmatrix} \widetilde{R} \\ 0^T \end{bmatrix},$$

by reducing y^T to 0^T from the left to right preserving the upper triangular form of R in $\widetilde{R}$.

2. If $y = \gamma b_{\max}$ or $y = \gamma b_{\min}$, for some γ, then proceed as described in §4, otherwise proceed to the next step.
3. Compute $\begin{bmatrix} Q_{21} & Q_{22} \end{bmatrix}$, given by (9), by forming

$$\begin{bmatrix} Q_{21}^T \\ Q_{22}^T \end{bmatrix} = Q_\rho^{1^T} Q_\rho^{2^T} \cdots Q_\rho^{n^T} \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

4. Compute the generalized eigenvalues $\lambda_{\max}, \lambda_{\min}$ and corresponding generalized eigenvectors of (16),

$$\begin{bmatrix} \alpha_{\max} \\ \beta_{\max} \end{bmatrix} \quad \text{and} \quad \begin{bmatrix} \alpha_{\min} \\ \beta_{\min} \end{bmatrix},$$

by solving the two corresponding generalized eigenvalue problems. Scale the generalized eigenvectors so that the expression in (15) is equal to 1 (this avoids having to normalize the $\widetilde{x}$'s).
5. Form $\widetilde{x}_{\max}$ and $\widetilde{x}_{\min}$ by computing

$$\begin{bmatrix} \widetilde{x}_{\max} & \widetilde{x}_{\min} \\ \theta_{\max} & \theta_{\min} \end{bmatrix} = Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 \begin{bmatrix} \alpha_{\max} x_{\max} & \alpha_{\min} x_{\min} \\ \beta_{\max} & \beta_{\min} \end{bmatrix}.$$

6. Compute

$$\widetilde{b}_{\max} = \alpha_{\max} b_{\max} + \rho \beta_{\max} y, \qquad \widetilde{b}_{\min} = \alpha_{\min} b_{\min} + \rho \beta_{\min} y.$$

7. Take as estimates

$$\sigma_{\min}^2\left(\widetilde{R}\right) \approx \lambda_{\min} = \left\|\widetilde{b}_{\min}\right\|_2^2, \qquad \sigma_{\max}^2\left(\widetilde{R}\right) \approx \lambda_{\max} = \left\|\widetilde{b}_{\max}\right\|_2^2,$$
$$\kappa_2\left(\widetilde{A}\right) = \kappa_2^2\left(\widetilde{R}\right) \approx \frac{\lambda_{\max}}{\lambda_{\min}}.$$

8. Replace

$$[R, x_{\max}, x_{\min}, b_{\max}, b_{\min}] \Leftarrow \left[\widetilde{R}, \widetilde{x}_{\max}, \widetilde{x}_{\min}, \widetilde{b}_{\max}, \widetilde{b}_{\min}\right],$$

input the new update vector y^T and ρ, and return to Step 1.

3.2. ACE and the modification of R^{-1}. In order to track the condition of the matrix R^{-1}, it is important to make use of the fact that the transformation matrices used in modifying R^{-1} (actually the transformations are applied to R^{-T}, but only to conform to the convention of multiplying on the left by transformation matrices) are the same as those used in modifying R. Thus, *ACE* can be applied by using the same information as was available for the application of *ACE* in modifying R.

Algorithm *ACE* for modifying R^{-1} with condition estimation. Given the current factor R^{-1}, update vector y, $\rho = \pm 1$, the $(x_{\max}, b_{\max})$ and $(x_{\min}, b_{\min})$ pairs with $\|b_{\max}\|_2 \approx \sigma_{\max}(R)$ and $\|b_{\min}\|_2 \approx \sigma_{\min}(R)$, the algorithm computes the updated factor $\widetilde{R}^{-1}$, the pairs $(\widetilde{x}_{\max}, \widetilde{b}_{\max})$ and $(\widetilde{x}_{\min}, \widetilde{b}_{\min})$, and the estimates $\|\widetilde{b}_{\max}\|_2$, $\|\widetilde{b}_{\min}\|_2$ for $\sigma_{\max}(\widetilde{R})$ and $\sigma_{\min}(\widetilde{R})$.

1. Compute the vector a below by the indicated matrix-vector multiplication

$$a = R^{-T} y.$$

2. If $y = \gamma b$, for some γ, then proceed as described in §4, otherwise proceed to the next step.
3. Choose orthogonal or hyperbolic plane rotations Q_ρ^i, rotating the ith row into the $(n+1)$st row, to form

$$Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 \begin{bmatrix} -\rho a \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ \delta \end{bmatrix}.$$

4. Apply these transformations to compute $\widetilde{R}^{-T}$ as follows:

$$\begin{bmatrix} \widetilde{R}^{-T} \\ \star \end{bmatrix} = Q_\rho^n Q_\rho^{n-1} \cdots Q_\rho^1 \begin{bmatrix} R^{-T} \\ 0^T \end{bmatrix}.$$

5. Perform steps 4 through 7 of the *ACE* algorithm for modifying R.
6. Replace

$$\left[R^{-1}, x_{\max}, x_{\min}, b_{\max}, b_{\min}\right] \Leftarrow \left[\widetilde{R}^{-1}, \widetilde{x}_{\max}, \widetilde{x}_{\min}, \widetilde{b}_{\max}, \widetilde{b}_{\min}\right],$$

input the new update vector y^T and ρ, and return to Step 1.

Thus in summary, we see that the condition of the factor R or R^{-1} can be monitored during the modification process for only $O(n)$ additional operations. In Table 1 we make a minor comparison between the condition estimators *ACE*, *ALE* [15], and *ICE* [6] in terms of operation counts (we have only counted multiplications, divisions, and square roots), so as to aid the reader in discerning the differences between these condition estimators.

TABLE 1
Operation counts for three condition estimators.

Method	Operations	Mode
ACE	$21n$ for R, $19n$ for R^{-1}	Adaptive
ALE	$4n^2 + O(n)$	Adaptive
ICE	$n^2 + O(n)$	Incremental

Note that in the application of *ACE* we test for the occurrence of the update vector y being a scalar multiple of either vector $b_{\max}$ or $b_{\min}$. Such an instance will result in a singular generalized eigenvalue problem. This situation can be quite effectively handled and the procedure for doing this is described in the next section.

4. Breakdown of ACE and how to fix it. In this section we examine the particular instance when the *ACE* method can theoretically break down. That is when the generalized eigenvalue problem

$$Kx = \lambda Mx$$

has a noninterpretable solution, such as complex λ's, infinite λ's, or when the generalized eigenvalue problem is singular (i.e., there exists an $x \neq 0$ such that $Kx = \lambda Mx$ for all λ's). In this section we show that the only real possibility is that K and M can be of rank one and share a null vector. This is remedied by deflating the generalized eigenvalue problem by restricting K and M to those subspaces complementary to the shared nullspace. Shroff and Bischof in [29] may have unknowingly proposed using

the solution of the deflated generalized eigenvalue problem. Our analysis sheds light on the effectiveness of their heuristics as well as indicating the impossibility of infinite generalized eigenvalues and clarifies their characterization of "bad" eigenvalues. We will present these results as a series of lemmas, in all of which we assume that

$$b = R^{-T}x, \qquad \|x\|_2 = 1, \qquad a = R^{-T}y, \qquad \delta = \sqrt{1 + \rho a^T a}.$$

Recall the definitions of the matrices K and M from (14) and (15), respectively:

$$K = \begin{bmatrix} b^T b & \rho b^T y \\ \rho b^T y & y^T y \end{bmatrix} \quad \text{and} \quad M = \begin{bmatrix} 1 - \rho(Q_{21}x)^2 & -\rho Q_{21} x Q_{22} \\ -\rho Q_{21} x Q_{22} & \rho - \rho Q_{22}^2 \end{bmatrix}.$$

LEMMA 4.1. *The following are equivalent:*

1. *K is singular.*
2. *K is a rank-one matrix.*
3. *$y = \pm\|a\|_2 b$.*

Proof. If K is singular, then since $b \neq 0$, the matrix K must be of rank one. If K is of rank one, we observe that since

$$K = \begin{bmatrix} b^T \\ \rho y^T \end{bmatrix} \begin{bmatrix} b & \rho y \end{bmatrix},$$

it follows that there exists a γ such that

$$y = \gamma b.$$

However,

$$a = R^{-T}y = \gamma R^{-T}b = \gamma x,$$

thus, since $\|x\|_2 = 1$,

$$\gamma = \pm\|a\|_2 = a^T x.$$

To conclude the proof, note that if $y = \pm\|a\|_2 b$, then K must be singular. □

LEMMA 4.2. *K is singular if and only if M is singular.*

Proof.

$$\begin{aligned} K \quad \text{is singular} &\iff y = \pm\|a\|_2 b \\ &\iff a = \pm\|a\|_2 x \\ &\iff Q_{21}^T = \mp\frac{\|a\|_2 x}{\delta} \\ &\iff x = \mp\frac{Q_{21}^T}{\|Q_{21}^T\|_2}. \end{aligned}$$

Now note that M is singular if and only if

$$(1 - \rho(Q_{21}x)^2)(\rho - \rho Q_{22}^2) - (Q_{21}x)^2(Q_{22})^2 = 0,$$

which is equivalent to

$$(Q_{21}x)^2 = \|Q_{21}\|_2^2.$$

Thus, by the Cauchy–Schwarz inequality, and the fact that $\|x\|_2 = 1$, we have established the lemma. □

LEMMA 4.3. *The null space of K, $\mathcal{N}(K)$, is the same as the null space of M, $\mathcal{N}(M)$.*

Proof. We have already established that M is singular if and only if K is singular. Thus when either K or M is nonsingular the other is as well, and their common null space is $\{0\}$.

Assume now that K and M are singular. Since

$$Q_{21}Q_{21}^T + \rho Q_{22}^2 = \rho,$$

Q_{21} and Q_{22} cannot both be zero, hence $M \neq 0$. (Note that $M = 0 \Rightarrow Q_{22}^2 = 1 \Rightarrow Q_{21} = 0 \Rightarrow$ the (1,1) entry of M is 1—a contradiction.) Therefore M is of rank one and the nullities of K and M agree. Furthermore,

$$K \begin{bmatrix} \rho\|a\|_2 \\ \mp 1 \end{bmatrix} = M \begin{bmatrix} \rho\|a\|_2 \\ \mp 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix},$$

where in the matrix K the vector y is given by $y = \pm\|a\|_2 b$. The rightmost relation is seen by noting that $Q_{21}x = \mp\|a\|_2/\delta$. Hence

$$\mathcal{N}(K) = \mathcal{N}(M) = \operatorname{span}\left\{ \begin{bmatrix} \rho\|a\|_2 \\ \mp 1 \end{bmatrix} \right\},$$

completing the proof. □

Then since ACE can only break down if the generalized eigenvalue problem has infinite eigenvalues or is singular, and since K and M are either both positive definite or share a common null space, we have the following result.

THEOREM 4.4. *ACE can break down if and only if the update vector y satisfies*

$$y = \pm\|R^{-T}y\|_2 b.$$

We now examine the heuristic "fixes" proposed by Shroff and Bischof in light of the understanding now gained on when ACE can break down. There are two heuristics proposed in [29]. The first recommends discarding those eigenvalue, eigenvector pairs (λ, Φ) for which $\Phi^T M \Phi$ is smaller than some threshold and using the remaining eigenvalue, eigenvector pair (i.e., the pair in the deflated problem). It should be noted though that from our analysis, only those pairs for which *both* $\Phi^T M \Phi$ and $\Phi^T K \Phi$ are small should be discarded. We propose using the value of the cosine of the angle ν between the vectors y and b (i.e., $\cos(\nu) = (y^T b)^2/(b^T b y^T y)$) as an indicator of when a singular (or nearly singular by using a threshold) generalized eigenvalue problem will arise. (Note that the quantities necessary for this test are also required to form K.) Thus one can solve the deflated problem immediately, thereby avoiding the cost of forming and solving the generalized eigenvalue problem, the results of which can be contaminated in the singular case. Note that by simultaneously diagonalizing K and M in (16) by the matrix

$$\begin{bmatrix} \rho\|a\|_2 & \pm 1 \\ \mp 1 & \rho\|a\|_2 \end{bmatrix},$$

one finds the finite generalized eigenvalue of the deflated problem is given by

$$\lambda = \|b\|_2^2(1 + \rho\|a\|_2^2) = b^T b \delta^2$$

and the corresponding generalized eigenvector is

$$\Phi = \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} \pm\delta/\mu \\ \rho\|a\|_2\delta/\mu \end{bmatrix}, \quad \text{where} \qquad \mu = \|a\|_2^2 + 1.$$

One can also show that

$$\tilde{b} = \delta b.$$

Hence this choice will always increase or decrease (depending on the value of ρ) the estimates $\lambda_{\max}$ and $\lambda_{\min}$. We call the use of this heuristic the first "fix."

The other heuristic "fix" suggested by Shroff and Bischof requires the use of the condition estimator *ICE* and results in the modification of the b vector, which removes the scalar dependency between b and y. It is possible, however, that *ICE* will not remove this dependency, and will only reproduce the same vector b. We think that this would be an extremely rare occurence and if it did occur one could always use an alternative condition estimator for the single step, which requires $O(n^2)$ operations such as the Cline, Conn, and Van Loan estimator described in [11]. In our computations, we have found that the first "fix" described above works quite adequately. These numerical experiments and others describing the reliability and accuracy of the estimator are contained in the following section.

5. Numerical experiments. In this section we report on some selected experiments designed to examine the performance of the adaptive condition estimation scheme *ACE*. We give only a few examples to illustrate the effectiveness of *ACE*, since similar results on different condition estimation problems have been reported elsewhere [14], [15], [26], [29] that indicate the reliability of *ACE*. It should be noted, though, that the $O(n^2)$ scheme *ALE* in [15] does track the condition number better than both *ICE* and *ACE*. It was also displayed in [15] that *ACE* reacted a bit slower to large changes in the condition number than either *ICE* or *ALE*, but still responded and tracked well, never off by more than a factor of three. Moreover, for most signal processing data computations, *ACE* tracked better than *ICE*.

In the experiments presented here, we update the Cholesky factor R in the computations and track the condition number of R over time; consideration of the inverse factor R^{-1} produces similar results. Reports on our experiments are given in Figs. 1–5. Here we report on the performance of experiments using random updates, with well- and ill-conditioned data, random updates with jumps in the condition number, and the effectiveness of our "fix" for handling the singular generalized eigenvalue problem. The last figure presented is a histogram summarizing the results of performing 50 random updates on 70 random matrices, for a total of 3500 updates or downdates.

All experiments were performed using the PRO-MATLAB system [22] on our Sun and Mac II workstations. The machine epsilon for PRO-MATLAB on these systems is approximately 2.2204×10^{-16}.

For each of the first four experiments we used 20×20 matrices with 100 downdates/updates. Except for the experiment with a widely varying condition number and the summary, the initial factor R was taken to be the identity. This choice, or a scalar multiple of the identity, is a quite common starting factor for many applications. For example, in secant update methods for unconstrained optimization problems, the initial Hessian approximation H_0 is most often chosen to be either the identity matrix I, or a scalar multiple of I [13]. It should also be noted that in general one can not use *any* downdate. That is, some downdates would cause the matrix to become indefinite.

In such circumstances we simply skip the downdate (this is a common practice in optimization where downdating the Hessian H_0 in, for example, the Symmetric Rank One (SR1) secant method, can force the H_0 to become indefinite [13], [30]). *ACE* has been shown by Ferng [14] to be useful in determining when the Hessian is about to become indefinite in some unconstrained optimization problems.

We note that in general *ACE* is somewhat sensitive to the choice of the initial starting vectors for $x_{\max}$ and $x_{\min}$. In fact, if a very poor starting vector for $x_{\max}$ or $x_{\min}$ is used it may take many update iterations before *ACE* produces estimates that have a ratio less than two. In any case, we recommend care in choosing the initial vectors.

In the first tests we choose a probability vector to initialize *ACE* (each component of the starting vectors for $x_{\max}$ and $x_{\min}$ is $1/\sqrt{n}$). We see from Fig. 1 that *ACE* is quite successful in tracking the condition of the random problem. Note that the ratio of the true condition number to the estimated condition number never exceeds two. One might assume that if the problem were more ill conditioned, then *ACE* would not track the condition numbers as well; however, from Fig. 2 we see that this is not the case at all. Again, *ACE* is quite successful in tracking the condition numbers within a factor of two.

In the next set of tests, illustrated in Fig. 3, we used update vectors that cause large changes in the condition number. In order to have a more ill conditioned matrix to start off with, the matrix R was chosen to be the diagonal matrix having the diagonal entries $10n, 10(n-1), \cdots, 20, 2$, so that the initial condition number is 100. The initial vectors $x_{\max}$ and $x_{\min}$, in this case, were chosen to be near the extreme singular vectors of R. One can see, even as changes in the condition number that exceed an order of magnitude are performed, that *ACE* effectively tracks the condition number resulting in an accuracy ratio always less than two.

The fourth test illustrates how effective the estimate is even when K and M are singular. Here, at every fifth update or downdate, the update vector y is chosen to be a multiple of the vector b. Poor initial vectors for $x_{\max}$ and $x_{\min}$ are used to force the initial singular value estimates away from the actual singular values. Again, as presented in Fig. 4, *ACE* performs undauntingly as it continues to estimate the condition numbers effectively.

For the last test we performed 50 updates/downdates to 70 random upper triangular matrices of orders 10 through 70. Figure 5 indicates the frequency of ratios Actual/Estimate, as well as the distribution of the actual condition numbers for the matrices associated with each of the 3500 updates. *ICE* was run on the initial triangular matrix to generate the starting vectors. Note that there are a few instances of the ratio exceeding 10. All of these instances occurred at the first of the 50 updates and can be attributed to a poor starting vector.

The results reported here are typical of the performance of the adaptive condition estimation scheme *ACE* on a variety of symmetric rank-one recursive matrix modification problems. Other numerical experiments with *ACE* have been reported in [15], [26] for problems arising in signal processing and recursive least squares, and experiments involving unconstrained optimization problems have been reported in [14]. These experiments also show *ACE* to be quite effective.

6. Concluding remarks. We have introduced a technique, *ACE*, that provides a fast, i.e., $O(n)$, adaptive condition estimator for recursive symmetric rank-one updates. Our numerical tests indicate that *ACE* is accurate and, with the modification scheme suggested in [29], very robust. Similar results for *ACE* on different condition

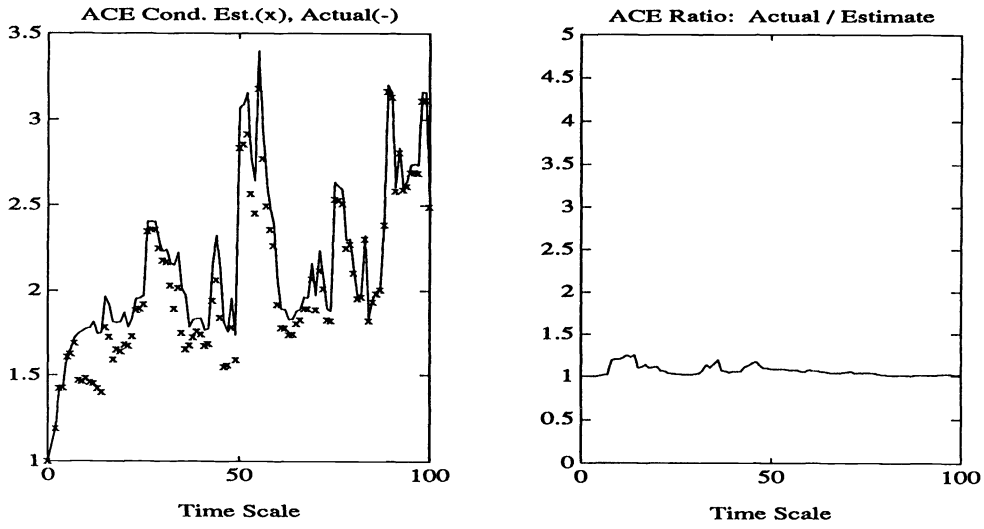


FIG. 1. *Performance of ACE with random updates and downdates on random data.*

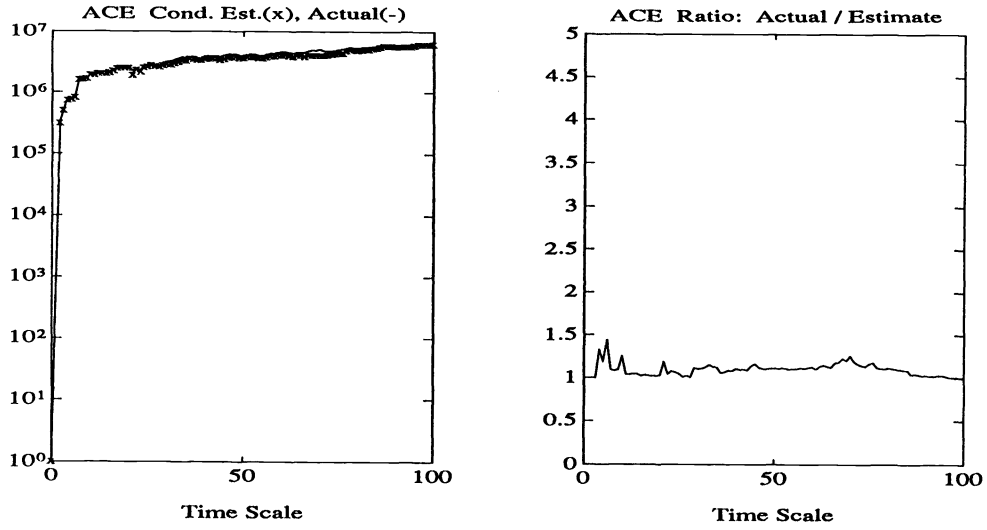


FIG. 2. *Performance of ACE with random updates and downdates on ill-conditioned data.*

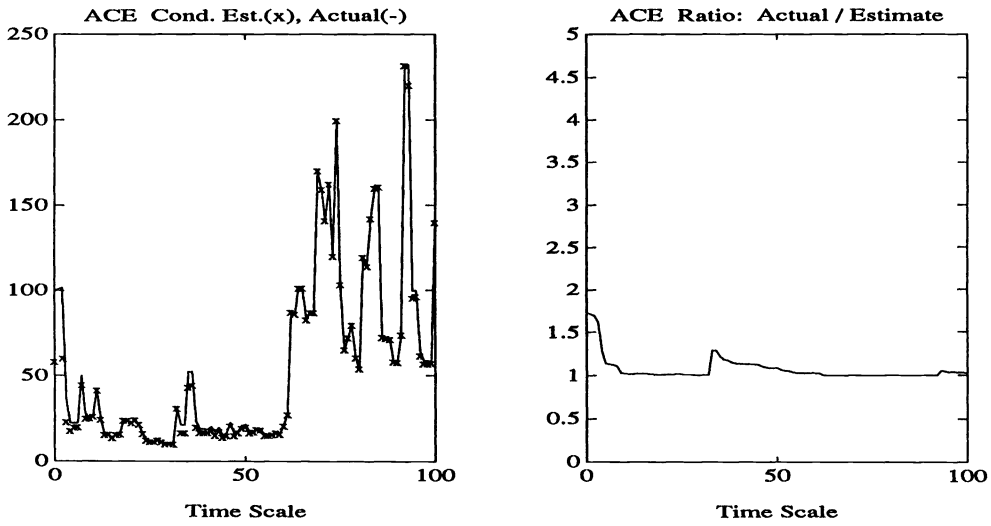


FIG. 3. *Performance of ACE with random updates and downdates and jumps in conditioning of data.*

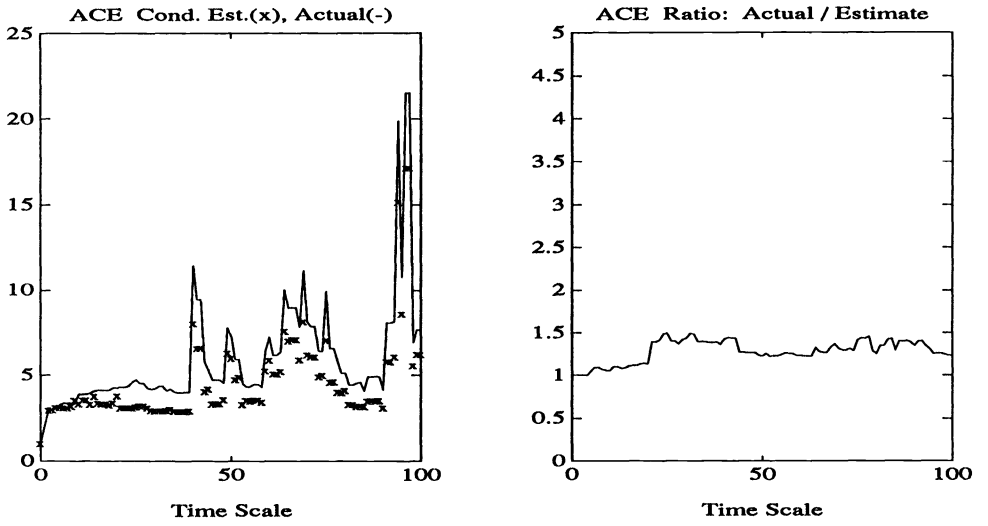


FIG. 4. *Performance of ACE with recovery scheme with random updates and downdates on data designed to force breakdowns.*

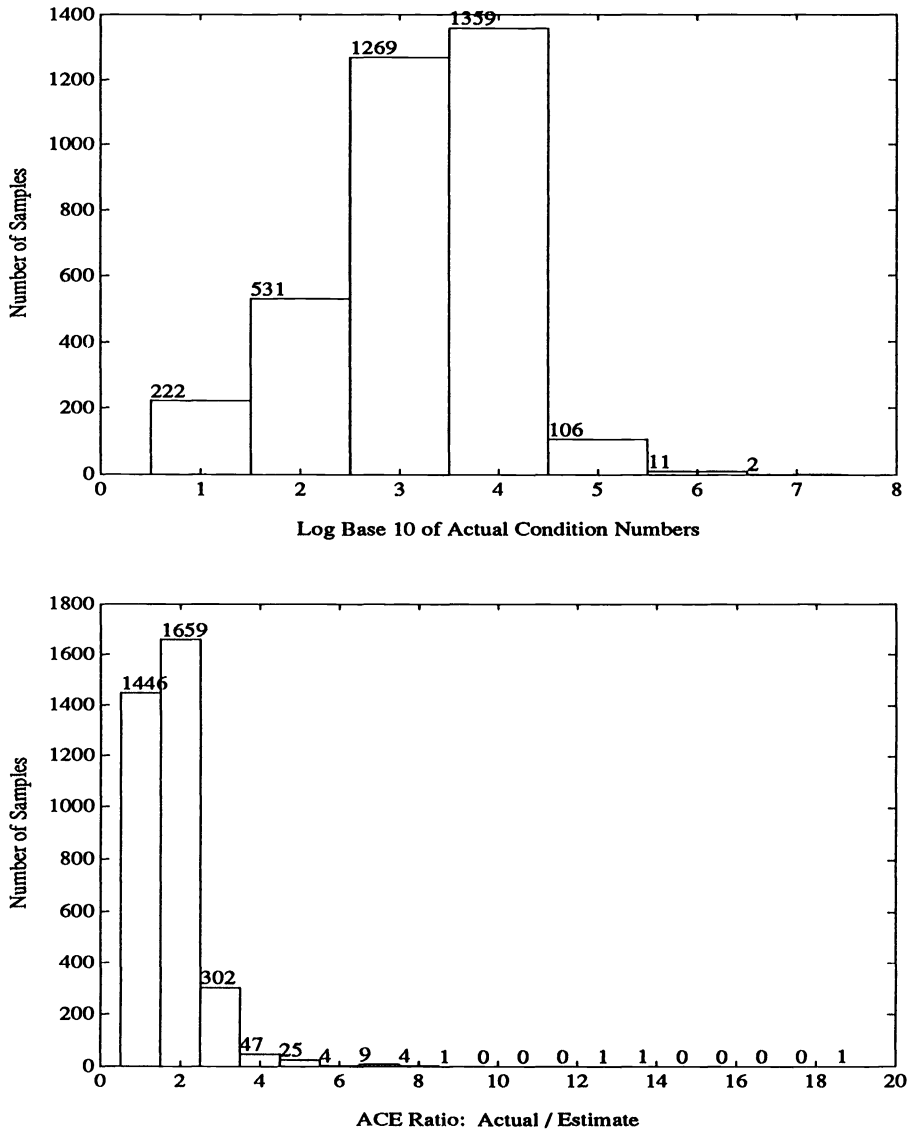


FIG. 5. *Performance of ACE on* 3500 *test cases.*

estimation problems have been reported elsewhere [14], [15], [26], [29]. A preliminary discussion of *ACE* along with some applications to recursive least squares methods in signal processing has been given in [26]. Comparisons of *ACE* with other recent condition estimation schemes for recursive least squares are given in [15].

Several extentions of this work are possible. For instance, rank $k > 1$ updates can also be handled by replacing the vector y with an $n \times k$ matrix of updates. In particular, if two vectors y and z are to be updates to the least squares process, the n-vector y is replaced by the $n \times 2$ matrix $[\ y \quad z\ ]$ and a is replaced by $R^{-T}[\ y \quad z\ ]$ in §3. The 2×2 matrices involved in the optimization formulas thus become 3×3. The algorithms are modified accordingly. Thus *ACE* may find applications in Hessian rank-two updating schemes such as BFGS in optimization [13], [14]. More generally, *ACE* can easily be incorporated into the rank-two matrix modification schemes described by Bartels and Kaufman [4]. Nonsymmetric rank-one updates as discussed in [29] can also be handled by our schemes. This is accomplished by first transforming the nonsymmetric update into a symmetric rank-two update problem (see, e.g., [13]) for the implicit normal equations. *ACE* is then applied to monitor the condition numbers of the recursive rank-two modifications. In fact, it can be shown that this approach has a smaller operation count than that of the scheme *GRACE* in [29], although the matrix modification schemes themselves may be less stable due to the squaring of the condition number in implicitly forming the normal equations.

It may also be possible to effectively utilize *ACE* as a type of incremental condition estimator for orthogonal factorization schemes. After an initial triangular factor candidate $\widehat{R}$ for an observation matrix X is determined and appropriate singular vectors $x_{\min}$, $x_{\max}$ computed (say, by using *ICE* [6]), then *ACE* could be used to monitor the conditioning as the remaining rows are reduced to zero by Givens rotations in order to obtain the Cholesky factor R. The use of *ACE* may be practical, for instance, when parallel pipelined Givens rotations are employed in orthogonal factorization.

Another important application of *ACE* is in sparse matrix least squares problems, where dense rows are held out to the end of the computation and incorporated into the solution process by applying the Sherman–Morrison formula. By use of this technique for handling dense rows, *ACE* fits naturally into updating the condition estimate that may have been determined initially in the sparse orthogonal decomposition phase by the use of the sparse incremental condition estimator, *SPICE*, due to Bischof, Lewis, and Pierce [8].

REFERENCES

[1] S. T. Alexander, *Adaptive Signal Processing*, Springer-Verlag, New York, 1986.

[2] S. T. Alexander, C.-T. Pan, and R. J. Plemmons, *Analysis of a recursive least squares hyperbolic rotation scheme for signal processing*, Linear Algebra Appl. Special Issue on Electrical Engrg., 98 (1988), pp. 3–40.

[3] K. J. Åström and B. Wittenmark, *Adaptive Control*, Addison–Wesley, Reading, MA, 1989.

[4] R. Bartels and L. Kaufman, *Cholesky factor updating techniques for rank-2 matrix modifications*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 557–592.

[5] G. J. Bierman, *Factorization Methods for Discrete Sequential Estimation*, Academic Press, New York, 1977.

[6] C. H. Bischof, *Incremental condition estimation*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 312–322.

[7] C. H. Bischof and G. M. Shroff, *On updating signal subspaces*, Tech. Report, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, 1989.

[8] C. H. Bischof, J. Lewis, and D. J. Pierce, *Incremental condition estimation for sparse matrices*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 644–659.

[9] Å. BJÖRCK, *Least squares methods*, in Handbook of Numerical Analysis, P. Ciarlet and J. Lions, eds., Elsevier/North Holland, Amsterdam, the Netherlands, 1989.

[10] ———, *Accurate downdating of least squares solutions*, paper presented at the SIAM Annual Meeting, San Diego, CA, 1989.

[11] A. K. CLINE, A. R. CONN, AND C. F. VAN LOAN, *Generalizing the Linpack condition number estimator*, Lecture Notes in Mathematics 909, Springer-Verlag, Berlin, New York, 1982, pp. 73–83.

[12] P. COMON AND G. H. GOLUB, *Tracking a few extreme singular values and vectors in signal processing*, Tech. Report NA-89-01, Department of Computer Science, Stanford University, Stanford, CA, 1989.

[13] J. E. DENNIS AND R. B. SCHNABEL, *Quasi-Newton Methods for Nonlinear Problems*, Prentice–Hall, Englewood Cliffs, NJ, 1983.

[14] R. FERNG, *Secant update methods for unconstrained minimization problems*, preprint, 1990.

[15] R. FERNG, G. H. GOLUB AND R. J. PLEMMONS, *Adaptive Lanczos methods for recursive condition estimation*, summary in Proc. SPIE Conference on Advanced Algorithms and Architectures for Signal Processing, San Diego, 1990; Math. Algorithms, to appear.

[16] P. E. GILL, G. H. GOLUB, W. MURRAY AND M. A. SAUNDERS, *Methods for modifying matrix factorizations*, Math. Comp., 28 (1974), pp. 505–535.

[17] G. H. GOLUB, *Matrix decompositions and statistical calculations*, in Statistical Computation, R. C. Milton and J. A. Nelder, eds., Academic Press, New York, 1969, pp. 365–395.

[18] G. H. GOLUB AND C. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[19] S. HAYKIN, *Adaptive Filter Theory*, Prentice–Hall, Englewood Cliffs, NJ, 1986.

[20] C. S. HENKEL AND R. J. PLEMMONS, *Recursive least squares computations on a hypercube multiprocessor using the covariance factorization*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 95–106.

[21] N. J. HIGHAM, *A survey of condition estimators for triangular matrices*, SIAM Rev., 29 (1987), pp. 575–596.

[22] C. MOLER, J. LITTLE, AND S. BANGERT, *PRO-MATLAB User's Guide*, The Mathworks, Sherborn, MA, 1987.

[23] M. MOONEN, P. VAN DOOREN, AND J. VANDEWALLE, *An SVD updating algorithm for subspace tracking*, preprint, 1989.

[24] M. MORF AND T. KAILATH, *Square root algorithms and least squares estimation*, IEEE Trans. Automat. Control, 20 (1975), pp. 487–497.

[25] C.-T. PAN AND R. J. PLEMMONS, *Least squares modifications with inverse factorizations: Parallel implications*, Computational and Applied Math., 27 (1989), pp. 109–127.

[26] D. J. PIERCE AND R. J. PLEMMONS, *Tracking the condition number for RLS in signal processing*, preprint, 1990; Math. Control Signals Systems, to appear.

[27] R. J. PLEMMONS, *Recursive least squares computations*, in Proc. Internat. Conference on Mathematics in Networks and Systems, 3, Birkhäuser, Boston, 1990, pp. 495-502.

[28] S. SASTRY AND M. BODSON, *Adaptive Control: Stability, Convergence, and Robustness*, Prentice–Hall, Englewood Cliffs, NJ, 1989.

[29] G. M. SHROFF AND C. H. BISCHOF, *Adaptive condition estimation for rank-one updates of* QR *factorizations*, preprint, 1990; SIAM J. Matrix Anal. Appl., 13 (1992), to appear.

[30] E. SPEDICATO, *Computational experience with symmetric rank-one positive definite quasi-Newton algorithms*, Optimization, 16 (1985), pp. 673–681.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 292–311, January 1992

021

PRECONDITIONERS FOR INDEFINITE SYSTEMS ARISING IN OPTIMIZATION*

PHILIP E. GILL†, WALTER MURRAY‡, DULCE B. PONCELEÓN§, AND MICHAEL A. SAUNDERS‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Methods are discussed for the solution of sparse linear equations $Ky = z$, where K is symmetric and indefinite. Since exact solutions are not always required, direct and iterative methods are both of interest. An important *direct* method is the Bunch–Parlett factorization $K = U^TDU$, where U is triangular and D is block-diagonal. A sparse implementation exists in the form of the Harwell code MA27. An appropriate *iterative* method is the conjugate-gradient–like algorithm SYMMLQ, which solves indefinite systems with the aid of a positive-definite preconditioner.

For any indefinite matrix K, it is shown that the U^TDU factorization can be modified at nominal cost to provide an "exact" preconditioner for SYMMLQ. Code is given for overwriting the block-diagonal matrix D produced by MA27.

The KKT systems arising in barrier methods for linear and nonlinear programming are studied, and preconditioners for use with SYMMLQ are derived.

For nonlinear programs a preconditioner is derived from the "smaller" KKT system associated with variables that are not near a bound. For linear programs several preconditioners are proposed, based on a square nonsingular matrix B that is analogous to the basis matrix in the simplex method. The aim is to facilitate solution of full KKT systems rather than equations of the form $AD^2A^T\Delta\pi = r$ when the latter become excessively ill conditioned.

Key words. indefinite systems, preconditioners, linear programming, nonlinear programming, numerical optimization, barrier methods, interior-point methods

AMS(MOS) subject classifications. 65F05, 65F10, 65F50, 65K05, 90C05

1. Introduction.

Symmetric indefinite systems of linear equations arise in many areas of scientific computation. We will discuss the solution of *sparse* indefinite systems $Ky = z$ by direct and iterative means.

The direct method we have in mind is the Bunch–Parlett factorization $K = U^TDU$, where U is triangular and D is block-diagonal with blocks of dimension 1 or 2 that may be indefinite. Such a factorization exists for any symmetric matrix K [BP71]. (We shall refer to it as the Bunch–Parlett factorization, while noting that the Bunch–Kaufman pivoting strategy is preferred in practice [BK77]. The principal sparse implementation to date is due to Duff and Reid [DR82], [DR83] in the Harwell code MA27. See also [DGRST89].)

The iterative method to be discussed is the Paige–Saunders algorithm SYMMLQ [PS75]. This is a conjugate-gradient–like method for indefinite systems that can make use of a positive-definite preconditioner.

1.1. Preconditioning indefinite systems.

One of our aims is to present a new and simple result that shows how to use the Bunch–Parlett factorization of an

* Received by the editors January 4, 1991; accepted for publication (in revised form) July 26, 1991. This paper was presented at the Second Asilomar Workshop on Progress in Mathematical Programming, February 1990. This research was supported in part by National Science Foundation grant DDM-8715153 and Office of Naval Research grant N00014-90-J-1242.

† Department of Mathematics, University of California at San Diego, La Jolla, California 92093 (peg@optimal.ucsd.edu).

‡ Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California 94305-4022 (mike@sol-michael.stanford.edu and walter@sol-walter.stanford.edu).

§ Apple Computer, Inc., 20525 Mariani Avenue, Cupertino, California 95014 (dulce@apple.com).

indefinite matrix to construct an exact preconditioner for an iterative method such as SYMMLQ. The intended use is as follows.

Given an indefinite system $Ky = z$ and a related indefinite matrix $\tilde{K}$, we expect that the Bunch–Parlett factorization $\tilde{K} = U^T D U$ will be computed, or will already be available. We show that D can be changed cheaply to provide a positive-definite matrix $M = U^T \bar{D} U$, such that SYMMLQ (with preconditioner M) will solve $\tilde{K}y = z$ in at most two iterations. Hence, M should be a good preconditioner for the original system involving K.

1.2. Optimization. As a source of indefinite systems, we are interested in *barrier methods* or *interior-point methods* for solving linear and nonlinear programs in the following standard form:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & c^T x \\ \text{subject to} & Ax = b, \qquad l \le x \le u, \end{array} \tag{1}$$

where $A \in \Re^{m \times n}$, and

$$\begin{array}{ll} \underset{x}{\text{minimize}} & F(x) \\ \text{subject to} & c(x) = 0, \qquad l \le x \le u, \end{array} \tag{2}$$

where $F(x)$ and $c(x)$ have continuous first and second derivatives. We assume that an optimal solution (x^*, π^*) exists, where π^* is a set of Lagrange multipliers for the constraints $Ax = b$ or $c(x) = 0$.

1.3. KKT systems. When barrier or interior-point methods are applied to these optimization problems, the Karush–Kuhn–Tucker optimality conditions lead to a set of equations of the form

$$\begin{pmatrix} H & A^T \\ A & \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta \pi \end{pmatrix} = \begin{pmatrix} -g \\ r \end{pmatrix}, \qquad K \equiv \begin{pmatrix} H & A^T \\ A & \end{pmatrix}, \tag{3}$$

whose solution usually dominates the total computation. The vectors Δx and $\Delta \pi$ are used to update the estimates of x^* and π^*.

For quadratic programs or general nonlinear programs, H is typically a general sparse matrix like A, and it is natural to solve the KKT system as it stands. The Harwell code MA27 has been used in this context by several authors, including Gill et al. [GMSTW86] and Turner [Tur87], [Tur90] for sparse linear programs, by Ponceleón [Pon90] for sparse linear and quadratic programs, and by Burchett [Bur88] for some large nonlinear programs arising in the electric power industry.

1.4. Avoiding AD^2A^T. If H is known to be nonsingular, it is common practice to use it as a block pivot and solve (3) according to the *range-space equations* of optimization:

$$AH^{-1}A^T \Delta \pi = AH^{-1}g + r, \qquad H\Delta x = A^T \Delta \pi - g.$$

For linear programs this is particularly attractive, since H is then a positive diagonal matrix. For example, in a typical primal barrier method, $H = \mu D^{-2}$ where D is diagonal and μ is the barrier parameter ($\mu > 0$) [GMSTW86]. The range-space equations reduce to

$$AD^2A^T \Delta \pi = AD^2 g + \mu r, \qquad \Delta x = \frac{1}{\mu} D^2 (A^T \Delta \pi - g), \tag{4}$$

and most of the work lies in solving the system involving AD^2A^T. When $r = 0$, the numerical properties may be improved by noting that the equation for $\Delta\pi$ reduces to the least-squares problem

$$\min_{\Delta\pi} \|Dg - DA^T\Delta\pi\|_2. \tag{5}$$

However, it is important to observe that *the range-space equations may not give a stable method for solving the* KKT *system if H is ill conditioned.*

1.5. Example. Let

$$A = \begin{pmatrix} 1 & 1 & & \\ 1 & & 1 & \\ 1 & & & 1 \end{pmatrix}, \qquad H = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & \mu & \\ & & & \mu \end{pmatrix}, \qquad x = \begin{pmatrix} \sqrt{\mu} \\ \sqrt{\mu} \\ 1 \\ 1 \end{pmatrix},$$

where $\mu \ll 1$, and consider the KKT system (3). This would arise when a primal barrier method is applied to a 3×4 LP problem (1) having $l = 0$, $u = \infty$, when x is the current estimate of x^* and μ is the current barrier parameter. Thus $H = \mu D^{-2}$, where $D = \operatorname{diag}(x_j)$.

The condition numbers of interest are $\operatorname{cond}(K) \approx 6$ (independent of μ) and $\operatorname{cond}(AH^{-1}A^T) = \operatorname{cond}(AD^2A^T) \approx 0.5/\mu$.[1] The latter becomes increasingly large as a solution is approached ($\mu \to 0$), even though K and the original linear program are very well conditioned.

Similar examples are easily constructed. (Indeed, K can be well conditioned even if H is singular.) Thus, we advocate direct or iterative solution of the full KKT system (3) even for linear programs, rather than (4) or (5) according to current practice.

Gay [Gay89, pp. 16–17] has already drawn attention to the lurking numerical difficulties and suggests a middle ground of working with AD^2A^T as long as possible, then switching to a more robust alternative such as direct solves with K.

1.6. Iterative methods and preconditioning. The KKT systems we are concerned with arise when Newton's method is applied to the nonlinear equations defining optimality conditions for barrier subproblems; see §3. In this context, there are not many KKT systems to be solved (compared to those in active-set methods), the systems need not be solved exactly [DES82], and the KKT matrix eventually does not change significantly. It is therefore appropriate to consider iterative methods and preconditioners for the indefinite matrix K.

Previous work on preconditioning for interior-point methods has focused on the LP case and the Schur-complement matrix AD^2A^T. Most authors have used approximate Cholesky factors of AD^2A^T; see, for example, [GMSTW86], [Kar87], [KR88], [Meh89a]. Exact LU factors of DA^T have also been investigated [GMS89].

The success of preconditioned conjugate-gradient methods in this context lends added promise to our proposed use of the much better conditioned KKT systems, now that it is known how to precondition indefinite systems.

1.7. Summary. In §2 we consider general indefinite systems and derive a preconditioner from the Bunch–Parlett factorization. In §3 we consider barrier methods for nonlinear programs, and propose factorizing just part of the KKT system to obtain a preconditioner for the whole system.

[1] We use the spectral condition number, $\operatorname{cond}(K) = \|K^{-1}\|_2\|K\|_2$.

Sections 4 to 6 deal with the LP case. In §4 we propose three preconditioners based on LU factors of a square nonsingular matrix B (analogous to the basis in the simplex method). Section 5 discusses some practical difficulties. Section 6 gives numerical results on the condition numbers of K and AD^2A^T in a typical sequence of barrier subproblems, and compares the preconditioned systems $C^{-1}KC^{-T}$ for several preconditioners CC^T.

2. Preconditioning indefinite systems. Let K be any symmetric nonsingular matrix, and let M be a given positive-definite matrix. Also, let "products with K" mean matrix-vector products of the form $u = Kv$, and "solves with M" mean solution of linear systems of the form $Mx = y$.

The Paige–Saunders algorithm as implemented in SYMMLQ [PS75] may be used to solve $Ky = z$ even if K is indefinite. As with other conjugate-gradient–like algorithms, the matrix is represented by a procedure for computing products with K (those generated by the symmetric Lanczos process).

The first steps towards accelerating the convergence of this algorithm were taken by Szyld and Widlund [SW78], [SW79]. Given a positive-definite matrix M as preconditioner, their algorithm used solves with M in the normal way, but was unconventional in also requiring products with M.

Subsequently, a variant of SYMMLQ was developed that requires only solves with M [Sau79]. To solve $Ky = z$, this variant regards the preconditioner as having the form $M = CC^T$ and implicitly applies the Paige–Saunders algorithm to the system

$$C^{-1}KC^{-T}w = C^{-1}z,$$

accumulating approximations to the solution $y = C^{-T}w$ (without approximating w, which isn't needed). An implementation is available from the *misc* chapter of *netlib* [DG85].

2.1. Use of the Bunch–Parlett factorization. Given any symmetric nonsingular matrix K, there exists a factorization of the form

$$K = P^TU^TDUP,$$

where P is a permutation, U is upper triangular, and D is block-diagonal with blocks of dimension 1 or 2 [BP71]. If K is indefinite, some of the blocks of D will have negative eigenvalues. Let the eigensystem of D be

$$D = Q\Lambda Q^T, \qquad \Lambda = \operatorname{diag}(\lambda_j),$$

and let

$$\bar{D} = Q\bar{\Lambda}Q^T, \qquad \bar{\Lambda} = \operatorname{diag}(|\lambda_j|),$$

be a closely related positive-definite matrix that can be obtained at minimal cost. If we define $C = P^TU^T\bar{D}^{1/2}$, it is easily verified that

$$\bar{K} \equiv C^{-1}KC^{-T} = \operatorname{diag}(\lambda_j/|\lambda_j|) = \operatorname{diag}(\pm 1).$$

This means that the "perfect" preconditioner for K is the matrix

$$M = CC^T = P^TU^T\bar{D}UP,$$

since the "preconditioned" matrix $\bar{K}$ has at most two distinct eigenvalues and the Paige–Saunders algorithm converges in at most two iterations.

In practice, M will be computed from the Bunch–Parlett factorization of an *approximation* to K.

2.2. Modification of D from MA27. The block-diagonal matrix D is packed in the MA27 data structure as a sequence of matrices of the form

$$\begin{pmatrix} \alpha \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} \alpha & \beta \\ \beta & \gamma \end{pmatrix}.$$

In the 1×1 case, we do nothing if $\alpha > 0$; otherwise we reverse its sign. In the 2×2 case, we do nothing if $\alpha\gamma > \beta^2$; otherwise we compute the eigensystem in the form

$$\begin{pmatrix} \alpha & \beta \\ \beta & \gamma \end{pmatrix} = \begin{pmatrix} c & s \\ s & -c \end{pmatrix} \begin{pmatrix} \lambda_1 & \\ & \lambda_2 \end{pmatrix} \begin{pmatrix} c & s \\ s & -c \end{pmatrix},$$

where $c^2 + s^2 = 1$. We then form the positive-definite matrix

$$\begin{pmatrix} \bar{\alpha} & \bar{\beta} \\ \bar{\beta} & \bar{\gamma} \end{pmatrix} = \begin{pmatrix} c & s \\ s & -c \end{pmatrix} \begin{pmatrix} |\lambda_1| & \\ & |\lambda_2| \end{pmatrix} \begin{pmatrix} c & s \\ s & -c \end{pmatrix},$$

and overwrite the appropriate three locations of MA27's storage.

The techniques for computing the 2×2 eigensystem are central to Jacobi's method for the symmetric eigenvalue problem. They were developed by Rutishauser [Rut66]. We have followed the description in Golub and Van Loan [GV89, p. 446] with minor changes to work with symmetric plane rotations.

A subroutine for modifying the D computed by MA27 is given in the Appendix.

2.3. Aasen's method. In general, Aasen's tridiagonalization method [Aas71] is considered competitive with the Bunch–Kaufman approach [BK77] for solving dense indefinite systems. Aasen's method computes a factorization of the form $K = U^T T U$ where T is tridiagonal.

We do not know of a sparse implementation, but in any event we note that it would not be ideal for producing a preconditioner in the manner described above, since the eigensystem for T would involve far more work than for the block-diagonal D of the Bunch–Parlett factorization.

On the other hand, we could compute a (very special) Bunch–Parlett factorization of T and modify the associated D as described above.

3. Barrier subproblems. We return now to the optimization problems (1)–(2). In barrier methods, the bounds $l \le x \le u$ are absorbed into the objective function and we solve a sequence of perturbed subproblems, typically of the form

$$\begin{aligned} &\underset{x}{\text{minimize}} && F_\mu(x) = F(x) - \mu \sum_{j=1}^{n} (\ln(x_j - l_j) + \ln(u_j - x_j)) \\ &\text{subject to} && c(x) = 0, \end{aligned} \tag{6}$$

where the barrier parameter μ takes decreasing positive values that are eventually very small. If $l_j = -\infty$ or $u_j = \infty$ for some j, the corresponding terms $\ln(x_j - l_j)$ or $\ln(u_j - x_j)$ are omitted. If x_j has no bounds, both terms may be omitted.

The following quantities are needed:

$L_\mu(x, \pi) = F_\mu(x) - \pi^T c(x)$	the Lagrangian function,
$g_\mu(x) = \nabla F_\mu(x)$	the gradient of the barrier function,
$g_L(x, \pi) = g_\mu(x) - A(x)^T \pi$	the gradient of the Lagrangian,
$H_L(x, \pi) = \nabla^2 F_\mu(x) - \sum \pi_i \nabla^2 c_i(x)$	the Hessian of the Lagrangian, and
$A(x) = \nabla c(x)$	the Jacobian of the constraints.

For convenience we assume that $A(x) \in \Re^{m \times n}$ has full row rank m, and that the scaling of the problem is reasonable, so that $\|A(x)\| \approx 1$.

3.1. Newton's method and the KKT system. The optimality conditions for (6) are the nonlinear equations

$$g_L(x, \pi) = 0, \tag{7}$$

$$c(x) = 0. \tag{8}$$

Newton's method may be applied directly, or to some equivalent system. Given suitable initial values for the primal and dual variables (x, π), the key set of equations for generating a search direction is the KKT system

$$\begin{pmatrix} H_L & A^T \\ A & \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta\pi \end{pmatrix} = - \begin{pmatrix} g_L \\ c \end{pmatrix}, \tag{9}$$

where the KKT matrix and right-hand side are evaluated at the current point (x, π). A positive steplength α is then chosen to reduce some measure of the size of the right-hand side (g_L, c), and the variables are updated according to $x \leftarrow x + \alpha\Delta x$, $\pi \leftarrow \pi + \alpha\Delta\pi$. (Sometimes a different α may be used for x and π.)

3.2. A preconditioner for K. In general, some of the variables converge to values near their upper or lower bounds. For such variables x_j, the Hessian H_L includes on its diagonal a term that becomes very large: $\mu/(x_j - l_j)^2$ or $\mu/(u_j - x_j)^2$, which are $O(1/\mu)$. Let the KKT matrix be partitioned accordingly:

$$K = \begin{pmatrix} K_1 & K_3^T \\ K_3 & K_2 \end{pmatrix}, \tag{10}$$

where K_1 is the part of H_L associated with variables near a bound, and K_2 looks like a smaller KKT system associated with the remaining variables. This partitioning is crucial to the sensitivity analysis in [Pon90]. Of course, the partition depends on the measure of closeness to a bound, but it is not critical here except that the dimension of K_1 should not exceed $n - m$.

One possible approximation to K is

$$\begin{pmatrix} D_1 & \\ & K_2 \end{pmatrix}, \tag{11}$$

where D_1 is a diagonal matrix containing the diagonals of K_1, which by construction are large and positive. Applying the method of §2, we can now obtain a positive-definite preconditioner for K as follows:

$$K_2 = U_2^T D_2 U_2, \qquad M = \begin{pmatrix} D_1 & \\ & U_2^T \bar{D}_2 U_2 \end{pmatrix}, \tag{12}$$

where $\bar{D}_2$ is obtained from D_2 at nominal cost.

3.3. Discussion. In broad terms, we need to estimate which variables are going to be "free" (away from their bounds) at a solution. If $m \ll n$, the KKT system K_2 associated with the free variables may be much smaller than the whole of K, and the cost of the Bunch–Parlett factorization of K_2 may be acceptably low.

For the early iterations of Newton's method, the estimate of K_2 will usually be poor, and the diagonal term D_1 will not be particularly large. However, following the inexact Newton approach [DES82], only approximate solutions to the KKT system are needed, and the iterative solver need not perform many iterations.

As the Newton iterations converge and the partition (10) becomes more sharply defined, the preconditioner should become increasingly powerful and produce the increasingly accurate solutions required at an acceptable cost.

3.4. Performance of SYMMLQ with the MA27 preconditioner. The approach of §§2 and 3 has been tested by Burchett [Bur89] within a barrier algorithm for solving some large nonlinear problems in optimal power flow.

Normally, MA27 is used to factorize K at each iteration of the barrier algorithm, with H_L and A in (9) changing each time. For experimental purposes, the factors of K at iteration k were used to construct a preconditioner for iteration $k+1$ (via subroutine `syprec` in the Appendix). The dimension of K was 6000 for one problem and 16,000 for another.

Initially, SYMMLQ required about 30 iterations to solve the KKT systems to moderate accuracy. As the barrier algorithm (and A) converged, the number of iterations required by SYMMLQ fell to about 10. This performance seems very promising.

4. Preconditioners for linear programming. For linear programs the structure of the partitioned KKT system (10) can be investigated more closely, given that optimal solutions are often associated with a vertex of the feasible region. We partition the constraint matrix into the form $A = (\,N \;\; B\,)$, where B is square and nonsingular, and N in some sense corresponds to the $n-m$ variables that are closest to a bound.

The Hessian for the barrier function is a diagonal matrix H, which we partition as $H = \operatorname{diag}(H_N, H_B)$. The KKT system is then

$$K = \begin{pmatrix} H_N & & N^T \\ & H_B & B^T \\ N & B & \end{pmatrix}.$$

As convergence occurs, the diagonals of $H_N \to \infty$ (and in general $\operatorname{cond}(K) \to \infty$). In degenerate cases, some diagonals of H_B may also become very large.

In various primal, dual, and primal-dual interior-point algorithms for LP, similar matrices K arise with varying definitions of H (e.g., [Meg86], [KMY88], [LMS89], [Meh89a], [Meh90]). The discussion hereafter applies to all such methods.

In the following sections we introduce a series of preconditioners of the form $M = CC^T$. To improve the convergence of SYMMLQ, the transformed matrices $\bar{K} = C^{-1}KC^{-T}$ should have a better condition than K or a more favorable distribution of eigenvalues (clustered near ± 1). We make use of the quantities

$$V = B^{-T}H_B B^{-1}, \qquad W = NH_N^{-1/2},$$

and are motivated by the fact that $V \to 0$ and $W \to 0$ in nondegenerate cases. The effects of degeneracy are discussed later.

4.1. The preconditioner M_1. The first preconditioner is diagonal and is intended to eliminate the large diagonals of K:

$$M_1 = C_1 C_1^T = \begin{pmatrix} H_N & & \\ & I & \\ & & I \end{pmatrix}, \tag{13}$$

$$\bar{K}_1 = C_1^{-1} K C_1^{-T} = \begin{pmatrix} I & & W^T \\ & H_B & B^T \\ W & B & \end{pmatrix}. \tag{14}$$

With diagonal preconditioning, there is no loss of precision in recovering solutions for the original system. Thus as H_N becomes large, the preconditioned matrix $\bar{K}_1$ tends to represent the true sensitivity of the KKT system with regard to solving linear equations.

We will use $\bar{K}_1$ later for comparing condition numbers.

4.2. The preconditioner M_2. The second preconditioner is block diagonal:

$$M_2 = C_2 C_2^T = \begin{pmatrix} H_N & & \\ & B^T B & \\ & & I \end{pmatrix}, \tag{15}$$

$$\bar{K}_2 = C_2^{-1} K C_2^{-T} = \begin{pmatrix} I & & W^T \\ & V & I \\ W & I & \end{pmatrix}. \tag{16}$$

Since V and W tend to become small, M_2 tends towards being an exact preconditioner for K. We see that a Bunch–Parlett factorization is no longer needed. In order to solve systems involving M_2, we may use any sparse factorization of B or B^T.

4.3. The preconditioner M_3. The third preconditioner is designed to eliminate the submatrix V in (16), for degenerate cases where V is not adequately small:

$$M_3 = C_3 C_3^T, \qquad C_3 = \begin{pmatrix} H_N^{1/2} & & \\ & B^T & \frac{1}{2} H_B B^{-1} \\ & & I \end{pmatrix}, \tag{17}$$

$$\bar{K}_3 = C_3^{-1} K C_3^{-T} = \begin{pmatrix} I & -\frac{1}{2} W^T V & W^T \\ -\frac{1}{2} VW & & I \\ W & I & \end{pmatrix}. \tag{18}$$

The off-diagonal term in (17) can be derived by observing that for a KKT matrix of the form

$$K = \begin{pmatrix} H & B^T \\ B & \end{pmatrix}, \qquad B \text{ square},$$

we would like $M = CC^T$ to satisfy

$$C^{-1} K C^{-T} = \begin{pmatrix} & I \\ I & \end{pmatrix} \equiv J,$$

or equivalently, $CJC^T = K$. Letting C be of the form

$$C = \begin{pmatrix} B^T & E \\ & I \end{pmatrix},$$

we find that E should satisfy $EB + B^T E^T = H$. The simplest choice is then to set $E = \frac{1}{2} H B^{-1}$.

Though V has been eliminated, we have now introduced the term $-\frac{1}{2} VW$, and solves with M_3 cost twice as much as solves with M_2. The expected benefit is that $\frac{1}{2} VW$ should be smaller than V itself.

4.4. The preconditioner M_4. The fourth preconditioner also eliminates V, using the factorization $B^T = LU$, where we intend that L be well conditioned:

$$M_4 = C_4 C_4^T, \qquad C_4 = \begin{pmatrix} H_N^{1/2} & & \\ & L & \frac{1}{2}H_B L^{-T} \\ & & U^T \end{pmatrix}, \tag{19}$$

$$\bar{K}_4 = C_4^{-1} K C_4^{-T} = \begin{pmatrix} I & -\frac{1}{2}\bar{W}^T\bar{V} & \bar{W}^T \\ -\frac{1}{2}\bar{V}\bar{W} & & I \\ \bar{W} & I & \end{pmatrix}, \tag{20}$$

where

$$\bar{V} = L^{-1} H_B L^{-T}, \qquad \bar{W} = U^{-T} N H_N^{-1/2}.$$

As before, letting C be of the form

$$C = \begin{pmatrix} L & E \\ & U^T \end{pmatrix}$$

and requiring $CJC^T = K$, we find that $EL^T + LE^T = H$, and we take $E = \frac{1}{2}HL^{-T}$.

Solves with M_4 are cheaper than with M_3. Comparing (18) and (20), a further advantage is that $\bar{V}\bar{W} = UVW$ tends to be smaller than VW, although $\bar{W} = U^{-T}W$ is probably larger than W.

4.5. The preconditioner M_D. We mention one further diagonal preconditioner that has appeared implicitly in the literature for the case $H = \mu D^{-2}$ with D diagonal. It does not depend on the N-B partitioning, and gives a transformed system that does not involve μ:

$$M_D = C_D C_D^T = \begin{pmatrix} \mu D^{-2} & \\ & \mu^{-1} I \end{pmatrix}, \tag{21}$$

$$\bar{K}_D = C_D^{-1} K C_D^{-T} = \begin{pmatrix} I & DA^T \\ AD & \end{pmatrix}. \tag{22}$$

The matrix $\bar{K}_D$ is associated with weighted least-squares problems of the form (5), as discussed in [GMSTW86]. Turner [Tur87], [Tur90] has investigated the use of MA27 to obtain exact factors of both K and $\bar{K}_D$. An important practical observation was that MA27 produced much sparser factors for $\bar{K}_D$ than for K.

Unfortunately, the numerical examples in §6 show that $\bar{K}_D$ has essentially the same condition as AD^2A^T, which tends to be much more ill conditioned than $\bar{K}_1$ (14). We therefore cannot recommend the use of $\bar{K}_D$.

Indeed, when $\|AD\| \approx 1$ as we have here, it can be shown that $\text{cond}(\bar{K}_D) \approx \text{cond}(AD)^2$. To improve the condition of $\bar{K}_D$ we should use

$$M_D = C_D C_D^T = \begin{pmatrix} \alpha^{-1}\mu D^{-2} & \\ & \alpha\mu^{-1} I \end{pmatrix}, \tag{23}$$

$$\bar{K}_D = C_D^{-1} K C_D^{-T} = \begin{pmatrix} \alpha I & DA^T \\ AD & \end{pmatrix} \tag{24}$$

for some $\alpha \in (0, \|AD\|_2]$, since it is known that $\text{cond}(\bar{K}_D) \approx \text{cond}(AD)$ can be achieved if $\alpha \approx \sigma_{\min}$, the smallest singular value of AD (Björck [Bjo67], [Bjo91]). Experiments in this direction have been performed by Arioli, Duff, and De Rijk [ADR89] (who also give error analyses) and by Fourer and Mehrotra [FM91].

For the sake of both direct and iterative methods for solving KKT systems, it is hoped that further development of MA27 will result in greatly improved sparsity in the factors of K and/or $\bar{K}_1$. At the time of writing, a new code MA47 holds much promise (see Duff et al. [DGRST89]), as does an analogous code described in [FM91].

4.6. Regularizing K and AD^2A^T. Since A often does not have full row rank, it is important to include a regularization parameter $\delta > 0$ in the KKT system. Thus (3) becomes

$$\begin{pmatrix} H & A^T \\ A & -\delta I \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta \pi \end{pmatrix} = \begin{pmatrix} -g \\ r \end{pmatrix}. \tag{25}$$

Systems of this type have been studied in the context of sequential quadratic programming by Murray [Mur69], Biggs [Big75], and Gould [Gou86].

In practice, a wide range of values of δ may be used without inhibiting convergence, particularly with methods that do not maintain primal feasibility ($A\Delta x = 0$). For example, we would recommend values in the range $10^{-8} \le \delta \le 10^{-4}$ on a machine with about 16 digits of precision, assuming $\|A\| \approx 1$.

Note that the corresponding system (4) becomes

$$(AD^2A^T + \mu\delta I)\Delta\pi = AD^2g + \mu r. \tag{26}$$

When μ is as small as 10^{-10} (say), one would have to choose a rather large δ (say, $\delta \ge 10^{-2}$) to achieve any degree of regularization of AD^2A^T. This constitutes a large perturbation to the underlying KKT system (25).

In other words, a much smaller δ is sufficient to regularize (25) than (26). Thus, KKT systems again show an advantage over AD^2A^T.

With regard to the preconditioners, δ introduces terms $-\delta I$, $-\delta I$, $-\delta U^{-T}U^{-1}$ into the bottom corner of $\bar{K}_2$, $\bar{K}_3$, $\bar{K}_4$, respectively. For $\bar{K}_4$ it appears that δ must be chosen quite small and that the choice of B must be flexible enough to prevent U from being excessively ill conditioned (see §5.3).

5. Use of LU factors. For linear programs, the "small" KKT matrix in (10) is of the form

$$K_2 = \begin{pmatrix} H_B & B^T \\ B & \end{pmatrix}.$$

As in the general nonlinear case we could obtain a preconditioner from a Bunch–Parlett factorization of K_2, and in practice this may prove to be a good approach.

The preconditioners M_2, M_3, and M_4 were derived on the assumption that it should be cheaper to compute sparse factors of just the matrix B. We propose to use the package LUSOL [GMSW87] to obtain $B^T = LU$, where L is a permuted lower triangle with unit diagonals. A user-defined tolerance limits the size of the off-diagonals of L (typically to 5, 10, or 100), thereby limiting the condition of L as required.

5.1. Choice of B. One of the main practical difficulties will be in choosing a "good" square matrix B at each stage. The current values of x and/or the estimated reduced costs $z = c - A^T\pi$ should provide some guidance. For example, the diagonal matrix H is defined in terms of these quantities, and the smallest $m + s$ diagonals of H could be used to pinpoint a submatrix $\bar{A}$ of A (for some moderate $s \geq 0$). LUSOL could then be used to obtain a rectangular factorization $\bar{A}^T = LU$. The first m pivot rows and columns may suggest a suitable B.

Alternative approaches to choosing B have been suggested by Gay [Gay89], Tapia and Zhang [TZ89], Mehrotra [Meh89b], and others. These remain to be explored.

5.2. The effects of degeneracy on V and W. In general, primal degeneracy will mean that certain elements of H_B do not tend to zero, so that not all of V or $\bar{V}$ will become small. Similarly, dual degeneracy will mean that certain elements of H_N will not become large, and not all of W or $\bar{W}$ will become small.

The main effect is that the preconditioners will be less "exact." Either form of degeneracy is likely to increase the number of SYMMLQ iterations required.

5.3. Singular systems. Whatever the method for choosing a square B, it is probable that B will be singular (since in many practical cases, A does not have full row rank). At present we propose to rely on the fact that LUSOL will compute a stable singular factorization of the form

$$B^T = \begin{pmatrix} L_1 \\ L_2 \end{pmatrix} \begin{pmatrix} U_1 & U_2 \end{pmatrix},$$

and the solve procedures will treat this as if it were the factorization of a nonsingular matrix

$$\bar{B}^T = \begin{pmatrix} L_1 & \\ L_2 & I \end{pmatrix} \begin{pmatrix} U_1 & U_2 \\ & I \end{pmatrix}.$$

User-defined tolerances determine how ill conditioned U_1 is allowed to be (and hence determine its dimension).

Alternatively, we may use the factorization $B_1^T = L_1U_1$ to transform most of K as already described. Certain rows of A will not be transformed in the preferred way, and again the effect will be to increase the number of SYMMLQ iterations required.

6. Numerical examples for the LP case. Here we investigate the effect of the preconditioners described in §4. For test purposes we have used MATLAB™ [MLB87] to implement a primal-dual interior-point algorithm for the standard LP problem min c^Tx subject to $Ax = b$, $x \geq 0$. The linear system to be solved each iteration is

$$\begin{pmatrix} H & A^T \\ A & -\delta I \end{pmatrix} \begin{pmatrix} \Delta x \\ -\Delta\pi \end{pmatrix} = \begin{pmatrix} -g \\ r \end{pmatrix}, \tag{27}$$

where $H = X^{-1}Z$, $X = \operatorname{diag}(x_j)$, $Z = \operatorname{diag}(z_j)$, $r = b - Ax$, $g = c - A^T\pi - \mu X^{-1}e$, and e is the vector of ones. The search direction for z is $\Delta z = X^{-1}(\mu e - Z\Delta x) - z$.

The rows and columns of A were scaled to give $\|A\| \approx 1$. The starting values were $x = e$, $z = e$, $\pi = 0$ (so that $H = I$ initially), and δ was fixed at 10^{-8}, with the machine precision on a DEC VAX system being around 16 digits. The parameter μ was reduced every iteration according to the steplengths for x and z: $\mu \leftarrow \mu - \alpha_\mu\mu$, where $\alpha_\mu = \min(\alpha_x, \alpha_z, 0.99)$, and α_x, α_z were limited in the usual way to be at most

TABLE 1
Condition numbers for problem exp1.

k	μ	AD^2A^T	$\bar{K}_D$	K	$\bar{K}_1$	$\bar{K}_2$	$\bar{K}_3$	$\bar{K}_4$	B rank-def
1	1.9e-2	1.2e1	1.1e1	1.1e1	1.1e1	3.1e1	2.6e1	1.1e1	
2	4.5e-3	6.5e1	3.1e1	6.8e2	1.9e1	5.5e1	2.7e1	2.7e1	
3	4.8e-4	1.2e3	2.2e3	4.4e4	6.9e1	4.6e2	5.7e1	1.6e1	
4	1.3e-4	1.7e5	3.1e5	1.0e6	4.2e3	4.4e3	1.5e3	1.6e3	2
5	3.2e-5	2.9e5	5.3e5	1.3e6	7.7e2	2.3e3	5.4e2	5.6e2	1
6	1.5e-5	4.0e5	5.8e5	3.1e5	6.3e2	2.8e3	7.4e2	7.5e2	1
7	4.6e-6	4.0e5	5.8e5	1.6e5	1.2e2	3.0e2	1.1e2	1.1e2	1
8	1.6e-6	4.7e5	6.3e5	3.0e5	3.9e1	1.4e2	1.9e1	8.4e0	
9	1.4e-7	9.1e5	1.1e6	4.9e5	2.0e1	4.7e0	1.3e0	1.5e0	
10	7.8e-9	8.4e5	1.0e6	3.5e6	1.7e1	1.1e0	1.1e0	1.3e0	

1 or 0.99 times the step to the boundaries $x > 0$, $z > 0$, respectively. See [KMY88], [MMS89], [LMS89], and [Meh90] for related details.

Condition numbers of various matrices were obtained using MATLAB's function *rcond*. The square matrices B for the preconditioners of §4 were obtained from the columns of A for which $H_{jj} \leq 20$. The diagonals H_{jj} were first sorted and up to $1.2m$ of the smallest were used to select a rectangular matrix $\bar{A}$ from A. In practice, a sparse LU factorization of $\bar{A}$ or $\bar{A}^T$ would extract a full-rank submatrix, but here we used MATLAB's function $qr(A)$ to elicit a full-rank set of columns (via a QR factorization with column interchanges), and a second QR factorization of part of $\bar{A}^T$ to pinpoint a full-rank set of rows. The dimension of the resulting matrix B is generally less than m. The "rank" was determined from the first QR factorization by requiring the diagonals of R to be greater than 10^{-4}.

6.1. A nondegenerate example. To illustrate ideal behavior of the preconditioners, we chose a nondegenerate problem *exp*1 [Bla82] in which A is 10 by 17 (including 10 unit columns associated with slack variables). The lack of primal or dual degeneracy means that near a solution, $m = 10$ diagonals of H are substantially less than 1, and $n - m = 7$ diagonals are significantly greater than 1. The choice of B is ultimately clear cut.

Table 1 lists various condition numbers for each iteration of the primal-dual algorithm. For interest, we include AD^2A^T and $\bar{K}_D$, which were defined in terms of $D = X = \text{diag}(x_j)$ (see §4.5) and incorporated the same regularization δ (§4.6). It may be seen that both $AD^2A^T + \mu\delta I$ and $\bar{K}_D$ become increasingly ill conditioned in step with K, in contrast to the "meaningful" condition of K reflected by $\bar{K}_1$ (in which the large diagonals of H have been scaled to 1).

The preconditioned systems $\bar{K}_2$, $\bar{K}_3$, and $\bar{K}_4$ show an increasing, though apparently mild, improvement over $\bar{K}_1$. Their effectiveness depends on the choice of B and whether or not it has dimension m. The column labeled "B rank-def" records the corresponding rank-deficiency. The conditions of B, L, and U were less than 25, 7, and 40, respectively, for all iterations.

Low conditions are always a good sign, but high ones tell an incomplete story. Figure 1 shows more clearly the increasing improvement of the preconditioners M_2, M_3, M_4 in terms of the clustering of the eigenvalues of $\bar{K}_2$, $\bar{K}_3$, $\bar{K}_4$ around ± 1. The KKT systems have dimension $m+n = 27$. Eigenvalues in the range $(-5, 5)$ are plotted exactly; the remainder are compressed into the ranges $(-6, -5)$ and $(5, 6)$. Thus, $\bar{K}_2$

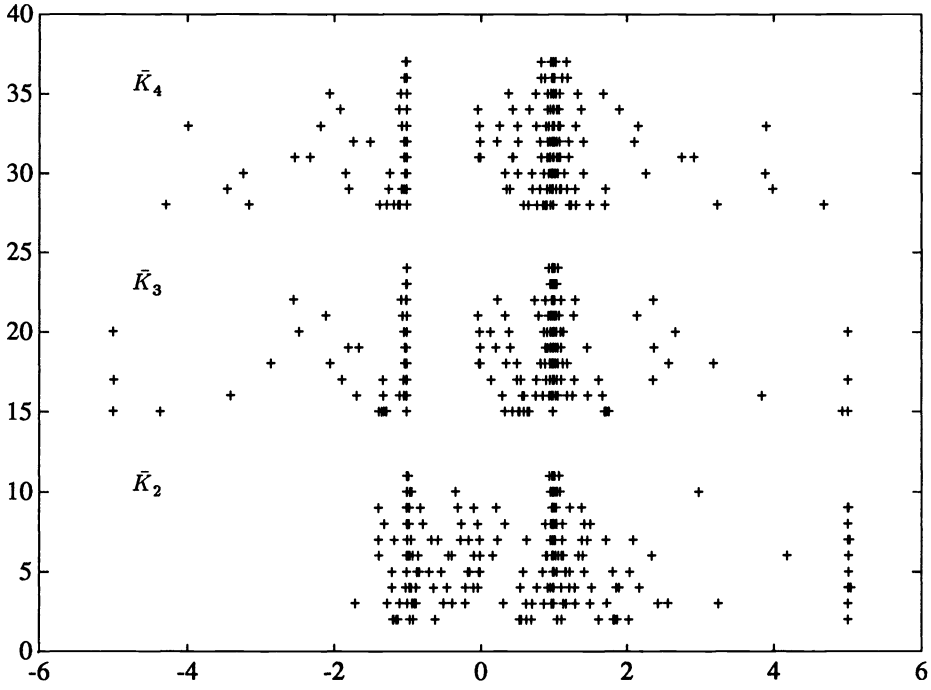


FIG. 1. *Eigenvalues for* $\bar{K}_2$, $\bar{K}_3$, $\bar{K}_4$ *for problem exp*1.

has one or two eigenvalues greater than 5 for the first eight iterations, whereas $\bar{K}_3$ has its eigenvalues inside $(-5, 5)$ at all times. (The vertical axis is "iteration number" shifted by 1 for $\bar{K}_2$, 14 for $\bar{K}_3$, and 27 for $\bar{K}_4$. Each horizontal line gives the spectrum of one of these matrices at the corresponding iteration.)

It is evident from Fig. 1 that $\bar{K}_3$ and $\bar{K}_4$ have more favorable eigenvalue distributions than $\bar{K}_2$, and that $\bar{K}_4$ is marginally better than $\bar{K}_3$, the main benefit being that it is more cheaply obtained. There is a striking absence of eigenvalues in the range $(-1+\beta, -\beta)$ for some small β, though we have no immediate explanation. This range broadens to $(-1+\beta, 1-\beta)$ for all systems at the final iteration, as we may expect.

6.2. A more typical example. Table 2 and Fig. 2 give similar results for the well-known problem *afiro* [Gay85]. The matrix A is 27 by 51, including 19 slack columns. We see that $AD^2A^T + \mu\delta I$ and $\bar{K}_D$ again become extremely ill conditioned in step with K.

The KKT systems have dimension 78. As before there is a clear division between large and small diagonals of H near a solution, but in this case only $m-5$ are substantially smaller than one. The rank of the corresponding columns of A is $m-7$, consistent with B's final rank-deficiency of 7. The conditions of B, L, and U were again low: less than 35, 13, and 34, respectively.

It is encouraging to observe that Fig. 2 is qualitatively similar to Fig. 1 in spite of the rank-deficiency in B. The main difference is two eigenvalues close to zero on the last iteration, in keeping with the difference between $m-5$ and $m-7$. Results

TABLE 2
Condition numbers for problem afiro.

k	μ	AD^2A^T	$\bar{K}_D$	K	$\bar{K}_1$	$\bar{K}_2$	$\bar{K}_3$	$\bar{K}_4$	B rank-def
1	2.6e-2	6.0e1	2.2e1	3.3e1	2.3e1	1.3e2	1.8e2	9.8e1	1
2	9.9e-3	3.1e2	2.0e2	1.8e3	1.2e2	7.0e2	1.1e2	4.4e1	1
3	1.8e-3	2.5e3	3.6e3	3.0e4	4.7e2	9.4e3	3.1e3	4.9e2	1
4	5.4e-4	1.7e4	3.4e4	1.1e6	2.2e3	4.1e4	1.1e4	7.0e3	1
5	2.9e-4	8.2e3	1.7e4	3.0e5	6.9e2	2.4e4	1.2e3	3.1e2	3
6	3.3e-5	8.5e3	1.8e4	4.5e4	1.0e2	1.1e3	4.6e2	2.9e2	
7	2.4e-5	2.0e4	3.9e4	1.4e6	3.8e2	3.0e3	6.5e2	4.8e2	1
8	8.5e-6	2.6e5	4.3e5	1.8e7	9.0e3	1.6e4	2.5e3	2.9e3	3
9	3.1e-6	1.7e7	2.7e7	1.1e8	7.8e4	5.3e3	2.5e3	2.2e3	6
10	4.3e-7	2.0e8	3.1e8	2.0e9	1.7e6	9.6e4	2.3e4	1.6e4	6
11	4.3e-9	2.e10	4.e10	4.e11	5.7e8	3.7e5	3.9e5	2.6e5	7

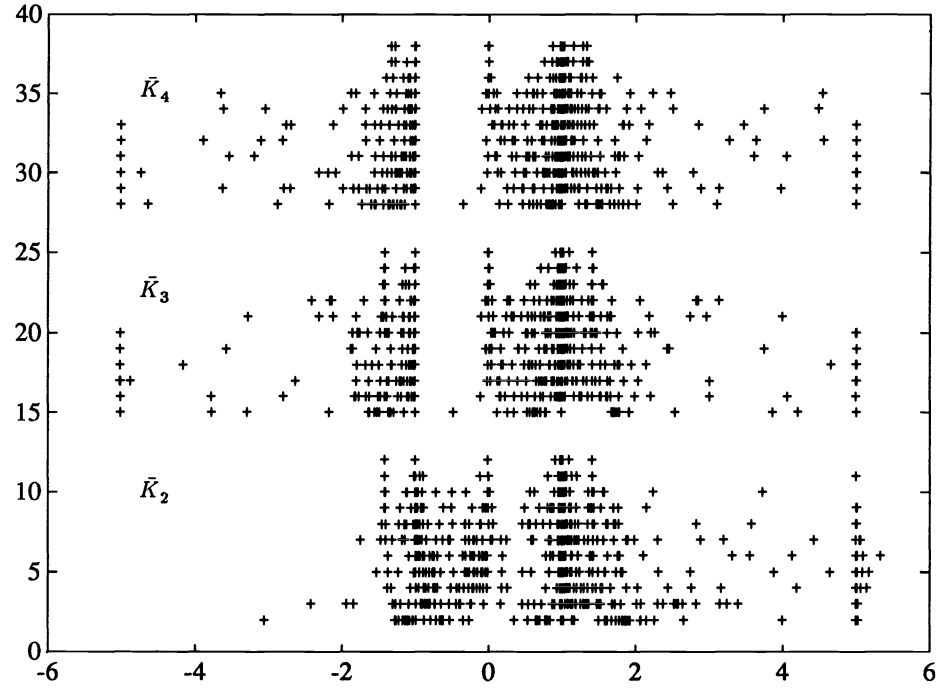


FIG. 2. *Eigenvalues for $\bar{K}_2$, $\bar{K}_3$, $\bar{K}_4$ for problem afiro.*

are similar for the second to last iteration. We can expect a low number of SYMMLQ iterations will be required as the barrier algorithm converges, as in the ideal nondegenerate case.

6.3. Performance of SYMMLQ with the LP preconditioners. As a further experiment we modified the barrier LP algorithm to solve (27) using SYMMLQ with the first four preconditioners M_1–M_4 of §4. We applied the LP algorithm to problem *exp*1 with and without scaling of the data. The iterations required by SYMMLQ

TABLE 3
SYMMLQ *iterations with various preconditioners on problem exp*1.

	Scaled				Unscaled			
k	M_1	M_2	M_3	M_4	M_1	M_2	M_3	M_4
1	17	21	24	17	44	21	25	82
2	28	25	21	21	72	80	68	98
3	29	27	19	19	95	76	72	91
4	31	29	17	21	88	73	69	83
5	31	23	18	22	55	51	52	57
6	27	19	14	16	51	36	29	42
7	25	14	12	14	42	27	25	31
8	24	12	9	11	37	20	20	26
9	23	8	8	9	35	12	12	18
10	23	6	6	7	32	9	9	13
11	21	4	4	4	29	6	6	7
12	19	3	3	3	28	4	4	6
13	18	3	3	3				
14	12	3	3	3				

are shown in Table 3 for each iteration k of the barrier algorithm. For simplicity the partition $A = (N \quad B)$ was chosen to be the same for all k, with B being the optimal nondegenerate basis determined by the simplex method. As expected, the preconditioners M_2–M_4 improve markedly as the LP solution is approached.

Iterative solution of each KKT system is easier for the scaled problem because $B^T = LU$ is better conditioned:

	Scaled	Unscaled
cond(B)	14	443
cond(L)	4	4
cond(U)	17	235

For the scaled problem the stopping tolerance for SYMMLQ was taken to be $rtol = 10^{-6}$ (a loose value since the KKT systems need not be solved accurately). However, *rtol* terminates solution of the *preconditioned* system. For the unscaled problem it was necessary to set $rtol = 10^{-10}$ to obtain sufficient accuracy in the search direction for the first few values of k. In general it seems that high precision would be needed for safety: $rtol \approx 10^{-15}$. (This appears to be a general difficulty. If the original system is $Kx = b$ and the preconditioner is CC^T, SYMMLQ terminates when $\|C^{-1}(b - Kx)\| \leq \|C^{-1}KC^{-T}\|\|C^Tx\| rtol$, since the terms involved can be estimated. There is no certainty that $\|b - Kx\| \leq \|K\|\|x\| rtol$, although $\|b - Kx\| \leq \|b\| rtol$ could be tested after the fact.)

On nondegenerate problems such as this, preconditioners M_2–M_4 can be expected to perform similarly, at least in the scaled case. We expected M_4 to show an advantage in the unscaled case, but this did not eventuate. Greater variation can be expected on degenerate problems when V does not become suitably small. Further experiments in this direction remain for the future.

7. Conclusions. For symmetric indefinite systems of linear equations, we have shown that the Bunch–Parlett factorization can be used to provide a preconditioner for the Paige–Saunders algorithm SYMMLQ (§2). This general result led us to consider iterative methods for the KKT systems arising in barrier methods for QP and

nonlinear programming. The preconditioner (12) should play an important role in future interior-point implementations for large-scale constrained optimization.

For linear programs, the sensitivity analysis associated with the partitioned KKT system (10) led us to consider the true sensitivity of K, as reflected by the preconditioner M_1 and the transformed system $\bar{K}_1$ (13), (14). In turn, the fact that $\text{cond}(\bar{K}_1)$ is typically much smaller than $\text{cond}(AD^2A^T)$ motivated development of the preconditioners M_2, M_3, M_4 (15), (17), (19).

Subject to effective methods for choosing B, we expect these KKT preconditioners to bring improved reliability to interior-point LP algorithms. Implementations based on direct or iterative solves with AD^2A^T are often remarkably effective, but the extreme ill-conditioning of the AD^2A^T systems as solutions are approached makes their use tantamount to walking the razor's edge.

A switch to the full KKT system should be beneficial as Gay [Gay89] suggests, particularly when A contains some relatively dense columns that prevent exact Cholesky factorization of AD^2A^T. Fortunately, since B becomes more sharply defined near a solution, the KKT preconditioners will become most effective when they are most needed.

Appendix: A preconditioner from the Bunch–Parlett factorization. The following Fortran 77 routine illustrates the construction of a positive-definite matrix $M = U^T\bar{D}U$ from the Bunch–Parlett factorization $A = U^TDU$ produced by the Harwell MA27 package of Duff and Reid [DR82], [DR83].

Subroutine syprec overwrites the representation of D in the MA27 data structure. A typical application would contain calls of the form

```
call ma27ad( n, nz, ... )
call ma27bd( n, nz, ... )
call syprec( n, la, ... )
```

to factorize A and compute $\bar{D}$, followed by multiple calls of the form

```
call ma27cd( n, a , ... )
```

to solve systems involving M.

```
*    ------------------------------------------------------------------
     subroutine syprec( n, la, liw, a, iw, neg1, neg2 )

     implicit           double precision ( a-h, o-z )
     double precision   a(la)
     integer*2          iw(liw)
     integer            neg1, neg2

*    ------------------------------------------------------------------
*    syprec (SYmmetric PREConditioner) takes the factors
*          A = U' D U
*    from Duff and Reid's Harwell subroutine MA27BD and changes the
*    block-diagonal matrix D to be a positive-definite matrix Dbar with
*    the same 1x1 and 2x2 block-diagonal structure.
*
*    The eigensystem D = Q E Q' is used to define Dbar = Q |E| Q',
*    where |E| contains the absolute values of the eigenvalues of D.
*    The matrix
*          Abar = U' Dbar U
*    is then an exact preconditioner for A, in the sense that SYMMLQ
*    would take only 2 iterations to solve Ax = b (or 1 iteration if
```

```
*     D = Dbar is already positive definite).
*
*     If the original matrix A is close to some other matrix K,
*     Abar should be a good preconditioner for solving K x = b.
*
*     Note that MA27 stores the elements of D(inverse) and ( - U )
*     within A and IW.  However, modifying a 2x2 block of D(inverse)
*     looks the same as modifying the 2x2 block itself.
*
*     10 Mar 1989: First version.
*     Systems Optimization Laboratory, Stanford University.
*    ------------------------------------------------------------------

      intrinsic          abs , sqrt
      integer            alen, apos
      logical            single
      parameter        ( zero = 0.0d+0,  one = 1.0d+0,  two = 2.0d+0 )

      neg1   = 0
      neg2   = 0
      nblk   = abs( iw(1) )
      ipos   = 2
      apos   = 1

      do 100, iblk = 1, nblk
         ncols     =   iw(ipos)

         if (ncols .lt. 0) then
            nrows  =   1
            ncols  = - ncols
         else
            ipos   =   ipos + 1
            nrows  =   iw(ipos)
         end if

*        Process the diagonals in this block.

         alen   =   ncols
         single =  .true.

         do 50, k = ipos + 1, ipos + nrows
            if ( single ) then
               alpha  =   a(apos)
               j      =   iw(k)
               single =   j .gt. 0

               if ( single ) then
                  if ( alpha .lt. zero ) then
*                    ---------------------------------------------
*                    The 1x1 diagonal is negative.
*                    ---------------------------------------------
                     neg1    =   neg1 + 1
                     a(apos) = - alpha
                  end if
```

```
               else
                  beta   =   a(apos+1   )
                  gamma  =   a(apos+alen)

                  if ( alpha * gamma  .lt.  beta**2 ) then
*                    ------------------------------------------------
*                    The 2x2 diagonal is indefinite.
*                    Find its eigensystem in the form
*
*                    ( alpha  beta  )  =  ( c   s ) ( e1    ) ( c  s )
*                    ( beta   gamma )     ( s  -c ) (    e2 ) ( s -c )
*                    ------------------------------------------------
                     tau    =   ( gamma - alpha ) / ( two * beta )
                     t      =   abs( tau )  +  sqrt( tau**2 + one )
                     t      = - one / t
                     if ( tau .lt. zero ) t = - t
                     c      =   one / sqrt( t**2 + one )
                     s      =   t * c
                     e1     =   alpha  +  beta * t
                     e2     =   gamma  -  beta * t

*                    Change e1 and e2 to their absolute values
*                    and then multiply the three 2x2 matrices
*                    to get the modified  alpha, beta and gamma.


                     if ( e1 .lt. zero ) then
                        neg2   =   neg2 + 1
                        e1     = - e1
                     end if
                     if ( e2 .lt. zero ) then
                        neg2   =   neg2 + 1
                        e2     = - e2
                     end if

                     alpha        =   c**2 * e1  +  s**2 * e2
                     beta         =   c*s  *(e1  -  e2)
                     gamma        =   s**2 * e1  +  c**2 * e2
                     a(apos     ) =   alpha
                     a(apos+1   ) =   beta
                     a(apos+alen) =   gamma
                  end if
               end if
            else
               single =  .true.
            end if

            apos   =   apos + alen
            alen   =   alen - 1
   50    continue

         ipos   =   ipos + ncols + 1
  100 continue
```

```
*     end of syprec
      end
```

Acknowledgments. We thank Dr. Ed Klotz for making the test problem *exp1* available to us, and Gerry Miranda for his help in translating SYMMLQ into MATLAB. We also thank the referees for their helpful suggestions.

REFERENCES

[Aas71] J. O. AASEN, *On the reduction of a symmetric matrix to tridiagonal form*, BIT, 11 (1971), pp. 233–242.

[ADR89] M. ARIOLI, I. S. DUFF, AND P. P. M. DE RIJK, *On the augmented system approach to sparse least-squares problems*, Numer. Math., 55 (1989), pp. 667–684.

[Big75] M. C. BIGGS, *Constrained minimization using recursive quadratic programming: Some alternative subproblem formulations*, in Towards Global Optimization, L. C. W. Dixon and G. P. Szegö, eds., North–Holland, Amsterdam, 1975, pp. 341–349.

[Bjo67] Å. BJÖRCK, *Iterative refinement of linear least squares solutions*, BIT, 7 (1967), pp. 257–278.

[Bjo91] ———, *Pivoting and stability in the augmented system method*, Report LiTH-MAT-R-1991-30, Department of Mathematics, Linköping University, Linköping, Sweden, 1991.

[Bla82] C. BLAIR, *Some linear programs requiring many pivots*, Faculty Working Paper No. 867, College of Commerce and Business Administration, University of Illinois, Urbana, IL, 1982.

[BK77] J. R. BUNCH AND L. KAUFMAN, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., 31 (1977), pp. 162–179.

[BP71] J. R. BUNCH AND B. N. PARLETT, *Direct methods for solving symmetric indefinite systems of linear equations*, SIAM J. Numer. Anal., 8 (1971), pp. 639–655.

[Bur88] R. C. BURCHETT, Power Engineering Associates, private communication, 1988.

[Bur89] ———, Power Engineering Associates, private communication, 1989.

[DES82] R. S. DEMBO, S. C. EISENSTAT, AND T. STEIHAUG, *Inexact Newton methods*, SIAM J. Numer. Anal., 19 (1982), pp. 400–408.

[DG85] J. J. DONGARRA AND E. GROSSE, *Distribution of mathematical software via electronic mail*, SIGNUM Newsletter, 20 (1985), pp. 45–47.

[DGRST89] I. S. DUFF, N. I. M. GOULD, J. K. REID, J. A. SCOTT, AND K. TURNER, *The factorization of sparse symmetric indefinite matrices*, CSS Report 236, Computer Science and Systems Division, AERE Harwell, Oxford, England, 1989.

[DR82] I. S. DUFF AND J. K. REID, MA27: *A set of Fortran subroutines for solving sparse symmetric sets of linear equations*, Report R-10533, Computer Science and Systems Division, AERE Harwell, Oxford, England, 1982.

[DR83] ———, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Trans. Math. Software, 9 (1983), pp. 302–325.

[FM91] R. FOURER AND S. MEHROTRA, *Performance of an augmented system approach for solving least-squares problems in an interior-point method for linear programming*, Mathematical Programming Society COAL Newsletter, 19 (1991), pp. 26–31.

[Gay85] D. M. GAY, *Electronic mail distribution of linear programming test problems*, Mathematical Programming Society COAL Newsletter, December 1985.

[Gay89] ———, *Stopping tests that compute optimal solutions for interior-point linear programming algorithms*, Numerical Analysis Manuscript 89-11, AT&T Bell Laboratories, Murray Hill, NJ, 1989.

[GMS89] P. E. GILL, W. MURRAY, AND M. A. SAUNDERS, *A dual barrier method for linear programming using* LU *preconditioning*, presented at the SIAM Annual Meeting, San Diego, CA, 1989.

[GMSTW86] P. E. GILL, W. MURRAY, M. A. SAUNDERS, J. A. TOMLIN, AND M. H. WRIGHT, *On projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method*, Math. Programming, 36 (1986), pp. 183–209.

[GMSW87] P. E. GILL, W. MURRAY, M. A. SAUNDERS, AND M. H. WRIGHT, *Maintaining* LU *factors of a general sparse matrix*, Linear Algebra Appl., 88/89 (1987), pp. 239–270.

[GV89] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1989.

[Gou86] N. I. M. GOULD, *On the accurate determination of search directions for simple differentiable penalty functions*, IMA J. Numer. Anal., 6 (1986), pp. 357–372.

[Kar87] N. K. KARMARKAR, *Recent developments in new approaches to linear programming*, presented at the SIAM Conference on Optimization, Houston, TX, 1987.

[KR88] N. K. KARMARKAR AND K. G. RAMAKRISHNAN, *Implementation and computational results of the Karmarkar algorithm for linear programming, using an iterative method for computing projections*, Extended abstract, presented at the 13th International Symposium on Mathematical Programming, Tokyo, Japan, 1988.

[KMY88] M. KOJIMA, S. MIZUNO, AND A. YOSHISE, *A primal-dual interior-point algorithm for linear programming*, in Progress in Mathematical Programming, N. Megiddo, ed., Springer-Verlag, New York, 1988, pp. 29–48.

[LMS89] I. J. LUSTIG, R. E. MARSTEN, AND D. F. SHANNO, *Computational experience with a primal-dual interior-point method for linear programming*, Report SOR 89-17, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ, 1989.

[MMS89] K. A. MCSHANE, C. L. MONMA, AND D. F. SHANNO, *An implementation of a primal-dual interior point method for linear programming*, ORSA J. Comput., 1 (1989), pp. 70–83.

[Meg86] N. MEGIDDO, *Pathways to the optimal set in linear programming*, in Progress in Mathematical Programming, N. Megiddo, ed., Springer-Verlag, New York, 1986, pp. 131–158.

[Meh89a] S. MEHROTRA, *Implementations of affine scaling methods: Approximate solutions of systems of linear equations using preconditioned conjugate-gradient methods*, Report 89-04, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1989.

[Meh89b] ———, *On finding a vertex solution using interior-point methods*, Report 89-22, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1989.

[Meh90] ———, *On the implementation of a primal-dual interior-point method*, Report 90-03, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL, 1990.

[MLB87] C. MOLER, J. LITTLE, AND S. BANGERT, *PRO-MATLAB User's Guide*, The MathWorks, Inc., Sherborn, MA, 1987.

[Mur69] W. MURRAY, *Constrained Optimization*, Ph.D. thesis, Computer Science Department, University of London, London, England, 1969.

[PS75] C. C. PAIGE AND M. A. SAUNDERS, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12, (1975), pp. 617–629.

[Pon90] D. B. PONCELEÓN, *Barrier methods for large-scale quadratic programming*, Ph.D. thesis, Computer Science Department, Stanford University, Stanford, CA, 1990.

[Rut66] H. RUTISHAUSER, *The Jacobi method for real symmetric matrices*, Numer. Math., 9 (1966), pp. 1–10.

[Sau79] M. A. SAUNDERS, *Fortran code: A modification of SYMMLQ to allow use of a positive-definite preconditioner*, 1979; Included in *misc* chapter of *netlib*, September 1985.

[SW78] D. B. SZYLD AND O. B. WIDLUND, *Fortran code: A modification of SYMMLQ to allow use of a positive-definite preconditioner*, private communication, 1978.

[SW79] ———, *Applications of conjugate-gradient-type methods to eigenvalue calculations*, in Advances in Computer Methods for Partial Differential Equations III, R. Vichnevetsky and R. S. Stepleman, eds., IMACS, 1979, pp. 167–173.

[TZ89] R. A. TAPIA AND Y. ZHANG, *A fast optimal basis identification technique for interior point linear programming methods*, Report TR89-1, Department of Mathematical Sciences, Rice University, Houston, TX, 1989.

[Tur87] K. TURNER, *Computational experience with the projective Karmarkar algorithm*, presented at ORSA/TIMS Joint National Meeting, St. Louis, MO, 1987.

[Tur90] ———, *Computing projections for the Karmarkar algorithm*, Report 49, Department of Mathematics and Statistics, Utah State University, Logan, UT, 1990.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 312–332, January 1992

022

ALGORITHMIC FAULT TOLERANCE USING THE LANCZOS METHOD*

DANIEL L. BOLEY†, RICHARD P. BRENT‡, GENE H. GOLUB§, AND FRANKLIN T. LUK¶

Dedicated by Boley, Brent, and Luk, to their teacher Gene Golub on the occasion of his 60th birthday

Abstract. This paper considers the problem of algorithm-based fault tolerance, and makes two major contributions. First, it shows how very general sequences of polynomials can be used to generate the checksums, so as to reduce the chance of numerical overflows. Second, it shows how the Lanczos process can be applied in the error location and correction steps, so as to save on the amount of work and to facilitate actual hardware implementation.

Key words. algorithmic fault tolerance, error correction, checksums, Lanczos method

AMS(MOS) subject classifications. 65F15, 68A10

1. Background.

Many important signal processing and control problems require computational solution in real time. Much research has gone into the development of special purpose algorithms and associated hardware. The latter are usually called systolic arrays in academia, and application-specific integrated circuits (ASICs) in industry. In many critical situations, so much depends on the ability of the combined software/hardware system to deliver reliable and accurate numerical results that fault tolerance is indispensable. Often, weight constraints forbid the use of multiple modular redundancy and one must resort to a software technique to handle errors. A top choice is *Algorithm-Based Fault Tolerance* (ABFT), originally developed by Abraham and students [9], [10], to provide a low-cost error protection for basic matrix operations. Their work was extended by Luk and others [11], [13], [14] to applications that include matrix equation solvers, triangular decompositions, and recursive least squares. A theoretical framework for error correction was developed for the cases of one error [10], two errors [1], and multiple errors [7]. Interestingly, the model in [7] turns out to be the Reed–Solomon code [17]. However, the procedure proposed in [7], and implicit in [17], is cumbersome in work and quite suspect in its numerical stability.

A lot has already appeared in the literature on fault tolerant matrix algorithms, e.g., [9], [10], [11], [13], [14], [16]. A simple example is matrix multiplication. Let A and B be given square matrices of order $n+1$, and C be the desired matrix product AB. A way to achieve fault tolerance is to append the matrix B with, say, $m+1$ checksum columns, with $m \leq n$, and to calculate a checksum matrix product. Details

* Received by the editors January 25, 1991; accepted for publication (in revised form) July 30, 1991.

† Department of Computer Science, University of Minnesota, Minneapolis, Minnesota 55455 (boley@cs.umn.edu). This author's work was supported in part by National Science Foundation grant CCR-8813693.

‡ Computer Sciences Laboratory, Australian National University, Canberra, ACT 2601, Australia (rpb@cslab.anu.edu.au).

§ Department of Computer Science, Stanford University, Stanford, California 94305 (golub@na-net.stanford.edu). This author's work was supported by Army Research Office contract DAAL03-90-G-0105.

¶ School of Electrical Engineering, Cornell University, Ithaca, New York 14853 (luk@jacobi.ee.cornell.edu). This author's work was supported by Army Research Office contract DAAL03-90G-0104.

can be found in [1] and [10]. Briefly, define

$$B_r \equiv (\, B \quad S_B \,) \quad \text{and} \quad C_r \equiv (\, C \quad S_C \,),$$

where S_B and S_C denote $(n+1)\times(m+1)$ checksum matrices on B and C, respectively. Then

$$C_r = AB_r,$$

and so

$$S_C = AS_B.$$

The matrix S_C is used to detect, locate, and correct errors in C. Fault tolerant matrix multiplication is simple in that the rows of C_r can be examined independently. Indeed, denote the ith row of C_r by

$$(\, \xi_0, \xi_1, \cdots, \xi_n, \eta_0, \eta_1, \cdots, \eta_m \,), \tag{1.1}$$

where the ξ_j's represent data and the η_k's represent checksums. In [9] it is explained how, even if only one processor in the parallel system malfunctions temporarily, multiple errors will be present in the computed matrix product. In [1] it is shown that, with a judicious choice of checksum coefficients, the use of $\eta_0, \eta_1, \cdots, \eta_m$ can detect up to $m+1$ errors in $\xi_0, \xi_1, \cdots, \xi_n$, and correct up to $\lfloor (m+1)/2 \rfloor$ errors therein. A method for handling these errors is presented in [7]. Unfortunately, the procedure is quite complex, for it includes determining the rank of a matrix (to calculate the number of errors) and solving a Hankel matrix equation (to locate the errors).

A major contribution of this paper is to show how a clever use of just the Lanczos algorithm suffices for fault tolerance. The Lanczos algorithm was originally devised by Lanczos as a procedure for reducing an arbitrary matrix to a tridiagonal form having the same eigenvalues as the original matrix. The method has been found to be particularly useful for large sparse matrices and has a number of optimality properties with respect to convergence in the symmetric case. For nonsymmetric matrices the algorithm has been less well understood, but in recent years there has been major progress in the development and understanding of the algorithm (see, e.g., [3] and references therein). Our work in this paper is important in at least two aspects: (1) only simple additional hardware is necessary to implement the Lanczos scheme; (2) through the use of orthogonal polynomials, our error correction problem is numerically well conditioned. As first pointed out in [11], exponential growth in the checksum coefficients leads directly to ill-conditioning of the associated checksum matrices, which in turn leads to loss of accuracy in the computations. It is well known (see, e.g., [8]) that in solving, for example, a set of linear equations in the presence of round-off errors, a condition number of 10^x leads to a loss of about x digits of accuracy in the solutions. Ways to alleviate this difficulty were attempted in [6], [11], and [16], albeit without the success here to achieve essentially "optimal conditioning" (see §1.3). Examples illustrating the importance of matrix conditioning in fault tolerant computing can be found in [12]. We stress here once more that our signal processing applications mandate the use of floating-point data types and operations.

This paper is organized as follows. The error correction problem is described in the next two subsections. Section 2 presents the recurrence relationships for the

polynomials that generate the checksum coefficients. Sections 3 and 4 introduce the notions of Krylov matrices and error locator polynomial, respectively. The application of the Lanczos procedure to the error correction problem is discussed in §5. The Lanczos process is further simplified to a column elimination scheme in §6, and two numerical examples illustrating our ideas are given in §7.

1.1. Problem description. The data $\{\xi_j\}$ and the checksums $\{\eta_i\}$ are related via

$$\eta_i = \sum_{j=0}^{n} \nu_{ij}\xi_j, \tag{1.2}$$

where $i = 0, 1, \cdots, m$, and the coefficients $\{\nu_{ij}\}$ are prechosen. Suppose now that faulty computation has given us possibly corrupted data $\{\hat{\xi}_j\}$, but that the checksum values $\{\eta_i\}$ stay intact. The general case that includes errors in checksums will be discussed in §1.3. Hence the errors ω_j can be defined by

$$\omega_j \equiv \hat{\xi}_j - \xi_j, \tag{1.3}$$

for $j = 0, 1, \cdots, n$. Define another set of checksums $\{\hat{\eta}_i\}$ from the faulty data:

$$\hat{\eta}_i = \sum_{j=0}^{n} \nu_{ij}\hat{\xi}_j, \tag{1.4}$$

where $i = 0, 1, \cdots, m$. We note here that a major difficulty in ABFT is the proper choice of the coefficients $\{\nu_{ij}\}$. Taking the difference of (1.2) and (1.4), we get

$$\hat{\eta}_i - \eta_i = \sum_{j=0}^{n} \nu_{ij}\left(\hat{\xi}_j - \xi_j\right).$$

Defining a set of *syndromes* $\{\sigma_i\}$ by

$$\sigma_i \equiv \hat{\eta}_i - \eta_i, \quad \text{for} \quad i = 0, 1, \cdots, m,$$

we get

$$\sigma_i = \sum_{j=0}^{n} \nu_{ij}\omega_j, \quad \text{for} \quad i = 0, 1, \cdots, m. \tag{1.5}$$

Furthermore, define an $(m+1)$-element syndrome vector $\mathbf{s}$, an $(m+1) \times (n+1)$ "generator" matrix G, and an $(n+1)$-element error vector $\mathbf{w}$ by

$$\mathbf{s} \equiv \begin{pmatrix} \sigma_0 \\ \sigma_1 \\ \vdots \\ \sigma_m \end{pmatrix}, \quad G \equiv \begin{pmatrix} \nu_{00} & \nu_{01} & \cdots & \nu_{0n} \\ \nu_{10} & \nu_{11} & \cdots & \nu_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ \nu_{m0} & \nu_{m1} & \cdots & \nu_{mn} \end{pmatrix}, \quad \text{and} \quad \mathbf{w} \equiv \begin{pmatrix} \omega_0 \\ \omega_1 \\ \vdots \\ \omega_n \end{pmatrix}.$$

We can write out (1.5) in matrix form:

$$\mathbf{s} = G\mathbf{w}. \tag{1.6}$$

Given $\mathbf{s}$ and G, our *problem* is to solve for $\mathbf{w}$. Analogously, we also define the data vectors

$$\mathbf{x} = \begin{pmatrix} \xi_0 \\ \xi_1 \\ \vdots \\ \xi_n \end{pmatrix}, \qquad \hat{\mathbf{x}} = \begin{pmatrix} \hat{\xi}_0 \\ \hat{\xi}_1 \\ \vdots \\ \hat{\xi}_n \end{pmatrix}.$$

In this paper, we choose G as a *generalized Vandermonde* matrix:

$$G = \begin{pmatrix} p_0(x_0) & p_0(x_1) & p_0(x_2) & \cdots & p_0(x_n) \\ p_1(x_0) & p_1(x_1) & p_1(x_2) & \cdots & p_1(x_n) \\ p_2(x_0) & p_2(x_1) & p_2(x_2) & \cdots & p_2(x_n) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ p_m(x_0) & p_m(x_1) & p_m(x_2) & \cdots & p_m(x_n) \end{pmatrix}, \tag{1.7}$$

where $p_i(x)$ denotes a polynomial of exact degree i, for $i = 0, 1, \cdots, m$, and $\{x_j\}$ denotes a set of distinct points that we call the *knots*. Hence

$$\nu_{ij} = p_i(x_j). \tag{1.8}$$

We will also scale the zero-degree polynomial to unity:

$$p_0(x) \equiv 1.$$

In most previous work, e.g., [6], [7], [9], [10], [11], the polynomials $\{p_i(x)\}$ were chosen to be the *monomials*, viz.,

$$p_i(x) \equiv x^i, \quad \text{for} \quad i = 0, 1, \cdots, m. \tag{1.9}$$

Then G is the ordinary Vandermonde matrix:

$$G = \begin{pmatrix} x_0^0 & x_1^0 & x_2^0 & \cdots & x_n^0 \\ x_0^1 & x_1^1 & x_2^1 & \cdots & x_n^1 \\ x_0^2 & x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^m & x_1^m & x_2^m & \cdots & x_n^m \end{pmatrix}. \tag{1.10}$$

Jou and Abraham [10] chose the knots as

$$x_j = 2^j, \quad \text{for} \quad j = 0, 1, \cdots, n.$$

Recognizing that such a choice could easily lead to numerical overflow of the coefficients $\{\nu_{ij}\}$, Luk [11] proposed that

$$x_j = j + 1, \quad \text{for} \quad j = 0, 1, \cdots, n, \tag{1.11}$$

which would grow at a somewhat slower pace. Brent, Luk, and Anfinson [6] showed how one could keep ν_{ij} from exceeding a prime number that is only a little bigger than n. However, their scheme is usable only for error detection, and not for error correction. Nair and Abraham [16] explored how standard codes over a finite field may be converted to corresponding codes over the reals with various properties. In this paper, we show how other sequences of polynomials $\{p_i(x)\}$ of exact degree i can be chosen that would yield coefficients $\{\nu_{ij}\}$ that are better scaled than those arising from the monomials. In this way an efficient Lanczos method can be used for error correction. Numerical issues, however, will not be discussed, even though the sensitivity of ABFT techniques to roundoff errors is well recognized; see, e.g., [5] and [12].

1.2. Error location and correction. In [1], [9], and [10] a linear algebraic model of the weighted checksum scheme is developed, allowing parallels to be drawn between algorithm-based fault tolerance and coding theory. An assumption that we must make for our correction procedure is that no errors occur in the checksums. We now show that (1.6) with G of the form (1.7) always has at least one solution.

THEOREM 1.1. *For any n, let the knots $x_0, x_1, \cdots, x_n$ be distinct. Choose $m \leq n$. For each i, where $i = 0, 1, \cdots, m$, let the polynomial $p_i(x)$ have exact degree i. Then the matrix G of* (1.7) *has full row rank, and so any $m+1$ columns of G form an $(m+1) \times (m+1)$ nonsingular matrix.*

Proof. We show that $\mathbf{v}^T G = 0$ implies $\mathbf{v} = 0$. For any $(m+1)$-vector $\mathbf{v}$, define the polynomial $q(x)$ of degree at most m as

$$q(x) \equiv \mathbf{v}^T \begin{pmatrix} p_0(x) \\ p_1(x) \\ \vdots \\ p_m(x) \end{pmatrix}.$$

There is a one-to-one correspondence between vectors $\mathbf{v}$ and such polynomials $q(x)$. Then $\mathbf{v}^T G$ is an $(n+1)$-vector whose entries are the values that $q(x)$ takes on at all the knots x_i:

$$(q(x_0), \cdots, q(x_n)) = \mathbf{v}^T G.$$

If $q(x_i) = 0$ for all i, then $q(x)$ must be the zero polynomial. Hence $\mathbf{v} = \mathbf{0}$. The submatrix formed by extracting any $m+1$ columns of G also has the form (1.7) with $m+1$ distinct knots, so this submatrix is square and has full rank. □

We say that a coding scheme has *detected* the presence of errors if the syndrome vector $\mathbf{s}$ is nonzero, and that it can *correct* the errors if $\mathbf{x}$ can be recovered from $\hat{\mathbf{x}}$. From Theorem 1.1 it follows that $\mathbf{s}$ is guaranteed to be nonzero as long as the number of nonzero ω_i's (or errors) is between one and $m+1$. Hence our coding scheme can detect up to $m+1$ errors. The problem of error correction is harder. If $m = n$, then

$$\mathbf{w} = G^{-1}\mathbf{s}.$$

But when $m < n$, the solution to (1.6) is no longer unique, for we can find $\mathbf{w}$ by inverting any $(m+1) \times (m+1)$ submatrix of G. A usual choice (cf. [1]) is to restrict the number of errors so that there is only one solution. Let

$$\gamma = \begin{cases} m+1 & \text{if} \quad m = n, \\ \lfloor (m+1)/2 \rfloor & \text{if} \quad m < n, \end{cases} \tag{1.12}$$

and assume that at most γ errors have occurred. We claim that this $\mathbf{w}$ is unique. Assume that there is a different vector $\tilde{\mathbf{w}}$ with at most γ nonzero entries, that also satisfies (1.6). So,

$$G\mathbf{w} = \mathbf{s} \qquad \text{and} \qquad G\tilde{\mathbf{w}} = \mathbf{s}.$$

But then

$$G(\mathbf{w} - \tilde{\mathbf{w}}) = \mathbf{0},$$

and the difference vector $(\mathbf{w} - \tilde{\mathbf{w}})$ will have between one and $m+1$ nonzero elements, contradicting Theorem 1.1. From here on, we will assume that at most γ errors are

present in the data, and in §6, a Lanczos method will be presented for finding the $\mathbf{w}$ that contains at most γ nonzero elements. We summarize our results as follows.

FACT 1.1. With G defined as in Theorem 1.1, our coding scheme can detect up to $m+1$ errors, for $m+1$ given checksums, or syndromes. This coding scheme can also correct up to γ errors, in the sense that for a given set of $m+1$ syndromes, there is at most one solution to (1.6) with between 1 and γ nonzero ω_j's.

1.3. Errors in checksums and matrix conditioning. Let us illustrate some of the numerical advantages of the extra flexibility we gain from using general sets of polynomials and knots. To account for errors also in the checksums, we append an equal number of "parity" values to each of the data rows of the matrix, where the parity values are set just so the checksums are zero. Specifically, we append an equal number of parity values $\{\pi_0, \pi_1, \cdots, \pi_m\}$ to the data vector and then compute the checksums from the $(n+m+2)$-element vector of data and parity values. To do this, we need $m+1$ extra knots $x_{n+1}, x_{n+2}, \cdots, x_{n+m+1}$ corresponding to the parity values. Thus, the checksums are computed by the formula

$$\begin{pmatrix} \eta_0 \\ \eta_1 \\ \vdots \\ \eta_m \end{pmatrix} = G \begin{pmatrix} \xi_0 \\ \xi_1 \\ \vdots \\ \xi_n \end{pmatrix} + F \begin{pmatrix} \pi_0 \\ \pi_1 \\ \vdots \\ \pi_m \end{pmatrix}, \tag{1.13}$$

where F is an $(m+1) \times (m+1)$ matrix whose (i,j)th entry is $F_{ij} = p_i(x_{n+j})$. The simplest choice is to set the η_k's to zero, in which case the parity values are computed from the data values by

$$\begin{pmatrix} \pi_0 \\ \pi_1 \\ \vdots \\ \pi_m \end{pmatrix} = -F^{-1}G \begin{pmatrix} \xi_0 \\ \xi_1 \\ \vdots \\ \xi_n \end{pmatrix}. \tag{1.14}$$

Since the parity values are related to the data in the same way as the original checksums via the new coefficient matrix $F^{-1}G$, they may be carried along with the data row during all the floating-point operations in the same way as the original checksums can. With this choice for the parity values, the checksums are identically zero and hence need not be computed at all. Thus the amount of data that must be carried during the computation using this set of parity values is the same as that using the former checksum scheme with no parity values, but we gain tolerance for errors among the error check (parity) values as well as among the original data. This scheme corresponds to a *systematic linear code* in the parlance of algebraic coding theory.

When using the monomials as in [10] and [11], the condition number of F can be high. As an example, with $m = 5$ and the knots of [11]: $x_j = j+1$, the condition number of the corresponding 6×6 matrix F will be at least 7×10^5, meaning that the computed parity values will have at least five fewer digits of accuracy than the original data. This would make it impossible to detect any errors occurring in the low-order five digits of any data item. For larger values of m, this effect is even more marked: the condition number of the 8×8 undermonde matrix using knots $1, 2, \cdots, 8$ is almost 10^9.

On the other hand, if we choose the polynomials to be the Chebyshev polynomials of the first kind, we can choose the knots to substantially reduce the condition number of F. This is illustrated in the first numerical example in §7.

The accuracy of the computed parity values will make a big difference in the ability to detect and correct errors that occur in the lower part of the mantissa part of the floating-point words. When errors approach the lower part, they begin to become indistinguishable from rounding errors, and if a severe loss of accuracy occurs during the computation of parity values, hardware errors in the corresponding last digits of the floating-point word will be undetectable or uncorrectable, as they will be indistinguishable from rounding errors.

2. Recurrence relations. Define $p_{n+1}(x)$ to be the monic polynomial of degree $n+1$ whose zeros are the given knots, viz.,

$$p_{n+1}(x) = \prod_{j=0}^{n} (x - x_j) = x^{n+1} + \sum_{j=0}^{n} \zeta_j x^j, \tag{2.1}$$

for some coefficients ζ_j. The two vectors $\mathbf{s}$ and $\mathbf{w}$ are related via

$$\mathbf{s} = G\mathbf{w} = GD_\omega \mathbf{e}, \tag{2.2}$$

where D_ω is an $(n+1) \times (n+1)$ diagonal matrix given by

$$D_\omega \equiv \operatorname{diag}(\omega_0, \omega_1, \cdots, \omega_n), \tag{2.3}$$

and $\mathbf{e}$ is an $(n+1)$-vector of all ones, viz., $\mathbf{e} \equiv (1, 1, \cdots, 1)^T$.

The polynomials $\{p_i(x)\}$ satisfy a set of recurrence relations that can be grouped into the matrix expression:

$$x\ (p_0(x), \cdots, p_n(x)) = (p_0(x), \cdots, p_n(x))\ Z + (0, \cdots, 0, 1)\ p_{n+1}(x)\ \zeta_{n+1}, \tag{2.4}$$

where ζ_{n+1} equals some unspecified scalar, and Z denotes an $(n+1)\times(n+1)$ *irreducible upper Hessenberg* matrix, i.e., a matrix whose immediate subdiagonal elements are all nonzero. If all the polynomials are monic then the subdiagonals of Z are all ones. Formula (2.4) expresses each polynomial $p_{k+1}(x)$ as a linear combination of $x\,p_k(x)$ and all the previous polynomials $p_0(x)$, $p_1(x)$, $\cdots, p_k(x)$. For example, if we choose the $\{p_i(x)\}$ as the monomials, then Z is the companion matrix for the polynomial $p_{n+1}(x)$:

$$Z = \begin{pmatrix} 0 & & & & & \zeta_0 \\ 1 & 0 & & & & \zeta_1 \\ & 1 & 0 & & & \zeta_2 \\ & & 1 & \cdot & & \zeta_3 \\ & & & \cdot & \cdot & \cdot \\ & & & & 0 & \zeta_{n-1} \\ & & & & 1 & \zeta_n \end{pmatrix}. \tag{2.5}$$

In the procedure that we will describe, these scalars ζ_i will play no role in the actual computation, and so the matrix Z functions essentially as the "shift-down" matrix for the case of the monomials.

From here until the middle of §5, we will *assume* that

$$m = n.$$

In §5.1 we will show how the algorithms will still work when $m < n$. If we evaluate (2.4) at each of the knots, we obtain a relation for G:

$$D_x G^T = G^T Z, \tag{2.6}$$

where

$$D_x \equiv \text{diag}\,(x_0, x_1, \cdots, x_n), \tag{2.7}$$

because $p_{n+1}(x_i) = 0$ for $i = 0, 1, \cdots, n$. Note that we have just used our assumption that $m = n$; the matrix G is now square in computing the product $Z^T\mathbf{s}$. Equation (2.6) yields the relations

$$D_x^j G^T = G^T Z^j \quad \text{for any } j, \tag{2.8}$$

and

$$q(D_x)G^T = G^T q(Z) \quad \text{for any polynomial } q(x). \tag{2.9}$$

Furthermore, from (2.2) and (2.6), we derive the relation:

$$Z^T\mathbf{s} = Z^T G D_\omega \mathbf{e} = G D_x D_\omega \mathbf{e} = G D_\omega D_x \mathbf{e}, \tag{2.10}$$

which yields the two equations:

$$(Z^T)^j\mathbf{s} = G D_\omega D_x^j \mathbf{e} = G D_\omega \begin{pmatrix} x_0^j \\ x_1^j \\ \vdots \\ x_n^j \end{pmatrix} \quad \text{for any } j, \tag{2.11}$$

and

$$q(Z^T)\mathbf{s} = G D_\omega\, q(D_x)\mathbf{e} \quad \text{for any polynomial } q(x). \tag{2.12}$$

3. Krylov matrices. We define two sequences of Krylov matrices $\{B_i\}$ and $\{C_i\}$, to be generated by the two matrices Z and Z^T. Let $\mathbf{e}_1$ denote the $(n+1)$-element first coordinate unit vector, viz., $(1, 0, \cdots, 0)^T$. The matrix B_j is $(n+1) \times (j+1)$, and given by

$$B_j = \left(\mathbf{e}_1,\, Z\mathbf{e}_1,\, Z^2\mathbf{e}_1,\, \cdots,\, Z^j\mathbf{e}_1 \right). \tag{3.1}$$

Since Z is an irreducible upper Hessenberg matrix, the matrix B_j has full column rank and is upper triangular. The column space of B_j is the same as the column space of the first $j+1$ columns of the identity matrix. The other matrix C_j has dimensions $(n+1) \times (j+1)$, and is defined by

$$C_j = \left(\mathbf{s},\, Z^T\mathbf{s},\, (Z^T)^2\mathbf{s},\, \cdots,\, (Z^T)^j\mathbf{s} \right). \tag{3.2}$$

Note again how we have used our assumption that $m = n$. Utilizing (2.11) we may write C_j as

$$C_j = G D_\omega V_j^T, \tag{3.3}$$

where

$$V_j = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_0 & x_1 & x_2 & \cdots & x_n \\ x_0^2 & x_1^2 & x_2^2 & \cdots & x_n^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_0^j & x_1^j & x_2^j & \cdots & x_n^j \end{pmatrix}.$$

So, V_j consists of the first $(j+1)$ rows of an ordinary $(n+1) \times (n+1)$ Vandermonde matrix.

Now, how do we determine how many errors have occurred? Suppose that the number is k. Recall our assumption that

$$k \leq \gamma. \tag{3.4}$$

In (3.3), the matrix D_ω has rank k, and from Theorem 1.1, the matrix G is nonsingular and V_j^T has full column rank, which equals $(j+1)$. Hence the rank of the matrix C_j is given by

$$\text{rank } (C_j) = \begin{cases} j+1 & \text{if } \ j+1 < k, \\ k & \text{if } \ j+1 \geq k. \end{cases} \tag{3.5}$$

It also follows from Theorem 1.1 and (3.3) that the first l rows of C_j have maximal rank given by $\min\{j+1, k\}$, for any $l \geq k$. In particular, we have the following result.

LEMMA 3.1. *Let k be the number of errors (nonzero ω_j's). Denote the first k rows of B_j and C_j by $B_j^{(k)}$ and $C_j^{(k)}$, respectively. Then*

$$\text{rank } \left(B_{k-1}^{(k)}\right) = \text{rank } (B_{k-1}) = k,$$

and

$$\text{rank } \left(C_{k-1}^{(k)}\right) = \text{rank } (C_{k-1}) = k.$$

Hence $C_{k-1}^T B_{k-1} = (C_{k-1}^{(k)})^T B_{k-1}^{(k)}$ is a $k \times k$ nonsingular matrix.

Again, for the special case where $\{p_i(x)\}$ are the monomials, we get that

$$B_j = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix} \tag{3.6}$$

and

$$C_j = \begin{pmatrix} \sigma_0 & \sigma_1 & \sigma_2 & \cdots & \sigma_j \\ \sigma_1 & \sigma_2 & \sigma_3 & \cdots & \sigma_{j+1} \\ \sigma_2 & \sigma_3 & \sigma_4 & \cdots & \sigma_{j+2} \\ \vdots & \vdots & \vdots & \iddots & \vdots \\ \sigma_{n-j} & \sigma_{n-j+1} & \sigma_{n-j+2} & \cdots & \sigma_n \\ \sigma_{n-j+1} & \sigma_{n-j+2} & \sigma_{n-j+3} & \cdots & \times \\ \vdots & \vdots & \vdots & \iddots & \vdots \\ \sigma_{n-2} & \sigma_{n-1} & \sigma_n & \cdots & \times \\ \sigma_{n-1} & \sigma_n & \times & \cdots & \times \\ \sigma_n & \times & \times & \cdots & \times \end{pmatrix},$$

where $\times$ denotes elements that may not interest us.

4. Error locator polynomial. We have just seen how the number of errors can be calculated from the rank of $\{C_j\}$. A more difficult task is to find out which k of the ω_j's are nonzero. Labelling the errors as $\omega_{j_1}, \omega_{j_2}, \cdots, \omega_{j_k}$, we will show how to find the indices $j_1, j_2, \cdots, j_k$ by determining the corresponding knots $x_{j_1}, x_{j_2}, \cdots, x_{j_k}$.

DEFINITION 4.1. The *error locator polynomial* is a polynomial whose zeros are precisely the knots corresponding to the nonzero ω_j's.

Consider the $k \times (k+1)$ homogeneous system

$$C_{k-1}^T B_k \mathbf{a} = 0. \tag{4.1}$$

Denote the elements of $\mathbf{a}$ by

$$\mathbf{a} \equiv (\alpha_0, \alpha_1, \cdots, \alpha_k)^T. \tag{4.2}$$

From Lemma 3.1, a nonzero solution with $\alpha_k \neq 0$ to (4.1) exists, and is unique up to scaling. For example, if the $\{p_i(x)\}$ were the monomials, then

$$C_{k-1}^T B_k = \begin{pmatrix} \sigma_0 & \sigma_1 & \sigma_2 & \cdots & \sigma_{k-2} & \sigma_{k-1} & \sigma_k \\ \sigma_1 & \sigma_2 & \sigma_3 & \cdots & \sigma_{k-1} & \sigma_k & \sigma_{k+1} \\ \sigma_2 & \sigma_3 & \sigma_4 & \cdots & \sigma_k & \sigma_{k+1} & \sigma_{k+2} \\ \vdots & \vdots & \vdots & \iddots & \vdots & \vdots & \vdots \\ \sigma_{k-2} & \sigma_{k-1} & \sigma_k & \cdots & \sigma_{2k-4} & \sigma_{2k-3} & \sigma_{2k-2} \\ \sigma_{k-1} & \sigma_k & \sigma_{k+1} & \cdots & \sigma_{2k-3} & \sigma_{2k-2} & \sigma_{2k-1} \end{pmatrix} \tag{4.3}$$

is a Hankel matrix of syndrome values, and thus (4.1) can be regarded as *permuted Yule–Walker* equations [8, p. 184], obtained by reversing the order of the rows. With other choices for the polynomials $\{p_i(x)\}$, we can consider (4.1) as a generalization of the permuted Yule–Walker equations.

Now, in association with (4.2), define a kth-degree polynomial $q(x)$ by

$$q(x) \equiv \alpha_0 + \alpha_1 x + \cdots + \alpha_{k-1} x^{k-1} + x^k. \tag{4.4}$$

We will show that the k zeros of $q(x)$ are precisely the knots $x_{j_1}, x_{j_2}, \cdots, x_{j_k}$, corresponding to the nonzero ω-values. That is, we will show that the polynomial $q(x)$ is the error locator polynomial. Using the identity

$$\begin{aligned} G^T B_k \mathbf{a} &= G^T \left(\mathbf{e}, Z\mathbf{e}, Z^2\mathbf{e}, \cdots, Z^k\mathbf{e}\right) \mathbf{a} \\ &= \left(\mathbf{e}, D_x\mathbf{e}, D_x^2\mathbf{e}, \cdots, D_x^k\mathbf{e}\right) \mathbf{a} \\ &= q(D_x)\,\mathbf{e}, \end{aligned} \tag{4.5}$$

we expand (4.1) as follows:

$$0 = C_{k-1}^T B_k \mathbf{a} = V_{k-1} D_\omega G^T B_k \mathbf{a} = V_{k-1} D_\omega q(D_x)\,\mathbf{e} = V_{k-1} \begin{pmatrix} \omega_0 q(x_0) \\ \omega_1 q(x_1) \\ \vdots \\ \omega_n q(x_n) \end{pmatrix}.$$

If we extract only those entries involving the nonzero ω-values, we obtain a $k \times k$ nonsingular, homogeneous system:

$$0 = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_{j_1} & x_{j_2} & x_{j_3} & \cdots & x_{j_k} \\ x_{j_1}^2 & x_{j_2}^2 & x_{j_3}^2 & \cdots & x_{j_k}^2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{j_1}^{k-1} & x_{j_2}^{k-1} & x_{j_3}^{k-1} & \cdots & x_{j_k}^{k-1} \end{pmatrix} \begin{pmatrix} \omega_{j_1} q(x_{j_1}) \\ \omega_{j_2} q(x_{j_2}) \\ \omega_{j_3} q(x_{j_3}) \\ \vdots \\ \omega_{j_k} q(x_{j_k}) \end{pmatrix}. \tag{4.6}$$

Hence

$$q(x_{j_i}) = 0 \quad \text{for} \quad i = 1, 2, \cdots, k,$$

as we desired.

We can also express the error locator polynomial as a linear combination of the original set of polynomials $\{p_i(x)\}$, which may be useful in the computational procedure. We define a polynomial $r(x)$ by

$$r(x) \equiv (p_0(x), p_1(x), \cdots, p_n(x))\ B_k \mathbf{a}. \tag{4.7}$$

From the upper triangular structure of the matrix B_k, we see that $r(x)$ is a polynomial of degree at most k. If we evaluate $r(x)$ at each knot, we see from (4.5) that it agrees with $q(x)$ at every knot:

$$\begin{pmatrix} r(x_0) \\ r(x_1) \\ \vdots \\ r(x_n) \end{pmatrix} = G^T B_k \mathbf{a} = q(D_x)\mathbf{e} = \begin{pmatrix} q(x_0) \\ q(x_1) \\ \vdots \\ q(x_n) \end{pmatrix}. \tag{4.8}$$

So,

$$r(x) \equiv q(x). \tag{4.9}$$

We summarize our results as follows.

FACT 4.1. Suppose that there are exactly k errors, and that the syndrome vector $\mathbf{s}$ has been computed using G, which is generated via the recurrence matrix Z. Let B_k and C_{k-1} be the two Krylov matrices generated by Z, e_1, and $\mathbf{s}$. Then the error locator polynomial $q(x)$ is defined by (4.4), where the vector of coefficients $\mathbf{a}$ is the unique (up to scaling) nonzero solution to (4.1). Furthermore, we may express $q(x)$ as a linear combination of the original polynomials $\{p_i(x)\}$, in which case the coefficients are simply the entries of the vector $B_k\mathbf{a}$, as proved in (4.7) and (4.9).

5. Lanczos process. The nonsymmetric Lanczos algorithm, described in detail in [3], is a recursive process that starts with the matrix A and two vectors $\mathbf{r}_0$ and $\mathbf{l}_0$, and generates two sequences of matrices $\{R_j\}$ and $\{L_j\}$, given by

$$R_j \equiv (\mathbf{r}_0, \mathbf{r}_1, \cdots, \mathbf{r}_j) \tag{5.1}$$

and

$$L_j \equiv (\mathbf{l}_0, \mathbf{l}_1, \cdots, \mathbf{l}_j), \tag{5.2}$$

for $j = 0, 1, \cdots, n$. Hence R_j and L_j are both $(n+1) \times (j+1)$ matrices. Let $\mathrm{Sp}(M)$ denote the column space of a matrix M. Then, for every j, the following four relations will be satisfied:

$$\mathrm{Sp}(R_j) = \mathrm{Sp}\left(\mathbf{r}_0,\ A\mathbf{r}_0,\ A^2\mathbf{r}_0, \cdots,\ A^j\mathbf{r}_0\right), \tag{5.3}$$

$$\mathrm{Sp}(L_j) = \mathrm{Sp}\left(\mathbf{l}_0,\ A^T\mathbf{l}_0,\ (A^T)^2\mathbf{l}_0, \cdots,\ (A^T)^j\mathbf{l}_0\right), \tag{5.4}$$

and, if $L_j^T R_j$ is nonsingular, then

$$\mathbf{l}_{j+1}^T R_j = 0, \tag{5.5}$$

$$\mathbf{r}_{j+1}^T L_j = 0. \tag{5.6}$$

Property (5.4) implies that $\mathbf{l}_{j+1}$ is a linear combination of $A^T\mathbf{l}_j$ and $\mathbf{l}_0, \mathbf{l}_1, \cdots, \mathbf{l}_j$. If the matrix $L_j^T R_j$ is nonsingular, then the particular linear combination is chosen to satisfy (5.5). Otherwise, we have some freedom in choosing $\mathbf{l}_{j+1}$, and we may pick $\mathbf{l}_{j+1} = A^T\mathbf{l}_j$. Other choices satisfying (5.3)–(5.4) are possible, but this particular one will lead to a computational simplification, as will be seen in the next section. We can derive similar results for the $\{\mathbf{r}_i\}$ vectors. The process terminates when either $\mathbf{l}_j = 0$ or $\mathbf{r}_j = 0$, for some j.

For our situation, we propose to use the Lanczos process with the matrix $A = Z$, and the starting vectors $\mathbf{r}_0 = \mathbf{e}_1$ and $\mathbf{l}_0 = \mathbf{s}$. With this choice, we get

$$\mathrm{Sp}(R_j) = \mathrm{Sp}(B_j) \tag{5.7}$$

and

$$\mathrm{Sp}(L_j) = \mathrm{Sp}(C_j) \tag{5.8}$$

for $j = 0, 1, \cdots, n$. Since the matrix Z is irreducible upper Hessenberg, the matrix R_j will be upper triangular and will have full column rank $j+1$ for every $j < k$. Hence the Lanczos process will terminate at the kth step with $\mathbf{l}_k = 0$, by (3.4). Since the matrix $C_{k-1}^T B_{k-1}$ is nonsingular, we get from (5.6) that $\mathbf{r}_k$ will be the vector in the column space of R_k that is orthogonal to L_{k-1}, or equivalently in the column space of B_k that is orthogonal to C_{k-1}. But this means that the vector $\mathbf{r}_k$ equals the vector $B_k\mathbf{a}$ defined by (4.1), up to a scaling constant. Hence the Lanczos algorithm may be used to generate the error locator polynomial $q(x)$ as a linear combination of the original set of polynomials $\{p_i(x)\}$.

FACT 5.1. Suppose that we have run the nonsymmetric Lanczos process with the matrix Z, and the starting vectors $\mathbf{e}_1$ and $\mathbf{s}$. The process will terminate at the kth step with $\mathbf{l}_k = 0$, and the vector $\mathbf{r}_k$ will equal the vector $B_k\mathbf{a}$, except possibly for a scaling factor. Hence the entries of $\mathbf{r}_k$ give us the coefficients of the error locator polynomial $q(x)$ in terms of the original set of polynomials $\{p_i(x)\}$, as shown in (4.7) and (4.9).

5.1. Case where $m < n$. We now examine the case where there are fewer syndromes than data values, i.e., $m < n$. Indeed in practice, usually $m << n$. As noted just below (3.4), we may check the rank of C_j for every j by just checking the first l rows, as long as $l \geq k$. If γ, the maximum number of errors, is known, then it suffices to examine the first γ rows of C_j, or equivalently of L_j, and the Lanczos process is guaranteed to terminate in at most γ steps. Note that each $\mathbf{l}_{j+1}$ in the Lanczos process is a linear combination of $Z^T\mathbf{l}_j$ and of $\mathbf{l}_0, \mathbf{l}_1, \cdots, \mathbf{l}_j$. Since Z^T is lower Hessenberg and $\mathbf{l}_j$ has j leading zero elements, to find the top ρ elements of $Z^T\mathbf{l}_j$, we need to know only the top $\rho+1$ elements of $\mathbf{l}_j$. It follows that the first $\gamma+1$ values of the generated vectors $\mathbf{l}_0, \cdots, \mathbf{l}_{\gamma-1}$ depend only upon the first 2γ values of the initial vector $\mathbf{l}_0 = \mathbf{s}$. Therefore, it suffices to compute only 2γ syndrome values in order to generate the coefficients in (4.1) that are needed to solve for $\mathbf{a}$. Recall from (1.12) our assumption that

$$\gamma = \lfloor (m+1)/2 \rfloor,$$

and so, given γ, one should choose m so that

$$m + 1 = 2\gamma.$$

FACT 5.2. If the number of errors is at most γ, then 2γ syndrome values are necessary and sufficient to determine the error locator polynomial and its zeros by means of (4.1).

6. Column elimination scheme. The Lanczos process as described somewhat simplifies, for the particular purpose we are using it here, that of computing the error locator polynomial. With our particular starting data, the right Lanczos matrix R_j will be upper triangular and will have full column rank for every j. The left Lanczos matrix L_j may assume several forms. We first examine the "generic" case that $L_j^T R_j$ is nonsingular for every j. Then the condition (5.5) is equivalent to forcing L_{j+1} to be lower triangular. In the Lanczos algorithm, this structure is obtained by subtracting multiples of previous columns $\{\mathbf{l}_i\}$ from the one that has just been generated as $Z^T\mathbf{l}_j$. That is, we perform "column operations" akin to "row operations" in ordinary Gaussian elimination. (Note that using another elimination scheme, such as an orthogonal decomposition, would destroy the properties (5.3) and (5.4) as well as the triangular structure of the generated matrices.) Thus, at stage j of the process, we generate $\mathbf{l}_{j+1}$ from $\mathbf{l}_{j-1}$ and $\mathbf{l}_j$ as follows. The vector $\mathbf{l}_{j-1}$ has $(j-1)$ leading zero entries, the vector $\mathbf{l}_j$ has j leading zero entries, and the vector $Z^T\mathbf{l}_j$ $(\equiv \tilde{\mathbf{l}}_{j+1})$ has $(j-1)$ zero entries:

$$\left(\mathbf{l}_{j-1},\quad \mathbf{l}_j,\quad \tilde{\mathbf{l}}_{j+1}\right) = \begin{pmatrix} 0 & 0 & 0 \\ \vdots & \vdots & \vdots \\ 0 & 0 & 0 \\ l_{j-1,j-1} & 0 & \tilde{l}_{j-1,j+1} \\ l_{j,j-1} & l_{j,j} & \tilde{l}_{j,j+1} \\ l_{j+1,j-1} & l_{j+1,j} & \tilde{l}_{j+1,j+1} \\ \vdots & \vdots & \vdots \end{pmatrix}.$$

We must eliminate the two elements $\tilde{l}_{j-1,j+1}$ and $\tilde{l}_{j,j+1}$ to obtain an $\mathbf{l}_{j+1}$ that has $j+1$ leading zero entries. These two eliminations are done by subtracting from $\tilde{\mathbf{l}}_{j+1}$ suitable multiples of $\mathbf{l}_{j-1}$ and $\mathbf{l}_j$, respectively.

We now examine the "nongeneric" case. Suppose that for some particular value of j, the matrix $L_j^T R_j$ is singular and $L_{j-1}^T R_{j-1}$ is nonsingular. (The following also applies for the case where $j=0$, i.e., $\mathbf{l}_0^T\mathbf{r}_0 = 0$.) This means that $\mathbf{l}_j$ must have more than j leading zero entries. Suppose there are i extra leading zero entries, for a total of $j+i$ leading zero entries:

$$\mathbf{l}_j = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ l_{j+i,j} \\ l_{j+i+1,j} \\ \vdots \end{pmatrix}. \tag{6.1}$$

Then the next $(i+1)$ vectors $\tilde{\mathbf{l}}_{j+1}, \tilde{\mathbf{l}}_{j+2}, \cdots, \tilde{\mathbf{l}}_{j+i+1}$ are defined simply by

$$\tilde{\mathbf{l}}_{j+l} = Z^T\tilde{\mathbf{l}}_{j+l-1} = (Z^T)^l\tilde{\mathbf{l}}_j$$

for $l = 1, 2, \cdots, i+1$. Due to the lower Hessenberg form of Z^T, these vectors have a lower antitriangular form, as illustrated below for $i=3$:

$$(6.2)\quad \left(\mathbf{l}_{j-1}, \mathbf{l}_j, \tilde{\mathbf{l}}_{j+1}, \tilde{\mathbf{l}}_{j+2}, \tilde{\mathbf{l}}_{j+3}, \tilde{\mathbf{l}}_{j+4}\right) =$$

$$\begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 \\ l_{j-1,j-1} & 0 & 0 & 0 & 0 & \tilde{l}_{j-1,j+4} \\ l_{j,j-1} & 0 & 0 & 0 & \tilde{l}_{j,j+3} & \tilde{l}_{j,j+4} \\ l_{j+1,j-1} & 0 & 0 & \tilde{l}_{j+1,j+2} & \tilde{l}_{j+1,j+3} & \tilde{l}_{j+1,j+4} \\ l_{j+2,j-1} & 0 & \tilde{l}_{j+2,j+1} & \tilde{l}_{j+2,j+2} & \tilde{l}_{j+2,j+3} & \tilde{l}_{j+2,j+4} \\ l_{j+3,j-1} & l_{j+3,j} & \tilde{l}_{j+3,j+1} & \tilde{l}_{j+3,j+2} & \tilde{l}_{j+3,j+3} & \tilde{l}_{j+3,j+4} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{pmatrix}.$$

We wish to "exhibit" the rank of this matrix by reducing it to a column permutation of a lower triangular matrix. To preserve the Krylov sequence property (5.4), we set

$$\mathbf{l}_j = \tilde{\mathbf{l}}_j,\ \mathbf{l}_{j+1} = \tilde{\mathbf{l}}_{j+1},\ \cdots,\ \mathbf{l}_{j+i} = \tilde{\mathbf{l}}_{j+i}.$$

Then to form $\mathbf{l}_{j+i+1}$ we must eliminate the leading $(i+2)$ nonzero entries of $\tilde{\mathbf{l}}_{j+i+1}$, namely, $\tilde{l}_{j-1,j+i+1}, \tilde{l}_{j,j+i+1}, \cdots, \tilde{l}_{j+i,j+i+1}$, by means of column operations. Note that (5.4) is still preserved for all the index values j, $j+1$, $\cdots, j+i+1$, and that this column elimination scheme is essentially the Berlekamp–Massey algorithm [2], [15] for solving the permuted Yule–Walker problem (4.1) when the coefficient matrix is given by (4.3). The scheme requires $i+2$ column operations, less than the $2(i+1)$ that would have been required for the generic case.

Whether the generic or nongeneric elimination scheme is used, the result after k steps is a full rank matrix L_{k-1}, which is either lower triangular or a column permutation of a lower triangular matrix. Therefore, to solve for $\mathbf{a}$ in (4.1), it suffices to choose an arbitrary nonzero value for $\tilde{\alpha}_k$ and solve the following system for $\tilde{\alpha}_0, \cdots, \tilde{\alpha}_{k-1}$:

$$(6.3)\qquad L_{k-1}^T (R_k \tilde{\mathbf{a}}) = \mathbf{0}.$$

As $\mathrm{Sp}(C_{k-1}) = \mathrm{Sp}(L_{k-1})$, and hence $\mathrm{Null}\left(C_{k-1}^T\right) = \mathrm{Null}\left(L_{k-1}^T\right)$, the right annihilating vector $B_k \mathbf{a}$ of C_{k-1}^T in (4.1) is the same as the right annihilating vector $R_k \tilde{\mathbf{a}}$ of L_{k-1}^T in (6.3). That is,

$$(6.4)\qquad B_k \mathbf{a} = R_k \tilde{\mathbf{a}},$$

except for a scaling constant. The matrix R_k is upper triangular, so it suffices to extract only the first $(k+1)$ rows of L_{k-1}. The $(k+1)$st row of L_{k-1} enters only into the part depending on $\tilde{\alpha}_k$, so (6.3) is a $k \times k$ system for the remaining $\tilde{\alpha}$-values. Since L_{k-1} is lower triangular (at least within a column permutation), solving (6.3) for $R_k \tilde{\mathbf{a}} = B_k \mathbf{a}$ requires a back-substitution step, and to obtain $\mathbf{a}$ itself requires another back-substitution step. If we are interested only in the locations of the zeros of the error locator polynomial, as opposed to the coefficients of the polynomial itself, it suffices to solve (6.3) for the right annihilating vector $B_k \mathbf{a}$ and substitute this result directly into (4.8), yielding directly the values of the error locator polynomial evaluated at every knot.

We summarize the steps to obtain $\mathbf{a}$ as follows:

0. Start with $\mathbf{l}_0 = \mathbf{s}$.

1. For $i = 0, 1, \cdots$, compute $\mathbf{l}_{i+1}$ by forming $\tilde{\mathbf{l}}_{i+1} = Z^T \mathbf{l}_i$, and annihilating the first two nonzero entries by two "column operations." (In the nongeneric case described above, we form the vectors as illustrated in (6.2) and follow the prescription described thereafter.)
2. The process in step 1 continues until $\mathbf{l}_k = 0$ for some value k. This value is the number of errors in the data.
3. Solve system (6.3) for $B_k \mathbf{a}$ using (6.4).
4. If the error locator polynomial itself is desired, as opposed to its zeros, then generate the Krylov matrix B_k, and back-solve to obtain $\mathbf{a}$.

Step 1 is guaranteed to terminate in at most γ steps. This requires that only the first $m+1$, which equals 2γ, syndromes be retained, and that only the first $m+1-j$ entries in each $\mathbf{c}_j$ vector be computed. Therefore, to carry out step 1 requires at most γ applications of the matrix Z^T and 2γ "column operations."

If the bandwidth of Z equals b, then each application of Z or Z^T to the leading $m+1$ entries of a vector costs $(m+1)b$ operations. Since only the leading principal submatrix of Z participates in the computations, the bandwidth of (2.5) is $b = 1$. Indeed, that particular choice would incur only shifting and no arithmetic costs. If the original set of polynomials $\{p_i(x)\}$ were the Chebyshev polynomials, or some other sequence of orthogonal polynomials, then Z would be tridiagonal and the bandwidth would be $b = 3$, as in the numerical example in the next section.

Only steps 1 and 3 require computation for the specific syndrome values: we approximate their costs using $k \leq \gamma$ and $m = 2\gamma - 1$.

Cost of step 1 $= k(mb + 2m) \leq 2\gamma^2(b+2)$.
Cost of step 3 = Cost of back-substitutions $= k^2/2 \leq \gamma^2/2$.
Total cost $\leq \gamma^2(2b + 4.5)$.

To summarize, we have described a method that computes a lower triangular basis for the Krylov space, $\mathrm{Sp}\,(C_{k-1})$. By recursively carrying out column eliminations as each new column is generated, we are able to exhibit the maximal rank of R_j and L_j at each stage j. We then use the lower triangular basis L_{k-1} to solve for the vector $B_k \mathbf{a}$ representing the error locator polynomial. In principle, we could use *any* basis for $\mathrm{Sp}\,(C_{k-1})$. If we used instead an orthonormal basis, we would enhance the numerical stability of the method at a cost of more arithmetic. Such an orthonormal basis can be generated recursively by an Arnoldi process (see, e.g., [4]), and the rank would be exhibited in the same way as in the above process. But in this paper we focus on the lower triangular basis because it is simple to compute and because it is closely related to the nonsymmetric Lanczos and Berlekamp–Massey algorithms.

7. Numerical examples. Except for the possible goal of reducing the condition number of the relevant matrices, the choice of polynomials and knots is arbitrary as long as the polynomials are of increasing degree and the knots are distinct. These are the only conditions required to apply the Lanczos-based paradigm. Different choices lead to a wide variety of different schemes, including many of the standard ones. In this section we illustrate our method with two particular numerical examples in which we use the Chebyshev polynomials and the monomials to generate the checksum coefficients and the knots. In printing the numbers, we have rounded them to the digits shown, even though the computations were carried out in Lisp on a Sun workstation with IEEE arithmetic using a precision of about 16 decimal digits. The first two Chebyshev polynomials are

$$p_0(x) = 1, \qquad p_1(x) = x,$$

and it is well known that the subsequent polynomials are generated by the recurrence

$$p_{i+1}(x) = 2x\,p_i(x) - p_{i-1}(x), \quad \text{for } i = 1, 2, \cdots.$$

The Chebyshev polynomials $p_0(x), p_1(x), \cdots$ are related via the recurrence (2.4) and the recurrence matrix

$$\frac{1}{2}\begin{pmatrix} 0 & 1 & 0 & 0 & \\ 2 & 0 & 1 & 0 & \ddots \\ 0 & 1 & 0 & 1 & \ddots \\ 0 & 0 & 1 & 0 & \ddots \\ 0 & 0 & 0 & 1 & \ddots \\ 0 & 0 & 0 & 0 & \ddots \\ & \ddots & \ddots & \ddots & \ddots \end{pmatrix}, \tag{7.1}$$

and it is well known that the zeros of the polynomial p_k are all real, simple, and are the same as the eigenvalues of the leading $k \times k$ principal submatrix of (7.1).

Example 1. We illustrate the process of determining the errors that might be present in a given row $(\xi_0, \xi_1, \cdots, \xi_n)$ of an $(n+1) \times (n+1)$ matrix A. In order to compute the checksums, we need $n+1$ knots $x_0, x_1, \cdots, x_n$, from which we determine the matrix G of checksum coefficients (1.7) using the Chebyshev polynomials.

In order to illustrate the process for also handling errors in the checksums, we suppose that $m+1 = 6$ parity values $\pi_0, \pi_1, \cdots, \pi_5$ have been appended to the matrix row, and that an extra six knots have been chosen. Corresponding to the parity values are six extra knots $x_{n+1}, x_{n+2}, \cdots, x_{n+6}$. Define the 6×6 matrix F by $F_{ij} = p_{i-1}(x_{n+j})$ for $i, j = 1, 2, \cdots, 6$. We then define the parity values by (1.14) so that the checksums (1.13) computed on the entire "data sequence"

$$\xi_0, \xi_1, \cdots, \xi_n, \pi_0, \pi_1, \cdots \pi_5 \tag{7.2}$$

are zero.

We may choose the knots to be any set of distinct numbers, so we make the following arbitrary choice. The last eight knots are chosen as the zeros of p_8:

$$\begin{aligned} x_{n-1} &= \cos 15\pi/16 = -0.980785 \\ x_{n+0} &= \cos 13\pi/16 = -0.831470 \\ x_{n+1} &= \cos 11\pi/16 = -0.555570 \\ x_{n+2} &= \cos\ 9\pi/16 = -0.195090 \\ x_{n+3} &= \cos\ 7\pi/16 = +0.195090 \\ x_{n+4} &= \cos\ 5\pi/16 = +0.555570 \\ x_{n+5} &= \cos\ 3\pi/16 = +0.831470 \\ x_{n+6} &= \cos\ \pi/16 = +0.980785 \end{aligned}$$

and the $n-1$ remaining knots are chosen as the zeros of p_{n-1}, which are all distinct from those of p_8 for any $n-1$ relatively prime to 8. All $n+7$ knots are guaranteed to be distinct. This is a modification of a periodic code in the sense of [16]. We have that $m = 5$ and $\gamma = 3$. The combined matrix $(G \mid F)$ of checksum coefficients is given by

$$G = \begin{pmatrix} \cdots & 1.0000 & 1.0000 \\ \cdots & -0.9808 & -0.8315 \\ \cdots & 0.9238 & 0.3827 \\ \cdots & -0.8315 & 0.1951 \\ \cdots & 0.7071 & -0.7071 \\ \cdots & 0.5556 & 0.9808 \end{pmatrix},$$

and

$$F = \begin{pmatrix} 1.0000 & 1.0000 & 1.0000 & 1.0000 & 1.0000 & 1.0000 \\ -0.5556 & -0.1951 & 0.1951 & 0.5556 & 0.8315 & 0.9808 \\ -0.3827 & -0.9238 & -0.9238 & -0.3827 & 0.3827 & 0.9238 \\ 0.9808 & 0.5556 & -0.5556 & -0.9808 & -0.1951 & 0.8315 \\ -0.7071 & 0.7071 & 0.7071 & -0.7071 & -0.7071 & 0.7071 \\ -0.1951 & -0.8315 & 0.8315 & 0.1951 & -0.9808 & 0.5556 \end{pmatrix}.$$

We note that the condition number of F is 89. If instead we use the polynomials (1.9) and knots (1.11), then G and F would both be ordinary Vandermonde matrices, and the condition number of F would be in excess of 7×10^5. This means that by using the Chebyshev polynomials, we may compute the parity values (1.14) to almost full machine accuracy, whereas when using (1.9) and (1.11), the last five digits of the computed parity values are guaranteed to be in error from the ill-conditioning of F. In the latter case, we would not be able to detect any errors that might occur in the last five digits of any data item. We note that if we choose the last six knots as the zeros of p_6 and the remaining $n+1$ knots as the zeros of p_{n+1} then it can be shown that the rows of F would be mutually orthogonal and the condition number of F would be reduced to only 2. We use the choice of p_8 and p_{n-1} to illustrate that many different choices can lead to much improvement in the condition number.

Given the $n+7$ knots, the checksum coefficients are generated by the first six Chebyshev polynomials, which satisfy the recurrence (2.4) where the recurrence matrix Z is just the leading 6×6 principal submatrix of (7.1). The Krylov sequence $\{B_j\}$ depends only on the recurrence matrix Z. In particular, B_5 is given by

$$B_5 = \begin{pmatrix} 1.0 & 0 & 0.5 & 0 & 0.375 & 0 \\ 0 & 1.0 & 0 & 0.75 & 0 & 0.6250 \\ 0 & 0 & 0.5 & 0 & 0.500 & 0 \\ 0 & 0 & 0 & 0.25 & 0 & 0.3125 \\ 0 & 0 & 0 & 0 & 0.125 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.0625 \end{pmatrix}. \tag{7.3}$$

Since the parity values were chosen just to make the checksums zero, the syndrome values are obtained by applying the checksum coefficient matrix $(G \mid F)$ to the augmented data (7.2). We suppose that the resulting six $(= 2\gamma)$ syndromes are

$$\mathbf{s} = \begin{pmatrix} -2.000000 \\ -2.443301 \\ 4.460885 \\ 2.612462 \\ -4.242641 \\ -1.364308 \end{pmatrix}.$$

The other Krylov sequence $\{C_j\}$ is calculated as

$$C_j = \begin{pmatrix} -2.0000 & -2.4433 & 1.2304 & \cdots \\ -2.4433 & 1.2304 & -1.1794 & \cdots \\ 4.4609 & 0.0845 & 0.6698 & \cdots \\ 2.6125 & 0.1091 & 0.3543 & \cdots \\ -4.2426 & 0.6241 & \times & \cdots \\ -1.3643 & \times & \times & \cdots \end{pmatrix},$$

where $j \geq 2$, and the symbol "$\times$" stands for entries depending on the further syndrome values that we do not have available, and do not need under the assumption that no

more than three errors have occurred. If we carry out column eliminations to reduce C_j to a lower triangular form, we obtain

$$L_j = \begin{pmatrix} -2.0000 & 0 & 0 & \cdots \\ -2.4433 & 4.2153 & 0 & \cdots \\ 4.4609 & -5.3651 & 0 & \cdots \\ 2.6125 & -3.0824 & 0 & \cdots \\ -4.2426 & 5.8071 & \times & \cdots \\ -1.3643 & \times & \times & \cdots \end{pmatrix},$$

where $j \geq 2$. Note that the third column is all zero, so number k of errors equals 2. We then use the top left 3×3 part of B_5 and the top left 3×2 part of L_2 to solve (6.3):

$$0 = \begin{pmatrix} -2.0000 & -2.4433 & 4.4609 \\ 0 & 4.2153 & -5.3651 \end{pmatrix} \begin{pmatrix} 1.0 & 0 & 0.5 \\ 0 & 1.0 & 0 \\ 0 & 0 & 0.5 \end{pmatrix} \begin{pmatrix} \alpha_0 \\ \alpha_1 \\ 1 \end{pmatrix}$$

for the 3-vector $B_2\mathbf{a}$. This is a 2×2 system of equations for α_0, α_1, but we can solve directly for $B_2\mathbf{a}$. The results are

$$B_2^{(3)}\mathbf{a} = \begin{pmatrix} 0.3378 \\ 0.6364 \\ 0.5000 \end{pmatrix} \quad \text{and} \quad \mathbf{a} = \begin{pmatrix} -0.1622 \\ 0.6364 \\ 1.0000 \end{pmatrix}.$$

Thus the error locator polynomial is

$$q(x) = x^2 + 0.6364x - 0.1622.$$

We can find the zeros of $q(x)$ directly, or substitute $B_2^{(3)}\mathbf{a}$ directly into (4.8) to obtain the vector of evaluations of the error locator polynomial at all the knots:

$$(G \mid F)^T B_2\mathbf{a} = \begin{pmatrix} \vdots & \vdots & \vdots \\ 1.0 & -0.9808 & 0.9238 \\ 1.0 & -0.8315 & 0.3827 \\ 1.0 & -0.5556 & -0.3827 \\ 1.0 & -0.1951 & -0.9238 \\ 1.0 & 0.1951 & -0.9238 \\ 1.0 & 0.5556 & -0.3827 \\ 1.0 & 0.8315 & 0.3827 \\ 1.0 & 0.9808 & 0.9238 \end{pmatrix} \begin{pmatrix} 0.3378 \\ 0.6364 \\ 0.5000 \end{pmatrix} = \begin{pmatrix} \vdots \\ 0.1756 \\ O(10^{-16}) \\ -0.2071 \\ -0.2483 \\ O(10^{-16}) \\ 0.5000 \\ 1.0583 \\ 1.4238 \end{pmatrix}.$$

The locations of the $O(10^{-16})$ entries indicate that the zeros of the error locator polynomial are -0.8315 and -0.1951, which are the knots corresponding to the locations of the nonzero ω-values, ω_n and ω_{n+3}, which in turn are the errors in the last data item ξ_n and the third parity value π_2, respectively. We can then extract the corresponding columns from (1.6) to obtain the 2×2 system to solve for the ω-values:

$$\begin{pmatrix} -2.0000 \\ -2.4433 \end{pmatrix} = \begin{pmatrix} 1.0000 & 1.0000 \\ -0.8315 & 0.1951 \end{pmatrix} \begin{pmatrix} \omega_n \\ \omega_{n+3} \end{pmatrix}$$

yielding the result

$$\begin{pmatrix} \omega_n \\ \omega_{n+3} \end{pmatrix} = \begin{pmatrix} 2.0000 \\ -4.0000 \end{pmatrix}.$$

Example 2. We consider a second numerical example illustrating the nongeneric procedure and the permuted lower triangular structure (6.2). To simplify the exposition, we do not use any parity values and ignore conditioning issues. We suppose we have knots $x_j = j+1$, $j = 0, 1, 2, \cdots$, and polynomials $p_i(x) = x^i$, $i = 0, 1, 2, \cdots$. The matrix G is the ordinary Vandermonde matrix

$$G = \begin{pmatrix} 1 & 1 & 1 & 1 & \cdots \\ 1 & 2 & 3 & 4 & \cdots \\ 1 & 4 & 9 & 16 & \cdots \\ 1 & 8 & 27 & 64 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}.$$

Suppose that we start with the syndrome vector

$$\mathbf{s} = (1, 0, 0, 0, -24, -240, -1560, -8400, \cdots)^T.$$

We have eight syndrome values, allowing up to four errors. Then the Krylov sequence would be generated by a "shift-down" matrix yielding

$$C_4 = \begin{pmatrix} 1 & 0 & 0 & 0 & -24 \\ 0 & 0 & 0 & -24 & -240 \\ 0 & 0 & -24 & -240 & -1560 \\ 0 & -24 & -240 & -1560 & -8400 \\ -24 & -240 & -1560 & -8400 & \times \\ -240 & -1560 & -8400 & \times & \times \\ -1560 & -8400 & \times & \times & \times \\ -8400 & \times & \times & \times & \times \end{pmatrix}.$$

The Krylov matrix B_4 of (3.6) is just the first five columns of an identity matrix. When we attempt to reduce C_4 to the lower triangular form L_4 by column operations, we find that the second column $\mathbf{c}_1$ has three leading zeros. Hence, we get

$$\mathbf{l}_1 = \mathbf{c}_1$$

without any elimination at all, and furthermore we have two additional leading zero elements. So we generate the next two $\mathbf{l}$ vectors by

$$\mathbf{l}_2 = \mathbf{c}_2 \quad \text{and} \quad \mathbf{l}_3 = \mathbf{c}_3.$$

The next vector $\mathbf{l}_4$ is obtained by eliminating the first four elements of $\tilde{\mathbf{l}}_4$ (in this case the same as $\mathbf{c}_4$) by means of column operations. The result is

$$L_4 = (\mathbf{l}_0, \mathbf{l}_1, \mathbf{l}_2, \mathbf{l}_3, \mathbf{l}_4) = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -24 & 0 \\ 0 & 0 & -24 & -240 & 0 \\ 0 & -24 & -240 & -1560 & 0 \\ -24 & -240 & -1560 & -8400 & \times \\ -240 & -1560 & -8400 & \times & \times \\ -1560 & -8400 & \times & \times & \times \\ -8400 & \times & \times & \times & \times \end{pmatrix}. \tag{7.4}$$

Since the last column is zero, the maximal rank k equals 4, so we have four errors. We then solve (4.1) for the coefficients $\mathbf{a}$ of the error locator polynomial $q(x)$. That is, we solve

$$\begin{pmatrix} 1 & 0 & 0 & 0 & -24 \\ 0 & 0 & 0 & -24 & -240 \\ 0 & 0 & -24 & -240 & -1560 \\ 0 & -24 & -240 & -1560 & -8400 \end{pmatrix} \mathbf{a} = \mathbf{0},$$

obtaining the error locator polynomial

$$q(x) = x^4 - 10x^3 + 35x^2 - 50x + 24.$$

The zeros of q are 1, 2, 3, and 4, indicating that the errors occur in the first four positions. To find the actual errors, we extract the top left 4×4 part of (1.6):

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & 2 & 3 & 4 \\ 1 & 4 & 9 & 16 \\ 1 & 8 & 27 & 64 \end{pmatrix} \begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix},$$

and solve this to obtain the errors

$$\begin{pmatrix} \omega_0 \\ \omega_1 \\ \omega_2 \\ \omega_3 \end{pmatrix} = \begin{pmatrix} 4 \\ -6 \\ 4 \\ -1 \end{pmatrix}.$$

Acknowledgments. This paper is dedicated to the memory of Jeffrey Speiser; we thank him for his invaluable role in cross-fertilizing the fields of numerical analysis, scientific computing, and signal processing.

REFERENCES

[1] C. J. Anfinson and F. T. Luk, *A linear algebraic model of algorithm-based fault tolerance*, IEEE Trans. Comput., Special Issue on Parallel and Distributed Algorithms, C-37 (1988), pp. 1599–1604.

[2] E. R. Berlekamp, *Algebraic Coding Theory*, McGraw–Hill, New York, 1968.

[3] D. L. Boley, S. Elhay, G. H. Golub, M. H. Gutknecht, *Nonsymmetric Lanczos and finding orthogonal polynomials associated with indefinite weights*, Numer. Algorithms, 1 (1991), pp. 21–44.

[4] D. L. Boley and G. H. Golub, *The Lanczos algorithm and controllability*, Systems Control Lett., 4 (1984), pp. 317–324.

[5] D. L. Boley, G. H. Golub, S. Makar, N. Saxena, and E. J. McCluskey, *Backward error assertions for checking solutions to systems of linear equations*, Report NA-89-12, Computer Science Department, Stanford University, Stanford, CA, 1989.

[6] R. P. Brent, F. T. Luk, and C. J. Anfinson, *Choosing small weights for multiple error detection*, Proc. SPIE, IS&T High Speed Computing II, 1058 (1989), pp. 130–136.

[7] ———, *Checksum schemes for fault tolerant systolic computing*, in Mathematics in Signal Processing II, J. G. McWhirter, ed., Clarendon Press, Oxford, 1990, pp. 791–804.

[8] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[9] K. H. Huang and J. A. Abraham, *Algorithm-based fault tolerance for matrix operations*, IEEE Trans. Comput., C-33 (1984), pp. 518–528.

[10] J. Y. Jou and J. A. Abraham, *Fault-tolerant matrix arithmetic and signal processing on highly concurrent computing structures*, Proc. IEEE, Special Issue on Fault Tolerance in VLSI, 74 (1986), pp. 732–741.

[11] F. T. Luk, *Algorithm-based fault tolerance for parallel matrix equation solvers*, in Proc. SPIE, Real Time Signal Processing VIII, 564 (1985), pp. 49–53.

[12] F. T. Luk and H. Park, *An analysis of algorithm-based fault tolerance techniques*, J. Parallel & Distrib. Comput., 5 (1988), pp. 172–184.

[13] ———, *Fault-tolerant matrix triangularizations on systolic arrays*, IEEE Trans. Comput., C-37 (1988), pp. 1434–1438.

[14] F. T. Luk, E. K. Torng, and C. J. Anfinson, *A novel fault tolerance technique for recursive least squares minimization*, J. VLSI Signal Process., 1 (1989), pp. 181–188.

[15] J. L. Massey, *Shift register synthesis and BCH decoding*, IEEE Trans. Inform. Theory, IT-15 (1967), pp. 122–127.

[16] V. S. S. NAIR AND J. A. ABRAHAM, *Real-number codes for fault-tolerant matrix operations on processor arrays*, IEEE Trans. Comput., Special Issue on Fault-Tolerant Computing, C-39 (1990), pp. 426–435.

[17] I. S. REED AND G. SOLOMON, *Polynomial codes over certain finite fields*, J. Soc. Indust. Appl. Math., 8 (1960), pp. 300–304.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 333–356, January 1992

SPARSE MATRICES IN MATLAB: DESIGN AND IMPLEMENTATION*

JOHN R. GILBERT†, CLEVE MOLER‡, AND ROBERT SCHREIBER§

Dedicated to Gene Golub on the occasion of his 60th birthday.

Abstract. The matrix computation language and environment MATLAB is extended to include sparse matrix storage and operations. The only change to the outward appearance of the MATLAB language is a pair of commands to create full or sparse matrices. Nearly all the operations of MATLAB now apply equally to full or sparse matrices, without any explicit action by the user. The sparse data structure represents a matrix in space proportional to the number of nonzero entries, and most of the operations compute sparse results in time proportional to the number of arithmetic operations on nonzeros.

Key words. MATLAB, mathematical software, matrix computation, sparse matrix algorithms

AMS(MOS) subject classifications. 65–04, 65F05, 65F20, 65F50, 68N15, 68R10

1. Introduction.

MATLAB is an interactive environment and programming language for numeric scientific computation [18]. One of its distinguishing features is the use of matrices as the only data type. In MATLAB, a matrix is a rectangular array of real or complex numbers. All quantities, even loop variables and character strings, are represented as matrices, although matrices with only one row, or one column, or one element are sometimes treated specially.

The part of MATLAB that involves computational linear algebra on dense matrices is based on direct adaptations of subroutines from LINPACK and EISPACK [5], [23]. An $m \times n$ real matrix is stored as a full array of mn floating point numbers. The computational complexity of basic operations such as addition or transposition is proportional to mn. The complexity of more complicated operations such as triangular factorization is proportional to mn^2. This has limited the applicability of MATLAB to problems involving matrices of order a few hundred on contemporary workstations and perhaps a few thousand on contemporary supercomputers.

We have now added sparse matrix storage and operations to MATLAB. This report describes our design and implementation.

Sparse matrices are widely used in scientific computation, especially in large-scale optimization, structural and circuit analysis, computational fluid dynamics, and, generally, the numerical solution of partial differential equations. Several effective Fortran subroutine packages for solving sparse linear systems are available, including SPARSPAK [11], the Yale Sparse Matrix Package [9], and some of the routines in the Harwell Subroutine Library [25].

* Received by the editors February 11, 1991; accepted for publication (in revised form) July 23, 1991.

† Xerox Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, California 94304 (gilbert@parc.xerox.com).

‡ The MathWorks, 325 Linfield Place, Menlo Park, California 94025 (moler@mathworks.com).

§ Research Institute for Advanced Computer Science, MS T045-1, NASA Ames Research Center, Moffett Field, California 94035 (schreibr@riacs.edu). This author's work was supported by the National Academy of Sciences Systems Division and Defense Advanced Research Projects Agency via Cooperative Agreement NCC 2-387 between National Aeronautics and Space Administration (NASA) and the University Space Research Association (USRA).

TABLE 1
Operations with the 4096 by 4096 discrete Laplacian.

	Sparse	Full
Memory	0.25 megabyte	128 megabytes
Compute Dx	0.2 seconds	30 seconds
Solve $Dx = b$	10 seconds	> 12 hours

Our work was facilitated by our knowledge of the techniques used in the Fortran sparse matrix packages, but we have not directly adapted any of their code. MATLAB is written in C and we wished to take advantage of the data structures and other programming features of C that would not be used in a simple translation of Fortran code. We also wanted to implement the full range of matrix operations that MATLAB provides; the Fortran packages do not generally have routines for simply adding or transposing sparse matrices, for example. And, finally, we wanted to incorporate some recent algorithmic ideas that are not used in the Fortran packages.

J. H. Wilkinson's informal working definition of a sparse matrix was "any matrix with enough zeros that it pays to take advantage of them." So sparsity is an economic issue. By avoiding arithmetic operations on zero elements, sparse matrix algorithms require less computer time. And, perhaps more importantly, by not storing many zero elements, sparse matrix data structures require less computer memory. In a sense, we have not added any new functionality to MATLAB; we have merely made some existing functionality more efficient in terms of both time and storage.

An important descriptive parameter of a sparse matrix S is $nnz(S)$, the number of nonzero elements in S. Computer storage requirements are proportional to nnz. The computational complexity of simple array operations should also be proportional to nnz, and perhaps also depend linearly on m or n, but be independent of the product mn. The complexity of more complicated operations involves such factors as ordering and fill-in, but an objective of a good sparse matrix algorithm should be:

> The time required for a sparse matrix operation should be proportional to the number of arithmetic operations on nonzero quantities.

We call this the "time is proportional to flops" rule; it is a fundamental tenet of our design.

With sparse techniques, it is practical to handle matrices involving tens of thousands of nonzero elements on contemporary workstations. As one example, let D be the matrix representation of the discrete five-point Laplacian on a square 64×64 grid with a nested dissection ordering. This is a 4096×4096 matrix with 20,224 nonzeros. Table 1 gives the memory requirements for storing D as a MATLAB sparse matrix and as a traditional Fortran or MATLAB full matrix, as well as the execution time on a Sun SPARCstation-1 workstation for computing a matrix-vector product and solving a linear system of equations by elimination.

Band matrices are special cases of sparse matrices whose nonzero elements all happen to be near the diagonal. It would be somewhat more efficient, in both time and storage, to provide a third data structure and collection of operations for band matrices. We have decided against doing this because of the added complexity, particularly in cases involving mixtures of full, sparse, and band matrices. We suspect that solving linear systems with matrices that are dense within a narrow band might be twice as fast with band storage as it is with sparse matrix storage, but that linear systems with matrices that are sparse within the band (such as those obtained

from two-dimensional grids) are more efficiently solved with general sparse matrix technology. However, we have not investigated these tradeoffs in any detail.

In this paper, we concentrate on elementary sparse matrix operations, such as addition and multiplication, and on direct methods for solving sparse linear systems of equations. These operations are now included in the "core" of MATLAB. Except for a few short examples, we will not discuss higher-level sparse matrix operations, such as iterative methods for linear systems. We intend to implement such operations as interpreted programs in the MATLAB language, so-called "m-files," outside the MATLAB core.

2. The user's view of sparse MATLAB.

2.1. Sparse matrix storage. We wish to emphasize the distinction between a matrix and what we call its *storage class*. A given matrix can conceivably be stored in many different ways—fixed point or floating point, by rows or by columns, real or complex, full or sparse—but all the different ways represent the same matrix. We now have two matrix storage classes in MATLAB, full and sparse.

Two MATLAB variables, `A` and `B`, can have different storage classes but still represent the same matrix. They occupy different amounts of computer memory, but in most other respects they are the same. Their elements are equal, their determinants and their eigenvalues are equal, and so on. The crucial question of which storage class to choose for a given matrix is the topic of §2.5.

Even though MATLAB is written in C, it follows its LINPACK and Fortran predecessors and stores full matrices by columns [5], [19]. This organization has been carried over to sparse matrices. A sparse matrix is stored as the concatenation of the sparse vectors representing its columns. Each sparse vector consists of a floating point array of nonzero entries (or two such arrays for complex matrices), together with an integer array of row indices. A second integer array gives the locations in the other arrays of the first element in each column. Consequently, the storage requirement for an $m \times n$ real sparse matrix with nnz nonzero entries is nnz reals and $nnz + n$ integers. On typical machines with 8-byte reals and 4-byte integers, this is $12nnz + 4n$ bytes. Complex matrices use a second array of nnz reals. Notice that m, the number of rows, is almost irrelevant. It is not involved in the storage requirements, nor in the operation counts for most operations. Its primary use is in error checks for subscript ranges. Similar storage schemes, with either row or column orientation, are used in the Fortran sparse packages.

2.2. Converting between full and sparse storage. Initially, we contemplated schemes for automatic conversion between sparse and full storage. There is a MATLAB precedent for such an approach. Matrices are either real or complex and the conversion between the two is automatic. Computations such as square roots and logarithms of negative numbers and eigenvalues of nonsymmetric matrices generate complex results from real data. MATLAB automatically expands the data structure by adding an array for the imaginary parts.

Moreover, several of MATLAB's functions for building matrices produce results that might effectively be stored in the sparse organization. The function `zeros(m,n)`, which generates an $m \times n$ matrix of all zeros, is the most obvious candidate. The functions `eye(n)` and `diag(v)`, which generate the $n \times n$ identity matrix and a diagonal matrix with the entries of vector v on the main diagonal, are also possibilities. Even `tril(A)` and `triu(A)`, which take the lower and upper triangular parts of a matrix A, might be considered. But this short list begins to demonstrate a difficulty—how far

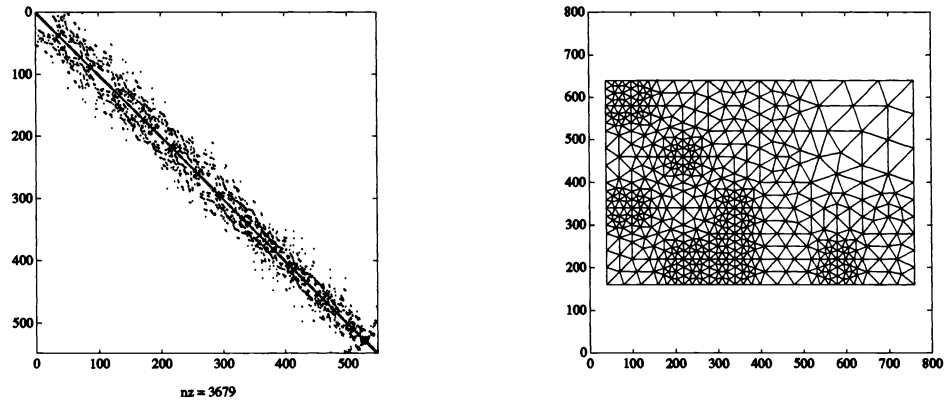


FIG. 1. *The Eppstein mesh as plotted by* `spy(A)` *and* `gplot(A,xy)`.

should "automatic sparsification" be carried? Is there some threshold value of sparsity where the conversion should be done? Should the user provide the value for such a sparsification parameter? We don't know the answers to these questions, so we decided to take another approach, which we have since found to be quite satisfactory.

No sparse matrices are created without some overt direction from the user. Thus, the changes we have made to MATLAB do not affect the user who has no need for sparsity. Operations on full matrices continue to produce full matrices. But once initiated, sparsity propagates. Operations on sparse matrices produce sparse matrices, and an operation on a mixture of sparse and full matrices produces a sparse result unless the operator ordinarily destroys sparsity. (Matrix addition is an example; more on this later.)

There are two new built-in functions, `full` and `sparse`. For any matrix A, `full(A)` returns A stored as a full matrix. If A is already full, then A is returned unchanged. If A is sparse, then zeros are inserted at the appropriate locations to fill out the storage. Conversely, `sparse(A)` removes any zero elements and returns A stored as a sparse matrix, regardless of how sparse A actually is.

2.3. Displaying sparse matrices. Sparse and full matrices print differently. The statement

```
A = [0 0 11; 22 0 0; 0 33 0]
```

produces a conventional MATLAB full matrix that prints as

```
A =
     0     0    11
    22     0     0
     0    33     0
```

The statement `S = sparse(A)` converts A to sparse storage, and prints

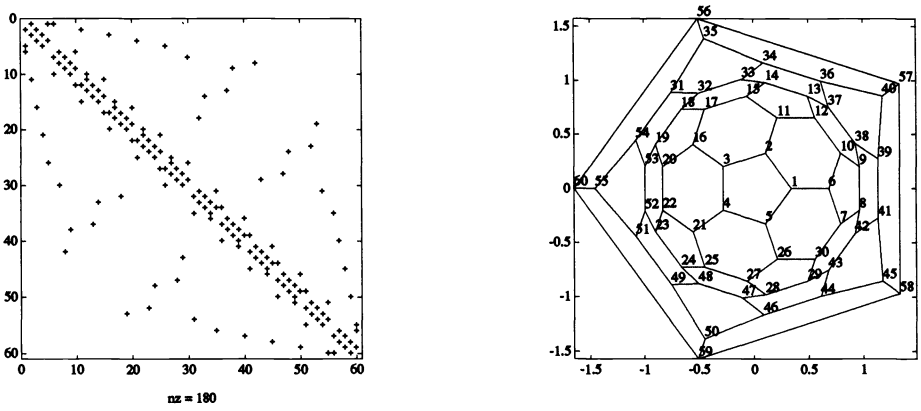


FIG. 2. *The buckyball as rendered by* `spy` *and* `gplot`.

```
S =
        (2,1)        22
        (3,2)        33
        (1,3)        11
```

As this illustrates, sparse matrices are printed as a list of their nonzero elements (with indices), in column major order.

The function `nnz(A)` returns the number of nonzero elements of A. It is implemented by scanning full matrices, and by access to the internal data structure for sparse matrices. The function `nzmax(A)` returns the number of storage locations for nonzeros allocated for A.

Graphic visualization of the structure of a sparse matrix is often a useful tool. The function `spy(A)` plots a silhouette of the nonzero structure of A. Figure 1 illustrates such a plot for a matrix that comes from a finite element mesh due to Eppstein. A picture of the graph of a matrix is another way to visualize its structure. Laying out an arbitrary graph for display is a hard problem that we do not address. However, some sparse matrices (from finite element applications, for example) have spatial coordinates associated with their rows or columns. If xy contains such coordinates for matrix A, the function `gplot(A,xy)` draws its graph. The second plot in Fig. 1 shows the graph of the sample matrix, which in this case is just the same as the finite element mesh. Figure 2 is another example: The `spy` plot is the 60×60 adjacency matrix of the graph of a Buckminster Fuller geodesic dome, a soccer ball, and a C_{60} molecule, and the `gplot` shows the graph itself.

Section 3.3.4 describes a function for visualizing the elimination tree of a matrix.

2.4. Creating sparse matrices. Usually one wants to create a sparse matrix directly, without first having a full matrix A and then converting it with `S = sparse(A)`. One way to do this is by simply supplying a list of nonzero entries and their indices. Several alternate forms of `sparse` (with more than one argument) allow this. The most general is

```
S = sparse(i,j,s,m,n,nzmax).
```

Ordinarily, i and j are vectors of integer indices, s is a vector of real or complex entries, and m, n, and $nzmax$ are integer scalars. This call generates an $m \times n$ sparse matrix, having one nonzero for each entry in the vectors i, j, and s, with $S(i(k), j(k)) = s(k)$, and with enough space allocated for S to have $nzmax$ nonzeros. The indices in i and j need not be given in any particular order.

If a pair of indices occurs more than once in i and j, **sparse** adds the corresponding values of s together. Then the sparse matrix S is created with one nonzero for each nonzero in this modified vector s. The argument s and one of the arguments i and j may be scalars, in which case they are expanded so that the first three arguments all have the same length.

There are several simplifications of the full six-argument call to **sparse**.

`S = sparse(i,j,s,m,n)` uses $nzmax = \text{length}(s)$.

`S = sparse(i,j,s)` uses $m = \max(i)$ and $n = \max(j)$.

`S = sparse(m,n)` is the same as `S = sparse([],[],[],m,n)`, where `[]` is MATLAB's empty matrix. It produces the ultimate sparse matrix, an $m \times n$ matrix of all zeros.

Thus, for example,

```
S = sparse([1 2 3], [3 1 2], [11 22 33])
```

produces the sparse matrix S from the example in §2.3, but does not generate any full 3×3 matrix during the process.

MATLAB's function `k = find(A)` returns a list of the positions of the nonzeros of A, counting in column major order. For sparse MATLAB we extended the definition of **find** to extract the nonzero elements together with their indices. For any matrix A, full or sparse, `[i,j,s] = find(A)` returns the indices and values of the nonzeros. (The square bracket notation on the left side of an assignment indicates that the function being called can return more than one value. In this case, **find** returns three values, which are assigned to the three separate variables i, j, and s.) For example, this dissects and then reassembles a sparse matrix:

```
[i,j,s] = find(S);
[m,n] = size(S);
S = sparse(i,j,s,m,n);
```

So does this, if the last row and column have nonzero entries:

```
[i,j,s] = find(S);
S = sparse(i,j,s);
```

Another common way to create a sparse matrix, particularly for finite difference computations, is to give the values of some of its diagonals. Two functions **diags** and **blockdiags** can create sparse matrices with specified diagonal or block diagonal structure.

There are several ways to read and write sparse matrices. The MATLAB **save** and **load** commands, which save the current workspace or load a saved workspace, have been extended to accept sparse matrices and save them efficiently. We have written a Fortran utility routine that converts a file containing a sparse matrix in the Harwell–Boeing format [6] into a file that MATLAB can load.

2.5. The results of sparse operations. What is the result of a MATLAB operation on sparse matrices? This is really two fundamental questions: what is the value of the result, and what is its storage class? In this section we discuss the answers that we settled on for those questions.

A function or subroutine written in MATLAB is called an *m-file*. We want it to be possible to write m-files that produce the same results for sparse and for full inputs. Of course, one could ensure this by converting all inputs to full, but that would defeat the goal of efficiency. A better idea, we decided, is to postulate that:

> The value of the result of an operation does not depend on the storage class of the operands, although the storage class of the result may.

The only exception is a function to inquire about the storage class of an object: `issparse(A)` returns 1 if A is sparse, 0 otherwise.

Some intriguing notions were ruled out by our postulate. We thought, for a while, that in cases such as `A ./ S` (which denotes the pointwise quotient of A and S) we ought not to divide by zero where S is zero, since that would not produce anything useful; instead we thought to implement this as if it returned $A(i,j)/S(i,j)$ wherever $S(i,j) \neq 0$, leaving A unchanged elsewhere. All such ideas, however, were dropped in the interest of observing the rule that the result does not depend on storage class.

The second fundamental question is how to determine the storage class of the result of an operation. Our decision here is based on three ideas. First, the storage class of the result of an operation should depend only on the storage classes of the operands, not on their values or sizes. (Reason: it is too risky to make a heuristic decision about when to sparsify a matrix without knowing how it will be used.) Second, sparsity should not be introduced into a computation unless the user explicitly asks for it. (Reason: the full matrix user should not have sparsity appear unexpectedly, because of the performance penalty in doing sparse operations on mostly nonzero matrices.) Third, once a sparse matrix is created, sparsity should propagate through matrix and vector operations, concatenation, and so forth. (Reason: most m-files should be able to do sparse operations for sparse input or full operations for full input without modification.)

Thus full inputs always give full outputs, except for functions like `sparse`, whose purpose is to create sparse matrices. Sparse inputs, or mixed sparse and full inputs, follow these rules (where S is sparse and F is full):

- Functions from matrices to scalars or fixed-size vectors, like `size` or `nnz`, always return full results.
- Functions from scalars or fixed-size vectors to matrices, like `zeros`, `ones`, and `eye`, generally return full results. Having `zeros(m,n)` and `eye(m,n)` return full results is necessary to avoid introducing sparsity into a full user's computation; there are also functions `spzeros` and `speye` that return sparse zero and identity matrices.
- The remaining unary functions from matrices to matrices or vectors generally return a result of the same storage class as the operand (the main exceptions are `sparse` and `full`). Thus, `chol(S)` returns a sparse Cholesky factor, and `diag(S)` returns a sparse vector (a sparse $m \times 1$ matrix). The vectors returned by `max(S)`, `sum(S)`, and their relatives (that is, the vectors of column maxima and column sums, respectively) are sparse, even though they may well be all nonzero.

- Binary operators yield sparse results if both operands are sparse, and full results if both are full. In the mixed case, the result's storage class depends on the operator. For example, `S + F` and `S \ F` (which solves the linear system $SX = F$) are full; `S .* F` (the pointwise product) and `S & F` are sparse.
- A block matrix formed by concatenating smaller matrices, like

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix},$$

  is written as `[A B ; C D]` in MATLAB. If all the inputs are full, the result is full, but a concatenation that contains any sparse matrix is sparse. Submatrix indexing on the right counts as a unary operator; `A = S(i,j)` produces a sparse result (for sparse S) whether i and j are scalars or vectors. Submatrix indexing on the left, as in `A(i,j) = S`, does not change the storage class of the matrix being modified.

These decisions gave us some difficulty. Cases like `~S` and `S >= T`, where the result has many ones when the operands are sparse, made us consider adding more exceptions to the rules. We discussed the possibility of "sparse" matrices in which all the values not explicitly stored would be some scalar (like 1) rather than zero. We rejected these ideas in the interest of simplicity.

3. Implementation. This section describes the algorithms for the sparse operations in MATLAB in some detail. We begin with a discussion of fundamental data structures and design decisions.

3.1. Fundamentals.

3.1.1. Data structure. A very important implementation decision is the choice of a data structure. The internal representation of a sparse matrix must be flexible enough to implement all the MATLAB operations. For simplicity, we ruled out the use of different data structures for different operations. The data structure should be compact, storing only nonzero elements, with a minimum of overhead storage for integers or pointers. Wherever possible, it should support matrix operations in time proportional to flops. Since MATLAB is an interpreted, high-level matrix language, efficiency is more important in matrix arithmetic and matrix-vector operations than in accessing single elements of matrices.

These goals are met by a simple column-oriented scheme that has been widely used in sparse matrix computation. A sparse matrix is a C record structure with the following constituents. The nonzero elements are stored in a one-dimensional array of double-precision reals, in column major order. (If the matrix is complex, the imaginary parts are stored in another such array.) A second array of integers stores the row indices. A third array of $n + 1$ integers stores the index into the first two arrays of the leading entry in each of the n columns, and a terminating index whose value is nnz. Thus a real $m \times n$ sparse matrix with nnz nonzeros uses nnz reals and $nnz + n + 1$ integers.

This scheme is not efficient for manipulating matrices one element at a time: access to a single element takes time at least proportional to the logarithm of the length of its column; inserting or removing a nonzero may require extensive data movement. However, element-by-element manipulation is rare in MATLAB (and is expensive even in full MATLAB). Its most common application would be to create

a sparse matrix, but this is more efficiently done by building a list $[i, j, s]$ of matrix elements in arbitrary order and then using `sparse(i,j,s)` to create the matrix.

The sparse data structure is allowed to have unused elements after the end of the last column of the matrix. Thus an algorithm that builds up a matrix one column at a time can be implemented efficiently by allocating enough space for all the expected nonzeros at the outset.

3.1.2. Storage allocation. Storage allocation is one of the thorniest parts of building portable systems. MATLAB handles storage allocation for the user, invisibly allocating and deallocating storage as matrices appear, disappear, and change size. Sometimes the user can gain efficiency by preallocating storage for the result of a computation. One does this in full MATLAB by allocating a matrix of zeros and filling it in incrementally. Similarly, in sparse MATLAB one can preallocate a matrix (using `sparse`) with room for a specified number of nonzeros. Filling in the sparse matrix a column at a time requires no copying or reallocation.

Within MATLAB, simple "allocate" and "free" procedures handle storage allocation. (We will not discuss how MATLAB handles its free storage and interfaces to the operating system to provide these procedures.) There is no provision for doing storage allocation within a single matrix; a matrix is allocated as a single block of storage, and if it must expand beyond that block it is copied into a newly allocated larger block.

MATLAB must allocate space to hold the results of operations. For a full result, MATLAB allocates mn elements at the start of the computation. This strategy could be disastrous for sparse matrices. Thus, sparse MATLAB attempts to make a reasonable choice of how much space to allocate for a sparse result.

Some sparse matrix operations, like Cholesky factorization, can predict in advance the exact amount of storage the result will require. These operations simply allocate a block of the right size before the computation begins. Other operations, like matrix multiplication and LU factorization, have results of unpredictable size. These operations are all implemented by algorithms that compute one column at a time. Such an algorithm first makes a guess at the size of the result. If more space is needed at some point, it allocates a new block that is larger by a constant factor (typically 1.5) than the current block, copies the columns already computed into the new block, and frees the old block.

Most of the other operations compute a simple upper bound on the storage required by the result to decide how much space to allocate—for example, the pointwise product `S .* T` uses the smaller of $nnz(S)$ and $nnz(T)$, and `S + T` uses the smaller of $nnz(S) + nnz(T)$ and mn.

3.1.3. The sparse accumulator. Many sparse matrix algorithms use a dense working vector to allow random access to the currently "active" column or row of a matrix. The sparse MATLAB implementation formalizes this idea by defining an abstract data type called the sparse accumulator, or SPA. The SPA consists of a dense vector of real (or complex) values, a dense vector of true/false "occupied" flags, and an unordered list of the indices whose occupied flags are true.

The SPA represents a column vector whose "unoccupied" positions are zero and whose "occupied" positions have values (zero or nonzero) specified by the dense real or complex vector. It allows random access to a single element in constant time, as well as sequencing through the occupied positions in constant time per element. Most matrix operations allocate the SPA (with appropriate dimension) at their beginning and free it at their end. Allocating the SPA takes time proportional to its dimension

(to turn off all the occupied flags), but subsequent operations take only constant time per nonzero.

In a sense, the SPA is a register and an instruction set in an abstract machine architecture for sparse matrix computation. MATLAB manipulates the SPA through some thirty-odd access procedures. About half of these are operations between the SPA and a sparse or dense vector, from a "spaxpy" that implements SPA := SPA+ax (where a is a scalar and x is a column of a sparse matrix) to a "spaeq" that tests elementwise equality. Other routines load and store the SPA, permute it, and access individual elements. The most complicated SPA operation is a depth-first search on an acyclic graph, which marks as "occupied" a topologically ordered list of reachable vertices; this is used in the sparse triangular solve described in §3.4.2.

The SPA simplifies data structure manipulation, because all fill occurs in the SPA; that is, only in the SPA can a zero become nonzero. The "spastore" routine does not store exact zeros, and in fact the sparse matrix data structure never contains any explicit zeros. Almost all real arithmetic operations occur in SPA routines, too, which simplifies MATLAB's tally of flops. (The main exceptions are in certain scalar-matrix operations like `2*A`, which are implemented without the SPA for efficiency.)

3.1.4. Asymptotic complexity analysis. A strong philosophical principle in the sparse MATLAB implementation is that it should be possible to analyze the complexity of the various operations, and that they should be efficient in the asymptotic sense as well as in practice. This section discusses this principle, in terms of both theoretical ideals and engineering compromises.

Ideally all the matrix operations would use time proportional to flops, that is, their running time would be proportional to the number of nonzero real arithmetic operations performed. This goal cannot always be met: for example, `[0 1] + [1 0]` does no nonzero arithmetic. A more accurate statement is that time should be proportional to flops or data size, whichever is larger. Here "data size" means the size of the output and that part of the input that is used nontrivially; for example, in `A*b` only those columns of A corresponding to nonzeros in b participate nontrivially.

This more accurate ideal can be realized in almost all of MATLAB. The exceptions are some operations that do no arithmetic and cannot be implemented in time proportional to data size. The algorithms to compute most of the reordering permutations described in §3.3 are efficient in practice but not linear in the worst case. Submatrix indexing is another example: if i and j are vectors of row and column indices, `B = A(i,j)` may examine all the nonzeros in the columns $A(:, j)$, and `B(i,j) = A` can at worst take time linear in the total size of B.

The MATLAB implementation actually violates the "time proportional to flops" philosophy in one systematic way. The list of occupied row indices in the SPA is not maintained in numerical order, but the sparse matrix data structure does require row indices to be ordered. Sorting the row indices when storing the SPA would theoretically imply an extra factor of $O(\log n)$ in the worst-case running times of many of the matrix operations. All our algorithms could avoid this factor—usually by storing the matrix with unordered row indices, then using a linear-time transposition sort to reorder all the rows of the final result at once—but for simplicity of programming we included the sort in "spastore."

The idea that running time should be susceptible to analysis helps the user who writes programs in MATLAB to choose among alternative algorithms, gives guidance in scaling up running times from small examples to larger problems, and, in a general-purpose system like MATLAB, gives some insurance against an unexpected worst-case

instance arising in practice. Of course, complete a priori analysis is impossible—the work in sparse LU factorization depends on numerical pivoting choices, and the efficacy of a heuristic reordering such as minimum degree is unpredictable—but we feel it is worthwhile to stay as close to the principle as we can.

In a technical report [14] we present some experimental evidence that sparse MATLAB operations require time proportional to flops and data size in practice.

3.2. Factorizations. The LU and Cholesky factorizations of a sparse matrix yield sparse results. MATLAB does not yet have a sparse QR factorization. Section 3.6 includes some remarks on sparse eigenvalue computation in MATLAB.

3.2.1. LU factorization. If A is a sparse matrix, `[L,U,P] = lu(A)` returns three sparse matrices such that $PA = LU$, as obtained by Gaussian elimination with partial pivoting. The permutation matrix P uses only $O(n)$ storage in sparse format. As in dense MATLAB, `[L,U] = lu(A)` returns a permuted unit lower triangular and an upper triangular matrix whose product is A.

Since sparse LU must behave like MATLAB's full LU, it does not pivot for sparsity. A user who happens to know a good column permutation Q for sparsity can, of course, ask for `lu(A*Q')`, or `lu(A(:,q))` where q is an integer permutation vector. Section 3.3 describes a few ways to find such a permutation. The matrix division operators \ and / do pivot for sparsity by default; see §3.4.

We use a version of the `GPLU` algorithm [15] to compute the LU factorization. This computes one column of L and U at a time by solving a sparse triangular system with the already finished columns of L. Section 3.4.2 describes the sparse triangular solver that does most of the work. The total time for the factorization is proportional to the number of nonzero arithmetic operations (plus the size of the result), as desired.

The column-oriented data structure for the factors is created as the factorization progresses, never using any more storage for a column than it requires. However, the total size of L or U cannot be predicted in advance. Thus the factorization routine makes an initial guess at the required storage, and expands that storage (by a factor of 1.5) whenever necessary.

3.2.2. Cholesky factorization. As in full MATLAB, `R = chol(A)` returns the upper triangular Cholesky factor of a Hermitian positive definite matrix A. Pivoting for sparsity is not automatic, but minimum degree and profile-limiting permutations can be computed as described in §3.3.

Our current implementation of Cholesky factorization emphasizes simplicity and compatibility with the rest of sparse MATLAB; thus it does not use some of the more sophisticated techniques, such as the compressed index storage scheme [11, § 5.4.2], or supernodal methods to take advantage of the clique structure of the chordal graph of the factor [2]. It does, however, run in time proportional to arithmetic operations with little overhead for data structure manipulation.

We use a slightly simplified version of an algorithm from the Yale Sparse Matrix Package [9], which is described in detail by George and Liu [11]. We begin with a combinatorial step that determines the number of nonzeros in the Cholesky factor (assuming no exact cancellation) and allocates a large enough block of storage. We then compute the lower triangular factor R^T one column at a time. Unlike YSMP and SPARSPAK, we do not begin with a symbolic factorization; instead, we create the sparse data structure column by column as we compute the factor. The only reason for the initial combinatorial step is to determine how much storage to allocate for the result.

3.3. Permutations. A permutation of the rows or columns of a sparse matrix A can be represented in two ways. A permutation matrix P acts on the rows of A as `P*A` or on the columns as `A*P'`. A permutation vector p, which is a full vector of length n containing a permutation of `1:n`, acts on the rows of A as `A(p,:)` or on the columns as `A(:,p)`. Here p could be either a row vector or a column vector.

Both representations use $O(n)$ storage, and both can be applied to A in time proportional to $nnz(A)$. The vector representation is slightly more compact and efficient, so the various sparse matrix permutation routines all return vectors—full row vectors, to be precise—with the exception of the pivoting permutation in LU factorization.

Converting between the representations is almost never necessary, but it is simple. If I is a sparse identity matrix of the appropriate size, then P is `I(p,:)` and P^T is `I(:,p)`. Also p is `(P*(1:n)')'` or `(1:n)*P'`. (We leave to the reader the puzzle of using `find` to obtain p from P without doing any arithmetic.) The inverse of P is `P'`; the inverse r of p can be computed by the "vectorized" statement `r(p) = 1:n`.

3.3.1. Permutations for sparsity: Asymmetric matrices. Reordering the columns of a matrix can often make its LU or QR factors sparser. The simplest such reordering is to sort the columns by increasing nonzero count. This is sometimes a good reordering for matrices with very irregular structures, especially if there is great variation in the nonzero counts of rows or columns.

The MATLAB function `p = colperm(A)` computes this column-count permutation. It is implemented as a two-line m-file:

```
[i,j] = find(A);
[ignore,p] = sort(diff(find(diff([0 j' inf]))));
```

The vector j is the column indices of all the nonzeros in A, in column major order. The inner `diff` computes first differences of j to give a vector with ones at the starts of columns and zeros elsewhere; the `find` converts this to a vector of column-start indices; the outer `diff` gives the vector of column lengths; and the second output argument from `sort` is the permutation that sorts this vector.

The symmetric reverse Cuthill–McKee ordering described in §3.3.2 can be used for asymmetric matrices as well; the function `symrcm(A)` actually operates on the nonzero structure of $A + A^T$. This is sometimes a good ordering for matrices that come from one-dimensional problems or problems that are in some sense long and thin.

Minimum degree is an ordering that often performs better than `colperm` or `symrcm`. The sparse MATLAB function `p = colmmd(A)` computes a minimum-degree ordering for the columns of A. This column ordering is the same as a symmetric minimum-degree ordering for the matrix A^TA, though we do not actually form A^TA to compute it.

George and Liu [10] survey the extensive development of efficient and effective versions of symmetric minimum degree, most of which is reflected in the symmetric minimum-degree codes in SPARSPAK, YSMP, and the Harwell Subroutine Library. The MATLAB version of minimum degree uses many of these ideas, as well as some ideas from a parallel symmetric minimum-degree algorithm by Gilbert, Lewis, and Schreiber [13]. We sketch the algorithm briefly to show how these ideas are expressed in the framework of column minimum degree. The reader who is not interested in all the details can skip to §3.3.2.

Although most column minimum-degree codes for asymmetric matrices are based on a symmetric minimum-degree code, our organization is the other way around: MATLAB's symmetric minimum-degree code (described in §3.3.2) is based on its column minimum-degree code. This is because the best way to represent a symmetric matrix (for the purposes of minimum degree) is as a union of cliques, or full submatrices. When we begin with an asymmetric matrix A, we wish to reorder its columns by using a minimum-degree order on the symmetric matrix A^TA—but each row of A induces a clique in A^TA, so we can simply use A itself to represent A^TA instead of forming the product explictly. Speelpenning [24] called such a clique representation of a symmetric graph the "generalized element" representation; George and Liu [10] call it the "quotient graph model." Ours is the first column minimum-degree implementation that we know of whose data structures are based directly on A, and which does not need to spend the time and storage to form the structure of A^TA. The idea for such a code is not new, however—George and Liu [10] suggest it, and our implementation owes a great deal to discussions with Ng and Peyton of Oak Ridge National Laboratories.

We simulate symmetric Gaussian elimination on A^TA, using a data structure that represents A as a set of vertices and a set of cliques whose union is the graph of A^TA. Initially, each column of A is a vertex and each row is a clique. Elimination of a vertex j induces fill among all the (so far uneliminated) vertices adjacent to j. This means that all the vertices in cliques containing j become adjacent to one another. Thus all the cliques containing vertex j merge into one clique. In other words, all the rows of A with nonzeros in column j disappear, to be replaced by a single row whose nonzero structure is their union. Even though fill is implicitly being added to A^TA, the data structure for A gets smaller as the rows merge, so no extra storage is required during the elimination.

Minimum degree chooses a vertex of lowest degree (the sparsest remaining column of A^TA, or the column of A having nonzero rows in common with the fewest other columns), eliminates that vertex, and updates the remainder of A by adding fill (i.e., merging rows). This whole process is called a "stage"; after n stages the columns are all eliminated and the permutation is complete. In practice, updating the data structure after each elimination is too slow, so several devices are used to perform many eliminations in a single stage before doing the update for the stage.

First, instead of finding a single minimum-degree vertex, we find an entire "independent set" of minimum-degree vertices with no common nonzero rows. Eliminating one such vertex has no effect on the others, so we can eliminate them all at the same stage and do a single update. George and Liu call this strategy "multiple elimination." (They point out that the resulting permutation may not be a strict minimum-degree order, but the difference is generally insignificant.)

Second, we use what George and Liu call "mass elimination": After a vertex j is eliminated, its neighbors in A^TA form a clique (a single row in A). Any of those neighbors whose own neighbors all lie within that same clique will be a candidate for elimination at the next stage. Thus, we may as well eliminate such a neighbor during the same stage as j, immediately after j, delaying the update until afterward. This often saves a tremendous number of stages because of the large cliques that form late in the elimination. (The number of stages is reduced from the height of the elimination tree to approximately the height of the clique tree; for many two-dimensional finite element problems, for example, this reduces the number of stages from about $\sqrt{n}$ to about $\log n$.) Mass elimination is particularly simple to implement in the column

data structure: after all rows with nonzeros in column j are merged into one row, the columns to be eliminated with j are those whose only remaining nonzero is in that new row.

Third, we note that any two columns with the same nonzero structure will be eliminated in the same stage by mass elimination. Thus we allow the option of combining such columns into "supernodes" (or, as George and Liu call them, "indistinguishable nodes"). This speeds up the ordering by making the data structure for A smaller. The degree computation must account for the sizes of supernodes, but this turns out to be an advantage for two reasons. The quality of the ordering actually improves slightly if the degree computation does not count neighbors within the same supernode. (George and Liu observe this phenomenon and call the number of neighbors outside a vertex's supernode its "external degree.") Also, supernodes improve the approximate degree computation described below. Amalgamating columns into supernodes is fairly slow (though it takes time only proportional to the size of A). Supernodes can be amalgamated at every stage, periodically, or never; the current default is every third stage.

Fourth, we note that the structure of A^TA is not changed by dropping any row of A whose nonzero structure is a subset of that of another row. This row reduction speeds up the ordering by making the data structure smaller. More significantly, it allows mass elimination to recognize larger cliques, which decreases the number of stages dramatically. Duff and Reid [8] call this strategy "element absorption." Row reduction takes time proportional to multiplying AA^T in the worst case (though the worst case is rarely realized and the constant of proportionality is very small). By default, we reduce at every third stage; again the user can change this.

Fifth, to achieve larger independent sets and hence fewer stages, we relax the minimum-degree requirement and allow elimination of any vertex of degree at most $\alpha d+\beta$, where d is the minimum degree at this stage and α and β are parameters. The choice of threshold can be used to trade off ordering time for quality of the resulting ordering. For problems that are very large, have many right-hand sides, or factor many matrices with the same nonzero structure, ordering time is insignificant and the tightest threshold is appropriate. For one-off problems of moderate size, looser thresholds like $1.5d+2$ or even $2d+10$ may be appropriate. The threshold can be set by the user; its default is $1.2d+1$.

Sixth and last, our code has the option of using an "approximate degree" instead of computing the actual vertex degrees. Recall that a vertex is a column of A, and its degree is the number of other columns with which it shares some nonzero row. Computing all the vertex degrees in A^TA takes time proportional to actually computing A^TA, though the constant is quite small and no extra space is needed. Still, the exact degree computation can be the slowest part of a stage. If column j is a supernode containing $n(j)$ original columns, we define its approximate degree as

$$d(j) = \sum_{a_{ij}\neq 0} (nnz(A(i,:)) - n(j)).$$

This can be interpreted as the sum of the sizes of the cliques containing j, except that j and the other columns in its supernode are not counted. This is a fairly good approximation in practice; it errs only by overcounting vertices that are members of at least three cliques containing j. George and Liu call such vertices "outmatched nodes," and observe that they tend to be rare in the symmetric algorithm. Computing approximate degrees takes only time proportional to the size of A.

Column minimum degree sometimes performs poorly if the matrix A has a few very dense rows, because then the structure of A^TA consists mostly of the cliques induced by those rows. Thus `colmmd` will withhold from consideration any row containing more than a fixed proportion (by default, 50 percent) of nonzeros.

All these options for minimum degree are under the user's control, though the casual user of MATLAB never needs to change the defaults. The default settings use approximate degrees, row reduction and supernode amalgamation every third stage, and a degree threshold of $1.2d + 1$, and withhold rows that are at least 50 percent dense.

3.3.2. Permutations for sparsity: Symmetric matrices. Preorderings for Cholesky factorization apply symmetrically to the rows and columns of a symmetric positive definite matrix. Sparse MATLAB includes two symmetric preordering permutation functions. The `colperm` permutation can also be used as a symmetric ordering, but it is usually not the best choice.

Bandwidth-limiting and profile-limiting orderings are useful for matrices whose structure is "one-dimensional" in a sense that is hard to make precise. The reverse Cuthill–McKee ordering is an effective and inexpensive profile-limiting permutation. MATLAB function `p = symrcm(A)` returns a reverse Cuthill–McKee permutation for symmetric matrix A. The algorithm first finds a "pseudo-peripheral" vertex of the graph of A, then generates a level structure by breadth-first search and orders the vertices by decreasing distance from the pseudo-peripheral vertex. Our implementation is based closely on the SPARSPAK implementation as described by George and Liu [11].

Profile methods like reverse Cuthill–McKee are not the best choice for most large matrices arising from problems with two or more dimensions, or problems without much geometric structure, because such matrices typically do not have reorderings with low profile. The most generally useful symmetric preordering in MATLAB is minimum degree, obtained by the function `p = symmmd(A)`. Our symmetric minimum-degree implementation is based on the column minimum degree described in §3.3.1. In fact, `symmmd` just creates a nonzero structure K with a column for each column of A and a row for each above-diagonal nonzero in A, such that K^TK has the same nonzero structure as A; it then calls the column minimum-degree code on K.

3.3.3. Nonzero diagonals and block triangular form. A square nonsingular matrix A always has a row permutation p such that $A(p,:)$ has nonzeros on its main diagonal. The MATLAB function `p = dmperm(A)` computes such a permutation. With two output arguments, the function `[p,q] = dmperm(A)` gives both row and column permutations that put A into block upper triangular form; that is, $A(p,q)$ has a nonzero main diagonal and a block triangular structure with the largest possible number of blocks. Notice that the permutations p returned by these two calls are likely to be different.

The most common application of block triangular form is to solve a reducible system of linear equations by block back-substitution, factoring only the diagonal blocks of the matrix. Figure 9 is an m-file that implements this algorithm. The m-file illustrates the call `[p,q,r] = dmperm(A)`, which returns p and q as before, and also a vector r giving the boundaries of the blocks of the block upper triangular form. To be precise, if there are b blocks in each direction, then r has length $b+1$, and the ith diagonal block of $A(p,q)$ consists of rows and columns with indices from $r(i)$ through $r(i+1)-1$.

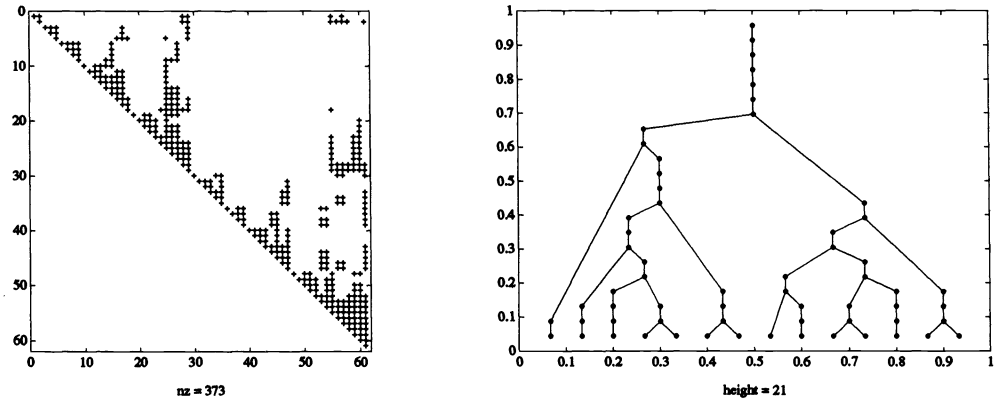


FIG. 3. *The Cholesky factor of a matrix and its elimination tree.*

Any matrix, whether square or not, has a form called the "Dulmage–Mendelsohn decomposition" [4], [20], which is the same as ordinary block upper triangular form if the matrix is square and nonsingular. The most general form of the decomposition, for arbitrary rectangular A, is `[p,q,r,s] = dmperm(A)`. The first two outputs are permutations that put $A(p,q)$ into block form. Then r describes the row boundaries of the blocks and s the column boundaries: the ith diagonal block of $A(p,q)$ has rows $r(i)$ through $r(i+1)-1$ and columns $s(i)$ through $s(i+1)-1$. The first diagonal block may have more columns than rows, the last diagonal block may have more rows than columns, and all the other diagonal blocks are square. The subdiagonal blocks are all zero. The square diagonal blocks have nonzero diagonal elements. All the diagonal blocks are irreducible; for the nonsquare blocks, this means that they have the "strong Hall property" [4]. This block form can be used to solve least squares problems by a method analogous to block back-substitution; see the references for more details.

3.3.4. Elimination trees. The elimination tree [21] of a symmetric positive definite matrix describes the dependences among rows or columns in Cholesky factorization. Liu [16] surveys applications of the elimination tree in sparse factorization. The nodes of the tree are the integers 1 through n, representing the rows of the matrix and of its upper triangular Cholesky factor. The parent of row i is the smallest $j > i$ such that the (i,j) element of the upper triangular Cholesky factor of the matrix is nonzero; if row i of the factor is zero after the diagonal, then i is a root. If the matrix is irreducible, then its only root is node n.

Liu describes an algorithm to find the elimination tree without forming the Cholesky factorization, in time almost linear in the size of the matrix. That algorithm is implemented as the MATLAB function `[t,q] = etree(A)`. The resulting tree is represented by a row vector t of parent pointers: $t(i)$ is the parent of node i, or zero if i is a root.

The optional second output q is a permutation vector that gives a postorder permutation of the tree, or of the rows and columns of A. This permutation reorders the tree vertices so that every subtree is numbered consecutively, with the subtree's root last. This is an "equivalent reordering" of A, to use Liu's terminology: the Cholesky factorization of $A(q,q)$ has the same fill, operation count, and elimination

tree as that of A. The permutation brings together the "fundamental supernodes" of A, which are full blocks in the Cholesky factor whose structure can be exploited in vectorized or parallel supernodal factorization [2], [17].

The postorder permutation can also be used to lay out the vertices for a picture of the elimination tree. The function `tspy(A)` plots a picture of the elimination tree of A, as shown in Fig. 3.

3.4. Matrix division. The usual way to solve systems of linear equations in MATLAB is not by calling `lu` or `chol`, but with the matrix division operators / and \. If A is square, the result of `X = A\B` is the solution to the linear system $AX = B$; if A is not square then a least squares solution is computed. The result of `X = A/B` is the solution to $A = XB$, which is `(B'\A')'`. Full MATLAB computes $A\backslash B$ by LU factorization with partial pivoting if A is square, or by QR factorization with column pivoting if not.

3.4.1. The sparse linear equation solver. Like full MATLAB, sparse MATLAB uses direct factorization methods to solve linear systems. The philosophy behind this is that iterative linear system solvers are best implemented as MATLAB m-files, which can use the sparse matrix data structures and operations in the core of MATLAB.

If A is sparse, MATLAB chooses among a sparse triangular solve, sparse Cholesky factorization, and sparse LU factorization, with optional preordering by minimum degree in the last two cases. The result returned has the same storage class as B. The outline of sparse $A\backslash B$ is as follows.

- If A is not square, solve the least squares problem.
- Otherwise, if A is triangular, perform a sparse triangular solve for each column of B.
- Otherwise, if A is a permutation of a triangular matrix, permute it and then perform a sparse triangular solve for each column of B.
- Otherwise, if A is Hermitian and has positive real diagonal elements, find a symmetric minimum-degree order p and attempt to compute the Cholesky factorization of $A(p,p)$. If successful, finish with two sparse triangular solves for each column of B.
- Otherwise (if A is not Hermitian with positive diagonal or if Cholesky factorization fails), find a column minimum-degree order p, compute the LU factorization with partial pivoting of $A(:,p)$, and perform two sparse triangular solves for each column of B.

Section 3.5 describes the sparse least squares method we currently use.

For a square matrix, the four possibilities are tried in order of increasing cost. Thus, the cost of checking alternatives is a small fraction of the total cost. The test for triangular A takes only $O(n)$ time if A is $n \times n$; it just examines the first and last row indices in each column. (Notice that a test for triangularity would take $O(n^2)$ time for a full matrix.) The test for a "morally triangular" matrix, which is a row and column permutation of a nonsingular triangular matrix, takes time proportional to the number of nonzeros in the matrix and is in practice very fast. (A Dulmage–Mendelsohn decomposition would also detect moral triangularity, but would be slower.) These tests mean that, for example, the MATLAB sequence

```
[L,U] = lu(A);
y = L\b;
x = U\y;
```

will use triangular solves for both matrix divisions, since L is morally triangular and U is triangular.

The test for Hermitian positive diagonal is an inexpensive guess at when to use Cholesky factorization. Cholesky is quite a bit faster than LU, both because it does half as many operations and because storage management is simpler. (The time to look at every element of A in the test is insignificant.) Of course it is possible to construct examples in which Cholesky fails only at the last column of the reordered matrix, wasting significant time, but we have not seen this happen in practice.

The function `spparms` can be used to turn the minimum-degree preordering off if the user knows how to compute a better preorder for the particular matrix in question.

MATLAB's matrix division does not have a block triangular preordering built in, unlike (for example) the Harwell `MA28` code. Block triangular preordering and solution can be implemented easily as an m-file using the `dmperm` function; see §4.3.

Full MATLAB uses the LINPACK condition estimator and gives a warning if the denominator in matrix division is nearly singular. Sparse MATLAB should do the same, but the current version does not yet implement it.

3.4.2. Sparse triangular systems. The triangular linear system solver, which is also the main step of LU factorization, is based on an algorithm of Gilbert and Peierls [15]. When A is triangular and b is a sparse vector, $x = A\backslash b$ is computed in two steps. First, the nonzero structures of A and b are used (as described below) to make a list of the nonzero indices of x. This list is also the list of columns of A that participate nontrivially in the triangular solution. Second, the actual values of x are computed by using each column on the list to update the sparse accumulator with a "spaxpy" operation (§3.1.3). The list is generated in a "topological" order, which is one that guarantees that x_i is computed before column i of A is used in a spaxpy. Increasing order is one topological order of a lower triangular matrix, but any topological order will serve.

It remains to describe how to generate the topologically ordered list of indices efficiently. Consider the directed graph whose vertices are the columns of A, with an edge from j to i if $a_{ij} \neq 0$. (No extra data structure is needed to represent this graph—it is just an interpretation of the standard column data structure for A.) Each nonzero index of b corresponds to a vertex of the graph. The set of nonzero indices of x corresponds to the set of all vertices of b, plus all vertices that can be reached from vertices of b via directed paths in the graph of A. (This is true even if A is not triangular [12].) Any graph-searching algorithm could be used to identify those vertices and find the nonzero indices of x. A depth-first search has the advantage that a topological order for the list can be generated during the search. We add each vertex to the list at the time the depth-first search backtracks from that vertex. This creates the list in the reverse of a topological order; the numerical solution step then processes the list backwards, in topological order.

The reason to use this "reverse postorder" as the topological order is that there seems to be no way to generate the list in increasing or decreasing order, and the time wasted in sorting it would often be more than the number of arithmetic operations. However, the depth-first search examines just once each nonzero of A that participates nontrivially in the solve. Thus generating the list takes time proportional to the number of nonzero arithmetic operations in the numerical solve. This means that LU factorization can run in time proportional to arithmetic operations.

3.5. Least squares and the augmented system. We have not yet written a sparse QR factorization for the core of MATLAB. Instead, linear least squares problems

of the form

$$\min \|b - Ax\|$$

are solved via the augmented system of equations

$$\begin{aligned} r + Ax &= b, \\ A^T r &= 0. \end{aligned}$$

Introducing a residual scaling parameter α, this can be written

$$\begin{pmatrix} \alpha I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} r/\alpha \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}.$$

The augmented matrix, which inherits any sparsity in A, is symmetric, but clearly not positive definite. We ignore the symmetry and solve the linear system with a general sparse LU factorization, although a symmetric, indefinite factorization might be twice as fast.

A recent note by Björck [3] analyzes the choice of the parameter α by bounding the effect of roundoff errors on the error in the computed solution x. The value of α that minimizes the bound involves two quantities, $\|r\|$ and the smallest singular value of A, which are too expensive to compute. Instead, we use an apparently satisfactory substitute,

$$\alpha = \max |a_{ij}|/1000.$$

This approach has been used by several other authors, including Arioli, Duff, and de Rijk [1], who do use a symmetric factorization and a similar heuristic for choosing α.

It is not clear whether augmented matrices, orthogonal factorizations, or iterative methods are preferable for least squares problems, from either an efficiency or an accuracy point of view. We have chosen the augmented matrix approach because it is competitive with the other approaches, and because we could use existing code.

3.6. Eigenvalues of sparse matrices. We expect that most eigenvalue computations involving sparse matrices will be done with iterative methods of Lanczos and Arnoldi type, implemented outside the core of MATLAB as m-files. The most time-consuming portion will be the computation of Ax for sparse A and dense x, which can be done efficiently using our core operations.

However, we do provide one almost direct technique for computing all the eigenvalues (but not the eigenvectors) of a real symmetric or complex Hermitian sparse matrix. The reverse Cuthill–McKee algorithm is first used to provide a permutation that reduces the bandwidth. Then an algorithm of Schwartz [22] provides a sequence of plane rotations that further reduces the bandwidth to tridiagonal. Finally, the symmetric tridiagonal QR algorithm from dense MATLAB yields all the eigenvalues.

4. Examples. This section gives the flavor of sparse MATLAB by presenting several examples. First, we show the effect of reorderings for sparse factorization by illustrating a Cholesky factorization with several different permutations. Then we give two examples of m-files, which are programs written in the MATLAB language to provide functionality that is not implemented in the "core" of MATLAB. These sample m-files are simplified somewhat for the purposes of presentation. They omit some of the error checking that would be present in real implementations, and they could be written to contain more flexible options than they do.

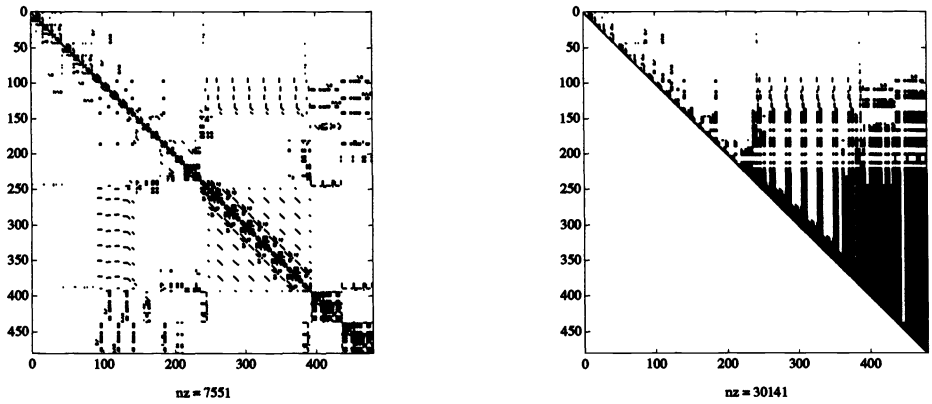


FIG. 4. *The structure of S and its Cholesky factor.*

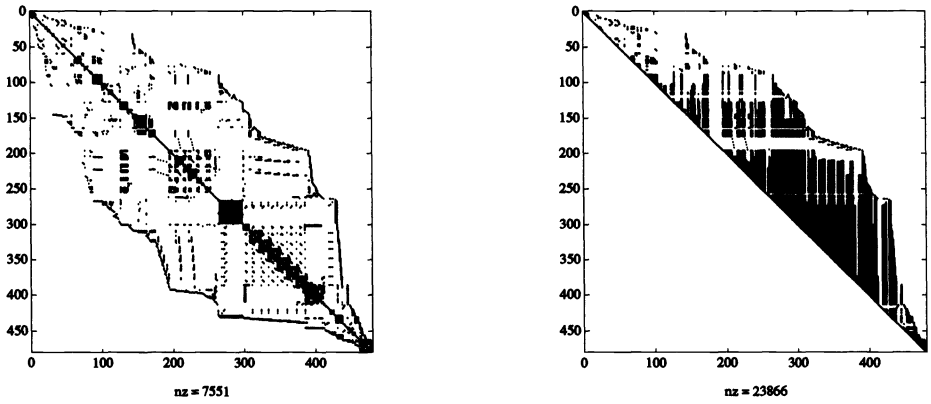


FIG. 5. *Matrix S and its Cholesky factor after reverse Cuthill–McKee reordering.*

4.1. Effect of permutations on Cholesky factors. This sequence of examples illustrates the effect of reorderings on the computation of the Cholesky factorization of one symmetric test matrix. The matrix is $S = WW^T$ where W is the Harwell–Boeing matrix WEST0479 [6], a model due to Westerberg of an eight-stage chemical distillation column.

TABLE 2
Effect of permutations on Cholesky factorization.

	nnz	Time
Original order	30141	5.64
Reverse Cuthill–McKee	23866	4.26
Column count	12675	1.91
Minimum degree	12064	1.75

There are four figures. Each figure shows two spy plots, first a particular symmet-

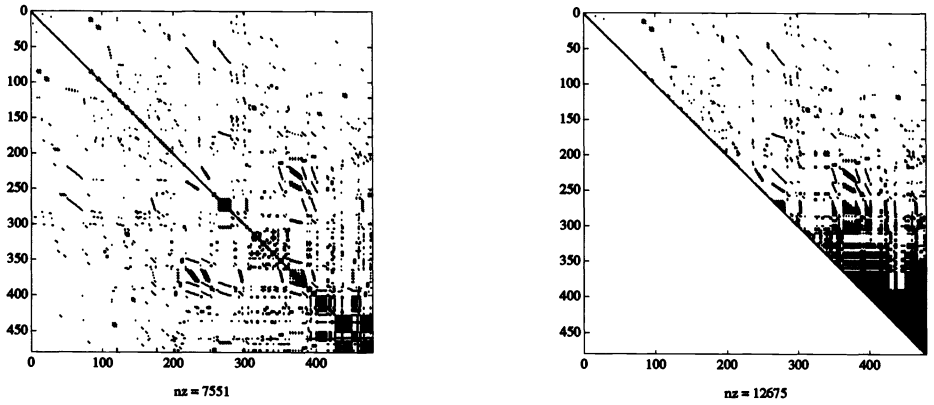


FIG. 6. *Matrix S and its Cholesky factor after column count reordering.*

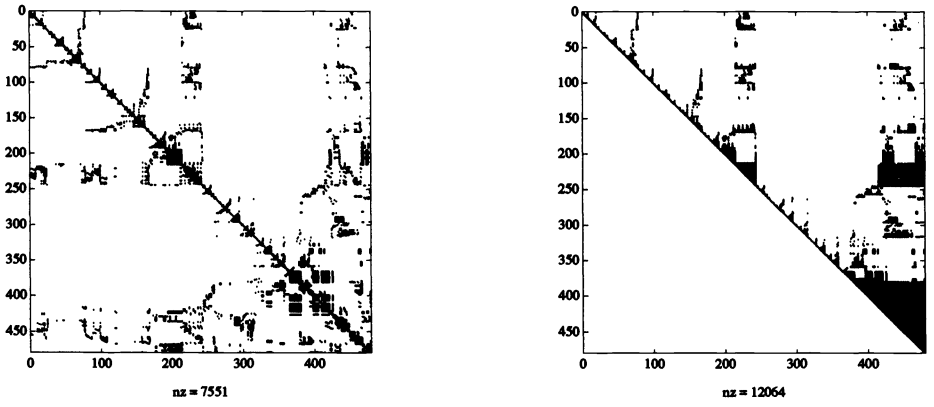


FIG. 7. *Matrix S and its Cholesky factor after minimum-degree reordering.*

ric permutation of S and then the Cholesky factor of the permuted matrix. Figure 4 is the original ordering; Fig. 5 uses symmetric reverse Cuthill–McKee, `symrcm`; Fig. 6 uses the column count permutation, `colperm`; Fig. 7 uses symmetric minimum degree, `symmmd`. Each of the `spy` plots shows a matrix profile that is typical for the underlying permutation: Cuthill–McKee shows an envelope; column count shows all the mass in the later rows and columns; and minimum degree shows a recursive pattern curiously similar to divide-and-conquer orderings like nested dissection.

The matrix S is of order 479 and has 7551 nonzeros. Table 2 shows the number of nonzeros and the execution time in seconds (on a Sun SPARCstation-1) required to compute the Cholesky factors for each of the permutations. The behavior of `symrcm` and `symmmd` is typical; both produce significant reductions in nnz and in the execution time. The behavior of `colperm` is less typical; its reductions are not usually this significant.

```
function x = cgsolve (A,b,tol)

% Solve A*x = b by the conjugate gradient method.
% Iterate until norm(A*x-b) / norm(b) <= tol.

    x = zeros(size(b));
    r = b;
    rtr = r'*r;
    p = zeros(size(b));
    beta = 0;
    while ( norm(r) > tol * norm(b) )
        p = r + beta * p;
        Ap = A * p;
        alpha = rtr / ( p' * Ap );
        x = x + alpha * p;
        r = r - alpha * Ap;
        rtrold = rtr;
        rtr = r'*r;
        beta = rtr / rtrold;
    end
```

FIG. 8. *Solving $Ax = b$ by conjugate gradients.*

```
function x = dmsolve (A,b)

% Solve A*x = b by permuting A to block
% upper triangular form and then performing
% block back substitution.

    % Permute A to block form.
    [p,q,r] = dmperm(A);
    nblocks = length(r)-1;
    A = A(p,q);
    x = b(p);

    % Block backsolve.
    for k = nblocks : -1 : 1

        % Indices above the kth block.
        i = 1 : r(k)-1;

        % Indices of the kth block.
        j = r(k) : r(k+1)-1;
        x(j) = A(j,j) \ x(j);
        x(i) = x(i) - A(i,j) * x(j);

   end;

   % Undo the permutation of x.
   x(q) = x;
```

FIG. 9. *Solving $Ax = b$ by block triangular back-substitution.*

4.2. The conjugate gradient method. Iterative techniques like the conjugate gradient method are often attractive for solving large sparse systems of linear equations. Figure 8 is an m-file for a conjugate gradient method. The code is somewhat simplified—a real code might use a more complicated criterion for termination, might compute Ap in a subroutine call in case A is not held explicitly, and might provide for preconditioning—but it illustrates an important point. Sparsity is never mentioned explicitly in the code. If the argument A is sparse, then `Ap = A*p` will be computed as a sparse operation; if A is full, then all the operations will be full.

In contrast to sparse direct methods, most iterative methods operate on matrices and vectors at a high level, typically using the coefficient matrix only in matrix-vector multiplications. This is the reason for our decision not to build an iterative linear solver into the core of MATLAB; such solvers can be more easily and flexibly written as m-files that make use of the basic sparse operations.

4.3. Solving reducible systems. If A is a reducible matrix, the linear system $Ax = b$ can be solved by permuting A to block upper triangular form (with irreducible diagonal blocks) and then performing block back-substitution. Only the diagonal blocks of the permuted matrix need to be factored, saving fill and arithmetic in the above-diagonal blocks. This strategy is incorporated in some existing Fortran sparse matrix packages, most notably Duff and Reid's code `MA28` in the Harwell Subroutine Library [7]. Figure 9 is an implementation as a MATLAB m-file. This function is a good illustration of the use of permutation vectors.

The call `[p,q,r] = dmperm(A)` returns a row permutation p and a column permutation q to put A in block triangular form. The third output argument r is an integer vector describing the boundaries of the blocks: the kth block of $A(p, q)$ includes indices from $r(k)$ to $r(k+1) - 1$. The loop has one iteration for each diagonal block; note that i and j are vectors of indices. The code resembles an ordinary triangular backsolve, but at each iteration the statement `x(j) = A(j,j) \ x(j)` solves for an entire block of x at once by sparse LU decomposition (with column minimum-degree ordering) of one of the irreducible diagonal blocks of A.

Again, this code is simplified a bit. A real code would merge every sequence of adjacent 1×1 diagonal blocks into a single triangular block, thus reducing the number of iterations of the main loop.

REFERENCES

[1] M. ARIOLI, I. S. DUFF, AND P. P. M. DE RIJK, *On the augmented system approach to least-squares problems*, Numer. Math., 55 (1989), pp. 667–684.

[2] C. ASHCRAFT, R. GRIMES, J. LEWIS, B. PEYTON, AND H. SIMON, *Progress in sparse matrix methods for large linear systems on vector supercomputers*, Internat. J. Supercomput. Appl., 1 (1987), pp. 10–30.

[3] A. BJÖRCK, *A note on scaling in the augmented system methods*, unpublished manuscript, 1991.

[4] T. F. COLEMAN, A. EDENBRANDT, AND J. R. GILBERT, *Predicting fill for sparse orthogonal factorization*, J. Assoc. Comput. Mach., 33 (1986), pp. 517–532.

[5] J. DONGARRA, J. BUNCH, C. MOLER, AND G. STEWART, *LINPACK Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1978.

[6] I. S. DUFF, R. G. GRIMES, AND J. G. LEWIS, *Sparse matrix test problems*, ACM Trans. Math. Software, 15 (1989), pp. 1–14.

[7] I. S. DUFF AND J. K. REID, *Some design features of a sparse matrix code*, ACM Trans. Math. Software, 5 (1979), pp. 18–35.

[8] ———, *The multifrontal solution of indefinite sparse symmetric linear equations*, ACM Trans. Math. Software, 9 (1983), pp. 302–325.

[9] S. C. EISENSTAT, M. H. SCHULTZ, AND A. H. SHERMAN, *Algorithms and data structures for sparse symmetric Gaussian elimination*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 225–237.

[10] A. GEORGE AND J. LIU, *The evolution of the minimum degree ordering algorithm*, SIAM Rev., 31 (1989), pp. 1–19.

[11] ———, *Computer Solution of Large Sparse Positive Definite Systems*, Prentice–Hall, Englewood Cliffs, NJ, 1981.

[12] J. R. GILBERT, *Predicting structure in sparse matrix computations*, Tech. Report 86–750, Cornell University, Ithaca, NY, 1986; SIAM J. Matrix Anal. Appl., submitted.

[13] J. R. GILBERT, C. LEWIS, AND R. SCHREIBER, *Parallel preordering for sparse matrix factorization*, in preparation.

[14] J. R. GILBERT, C. MOLER, AND R. SCHREIBER, *Sparse matrices in Matlab: Design and implementation*, Tech. Report CSL 91–4, Xerox Palo Alto Research Center, Palo Alto, CA, 1991.

[15] J. R. GILBERT AND T. PEIERLS, *Sparse partial pivoting in time proportional to arithmetic operations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 862–874.

[16] J. W. H. LIU, *The role of elimination trees in sparse factorization*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 134–172.

[17] J. W. H. LIU, E. NG, AND B. W. PEYTON, *On finding supernodes for sparse matrix computations*, Tech. Report ORNL/TM-11563, Oak Ridge National Laboratory, Oak Ridge, TN, 1990; SIAM J. Matrix Anal. Appl., 14 (1992), to appear.

[18] THE MATHWORKS, *Pro-Matlab User's Guide*, South Natick, MA, 1990.

[19] C. MOLER, *Matrix computations with Fortran and paging*, Comm. ACM, 15 (1972), pp. 268–270.

[20] A. POTHEN AND C.-J. FAN, *Computing the block triangular form of a sparse matrix*, ACM Trans. Math. Software, 16 (1990), pp. 303–324.

[21] R. SCHREIBER, *A new implementation of sparse Gaussian elimination*, ACM Trans. Math. Software, 8 (1982), pp. 256–276.

[22] H. SCHWARTZ, *Tridiagonalization of a symmetric band matrix*, Numer. Math., 12 (1968), pp. 231–241; Also in [26, pp. 273–283].

[23] B. SMITH, J. BOYLE, Y. IKEBE, V. KLEMA, AND C. MOLER, *Matrix Eigensystem Routines: EISPACK Guide*, Second Edition, Springer-Verlag, New York, 1970.

[24] B. SPEELPENNING, *The generalized element method*, Tech. Report UIUCDCS–R–78–946, University of Illinois, Urbana, IL, 1978.

[25] UNITED KINGDOM ATOMIC ENERGY AUTHORITY, *Harwell subroutine library: A catalogue of subroutines*, Tech. Report AERE R 9185, Harwell Laboratory, Oxfordshire, Great Britain, 1988.

[26] J. WILKINSON AND C. REINSCH, EDS., *Linear Algebra, Vol. 2, Handbook for Automatic Computation*, Springer-Verlag, New York, 1971.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 357–385, January 1992

024

IMPLICIT APPLICATION OF POLYNOMIAL FILTERS IN A K-STEP ARNOLDI METHOD*

D. C. SORENSEN†

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. The Arnoldi process is a well-known technique for approximating a few eigenvalues and corresponding eigenvectors of a general square matrix. Numerical difficulties such as loss of orthogonality and assessment of the numerical quality of the approximations, as well as a potential for unbounded growth in storage, have limited the applicability of the method. These issues are addressed by fixing the number of steps in the Arnoldi process at a prescribed value k and then treating the residual vector as a function of the initial Arnoldi vector. This starting vector is then updated through an iterative scheme that is designed to force convergence of the residual to zero. The iterative scheme is shown to be a truncation of the standard implicitly shifted QR-iteration for dense problems and it avoids the need to explicitly restart the Arnoldi sequence. The main emphasis of this paper is on the derivation and analysis of this scheme. However, there are obvious ways to exploit parallelism through the matrix-vector operations that comprise the majority of the work in the algorithm. Preliminary computational results are given for a few problems on some parallel and vector computers.

Key words. Arnoldi method, eigenvalues, polynomial filter, iterative refinement, parallel computing

AMS(MOS) subject classifications. 65F15, 65G05

1. Introduction.

Large scale eigenvalue problems arise in a variety of settings. Often these very large problems arise through the discretization of a linear differential operator in an attempt to approximate some of the spectral properties of the operator. However, there are a considerable number of sources other than PDE. Saad gives a number of examples in [28].

The Lanczos method [19] is a popular algorithm for solving large symmetric eigenvalue problems. The Arnoldi process [1] is a generalization of the Lanczos method which is appropriate for finding a few eigenvalues and corresponding eigenvectors of a large nonsymmetric matrix. These methods only require one to compute action of the matrix on a vector through a matrix-vector product. Often this may be accomplished without explicit storage of the matrix and this property, along with a number of theoretical and computational features, has contributed to the widespread appeal of these methods. However, both of these share some inherent numerical difficulties, which have been the subject of considerable research over the last two decades [8], [16], [25], [27].

In this paper these methods will be discussed from a new perspective. The goal is to address the nonsymmetric problem and thus the focus is on the Arnoldi algorithm. However, since the Arnoldi method reduces to the Lanczos method when the matrix is symmetric, everything that is developed here is applicable to the symmetric case as well, with obvious savings in computational effort available through the exploitation of symmetry. Traditionally, the point of view has been to let the Arnoldi or the Lanczos sequence develop without bound while monitoring error estimates associated with the Ritz vectors to identify converged eigenvalues. However, if one explores the

* Received by the editors October 29, 1990; accepted for publication (in revised form) July 30, 1991.

† Department of Mathematical Sciences, Rice University, Houston, Texas 77251-1829 (sorensen@rice.edu). This research was supported in part by National Science Foundation cooperative agreement CCR-8809615, Department of Energy contract DE-FG0F-91ER25103, and the Research Institute for Advanced Computer Science.

relation with the QR-iteration it is apparent that the Arnoldi (Lanczos) method is really a truncated reduction of the given matrix into upper Hessenberg (tridiagonal) form. The iterative phase of the QR-method does not have an analogy within the traditional treatment of these algorithms.

A variant of the Arnoldi method which includes such an iterative phase is developed here by analogy to the well-known implicitly shifted QR-iteration [14], [33], [35] for dense matrices. Such an analogy may be developed if one treats the residual vector as a function of the initial Arnoldi (Lanczos) vector, and then attempts to iteratively improve this vector in such a way as to force the residual vector to zero.

In §2 we develop the Arnoldi factorization, expose the functional dependence of the residual on the starting vector, and give necessary and sufficient conditions for a starting vector to produce a zero residual. In §3 we show how to update the starting vector through implicit application of a polynomial filter to this vector on each iteration. The implicit application of this polynomial filter is accomplished through a truncated version of the implicitly shifted QR-iteration. Within this context, an updating scheme is developed which preserves an Arnoldi (Lanczos) factorization of predetermined size. The method generalizes explicit restart methods and as shown in §4, it is possible to implement a mathematically equivalent implicit method corresponding to all of the explicitly restarted methods that we are aware of. Convergence results for specific restart strategies are given in §5. Extension to the generalized problem is discussed in §6 and preliminary computational results are presented in §7.

The idea of iteratively forcing the residual to zero is not new. Variants of this idea were introduced early by Karush in [18]. Cullum and her colleagues have investigated explicit restart methods for the symmetric case [5], [6], [8]. Most recently the idea has been explored by Saad in [28], [29], by Chatelin and Ho in [2], and by Chronopoulos in [3] for the nonsymmetric case. All of these techniques use eigensystem information from the projected matrix to construct an updated starting vector for the Arnoldi (Lanczos) process, and then *restart* this process from scratch. Here, a computational framework is developed that updates the Arnoldi factorization instead of restarting it.

This approach has several advantages over more traditional approaches. The number of eigenvalues that are sought is prespecified. This fixes the storage requirements instead of allowing them to become arbitrarily large. It is expected that the number of eigenvalues that are sought will be modest, and in this situation, orthogonality of the Arnoldi (Lanczos) basis for the Krylov subspace can be maintained. Therefore, the questions of spurious eigenvalues and selective reorthogonalization do not enter. Finally, the well-understood deflation rules associated with the QR-iteration may be carried over directly to the technique.

2. The Arnoldi factorization. The Arnoldi factorization may be viewed as a truncated reduction of an $n \times n$ matrix A to upper Hessenberg form. After k steps of the factorization one has

$$AV = VH + re_k^T, \tag{2.1}$$

where $V \in \mathbf{R}^{n\times k}$, $V^TV = I_k$, $H \in \mathbf{R}^{k\times k}$ is upper Hessenberg, $r \in \mathbf{R}^n$ with $0 = V^Tr$. An alternative way to write (2.1) is

$$AV = (V, v)\begin{pmatrix} H \\ \beta e_k^T \end{pmatrix} \quad \text{where } \beta = \|r\| \quad \text{and} \quad v = \frac{1}{\beta}r\,. \tag{2.2}$$

From this representation, it is apparent that (2.2) is just a truncation of the complete reduction

$$A(V,\hat{V}) = (V,\hat{V}) \begin{pmatrix} H & M \\ \beta e_1 e_k^T & \hat{H} \end{pmatrix} \tag{2.3}$$

where $(V,\hat{V})$ is an orthogonal $n \times n$ matrix and $\hat{H}$ is an upper Hessenberg matrix of order $n-k$. Equation (2.2) and hence (2.1) may be derived from (2.3) by equating the first k columns of both sides and setting $v = \hat{V}e_1$.

Approximate eigenvalues and eigenvectors are readily available through this factorization. If $Hy = y\theta$ is an eigenpair for H, then the vector $x = Vy$ satisfies

$$\|Ax - x\theta\| = \|(AV - VH)y\| = |\beta e_k^T y|.$$

We call the vector x a Ritz vector and the approximate eigenvalue θ a Ritz value and note that the smaller $|\beta e_k^T y|$ is, the better these approximations are.

The factorization (2.1) may be advanced one step through the following recursion formulas:

$$\beta = \|r\|, \qquad v = \frac{1}{\beta} r, \tag{2.3.1}$$

$$V_+ = (V, v), \tag{2.3.2}$$

$$w = Av, \qquad \begin{pmatrix} h \\ \alpha \end{pmatrix} = {V_+}^T w, \tag{2.3.3}$$

$$H_+ = \begin{pmatrix} H & h \\ \beta e_k^T & \alpha \end{pmatrix}, \tag{2.3.4}$$

$$r_+ = w - V_+ \begin{pmatrix} h \\ \alpha \end{pmatrix} = (I - V_+ V_+^T) w. \tag{2.3.5}$$

From this development it is easily seen that

$$AV_+ = V_+ H_+ + r_+ e_{k+1}^T, \qquad V_+^T V_+ = I_{k+1}, V_+^T r_+ = 0.$$

In a certain sense, computation of the projection indicated at step (2.3.5) has been the main source of research activity in this topic. The computational difficulty stems from the fact that $\|r\| = 0$ if and only if the columns of V span an invariant subspace of A. When V "nearly" spans such a subspace $\|r\|$ will be small. Typically, in this situation, a loss of significant digits will take place at step (2.3.5) through numerical cancellation unless special care is taken. On the one hand, it is a delightful situation when $\|r\|$ becomes small because this indicates that the eigenvalues of H are accurate approximations to the eigenvalues of A. On the other hand, this "convergence" will indicate a probable loss of numerical orthogonality in V. The identification of this phenomenon in the symmetric case and the first rigorous numerical treatment is due to Paige [22], [23]. There have been several approaches to overcoming this problem in the symmetric case. They include: (1) complete reorthogonalization which may be accomplished through maintaining V in product Householder form [15], [34] or through the modified Gram–Schmidt processes with reorthogonalization [9], [26]; (2) Selective reorthogonalization, which has been proposed by Parlett and has been

heavily researched by him and his students. Most notably, the thesis and subsequent papers and computer codes of Scott have developed this idea [24], [25], [31]; (3) No reorthogonalization, which has been developed by Cullum and her colleagues. This last option introduces the almost certain possibility of introducing spurious eigenvalues. Various techniques have been developed to detect and deal with the presence of spurious eigenvalues [7], [8].

The appearance of spurious eigenvalues may be avoided through complete reorthogonalization of the Arnoldi (or Lanczos) vectors. Computational cost has been cited as the reason for not employing this option. However, the cost will be reasonable if one is able to fix k at a modest size and then update the starting vector $v_1 = Ve_1$ while repeatedly doing k-Arnoldi steps. This approach has been explored to some extent in [2], [28]. In the symmetric case Cullum [6] relates a variant of this approach (which has been termed an s-step method) to applying a fixed number of conjugate gradient steps to minimize (maximize) $\langle VV^T, A\rangle$ where $\langle B, A\rangle = \text{trace}(B^T A)$ is the Frobenius product functional with V restricted to the generalized block Krylov subspace. However, while this argument gives considerable credence to the restart procedure, it does not establish convergence.

Throughout the remainder of this paper, the k-step approach will be developed from a different point of view. An attempt will be made to iteratively update v_1 in order to force the residual vector $r(v_1)$ to zero. In order to make sense of this it will be necessary to understand when r is indeed a function of v_1 and also to determine its functional form and characterize the zeros of this function.

The classic simple result that explains when r is a function of v_1 is the implicit Q-theorem.

THEOREM 2.4. *Suppose*

$$AV = VH + re_k^T,$$
$$AQ = QG + fe_k^T,$$

where Q, V have orthonormal columns and G, H are both upper Hessenberg with positive subdiagonal elements. If $Qe_1 = Ve_1$ and $Q^T f = V^T r = 0$, then $Q = V$, $G = H$, and $f = r$.

Proof. There is a straightforward inductive proof (or see [16, p. 367]). □

Of course the Krylov space

$$\mathcal{K}_k(A, v_1) = \text{Span}\,\{v_1, Av_1, A^2v_1, \cdots, A^{k-1}v_1\}$$

plays an important role along with the Krylov matrix

$$K = (v_1, Av_1, \cdots, A^{k-1}v_1)\,.$$

An alternate derivation of the Arnoldi process is to consider the companion (or Frobenius) matrix

$$F \equiv \begin{pmatrix} 0 & \gamma_0 \\ I & \hat{g} \end{pmatrix} = \begin{pmatrix} 0 & & & \gamma_0 \\ 1 & & & \gamma_1 \\ & 1 & & \\ & & \ddots & \vdots \\ & & 1 & \gamma_{k-1} \end{pmatrix}$$

and to observe that

$$AK - KF = \hat{r}e_k^T \tag{2.5}$$

where $\hat{r} = A^k v_1 - Kg$ with $g^T = (\gamma_o, \hat{g}^T)$. Note that $\hat{r} = \hat{p}(A)v_1$ where $\hat{p}(\lambda) = \lambda^k + \sum_{j=0}^{k-1} \gamma_j \lambda^j$, and also that $\hat{p}(\lambda)$ is the characteristic polynomial of F. If g is chosen to solve $\min \|A^k v_1 - Kg\|_2$ then $\hat{r}$ is orthogonal to all vectors in $\mathcal{K}_k(A, v_1)$. Moreover, $\hat{p}$ solves $\min_{p \in \mathcal{PM}_k}\{\|p(A)v_1\|\}$ where $\mathcal{PM}_k$ is the set of all monic polynomials of degree k.

To solve the minimization problem in (2.5), one would factor $K = QR$ where Q is orthogonal and R is upper triangular. Note that R is nonsingular if and only if K has linearly independent columns and that Q may be constructed so that $\rho_{jj} = e_j^T R e_j > 0$. One then solves

$$g = R^{-1} Q^T A^k v_1 \ .$$

This choice of g will minimize the residual and also will assure that $0 = Q^T \hat{r}$. Multiplying (2.5) on the right by R^{-1} gives

$$A(KR^{-1}) - (KR^{-1})RFR^{-1} = \hat{r} e_k^T R^{-1} \ ,$$

i.e.,

$$AQ - QG = f e_k^T \tag{2.6}$$

where $Q = KR^{-1}$, $G = RFR^{-1}$ is upper Hessenberg with the same characteristic polynomial as F, and $f = (1/\rho_{kk})\hat{r}$. It is easily verified that $v_1 = Qe_1 = Ve_1$ and $0 = Q^T f$. Thus, the implicit Q-theorem will imply that $Q = V$, $G = H$, and $f = r$. Putting $H = G$ yields

$$\beta_j \equiv e_{j+1}^T H e_j = e_{j+1}^T RFR^{-1} e_j = \frac{\rho_{j+1,j+1}}{\rho_{jj}} \ .$$

Moreover,

$$\frac{1}{\rho_{jj}} \|\hat{p}_j(A)v_1\| = \beta_j = \frac{\rho_{j+1,j+1}}{\rho_{jj}}$$

gives

$$\rho_{j+1,j+1} = \|\hat{r}_j\| = \|\hat{p}_j(A)v_1\| \ .$$

This discussion establishes the following.

THEOREM 2.7. *Let $AV_j = V_j H_j + r_j e_j^T$ be a sequence of successive Arnoldi steps $1 \le j \le k$ and suppose that $\dim(\mathcal{K}_k(A, v_1)) = k$. Then*

$$r_j = \frac{1}{\|\hat{p}_{j-1}(A)v_1\|} \hat{p}_j(A)v_1 \ , \qquad \beta_j = \frac{\|\hat{p}_j(A)v_1\|}{\|\hat{p}_{j-1}(A)v_1\|} \tag{2.7.1}$$

where $\hat{p}_j(\lambda)$ is the characteristic polynomial of H_j. Moreover,

$$\hat{p}_j \text{ solves } \min_{p \in \mathcal{PM}_j} \{\|p(A)v_1\|\} \tag{2.7.2}$$

for $1 \le j \le k$.

The development leading to Theorem 2.7 follows and builds upon the development by Ruhe in [27]. The fact that $\|\hat{p}_k(A)v_1\|$ (the characteristic polynomial of H_k acting on v_1) will minimize $\|p(A)v_1\|$ over all monic polynomials of degree k was proved

by Saad in [29]. Theorem 2.7 points out that the structure of β_j is a somewhat complicated function of A and v_1. This theorem will provide the foundation for a convergence result to be presented later in §5.

The final result of this section will develop necessary and sufficient conditions for a particular starting vector to generate a k-dimensional invariant subspace.

THEOREM 2.8. *Let $AV_k - V_kH_k = r_ke_k^T$ be a k-step Arnoldi factorization of A, with H unreduced (i.e., $r_j \neq 0$, $1 \leq j \leq k-1$). Then $r_k = 0$ if and only if $v_1 = Xy$ where $AX = XJ$ with $\operatorname{rank}(X) = k$ and J a Jordan matrix of order k (i.e., the direct sum of Jordan blocks).*

Proof. If $r_k = 0$, let $H\hat{X} = \hat{X}J$ be the Jordan canonical form of H and put $X = V_k\hat{X}$. Then $AX = XJ$, $\operatorname{rank}(X) = k$ and

$$v_1 = V_ke_1 = V_k\hat{X}\hat{X}^{-1}e_1 = Xy, \quad \text{with } y = \hat{X}^{-1}e_1.$$

Suppose now that $AX = XJ$, $\operatorname{rank}(X) = k$, and $v_1 = Xy$. Then $A^mX = XJ^m$ for any nonnegative integer m and it follows that

$$A^mv_1 = A^mXy = XJ^my \in \operatorname{Range}(X)$$

for all m. Hence, $\dim \mathcal{K}_{k+1}(A, v_1) \leq \operatorname{rank}(X) = k$. Now, H unreduced implies $\dim \mathcal{K}_j(A, v_1) = j$ for $1 \leq j \leq k-1$ and it follows from Theorem 2.7 that $r_k = 0$. □

A similar result may be formulated in terms of Schur vectors instead of generalized eigenvectors. This result will be stated without its proof, which is very similar to the proof of the previous result.

THEOREM 2.9. *Let $AV_k - V_kH_k = r_ke_k^T$ be a k-step Arnoldi factorization of A, with H unreduced. Then $r_k = 0$ if and only if $v_1 = Qy$ where $AQ = QR$ with $Q^HQ = I_k$ and R upper triangular of order k.*

Theorem 2.8 provides the motivation for the algorithms we shall develop. It suggests that one might find an invariant subspace by iteratively replacing the starting vector with a linear combination of approximate eigenvectors corresponding to eigenvalues of interest. Such approximations are readily available through the Arnoldi factorization. This theorem also indicates that it will be impossible to force the residual to zero if v_1 has a component of a generator of a cyclic subspace of dimension greater than k. Theorem 2.9 indicates that our computations can be carried out within the framework of a truncated Schur decomposition and this leads to the development of an implicit restart method that is analogous to the implicitly shifted QR iteration.

3. Updating the Arnoldi factorization via QR-iterations. In this section a direct analogue of the implicitly shifted QR-iteration will be derived in the context of the k-step Arnoldi factorization. This will lead to an updating formula that may be used to implement iterative techniques designed to drive the residual r_k to zero by iteratively forcing v_1 into a subspace spanned by k Schur vectors of A.

Throughout this discussion, the integer k should be thought of as a fixed prespecified integer of modest size. Let p be another positive integer, and consider the result of $k+p$ steps of the Arnoldi process applied to A, which has resulted in the construction of an orthogonal matrix V_{k+p} such that

$$\begin{aligned} AV_{k+p} &= V_{k+p}H_{k+p} + r_{k+p}e_{k+p}^T \\ &= (V_{k+p}, v_{k+p+1}) \begin{pmatrix} H_{k+p} \\ \beta_{k+p}e_{k+p}^T \end{pmatrix}. \end{aligned} \tag{3.1}$$

An analogy of the explicitly shifted QR-algorithm may be applied to this truncated factorization of A. It consists of the following four steps. Let μ be a shift and let $(H - \mu I) = QR$ with Q orthogonal and R upper triangular. Then (putting $V = V_{k+p}$, $H = H_{k+p}$)

$$(A - \mu I)V - V(H - \mu I) = r_{k+p}e_{k+p}^T, \tag{3.1.1}$$

$$(A - \mu I)V - VQR = r_{k+p}e_{k+p}^T, \tag{3.1.2}$$

$$(A - \mu I)(VQ) - (VQ)(RQ) = r_{k+p}e_{k+p}^T Q, \tag{3.1.3}$$

$$A(VQ) - (VQ)(RQ + \mu I) = r_{k+p}e_{k+p}^T Q. \tag{3.1.4}$$

Let $V_+ = VQ$ and $H_+ = RQ + \mu I$. Then H_+ is upper Hessenberg and applying the matrices in (3.1.2) to the vector e_1 to expose the relationship of their first columns gives

$$(A - \mu I)v_1 = v_1^+ \rho_{11}$$

where $\rho_{11} = e_1^T R e_1$, $v_1^+ = V_+ e_1$.

This idea may be extended for up to p shifts being applied successively. The development will continue using the implicit shift strategy. The application of a QR-iteration corresponding to an implicit shift μ produces an upper Hessenberg orthogonal $Q \in \mathbf{R}^{k+p}$ such that

$$AV_{k+p}Q = (V_{k+p}Q\,, v_{k+p+1}) \begin{pmatrix} Q^T H_{k+p} Q \\ \beta_{k+p} e_{k+p}^T Q \end{pmatrix}.$$

An application of p implicit shifts therefore results in

$$AV_{k+p}^+ = (V_{k+p}^+\,, v_{k+p+1}) \begin{pmatrix} H_{k+p}^+ \\ \beta_{k+p} e_{k+p}^T \hat{Q} \end{pmatrix} \tag{3.2}$$

where $V_{k+p}^+ = V_{k+p}\hat{Q}$, $H_{k+p}^+ = \hat{Q}^T H_{k+p}\hat{Q}$, and $\hat{Q} = Q_1 Q_2 \cdots Q_p$, with Q_j the orthogonal matrix associated with the shift μ_j.

Now, partition

$$V_{k+p}^+ = (V_k^+, \hat{V}_p)\,, \qquad H_{k+p}^+ = \begin{pmatrix} H_k^+ & M \\ \hat{\beta}_k e_1 e_k^T & \hat{H}_p \end{pmatrix}, \tag{3.3}$$

and note

$$\beta_{k+p} e_{k+p}^T \hat{Q} = (\underbrace{0, 0 \cdots \tilde{\beta}_{k+p}}_{k}, \underbrace{b^T}_{p}).$$

Substituting into (3.2) gives

$$A(V_k^+, \hat{V}_p) = (V_k^+, \hat{V}_p, v_{k+p+1}) \begin{bmatrix} H_k^+ & M \\ \hat{\beta}_k e_1 e_k^T & \hat{H}_p \\ \tilde{\beta}_{k+p} e_k^T & b^T \end{bmatrix}. \tag{3.4}$$

Equating the first k columns on both sides of (3.4) gives

$$AV_k^+ = V_k^+ H_k^+ + r_k^+ e_k^T \tag{3.5}$$

so that

$$AV_k^+ = (V_k^+, v_{k+1}^+) \begin{pmatrix} H_k^+ \\ \beta_k^+ e_k^T \end{pmatrix} \tag{3.6}$$

where $v_{k+1}^+ = (1/\beta_k^+) r_k^+$, $r_k^+ \equiv (\hat{V}_p e_1 \hat{\beta}_k + v_{k+p+1}\tilde{\beta}_{k+p})$, and $\beta_k^+ = \|r_k^+\|$. Note that $(V_k^+)^T \hat{V}_p e_1 = 0$ and $(V_k^+)^T v_{k+p+1} = 0$, so $(V_k^+)^T v_{k+1}^+ = 0$. Thus (3.6) is a legitimate Arnoldi factorization of A. Using this as a starting point it is possible to use p additional steps of the Arnoldi recursions (2.3.1)–(2.3.5) to return to the original form (3.1). This requires only p evaluations of a matrix-vector product involving the matrix A and the p-new Arnoldi vectors. This is to be contrasted with the Tchebyshev–Arnoldi method of Saad [28] where the entire Arnoldi sequence is restarted. From the standpoint of numerical stability this updating scheme has several advantages:

(1) Orthogonality can be maintained since the value of k is modest.
(2) There is no question of spurious eigenvalues.
(3) There is a fixed storage requirement.
(4) Deflation techniques similar to those associated with the QR-iteration for dealing with numerically small subdiagonal elements of H_k may be taken advantage of directly.

For the sake of clarity, the Arnoldi iteration and the updating procedure will be defined.

ALGORITHM 3.7.

function $[H, V, r] = Arnoldi\ (A, H, V, r, k, p)$

Input: $AV - VH = re_k^T$ *with* $V^TV = I_k$, $V^Tr = 0$.

Output: $AV - VH = re_{k+p}{}^T$ *with* $V^TV = I_{k+p}$, $V^Tr = 0$.

(1) *For* $j = 1, 2, \cdots, p$
  (1) $\beta \leftarrow \|r\|$; *if* $\beta < tol$ *then stop;*
  (2) $H \leftarrow \begin{pmatrix} H \\ \beta e_{k+j-1}^T \end{pmatrix}$; $v \leftarrow \frac{1}{\beta} r$; $V \leftarrow (V, v)$;
  (3) $w \leftarrow Av$;
  (4) $h \leftarrow V^T w$; $H \leftarrow (H, h)$;
  (5) $r \leftarrow w - Vh$;
  (6) *while* $\|s\| > \epsilon \|r\|$;
    (1) $s = V^T r$;
    (2) $r \leftarrow r - Vs$;
    (3) $h \leftarrow h + s$;

Remark 1. Step (1.6) is Gram–Schmidt with iterative refinement to assure orthogonality [9]. For details of implementation, see Reichel and Gragg [26]. Computational experience with this device indicates that it is sufficient to do just one step of iterative refinement.

With the basic Arnoldi factorization defined, it is possible to describe the complete iteration.

ALGORITHM 3.8.

function $[V, H, r] = Arnupd\ (A, k, p, tol)$.

(1) *initialize* $V(:, 1) = v_1$; $H \leftarrow (v_1{}^T A v_1)$; $r \leftarrow Av_1 - v_1 H$;
(2) $[H, V, r] \leftarrow Arnoldi\ (A, H, V, r, 1, k)$
(3) *For* $m = 1, 2, \cdots$
  (1) *if* $(\|r\| < tol)$ *then stop;*
  (2) $[V, H, r] \leftarrow Arnoldi\ (A, H, V, r, k, p)$;

(3) $u = Shifts\ (H, p)$; $\quad (defined\ below)$
(4) $Q \leftarrow I_{k+p}$;
(5) *for* $j = 1, 2, \cdots, p$
  (1) $H \leftarrow Q_j^T H Q_j$; (*Bulge–Chase corresponding to shift* $\mu_j = u(j)$)
  (2) $Q \leftarrow QQ_j$;
(6) $v \leftarrow (VQ)e_{k+1}$; $V \leftarrow (VQ)\binom{I_k}{0}$;
(7) $r \leftarrow (v\beta_k + r\sigma_k)$; *where* $\beta_k = e_{k+1}^T H e_k$, $\sigma_k = e_{k+p}^T Q e_k$;

Remark 2. The Bulge–Chase at step (3(5(1))) is defined implicitly as usual so that $H - \mu_j I = Q_j R_j$; if the shifts are in complex conjugate pairs then the implicit double shift can be implemented to avoid complex arithmetic.

Remark 3. During a Bulge–Chase sweep at step (3(5(1))), it may happen that a subdiagonal element β_j becomes small. The deflation strategies associated with the QR-algorithm are then employed. In this case, the matrix H is split, giving

$$H = \begin{pmatrix} H_j & M \\ \beta_j e_1 e_j{}^T & \hat{H}_j \end{pmatrix} \simeq \begin{pmatrix} H_j & M \\ 0 & \hat{H}_j \end{pmatrix}, \qquad VQ = (V_j, \hat{V}_j) .$$

Thus, an invariant subspace of dimension j has been found. If $j \geq k$ and all the shifts have been applied, then the iteration is halted. Otherwise H_j, V_j are retained and the iteration proceeds with $\hat{V}_j, \hat{H}_j$ filling the role of V, H, respectively. However, H_j continues to participate in the shift selection strategy on subsequent iterations. That is, all of the eigenvalues of H are considered in the selection process. If some of the eigenvalues of H_j are selected as shifts then these are applied implicitly to H_j to split this matrix and the unwanted portion is discarded to form a submatrix of smaller size. If the matrix is nonsymmetric the factorization must be explicitly restarted at the $j+1$ position with a vector that is orthogonal to the first j basis vectors. If the matrix A is symmetric then the corresponding columns of the (updated) matrix V_j are discarded and then $\hat{V}_j$ and $\hat{H}_j$ are moved (concatenated) to the left. The remaining shifts are applied implicitly to $\hat{H}_j$ and then the Arnoldi factorization is completed to fill out the remainder of the $k+p$ columns of V. In this way the iteration is not terminated by deflation until the appropriate approximation to the wanted spectrum has appeared.

As discussed at the beginning of this section, each application of an implicit shift μ_j will replace the starting vector v_1 with $(A - \mu_j I)v_1$. Thus after completion of each cycle of the loop at step (3) in Algorithm 3.8:

$$Ve_1 = v_1 \leftarrow \psi(A)v_1 ;$$

where $\psi(\lambda) = (1/\tau)\prod_{j=1}^{p}(\lambda - \mu_j)$ with τ a normalization factor. Numerous choices are possible for the selection of these p shifts. Some possibilities will be discussed in §5. However, there is one immediate possibility to discuss and that is the case of choosing p "exact" shifts with respect to H. Thus the selection process might be as in Algorithm 3.9.

ALGORITHM 3.9.
function $[u] = Shifts\ (H, p)$
(1) *Compute* $\lambda(H)$ (*by QR, for example*)
(2) *Select* p unwanted *eigenvalues* $\{u(j) \leftarrow \mu_j : 1 \leq j \leq p\} \subset \lambda(H)$

Some obvious criteria for this selection might be
(i) Sort $\lambda(H)$ according to the algebraically largest real part and select the p eigenvalues with the smallest real part as shifts.

(ii) Sort $\lambda(H)$ according to the largest modulus and select the p eigenvalues with the smallest modulus as shifts.

Selecting these exact shifts has interesting consequences in the iteration.

LEMMA 3.10. *Let* $\lambda(H) = \{\theta_1, \cdots, \theta_k\} \cup \{\mu_1, \cdots, \mu_p\}$ *be a disjoint partition of the spectrum of* H *and let*

$$H_+ = Q^T H Q,$$

where $Q = Q_1 Q_2 \cdots Q_p$ *with* Q_j *implicitly determined by the shift* μ_j. *If* $\beta_j \neq 0, \quad 1 \leq j \leq k-1$, *then* $\beta_k = 0$ *and*

$$H_+ = \begin{pmatrix} H_k^+ & M^+ \\ 0 & R_p \end{pmatrix},$$

where $\lambda(H_k^+) = \{\theta_1, \cdots, \theta_k\}$, $\lambda(R_p) = \{\mu_1, \mu_2, \cdots, \mu_p\}$. *Moreover,*

$$v_1^+ = VQe_1 = \sum x_j,$$

where each x_j *is a Ritz vector corresponding to the Ritz value* θ_j, *i.e.,* $x_j = Vy_j$, *where* $Hy_j = y_j\theta_j, \quad 1 \leq j \leq k$.

Proof. After applying the p implicit shifts we have

$$HQ = QH_+$$

with

$$q_1 \equiv Qe_1 = \psi(H)e_1\,, \qquad \psi(\lambda) = \frac{1}{\tau}\prod_{j=1}^{p}(\lambda - \mu_j)\,.$$

Therefore $q_1 = \sum_{j=1}^{k} y_j\zeta_j$ where $Hy_j = y_j\theta_j$, since $q_1 = \psi(H)e_1$ has annihilated any component of e_1 along an eigenvector of H associated with $\mu_j\,, \quad 1 \leq j \leq p$. As a consequence of Theorem 2.8, $\beta_k = 0$ must hold. Moreover, $v_1^+ = VQe_1 = Vq_1 = \sum_{j=1}^{k} Vy_j\zeta_j = \sum_{j=1}^{k} x_j\zeta_j$. □

This lemma provides a very nice interpretation of the iteration when exact shifts are chosen. Casting out the unwanted set of eigenvalues using exact shifts is mathematically equivalent to restarting the Arnoldi factorization from the beginning after updating $v_1 \leftarrow \sum x_j\zeta_j$, a linear combination of Ritz vectors associated with the "wanted" eigenvalues. Thus the updated starting vector has been implicitly replaced by the sum of k approximate eigenvectors.

If A is symmetric and the p algebraically smallest eigenvalues of H are selected for deletion, then this method is similar to the single vector s-step Lanczos process described by Cullum and Donath in [5] and expanded on in [6], [8]. The particular linear combination is apparently different. This variant has the advantage that a restart of the entire Lanczos sequence is not required. Approximate eigenvectors from a Krylov subspace of dimension $k+p$ are available at each iteration for a cost of p rather than $k+p$ matrix-vector products per iteration.

4. Some polynomial filters. The previous discussion has indicated that it would be advantageous to construct polynomials $\psi(\lambda)$ of degree p that filter out certain portions of the spectrum of A. Several researchers have considered such schemes [5], [8], [28]. Related ideas appear throughout the literature of iterative methods for linear systems [17], [21], [30].

We have just described the use of exact shifts to construct such filters. Another particularly appealing polynomial filter may be constructed using Tchebyshev polynomials. In this case, one constructs an ellipse containing the unwanted eigenvalues of H, then at step (3(3)) of Algorithm 3.8 the shifts μ_j are taken to be the zeroes of the Tchebyshev polynomial of degree p associated with this ellipse (i.e., the polynomial of degree p that gives the best approximation to 0 in the max norm). Construction of such an ellipse and the associated polynomials is discussed by Saad [29] and is based on Manteuffel's scheme [20]. Variants of this are presented and discussed by Chatelin and Ho in [2]. Since each of these schemes specifies the filter polynomials by their roots, they may all be implemented within the framework of the algorithms developed in the previous section. At some point in the iteration one might consider fixing the roots used as shifts and continuing with a stationary iteration. The convergence of this strategy is analyzed in the following section.

One may observe that these filters each have the feature of weighting the eigenvalues of the wanted spectrum quite unevenly. For example, in the symmetric case where the wanted spectrum consists of the k largest eigenvalues, the eigenvalues closest to the right end of the spectrum are weighted most heavily. An alternative is to construct polynomial approximations to step functions which take the value zero in unwanted regions and one in wanted regions of the complex plane. One also might construct polynomials that produce an updated $v_1{}^+$, which is a weighted linear combination of approximate eigenvectors corresponding to the wanted eigenvalues.

In order to construct these sorts of filters it is advantageous to be able to apply the filter polynomial which is specified by its coefficients when expanded in the basis of polynomials constructed through the Arnoldi (Lanczos) process. To make this more precise, suppose ψ is any polynomial of degree less than or equal to p. Then expand ψ in the form

$$\psi(\lambda) = \sum_{j=1}^{p+1} \eta_j p_{j-1}(\lambda),$$

where $\{p_j\}$ are the Arnoldi (Lanczos) polynomials. Observe that

$$\psi(A)v_1 = Vy$$

where $y^T = (\eta_1, \eta_2, \cdots, \eta_{p+1}, 0, 0, \cdots, 0)$ since

$$Vy = \sum_{j=1}^{p+1} v_j \eta_j = \sum_{j=1}^{p+1} \eta_j p_{j-1}(A)v_1, \qquad v_j = p_{j-1}(A)v_1.$$

The technique developed in §3 for the implicit application of $\psi(A)$ to v_1 is not directly applicable because the roots of ψ are unknown. One could perhaps compute these roots and then apply the scheme of §3. However, there is an alternate way to implicitly apply this polynomial directly from its expansion in the Arnoldi basis. Assume that $\|y\| = 1$ and construct a vector w_o such that

$$\left(I - 2w_o {w_o}^T\right) e_1 = y. \tag{4.1}$$

Replace H by

$$\hat{H} = \left(I - 2w_o {w_o}^T\right) H \left(I - 2w_o {w_o}^T\right). \tag{4.2}$$

Now, apply the Householder reduction of $\hat{H}$ to upper Hessenberg form so that

$$\hat{H} \leftarrow Q^T H Q$$

where

$$Q = \left(I - 2w_o w_o^T\right)\left(I - 2w_1 w_1^T\right)\cdots\left(I - 2w_{k+p-2} w_{k+p-2}^T\right), \tag{4.3}$$

with each $\left(I - 2w_j w_j^T\right)$ being a Householder transformation constructed to introduce zeros below the $(j+1)$st element of the jth column. Now, consider the application of Q to the Arnoldi factorization:

$$AVQ - VQ(Q^T H Q) = r e_{k+p}^T Q.$$

In order to fit within the updating framework developed in §3, the condition

$$e_{k+p}^T Q e_j = 0, \qquad 1 \le j < k$$

must hold. This is established by the following lemma.

LEMMA 4.4. *The matrix Q displayed in* (4.3) *satisfies $e_{k+p}^T Q e_j = 0$, $1 \le j < k$.*

Proof. Let $Q_j = I - 2w_j w_j^T$ for $0 \le j \le k+p-2$, and let $H^{(j+1)} = Q_j^T H^{(j)} Q_j$ with $H^{(o)} = H$. From (4.1) it follows that $w_o = \theta(y - e_1)$, with $\frac{1}{\theta} = \|y - e_1\|$. Thus, $e_i^T Q_o = e_i^T$ for $i > p+1$. Since

$$Q_o H^{(1)} = H Q_o$$

and since H is upper Hessenberg, it follows that

$$e_i^T H^{(1)} = e_i^T H Q_o = e_i^T H$$

for $i > p+2$. From this one may conclude that $e_i^T w_1 = 0$ for $i > p+2$ and thus $e_i^T Q_1 = e_i^T$ for $i > p+2$. Now, suppose that $e_i^T Q_j = e_i^T$ and that $e_i^T H^{(j)} = e_i^T H$ for $i > p+j+1$. Since $Q_j H^{(j+1)} = H^{(j)} Q_j$, it follows that

$$e_i^T H^{(j+1)} = e_i^T H^{(j)} Q_j = e_i^T H$$

for $i > p+j+2$, and again, one may conclude that $e_i^T w_{j+1} = 0$ so that $e_i^T Q_{j+1} = e_i^T$ for $i > p+j+2$. This inductive argument continues to hold until $j = k-1$. Hence,

$$e_{k+p}^T Q = e_{k+p}^T Q_{k-1} Q_k \cdots Q_{k+p-2}.$$

Now, observe that $Q_i e_j = e_j$ for $k-1 \le i \le k+p-2$ and for $1 \le j < k$ to establish the result. □

This observation allows the application of any polynomial filter of degree p when the polynomial is expanded in the Arnoldi basis.

Moreover, it provides the means for implicit application of all of the suggested restart methods known to this author which are not specified by roots of polynomials.

5. Some convergence results. In this section, we analyze the algorithm just developed with respect to two strategies for constructing the filter polynomials. The first analysis applies to general (nonsymmetric) matrices but the filter polynomials are stationary. The second analyzes the "exact shift" polynomials but applies only to symmetric matrices. In this sense, each of the two is incomplete. However, they both give insight to the nature of the convergence of this method.

Let us begin with an analysis of the stationary iteration. To this end we define

$$\lambda(A) = \lambda_W(A) \cup \lambda_U(A)$$

where $\lambda_W(A) = \{\lambda_1, \lambda_2, \cdots, \lambda_k\}$ and $\lambda_U(A) = \{\lambda_{k+1}, \lambda_{k+2}, \cdots, \lambda_n\}$. By a stationary iteration, we mean that the set of shifts $\{\mu_1, \mu_2, \cdots, \mu_p\}$ used to construct the filter polynomial remains fixed. This means that the filter polynomials are all multiples of a fixed (i.e., stationary) polynomial $\psi(\lambda) = \prod_{i=1}^{k}(\lambda - \mu_i)$.

We define

$$v_1^{(0)} = v \quad \text{and} \quad v_1^{(j)} = \psi(A)v_1^{(j-1)}/\|\psi(A)v_1^{(j-1)}\|,$$

where v is an arbitrary starting vector of length one. We define $\pi_j = \beta_1{}^{(j)}\beta_2{}^{(j)} \cdots \beta_k{}^{(j)}$ where the $\beta_i{}^{(j)}$ are the subdiagonal elements of H in the Arnoldi factorization resulting from the starting vector $v_1^{(j)}$.

THEOREM 5.1. *Assume that the fixed polynomial ψ satisfies*

$$|\psi(\lambda_1)| \geq |\psi(\lambda_2)| \geq \cdots \geq |\psi(\lambda_k)| > |\psi(\lambda_{k+1})| \geq \cdots \geq |\psi(\lambda_n)| \ ,$$

with

$$\gamma = |\psi(\lambda_{k+1})|/|\psi(\lambda_k)| < 1,$$

and assume that the starting vector v_1 is not a member of the invariant subspace corresponding to $\{\lambda_{k+1}, \cdots, \lambda_n\}$. Then the sequence $\{\pi_j\}$ converges to zero. Moreover, there is a fixed constant K and a positive integer J such that

$$0 \leq \pi_j \leq \gamma^j K$$

for all $j > J$.

Proof. Let

$$A(Q_1, Q_2) = (Q_1, Q_2) \begin{pmatrix} R_1 & M \\ 0 & R_2 \end{pmatrix} \tag{5.2}$$

be a Schur decomposition of A with $\lambda(R_1) = \lambda_W(A)$. The hypothesis on ψ assures $\lambda_W(A) \cap \lambda_U(A)$ is empty and thus there is a unique $k \times (n-k)$ matrix solution Γ to the Sylvester equation

$$\Gamma R_2 - R_1\Gamma = M \ .$$

Since

$$\begin{pmatrix} I & \Gamma \\ 0 & I \end{pmatrix} \begin{pmatrix} R_1 & 0 \\ 0 & R_2 \end{pmatrix} = \begin{pmatrix} R_1 & M \\ 0 & R_2 \end{pmatrix} \begin{pmatrix} I & \Gamma \\ 0 & I \end{pmatrix},$$

it follows that

$$\phi(A)(Q_1, Q_1\Gamma + Q_2) = (Q_1, Q_1\Gamma + Q_2) \begin{pmatrix} \phi(R_1) & 0 \\ 0 & \phi(R_2) \end{pmatrix}$$

for any polynomial ϕ. Let $\hat{Q}_2 = Q_1\Gamma + Q_2$ and let $v_1 = Q_1 y_1 + \hat{Q}_2 y_2$. Then,

$$\psi^j(A)v = Q_1\psi^j(R_1)y_1 + \hat{Q}_2\psi^j(R_2)y_2 \equiv Q_1 y_1{}^{(j)} + \hat{Q}_2 y_2{}^{(j)}.$$

Recall that $v_1^{(j)} = \psi^j(A)v_1/\|\psi^j(A)v_1\|$ and that from Theorem 2.7, one obtains $\pi_j = \min_{p\in\mathcal{PM}_k}\{\|p(A)v_1^{(j)}\|\}$. Thus, putting $\hat{p}$ to be the characteristic polynomial of R_1 gives

$$\begin{aligned}\pi_j\|\psi^j(A)v_1\| &\le \|\hat{p}(A)\psi^j(A)v_1\| \le \|\hat{p}(A)Q_1{y_1}^{(j)} + \hat{p}(A)\hat{Q}_2{y_2}^{(j)}\| \\ &= \|Q_1\hat{p}(R_1){y_1}^{(j)} + \hat{p}(A)\hat{Q}_2{y_2}^{(j)}\| = \|\hat{p}(A)\hat{Q}_2{y_2}^{(j)}\|.\end{aligned}$$

Dividing both sides by $|\psi^j(\lambda_k)|$ gives

$$\pi_j(\|\psi^j(A)v_1\|/|\psi^j(\lambda_k)|) \le \|\hat{p}(A)\hat{Q}_2{y_2}^{(j)}\|/|\psi^j(\lambda_k)| \le \left\|\hat{p}(A)\hat{Q}_2\left[\frac{1}{\psi(\lambda_k)}\psi(R_2)\right]^j y_2\right\|.$$

Note the spectral radius of $(1/\psi(\lambda_k))\psi(R_2)$ is less than the number γ according to the hypothesis. Using the fact that there is a consistent matrix norm ν such that $\nu([(1/\psi(\lambda_k))\psi(R_2)]^j) \le \gamma^j$, together with the equivalence of matrix norms, implies the existence of a positive constant K_o such that

$$\left\|\hat{p}(A)Q_2\left[\frac{1}{\psi(\lambda_k)}\psi(R_2)\right]^j y_2\right\| \le K_o\gamma^j\|y_2\| \ .$$

Thus

$$\pi_j \le \frac{K_o\gamma^j\|y_2\|}{(\|\psi^j(A)v_1\|/|\psi^j(\lambda_k)|)} \ .$$

Moreover, by hypothesis v_1 is not in the invariant subspace corresponding to $\{\lambda_{k+1},\cdots,\lambda_n\}$, and it follows that $y_1 \neq 0$. Since the spectral radius of $\psi(R_2)/\psi(\lambda_k)$ is less than one and since every eigenvalue (i.e., every diagonal element) of the triangular matrix $\psi(R_1)/\psi(\lambda_k)$ is greater than or equal to one, there is a J such that $j > J$ implies

$$\begin{aligned}\|\psi^j(A)v_1\|/|\psi^j(\lambda_k)| &= \|Q_1\psi^j(R_1)y_1 + \hat{Q}_2\psi^j(R_2)y_2\|/|\psi^j(\lambda_k)| \\ &\ge \|Q_1\psi^j(R_1)y_1\|/|\psi^j(\lambda_k)| - \|\hat{Q}_2\psi^j(R_2)y_2\|/|\psi^j(\lambda_k)| \\ &\ge |\eta| - \tfrac{1}{2}|\eta\ |,\end{aligned}$$

where η is the last nonzero component of y_1. Thus, the result is established with $K = 2K_o(\|y_2\|/|\eta|)$. □

This result applies directly and generally to cases where there is a priori information about the location of the spectrum and where a single polynomial might be constructed (e.g., Tchebyshev polynomials). The analysis does not apply to the adaptive algorithms. However, it does give an indication of how these might behave near the final stages of the iteration where the filter polynomials tend to become stationary.

Next, we analyze the "exact shift" filter polynomials in the symmetric case. This analysis makes heavy use of the interlace property of eigenvalues of symmetric matrices modified by low-rank changes. The analysis relies on two technical lemmas, which we must establish first.

LEMMA 5.3. *Let*

$$M = \begin{pmatrix} T & \beta e_k \\ \beta e_k^T & \alpha \end{pmatrix}$$

be a symmetric tridiagonal matrix. Then the roots of the equation

$$\beta^2 e_k^T (T - \lambda I)^{-1} e_k = \alpha - \lambda \tag{5.4}$$

are eigenvalues of M.

See [25] for a proof.

If $T = Y\Theta Y^T$ is the eigendecomposition of T then (5.4) becomes

$$\beta^2 \sum_{i=1}^{k} \frac{\eta_i^2}{(\theta_i - \lambda)} = \alpha - \lambda \tag{5.5}$$

where $(\eta_1, \eta_2, \cdots, \eta_k) = e_k^T Y$ and $\Theta = \operatorname{diag}(\theta_1, \theta_2, \cdots, \theta_k)$. It is easily verified that this equation has exactly one root in each of the intervals

$$(-\infty, \theta_1),\ (\theta_1, \theta_2), \cdots (\theta_{k-1}, \theta_k),\ (\theta_k, \infty)$$

and that the k largest roots $\hat{\theta}_i$ of this equation satisfy

$$\theta_i < \hat{\theta}_i \quad \text{for} \quad i = 1, \cdots, k.$$

Also, we note for the sequal that if the subdiagonals of T are all nonzero, then none of the η_j are zero and the θ_j are distinct.

The next lemma shows that when the starting vector v_1 nears a subspace of dimension less than k, then deflation must occur.

LEMMA 5.6. *Suppose* $AV = VH + re_k^T$ *is an Arnoldi factorization of* A *and let* β_j *be the* j*th subdiagonal element of* H. *If* $v_1 = q\gamma + w\sigma$ *with* $\gamma^2 + \sigma^2 = 1$, $\|q\| = \|w\| = 1$, $q^T w = 0$, *and* $q = \sum_{j=1}^{i} q_j \gamma_j, Aq_j = q_j \lambda_j$ (*where* $\{\lambda_j\}$ *are an arbitrary set of* i *eigenvalues of* A), *then*

$$\prod_{j=1}^{i} \beta_j \le \sigma \left\| \prod_{j=1}^{i} (A - \lambda_j I) \right\|.$$

Proof. From Theorem 2.7 and the fact that $\prod_{j=1}^{i} (A - \lambda_j I) q = 0$, we have

$$\begin{aligned}
\prod_{j=1}^{i} \beta_j &= \min_{p \in \mathcal{PM}_i} \|p(A) v_1\| \\
&\le \left\| \prod_{j=1}^{i} (A - \lambda_j I)(q\gamma + w\sigma) \right\| \\
&= \left\| \prod_{j=1}^{i} (A - \lambda_j I) w\sigma \right\| \\
&\le \sigma \left\| \prod_{j=1}^{i} (A - \lambda_j I) \right\|.
\end{aligned}$$

□

With these technical lemmas established it will be possible to analyze the iterations using polynomial filters constructed from exact shifts in the symmetric case.

We assume throughout the remainder of this section that the matrix A is symmetric and hence that $H = T$ is tridiagonal. The selection rule to be analyzed is to retain the k largest eigenvalues of T_{k+p}. Let m denote the iteration number. Then $v_1^{(m)}$ is the starting vector, and

$$AV_{k+p}^{(m)} - V_{k+p}^{(m)} T_{k+p}^{(m)} = r_{k+p}^{(m)} e_{k+p}^T .$$

Let

$$T_{k+p}^{(m)} = \begin{pmatrix} T_k^{(m)} & \beta_k^{(m)} e_k e_1^T \\ \beta_k^{(m)} e_1 e_k^T & \hat{T}^{(m)} \end{pmatrix}$$

have eigenvalues

$$\mu_{1,m+1} < \cdots < \mu_{p,m+1} < \theta_{1,m+1} < \cdots < \theta_{k,m+1}$$

and let $T_k^{(m)}$ have eigenvalues

$$\theta_{1m} < \theta_{2m} < \cdots < \theta_{km}.$$

Then, the exact implicit shift strategy provides

$$Q^{(m)T} T_{k+p}^{(m)} Q^{(m)} = \begin{pmatrix} T_k^{(m+1)} & 0 \\ 0 & \Omega_p^{(m+1)} \end{pmatrix}$$

where $Q^{(m)} = Q_{1m} Q_{2m} \cdots Q_{pm}$ are the orthogonal matrices constructed to apply the implicit shifts $\mu_{1m} \cdots \mu_{pm}$ at step (3(5)) of Algorithm 3.8. Step (3(6)) gives

$$V_k^{(m+1)} = [V_{k+p}^{(m)} Q^{(m)}] \begin{bmatrix} I_k \\ 0 \end{bmatrix} .$$

LEMMA 5.7. *Each* $\{\theta_{j,m} : m = 1, 2, \cdots\}$ *is an increasing convergent sequence for each* $j = 1, 2, \cdots, k$.

Proof. Since $T_{k+p}^{(m)}$ is obtained through a succession of p borderings of $T_k^{(m)}$, it follows from p successive applications of Lemma 5.3 that

$$\theta_{j,m} < \theta_{j,m+1} \quad \text{for} \quad j = 1, \cdots, k.$$

Since each $\theta_{j,m}$ is a Raleigh quotient with respect to A it follows that $\lambda_1 \leq \theta_{j,m} \leq \lambda_n$ for all j, m. Since bounded increasing sequences are convergent the lemma is established. □

We now establish that the limits of the convergent sequences $\theta_{j,m}$ are eigenvalues of the matrix A.

LEMMA 5.8. *Let* $T_k^{(m)} = Y^{(m)} \Theta^{(m)} Y^{(m)T}$ *where* $(\eta_1^{(m)}, \eta_2^{(m)}, \cdots, \eta_k^{(m)}) = e_k^T Y^{(m)}$. *Assume* θ_j *are distinct, where* $\theta_{j,m} \to \theta_j$. *Then*

$$\beta_k^{(m)} \eta_j^{(m)} \to 0 \quad \text{as } m \to \infty, \quad \text{for } j = 1, \cdots, k$$

and as a consequence

$$\|AV^{(m)} y_j^{(m)} - y_j^{(m)} \theta_{j,m}\| = |\beta_k^{(m)} \eta_j^{(m)}| \to 0,$$

where $y_j^{(m)} = Y^{(m)} e_j$ *for* $j = 1, \cdots, k$.

Proof. Consider the leading $(k+1) \times (k+1)$ submatrix of $T_{k+p}^{(m)}$

$$M^{(m)} = \begin{pmatrix} T_k^{(m)} & \beta_k^{(m)} e_k \\ \beta_k^{(m)} e_k^T & \alpha^{(m)} \end{pmatrix}.$$

From Lemma 5.3 it follows that the k largest eigenvalues $\hat{\theta}_{j,m}$ of $M^{(m)}$ satisfy

$$\theta_{j,m} < \hat{\theta}_{j,m} < \theta_{j,m+1}.$$

Moreover, a straightforward algebraic manipulation of (5.5) gives

$$(\beta \eta_j)^2 = (\theta_j - \lambda) \left[\frac{(\alpha - \lambda) - \beta^2 \sum_{i=j+1}^{k} \frac{\eta_i^2}{(\theta_i - \lambda)}}{1 + \sum_{i=1}^{j-1} \frac{\eta_i^2 (\theta_j - \lambda)}{\eta_j^2 (\theta_i - \lambda)}} \right]$$

for any root λ. Substituting the appropriate quantities indexed by m from the matrix $M^{(m)}$ and putting $\lambda = \hat{\theta}_{j,m}$ gives

$$(\beta_k^{(m)} \eta_j^{(m)})^2 < |(\theta_{j,m} - \hat{\theta}_{j,m})| \left| (\alpha^{(m)} - \hat{\theta}_{j,m}) - {\beta_k^{(m)}}^2 \sum_{i=j+1}^{k} \frac{{\eta_i^{(m)}}^2}{(\theta_{i,m} - \hat{\theta}_{j,m})} \right|.$$

The assumption that the limits θ_j are distinct implies that the quantities

$$\left| (\alpha^{(m)} - \hat{\theta}_{j,m}) - {\beta_k^{(m)}}^2 \sum_{i=j+1}^{k} \frac{{\eta_i^{(m)}}^2}{(\theta_{i,m} - \hat{\theta}_{j,m})} \right|$$

have finite limits for each j. Hence, for m sufficiently large there is a positive constant K such that

$$(\beta_k^{(m)} \eta_j^{(m)})^2 < K |\theta_{j,m} - \hat{\theta}_{j,m}| < K |\theta_{j,m} - \theta_{j,m+1}| \to 0$$

as $m \to \infty$. □

The final task is to show that not only will the limits θ_j be eigenvalues of A, but they will be the k eigenvalues of interest. To do this, we shall show that if deflation does not occur, then the coefficients $\gamma_j^{(m)} = q_j^T v_1^{(m)}$ must converge to zero for $j = 1, \cdots, n-k$ where q_j is the eigenvector of A corresponding to the eigenvalue λ_j and $v_1^{(m)}$ is the Arnoldi starting vector for the mth iteration.

Define $\psi_\ell(\lambda) = \prod_{i=1}^{p} (\lambda - \mu_{i,\ell})$ and $\Psi_m(\lambda) = \prod_{i=1}^{m} \psi_i(A)$. Then note that $v_1^{(m)} = (\Psi_m(A) v_1 / \|\Psi_m(A) v_1\|)$.

THEOREM 5.9. *Suppose that the initial starting vector* v_1 *satisfies* $q_j^T v_1 = \gamma_j \neq 0$ *for* $j = n-k+1, \cdots, n$, *where* q_j *is the eigenvector of* A *corresponding to the eigenvalue* λ_j *with the eigenvalues of* A *listed in increasing order. Let* $\beta_i^{(m)}$ *be the* i*th subdiagonal element of* $T_k^{(m)}$ *and assume that* $\beta_i^{(m)} > \epsilon > 0$ *for all* i, m. *Then the sequences*

$$\theta_{j,m} \to \theta_j = \lambda_{n-k+j} \quad \text{as } m \to \infty.$$

Proof. The assumption $\beta_i^{(m)} > \epsilon > 0$ assures that separation of the $\theta_{j,m}$ is uniform over all m so that the limits θ_j are distinct. This implies that each θ_j is an eigenvalue

of A . Moreover, the assumption implies a uniform lower bound on the quantities $|\eta_j^{(m)}|$ and thus $\beta_k^{(m)} \to 0$. (All due to remarks following Lemma 5.3.)

Let $\phi_m(\lambda) = \prod_{i=1}^{k}(\lambda - \theta_{j,m})$, and let $\phi(\lambda) = \prod_{i=1}^{k}(\lambda - \theta_j)$ be the limit polynomial of the ϕ_m. Then

$$\|\phi_m(A)v_1^{(m)}\| = \prod_{j=1}^{k} \beta_j^{(m)} \to 0$$

and thus

$$\phi_m(\lambda_j)\gamma_j^{(m)} = q_j^T \phi_m(A)v_1^{(m)} \to 0.$$

Hence, either

$$\phi(\lambda_j) = 0 \quad \text{or} \quad \gamma_j^{(m)} \to 0$$

for $j = 1, \cdots, n$. This means that $n - k$ of the expansion coefficients $\gamma_j^{(m)}$ tend to 0 as $m \to \infty$. Moreover, Lemma 5.7 implies that the k expansion coefficients corresponding to the eigenvalues θ_j must all be bounded away from zero due to the assumption $\beta_j^{(m)} > \epsilon > 0$ for all j, m.

Now, suppose that $\lambda_{j_k} = \theta_k < \lambda_n$. Then the expansion coefficient

$$\gamma_{j_k}^{(m)} = q_{j_k}^T v_1^{(m)} = q_{j_k}^T \frac{\Psi_m(A)v_1}{\|\Psi_m(A)v_1\|} = \frac{\gamma_{j_k}\Psi_m(\theta_k)}{\sqrt{\sum_{i=1}^{n} \gamma_i^2 \Psi_m^2(\lambda_i)}}.$$

Hence

$$(\gamma_{j_k}^{(m)})^2 = \frac{(\gamma_{j_k}\Psi_m(\theta_k)/\Psi_m(\lambda_n))^2}{\gamma_n^2 + \sum_{i=1}^{n-1} \gamma_i^2 \Psi_m^2(\lambda_i)/\Psi_m^2(\lambda_n)} \leq \left(\frac{\gamma_{j_k}}{\gamma_n}\frac{\Psi_m(\theta_k)}{\Psi_m(\lambda_n)}\right)^2$$

where the γ_i are the expansion coefficients of $v_1^{(o)}$. Now, the roots $\mu_{i,m}$ of the filter polynomials all satsify $\lambda_1 \leq \mu_{i,m} < \theta_{k,m} \leq \theta_k < \lambda_n$ so that

$$0 \leq \frac{\Psi_m(\theta_k)}{\Psi_m(\lambda_n)} = \prod_{\ell=1}^{m}\left(\prod_{i=1}^{p}\left(\frac{\theta_k - \mu_{i\ell}}{\lambda_n - \mu_{i\ell}}\right)\right) \leq \left(\frac{\theta_k - \lambda_1}{\lambda_n - \lambda_1}\right)^{mp} \to 0,$$

since $(\theta_k - \lambda_1/\lambda_n - \lambda_1) < 1$. This is a contradiction. We conclude that $\theta_k = \lambda_n$.

A similar argument may be carried out for each j in turn for the cases $\theta_j < \lambda_{n-k+j}$ and this concludes the proof. □

6. The generalized eigenvalue problem. In this section the generalized eigenvalue problem will briefly be discussed. The generalized problem is to find (x, λ) such that

$$Ax = \lambda Mx\ .$$

In many cases the matrix M is symmetric and positive definite, and this condition shall be assumed in this section. The basic iterative method will carry over to this setting with very little modification. In this setting we maintain and update a factorization of the form

$$AV - MVH = re_k^T \tag{6.1}$$

where

$$V^T M V = I \quad \text{and} \quad V^T r = 0 .$$

It is easily seen that one may apply the algorithm for the standard case to the matrix $M^{-1}A$ in place of A. Of course this would be implemented through factorization of M at the outset and solution of the appropriate linear system instead of applying M^{-1}.

There are two key consequences of maintaining the form (6.1):

1. $Q^T V^T M V Q = I$ is preserved so the implicit QR-shift strategy may be applied.
2. If $A = A^T$ is symmetric, then

$$H = V^T A V$$

follows from $V^T M V = I$, $V^T r = 0$ so that $H = H^T$ will be symmetric and tridiagonal when A is symmetric.

With these observations, it is straightforward to adapt the algorithms previously discussed to solve the generalized eigenproblem. Some limited computational experience with this approach is the subject of the following section.

7. Computational results and conclusions. Computational results for this technique are quite promising but are certainly preliminary. There is a Fortran implementation of the algorithms developed here. Two versions of the code have been produced. One of these implements the strategy for the generalized symmetric eigenvalue problem as described in §6. The other implements the algorithm for the standard nonsymmetric eigenproblem. In addition to exhibiting behavior on some test problems, two experiences with applications will be discussed. Finally, some very interesting illustrations of the shapes of the filter polynomials that are constructed through exact shifts shall be reported.

There are some important details of the Fortran implementation of Algorithm 3.7. Step (3) requires a user-supplied matrix-vector product. Steps (4) and (5) are implemented through calls to the level-two BLAS [11], [12] routine DGEMV. One step of iterative refinement is carried out at step (6) of Algorithm 3.7 rather than iterating until the test $\|s\| \leq \epsilon \|r\|$ is passed. Steps (6(1)) and (6(2)) were also implemented through calls to DGEMV. In all of the computations observed, there was never a loss of orthogonality in the columns of V. In all cases $\|V^T V - I\|$ was on the order of unit roundoff error. Eigenvalue calculations used a slight modification of EISPACK [32] subroutines TQL in the symmetric case and HQR in the nonsymmetric case. These may be replaced by the corresponding block routines from LAPACK [10] to enhance performance in the future.

Expressing the algorithm in terms of the level-two BLAS has provided the means to achieving high performance portable Fortran code. The code has been run on SUN SPARC, CONVEX C1, Stardent Titan, CRAY 2, and CRAY YMP computers. The cost of operations were clearly dominated by the user-supplied matrix-vector products (and system solves in the generalized problem). The time spent in the user-supplied portion was orders of magnitude over the time spent in the other parts of the eigenvalue calculations. This performance characteristic is a direct consequence of the performance of DGEMV on the architectures of the machines listed above. The crucial point for improving the algorithm is to better understand the construction of

TABLE 7.1
Discrete Laplacian.

Dimension	Niters	$\|r\|$	$\|Ax - x\lambda\|$
100	12	1.4-06	3D-15
256	23	3.4-06	5D-15
400	29	6.5-06	5D-15
625	25	7.1-06	3D-14
900	29	6.2-06	2D-14
1600	43	2.9-06	6D-14
2500	50	1.1-05	9D-13
3600	63	9.9-06	4D-11
4900	92	8.9-06	1D-11
8100	237	1.1-05	1D-11
10000	165	1.1-05	8D-12

the filter polynomials in order to reduce the required number of user-supplied matrix-vector products. Parallelism may be invoked through the level-two BLAS and also through the user-supplied matrix-vector product.

In all of the results reported below, exact shifts were used as described in §3. The iteration was halted when $\|(e_k^T y_j) r_k\| < 10^{-7}, 1 \le j \le k-3$, where y_j is the jth Ritz vector corresponding to Ritz values approximating the wanted spectrum. This ad hoc stopping rule allowed the iteration to halt quite early in cases where it was difficult to make a clean separation between the wanted and unwanted spectrum. This ad hoc criterion will have to be replaced with a more rigorous one in the future.

In the first set of test problems the matrix A arises from a standard five-point discretization of the convection-diffusion operator on the unit square Ω. The partial differential equations (PDE) is

$$-\Delta u + \rho u_x = \lambda u, \quad \text{in } \Omega, \quad u|_{\partial\Omega} = 0.$$

When $\rho = 0$ the matrix A is the discrete Laplacian and, for $\rho > 0$, A has distinct complex eigenvalues that appear in a rectangular grid in the complex plane when the cell size $h = 1/(n+1)$ is large enough with respect to the parameter ρ. However, the boundary conditions of the continuous problem do not admit eigenfunctions corresponding to complex eigenvalues, so the eigenvalues of the matrix A become real when the mesh size becomes small enough. The order of the discrete operator A is $N = n^2$ and since its eigenvalues are distinct, it is diagonalizable. These problems allowed testing of the algorithm for accuracy and performance in some interesting but well-understood cases. In both Tables 7.1 and 7.2, the values $k = 10$ and $p = 10$ were used. The two columns on the right of the tables give the norm of the residual vector r and the norm of the true residual $\|Ax - x\lambda\|$ for the sixth eigenvalue. Typically, the eigenvalues of smaller index had residuals that were smaller than this one. For the symmetric problems the residual estimates were uniformly small for the eight smallest eigenvalues.

In Table 7.2 below, the problems of orders 256 and 400 did not satisfy the convergence test before the maximum number of iterations allowed had been reached. In all cases the ten eigenvalues of smallest real part were sought. In both of the problems just mentioned, five or more eigenvalues had been determined to high accuracy. In all cases the iterations could have halted much earlier if a better stopping criterion were devised.

The second set of results will briefly describe two problems that arise in the context of solving PDEs. The first of these involves a discretization of a membrane

TABLE 7.2
Convection diffusion.

Dimension	Niters	$\|r\|$	$\|Ax - x\lambda\|$
100	61	5.3-06	1D-12
256	100	.23	1D-5
400	100	5.2-03	2D-10
625	77	2.3-06	8D-12
900	153	8.9-06	2D-14
1600	103	7.4-06	6D-14

problem in which the membrane is composed of two materials. On an open bounded connected set $\Omega \subset \mathbf{R}^2$, we consider

$$-\Delta u = \lambda \rho u, \quad \text{in } \Omega, \quad u|_{\partial\Omega} = 0,$$

where the density ρ is of the form

$$\rho = \alpha\chi_S + \beta(1 - \chi_S),$$

where χ_S is the characteristic function of a subset $S \subset \Omega$ with area γ. The problem is to determine the density function ρ that minimizes the lowest eigenvalue $\lambda_1(\rho)$ of this PDE. Here α and β are the known (constant) densities of two given materials in respective volume fractions $\gamma/|\Omega|$ and $1 - \gamma/|\Omega|$ and the set S is occupied by the material with density α. Cox [4] has formulated an algorithm to solve this minimization problem. The algorithm generates a sequence of symmetric generalized eigenvalue problems

$$Av = \lambda M(\rho)v,$$

which arise through a bilinear finite element discretization of the PDE. The density function ρ is modified at each iteration with the set S determined through level sets of the corresponding eigenfunction. The matrix A is positive definite and independent of the density function ρ, so the problem was cast in the form

$$M(\rho)v = \frac{1}{\lambda}Av.$$

Since only matrix-vector products are required of M the dependence on ρ presented no additional computational burden. The matrix A was factored once and this factorization was subsequently used repeatedly to compute $A^{-1}M(\rho)v$ for all ρ. The eigenvalue iteration also benefited from the reuse of the converged starting vector from the previous problem but this did not appear to be of great consequence in this case. Table 7.3 gives results for the same subproblem on a variety of machines.

TABLE 7.3
Membrane problem on various machines.

	Sun	Convex	Titan	Y-MP
Time (secs)	240	81	40.9	5.4
Matrix-vector	40	40	40	40
$\|V^TW - I\|$	10^{-14}	10^{-14}	10^{-14}	10^{-11}

The overall performance was excellent on this problem. Grid sizes of 64 by 64, 100 by 100, and 200 by 200 were used. Both minimization of $\lambda_1(\rho)$ and $\lambda_2(\rho)$ were done.

The number of matrix-vector products was typically around 32–40 regardless of the dimension of the matrix. That is, with $k = 8$ and $p = 8$ the eigenvalue solver required three to four iterations, with three being the usual number. The Ritz estimates for $\|Ax - M(\rho)x\lambda\|$ were on the order of $10D - 14$ for the lowest six eigenvalues.

The second application leads to a nonsymmetric eigenvalue problem. The PDE arises in a study of bifurcations in a Couette–Taylor wavy vortex instability calculation. This work, described in [13], is based upon a method of Edwards and Tuckerman that is designed to study these bifurcations from Taylor vortices to wavy vortices. The discrete problem is obtained by first linearizing the Navier–Stokes equations about a (numerically) known steady state solution U corresponding to Taylor vortices. The perturbation u corresponding to wavy vortices is found by solving the linearized Navier–Stokes problem

$$\frac{\partial u}{\partial t} = -(U \cdot \nabla)u - (u \cdot \nabla)U - \nabla p + \nu \nabla^2 u$$

with

$$\nabla \cdot u = 0 \quad \text{and} \quad u|_{\partial\Omega} = 0,$$

where Ω is the annular region between two concentric rotating cylinders. This PDE is discretized to then yield a nonsymmetric eigenvalue problem

$$A(\nu)v = \lambda v.$$

Since a pseudospectral method is used, the discrete matrix is dense rather than sparse. However, matrix-vector products can still be performed rapidly using Fourier transforms. The discrete problem involved a matrix of order 2380. The eigenvalue code with $k = 16$ and $p = 40$ required 60 iterations to produce eight eigenvalues and corresponding eigenvectors with largest real part. This entailed about 2400 matrix-vector products. The accuracy of these were confirmed to be at least five significant digits.

This behavior of the algorithm on these two problems seems to be typical of more difficult problems. The number of matrix-vector products tends to be near n for difficult nonsymmetric problems. Symmetric generalized eigenvalue problems from finite element analysis of structures or membranes seem to be solved very rapidly if posed in terms of finding the largest eigenvalues.

To close this section, the interesting behavior of filtering polynomials associated with the choice of exact shifts will be presented. Two problems will be discussed. The first example arises from the convection diffusion above with $\rho = 40$. The grid size was $1/30$ leading to a nonsymmetric matrix of order 900. The results for this problem are displayed in Figs. 7.1 and 7.2. The second example is the banded Toeplitz matrix used for test purposes by Grcar [17]. This matrix is nonnormal and has a nontrivial pseudospectrum, as discussed in [21]. (The ϵ pseudospectrum of a matrix A is $\{\lambda \in \mathbf{C} : \|(\lambda I - A)^{-1}\| \geq \epsilon^{-1}\}$.) The matrix is a five-diagonal matrix with the value -1 on the first subdiagonal and the value 1 on the main diagonal and the next three superdiagonals. The results for this problem are displayed in Figs. 7.3 and 7.4.

The graphs shown below depict the filter polynomial $\psi(\lambda)$ for values of λ over a region containing the eigenvalues of A. The surface plot is of $|\psi|$ and the contour plots are of $\log(|\psi|)$. The $+$ symbols show the location of the true eigenvalues of A. The o symbols mark the location of the eigenvalues of H that are "wanted." These will eventually converge to eigenvalues of A. The $*$ symbols show the roots of the polynomial ψ.

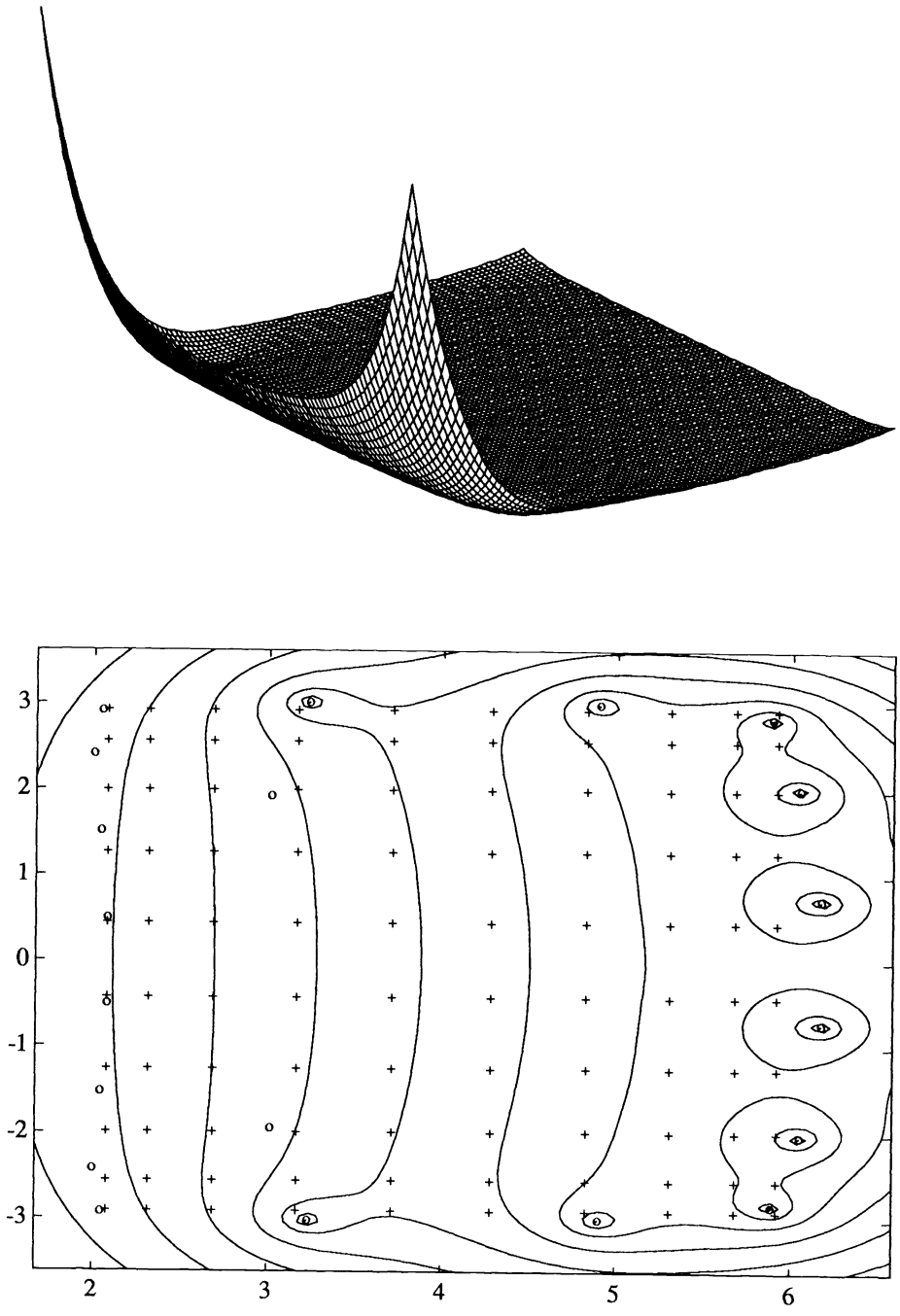


FIG. 7.1. *Convection diffusion: Iteration* 1.

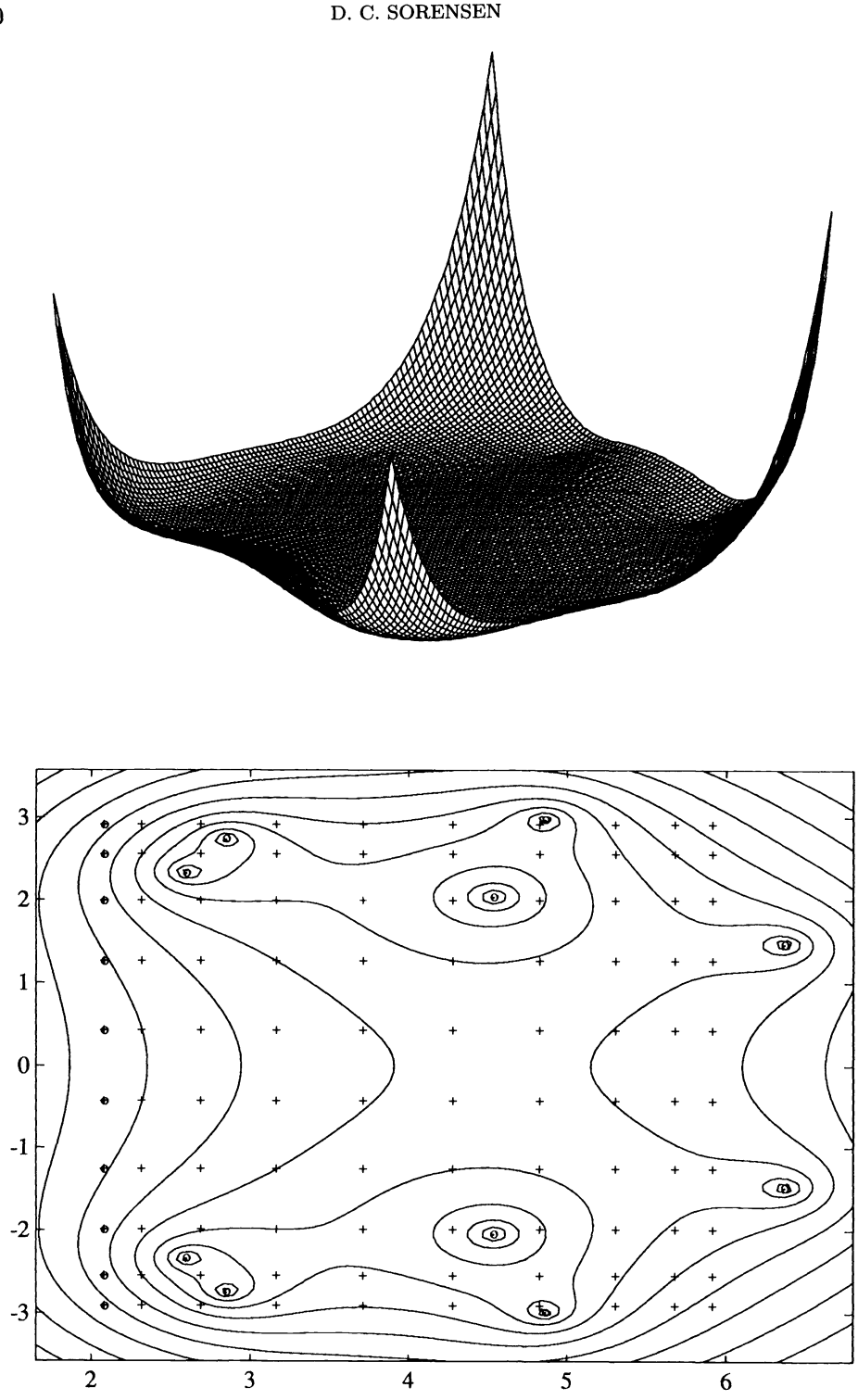


FIG. 7.2. *Convection diffusion: At convergence.*

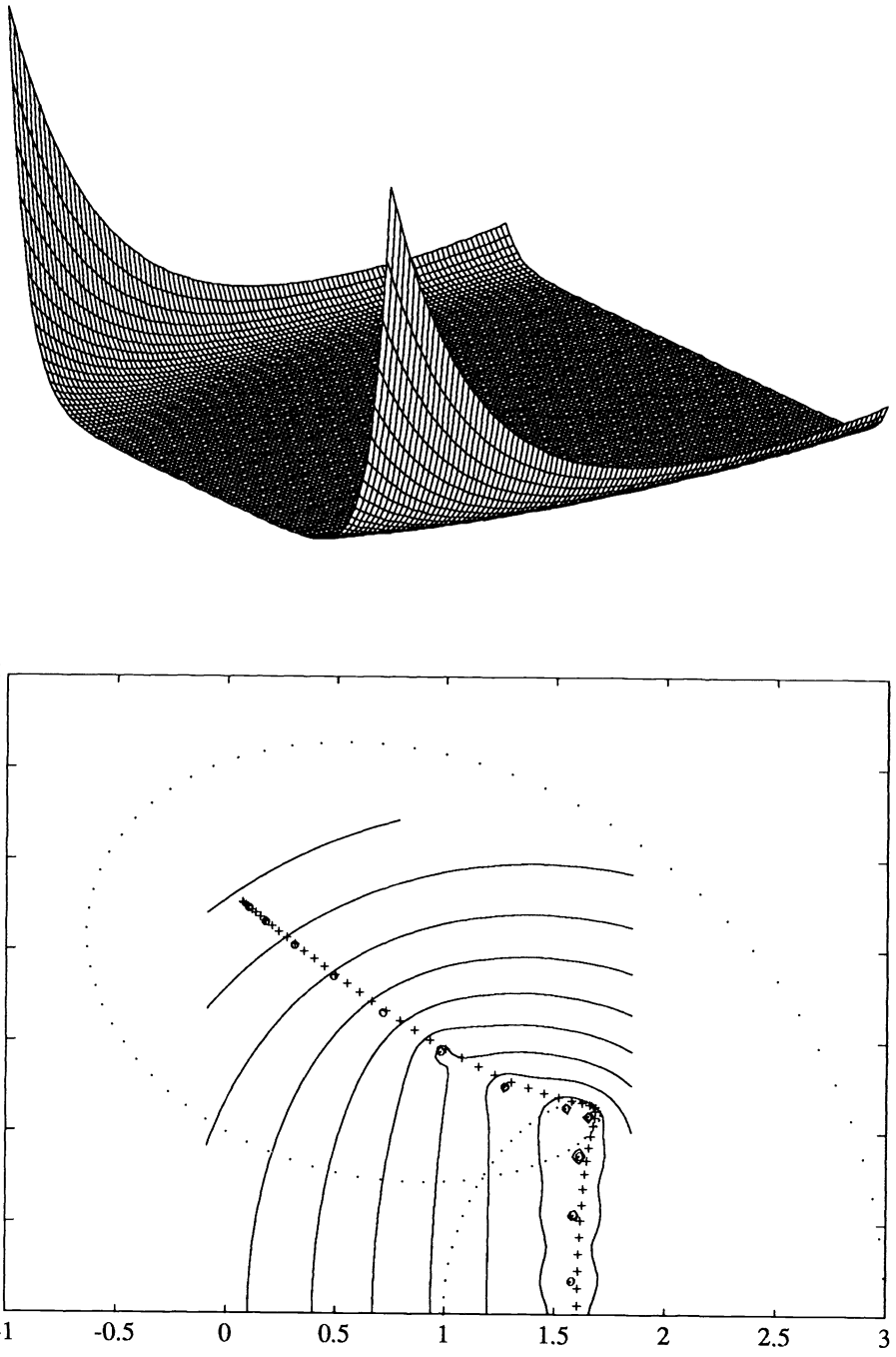
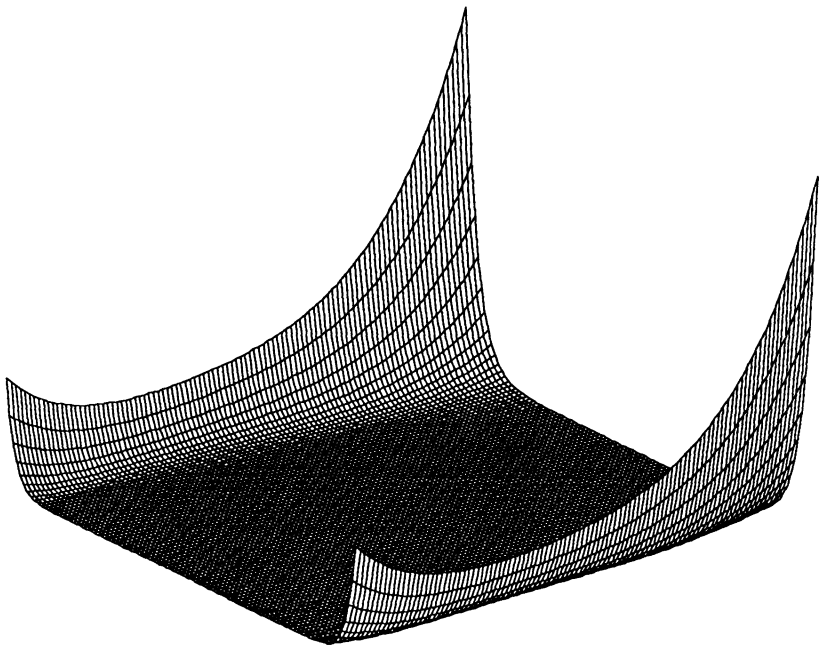


FIG. 7.3. *Grcar matrix: Iteration* 1.



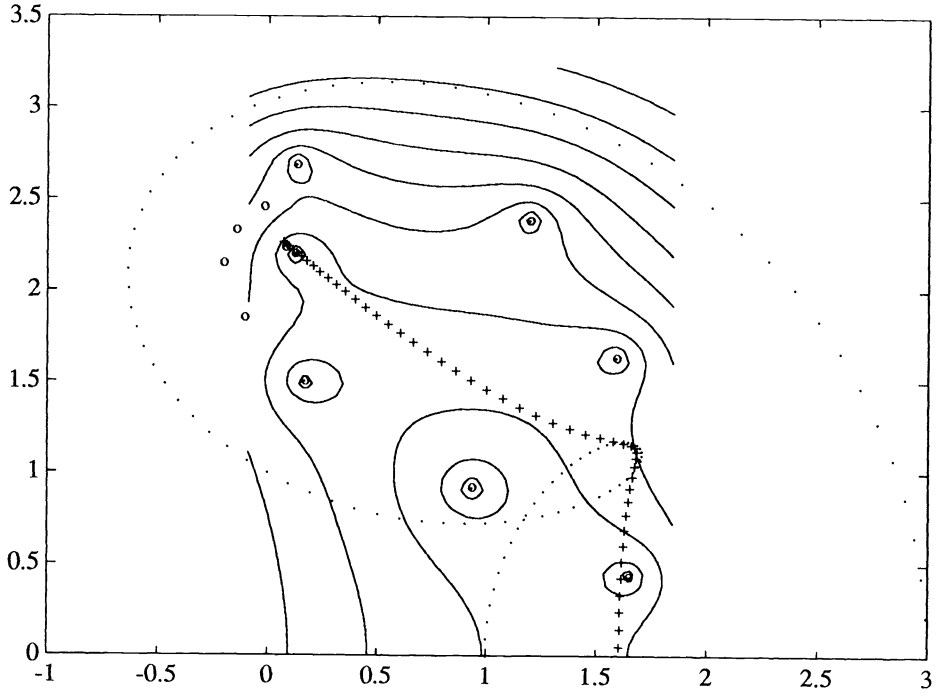


FIG. 7.4. *Grcar matrix: At convergence.*

In Figs. 7.1 and 7.2 the values $k = 10$, $p = 10$ were used. One may observe convergence by looking at the 10 leftmost o symbols enclosing the + symbols. The interesting features of these filter polynomials is that they are remarkably well behaved in terms of being very flat in the region that is to be damped and very steep outside that region. The reason for this desirable behavior is not completely understood at the moment.

In Figs. 7.3 and 7.4 the corresponding behavior of the filter polynomials is shown. In these figures only the upper half-plane is shown. The dotted line shows the boundary of the practical spectrum [21] for this matrix. It is interesting to note how the contours of the filter polynomial obtained through the exact shifts mimic the shape of this boundary. The algorithm claimed convergence of the leftmost eigenvalues (i.e., the ten eigenvalues of smallest real part). However, as demonstrated in the figure, these are pseudoeigenvalues. Interestingly enough, HQR from Eispack will give the same behavior if applied to the transpose of the Grcar matrix. HQR will give the correct eigenvalues when applied to the Grcar matrix directly and it was used to calculate the values of the "true" spectrum shown above.

In conclusion, it seems that this is quite a promising approach. A direct relationship to the implicitly shifted QR-iteration has been established and several problems inherent to the traditional Arnoldi method have been addressed through this new approach. The most important of these are the fixed storage, maintenance of orthogonality, and avoidance of spurious eigenvalues. The computational results are clearly preliminary. The limited experience indicates that research is needed in constructing filter polynomials that have better properties with respect to the wanted part of the spectrum. Moreover, a better understanding of the Ritz convergence estimates in the nonsymmetric case would be helpful. These estimates have been very important in terminating the iteration early (i.e., before the residual is very small) in the symmetric (generalized) eigenproblem. A criterion for choosing the values of k and p is also required. At present, ad hoc choices are made and there is little understanding of the relation of these two parameters to each other and to the given problem. They have been chosen through experimentation for these results.

Future research on this topic might include a blocked variant to better deal with multiple eigenvalues. Investigations of the use of a preconditioner would also be interesting. Finally, extensions of this idea to other settings, such as the solution of linear systems, would seem to be a promising area of research as well. These investigations are under way and will be the topic of subsequent papers.

Acknowledgments. I would like to acknowledge Dr. Phuong Vu and Cray Research for providing access to CRAY-2 and CRAY Y-MP computers and for help in performing a number of numerical experiments with the computer codes. The deflation strategy described in Remark 3 of §3 arose from a discussion with Dr. Vu. I would also like to thank Dr. S. Parter, Dr. L. Reichel and Dr. P. Vu for reading the manuscript and making some useful comments and corrections.

REFERENCES

[1] W. E. Arnoldi, *The principle of minimized iterations in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.

[2] F. Chatelin and D. Ho, *Arnoldi–Tchebychev procedure for large scale nonsymmetric matrices*, Math. Model. Numer. Anal., 24 (1990), pp. 53–65.

[3] A. T. CHRONOPOULOS, *s-Step Orthomin and* GMRES *implemented on parallel computers*, Report TR 90-15, Department of Computer Science, University of Minnesota, Minneapolis, MN, 1989.

[4] S. COX, *The symmetry, regularity, and computation of a membrane interface*, Department of Mathematics and Science Report, Rice University, Houston, TX, 1990.

[5] J. CULLUM AND W. E. DONATH, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace for large, sparse symmetric matrices*, in Proc. 1974 IEEE Conference on Decision and Control, New York, 1974, pp. 505–509.

[6] J. CULLUM, *The simultaneous computation of a few of the algebraically largest and smallest eigenvalues of a large, symmetric, sparse matrix*, BIT, 18 (1978), pp. 265–275.

[7] J. CULLUM AND R. A. WILOUGHBY, *Computing eigenvalues of very large symmetric matrices—an implementation of a Lanczos algorithm with no reorthogonalization*, J. Comput. Phys., 434 (1981), pp. 329–358.

[8] ———, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations*, Vol. I. *Theory*, Birkhäuser, Boston, 1985.

[9] J. DANIEL, W. B. GRAGG, L. KAUFMAN, AND G. W. STEWART, *Reorthogonalization and stable algorithms for updating the Gram–Schmidt* QR *factorization*, Math. Comp., 30 (1976), pp. 772–795.

[10] J. W. DEMMEL, J. J. DONGARRA, J. DU CROZ, A. GREENBAUM, S. HAMMARLING, AND D. SORENSEN, *A prospectus for the development of a linear algebra library for high performance computers*, Report ANL-MCS-TM-97, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, 1987.

[11] J. J. DONGARRA, J. DU CROZ, S. HAMMARLING, AND R. J. HANSON, *An extended set of fortran basic linear algebra subprograms*, ACM Trans. Math. Software 14 (1988), pp. 1–17.

[12] ———, *Algorithm* 656. *An extended set of fortran basic linear algebra subprograms: Model implementation and test programs*, ACM Trans. Math. Software, 14 (1988), pp. 18-32.

[13] W. S. EDWARDS, S. R. BEANE, AND S. VARMA, *Onset of wavy vortices in the finite-length Couette–Taylor problem*, Phys. Fluids, submitted.

[14] J. G. F. FRANCIS, *The* QR *transformation: A unitary analogue to the* LR *transformation, Parts* I *and* II, Comput. J., 4 (1961), pp. 265–272; 332–345.

[15] G. H. GOLUB, R. UNDERWOOD, AND J. H. WILKINSON, *The Lanczos algorithm for the symmetric* $Ax = \lambda Bx$ *problem*, Report STAN-CS-72-270, Department of Computer Science, Stanford University, Stanford, CA, 1972.

[16] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

[17] J. F. GRCAR, *Operator coefficient methods for linear equations*, Sandia National Laboratory Report SAND89-8691, Livermore, CA, 1989.

[18] W. KARUSH, *An iterative method for finding characteristic vectors of a symmetric matrix*, Pacific J. Math., 1 (1951), pp. 233–248.

[19] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45 (1950), pp. 255–282.

[20] T. A. MANTEUFFEL, *Adaptive procedure for estimating parameters for the nonsymmetric Tchebychev iteration*, Numer. Math., 31 (1978), pp. 183–208.

[21] N. NACHTIGAL, L. REICHEL, AND L. N. TREFETHEN, *A hybrid* GMRES *algorithm for nonsymmetric matrix iterations*, Numerical Analysis Report 90-7, Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, 1990.

[22] C. C. PAIGE, *The computation of eigenvalues and eigenvectors of very large sparse matrices*, Ph.D. thesis, University of London, London, UK, 1971.

[23] ———, *Computational variants of the Lanczos method for the eigenproblem*, J. Inst. Math. Appl., 10 (1972), pp. 373–381.

[24] B. N. PARLETT AND D. S. SCOTT, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 311–328.

[25] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice–Hall, Englewood Cliffs, NJ, 1980.

[26] L. REICHEL AND W. B. GRAGG, *Fortran subroutines for updating the QR decomposition of a matrix*, ACM Trans. Math. Software, to appear.

[27] A. RUHE, *Rational Krylov sequence methods for eigenvalue computation*, Linear Algebra Appl., 58 (1984), pp. 391–405.

[28] Y. SAAD, *Chebyshev acceleration techniques for solving nonsymmetric eigenvalue problems*, Math. Comp., 42 (1984), pp. 567–588.

[29] ———, *Projection methods for solving large sparse eigenvalue problems*, in Matrix Pencil Proceedings, B. Kagstrom and A. Ruhe, eds., Springer-Verlag, Berlin, 1982, pp. 121–144.

[30] Y. SAAD AND M. SCHULTZ, GMRES: *A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[31] D. SCOTT, *Analysis of the symmetric Lanczos algorithm*, Ph.D. thesis, Department of Mathematics, University of California, Berkeley, CA, 1978.
[32] B. T. SMITH, J. M. BOYLE, J. J. DONGARRA, B. S. GARBOW, Y. IKEBE, V. C. KLEMA, AND C. B. MOLER, *Matrix eigensystem routines*—EISPACK *guide*, Lecture Notes in Computer Science, 6, Second Edition, Springer-Verlag, Berlin, 1976.
[33] G. W. STEWART, *Introduction to Matrix Computations*, Academic Press, New York, 1973.
[34] H. F. WALKER, *Implementation of the* GMRES *method using Householder transformations*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 152–163.
[35] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, UK, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 386–401, January 1992

025

EFFICIENT MATRIX MULTIPLICATION ON SIMD COMPUTERS*

P. BJØRSTAD†, F. MANNE†, T. SØREVIK†, AND M. VAJTERŠIC†‡

Dedicated to Gene H. Golub on the occasion of his 60th birthday

Abstract. Efficient algorithms are described for matrix multiplication on SIMD computers. SIMD implementations of Winograd's algorithm are considered in the case where additions are faster than multiplications. Classical kernels and the use of Strassen's algorithm are also considered. Actual performance figures using the MasPar family of SIMD computers are presented and discussed.

Key words. matrix multiplication, Winograd's algorithm, Strassen's algorithm, SIMD computer, parallel computing

AMS(MOS) subject classifications. 15-04, 65-04, 65F05, 65F30, 65F35, 68-04

1. Introduction.

One of the basic computational kernels in many linear algebra codes is the multiplication of two matrices. It has been realized that most problems in computational linear algebra can be expressed in block algorithms and that matrix multiplication is the most important kernel in such a framework. This approach is essential in order to achieve good performance on computer systems having a hierarchical memory organization. Currently, computer technology is strongly directed towards this design, due to the imbalance between the rapid advancement in processor speed relative to the much slower progress towards large, inexpensive, fast memories. This algorithmic development is highlighted by Golub and Van Loan [13], where the formulation of block algorithms is an integrated part of the text. The rapid development within the field of parallel computing and the increased need for cache and multilevel memory systems are indeed well reflected in the changes from the first to the second edition of this excellent text and reference book. Their notation and analysis of the "level-3 fraction" of a given matrix algorithm emphasizes the importance of efficient computational kernels for BLAS-3–type [11] operations.

Today, hierarchical memories provide the same motivation as small memories and secondary storage did in the sixties and seventies. The organization of linear algebra computations in terms of a complete library of block algorithms with a corresponding set of routines operating on individual blocks is more than twenty years old (see, for example, [2]).

Matrix multiplication is a very compute-intensive task, but also rich in computational parallelism, and hence well suited for parallel computation. The problem has a simple structure and well-understood mathematical properties. It is therefore often used as a benchmark for parallel computers. Despite this, the task of writing an efficient implementation of the BLAS-3 kernels for any particular advanced architecture machine is often nontrivial [3]. We will in this paper do a careful study of the matrix multiplication problem on SIMD computers. In order to appreciate the often subtle architectural differences between different computers, one must relate this type

* Received by the editors May 16, 1991; accepted for publication (in revised form) August 2, 1991.
† Institutt for Informatikk, Universitetet i Bergen, Høyteknologisenteret, N-5020 Bergen, Norway.
This work was supported in part by Norwegian Research Council for Science and the Humanities grant 413.89/024 and in part by The Royal Norwegian Council for Scientific and Industrial Research contract IT0228.20484.
‡ This author is on leave from the Slovak Academy of Sciences, Bratislava, Czechoslovakia.
This author's research was supported by The Royal Norwegian Council for Scientific and Industrial Research.

of work to a particular computer. We will use the MasPar MP-1 computer for our actual implementations; a brief description of this computer can be found in §2. Most of the discussion is relevant for other data parallel machines (like the AMT DAP or the CM-2), but actual parameters will of course be different.

In particular, we will consider the relative speed of addition and multiplication as well as the relative speed of arithmetic and communication, in order to find efficient algorithms. We show that nonstandard algorithms like the one proposed by Winograd [25] and the fast method of Strassen [23] can be efficiently implemented on SIMD computers. Winograd's algorithm is attractive in the case where additions are faster than multiplications.

As was observed by Brent [7], the exchange of multiplications with additions can give significant speedup, provided that floating point addition is executed faster than floating point multiplication. This was indeed the case in the late sixties and early seventies, but the difference decreased in the following years. Quite recently, this trend has been partially changed, resulting in new computer systems where, again, additions are less expensive than multiplications.

In the MP-1 computer, each processor is only four bits wide. Arithmetic must then be implemented using four bit "nibbles," and while addition is linear in the number of nibbles, multiplication is quadratic in the number of nibbles of the mantissa. Similarly, the AMT DAP is based on single bit processors while the CM-2 has special hardware for floating point arithmetic. It may also be expected that individual SIMD processors will become more complex in future generations. This will increase the floating point speed and tend to reduce the time difference between addition and multiplication. On the other hand, this difference may also be present in modern high performance microprocessors. An example is the Intel i860 chip [16], where 64 bit additions can be executed at a peak rate of 40 Mflops, while 64 bit multiplications can be performed in parallel, but at a maximum rate of 20 Mflops.

On any distributed memory computer performing matrix computations, important questions include how to map the matrices onto the computer and how to design an efficient data flow between processors. On massively parallel systems this issue is critical. The problem has attracted much interest and a number of systolic arrays have been proposed for the problem (see [17], [19] and their references). Some systolic algorithms impose a very specific design on the hardware that can be used; we focus on algorithms that can be implemented efficiently on general purpose SIMD computers.

After a brief description of the MasPar MP-1 computer, we focus, in §3, on the case with N^2 processors where all matrices are $N \times N$. These algorithms are the computational kernels for various block algorithms needed to handle the multiplication of arbitrary-sized matrices. In §4, we discuss the case where each matrix dimension is of the form kN, for $k = 2, 3, \cdots$. In §5, we briefly discuss some of the questions related to the case where the dimensions are arbitrary.

2. Some basic features of the MasPar MP-1 computer. The MasPar MP-1 system is a massively parallel SIMD computer system. The system consists of a high performance UNIX workstation (FE) and a data parallel unit (DPU). The DPU consist of at least 1024 processor elements (PEs), each with 16Kb of memory and 192 bytes of register space. All processors execute instructions broadcast by an array control unit (ACU) in lockstep, but each processor can disable itself based on logical expressions for conditional execution. It should be noted that the individual processors may operate not only on different data, but also in different memory locations,

thus supporting an indirect addressing mode.

There are three different communication mechanisms available: the Xnet, the router, and the global or-tree.

The PEs are interconnected by a two-dimensional toroidal mesh that also allows for diagonal communication. In the MasPar terminology this is called the Xnet. The Xnet operates in three modes:

— Xnet: time is: startup + #bits * distance,
— XnetP(ipe): time is: startup + #bits + distance,
— XnetC(opy): time is: startup + #bits + distance.

The two last modes are useful for regular, nonlocal communication, but require that the processors between any pair of communicating processors be inactive. Thus for sending over longer distance, XnetP is much faster than basic Xnet. XnetC is similar to XnetP, but it leaves a copy of the transmitted variable on all the intermediate processors. The notation Xnet[k] means that the communication distance is k with respect to the processor mesh.

MasPar also supports arbitrary node to node communication through a three-stage switch called the router. For our purpose and for the current range of available models, the router communication cost is constant, independent of the size of the machine. This means that the router, despite its much higher startup time, becomes more competitive compared with Xnet as the machine scales up in size, for all data movements where the communication distance scales with the size of the machine.

The global or-tree can move data from the individual processors to the ACU. If many processors send data at the same time a global reduction results. We take advantage of this mechanism to find the maximum data value of an array at a very small cost in time.

MasPar currently supports Fortran, including a substantial part of the new F90 standard [21] and C based on Kernighan and Ritchie [18], extended to take advantage of the MasPar architecture.

Floating point is implemented in software. We define the average time of a floating point instruction: $\alpha = \frac{1}{2}(Mult + Add)$. Measured in units of α, the floating point performance of the MP-1 corresponds to a peak speed of 0.0355 Mflops in 64 bit arithmetic per processor, or 290 Mflops for a machine having 8192 processors. The processors can access memory in parallel with the execution of arithmetic or communication instructions. We define the ratio

$$\delta = \frac{Load}{\alpha}, \tag{1}$$

where "Load" is the communication time between local memory and registers to load or store one (64 bit) word. Expressing the relative speed of memory access and floating point arithmetic, we expect $\delta \leq 1$ on balanced systems. Due to the asynchronous nature of this operation on the MP-1, δ varies in the interval $(0.05 - 0.5)$ depending on the algorithm. Also define the ratio

$$\gamma = \frac{Xnet[1]}{\alpha}, \tag{2}$$

expressing the time of nearest neighbor communication relative to the time of an average floating point operation. On the MasPar MP-1, $\gamma \approx 0.2$ and a floating point multiplication takes approximately three times the time for a corresponding floating point addition, all in 64 bit precision.

A more detailed general description of the MasPar MP-1 computer can be found in [4], [9], and [22].

3. Multiplying $N \times N$ matrices on an $N \times N$ processor array. To emphasize the algorithmic structure, we first describe the basic algorithms for the special case of square $N \times N$ matrices that fit exactly on an N^2 processor machine. We assume (as is the case on current machines) that N is a power of two. Later, we discuss the modifications necessary to obtain fast algorithms for larger matrix problems.

3.1. Cannon's data flow for the standard algorithm. The standard definition of matrix multiplication, $C = AB$, as

$$c_{i,j} = \sum_k a_{i,k} b_{k,j} \quad \forall i, j \tag{3}$$

provides an obvious method for the computation. Evaluating each of the N^2 elements requires exactly N multiplications and $N - 1$ additions, a sequential complexity of $2N^3 - N^2$. If N^3 processors are available, the N^3 multiplications may be done in one step and the N^2 sums of N terms in $\log N$ steps. On a local memory machine one must, however, take communication costs into account. On a two-dimensional mesh of processors with nearest neighbor communication only, Gentleman [12] has proved that there does not exist any parallel algorithm with communication complexity of order less than $O(N)$. This result is independent of the number of processors available. Thus unless we use the router communication, the largest number of processors for which we can hope to achieve optimal efficiency is $O(N^2)$.[1]

A data flow scheme for the evaluation of (3) on a two-dimensional mesh of processors where the matrices fit exactly on the processor grid was designed by Cannon [8]. The algorithm is well described in [13], but since it has similarities with the alternative algorithms to be described and since it serves to introduce our notation, we briefly describe it in the next paragraph.

Only one element from each matrix is stored on each processor. In order to keep all the processors busy, we need to assure that each processor has elements from A and B that form a product term (i.e., $a_{i,k}$ and $b_{k,j}$ for some k). In Cannon's scheme this is done by an initial preskewing of the matrices. The A matrix is preskewed by rows, while the B matrix is preskewed by columns, as in the 4×4 example shown in Fig. 1.

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{11} & a_{12} & a_{13} & a_{10} \\ a_{22} & a_{23} & a_{20} & a_{21} \\ a_{33} & a_{30} & a_{31} & a_{32} \end{pmatrix} \begin{pmatrix} b_{00} & b_{11} & b_{22} & b_{33} \\ b_{10} & b_{21} & b_{32} & b_{03} \\ b_{20} & b_{31} & b_{02} & b_{13} \\ b_{30} & b_{01} & b_{12} & b_{23} \end{pmatrix}$$

FIG. 1. *Standard preskewing.*

With this preskewing, a very simple data flow scheme guarantees that each processor gets appropriate pairs of elements in each step. The entire multiplication is described by the algorithm below.

In all our algorithms we use $\leftarrow$ to denote assignment, while $\Leftarrow$ denotes data transmission. All operations are performed on matrix elements. Subscripts that occur in algorithms *do not* represent the indices of the matrix elements, but a processor

[1] Note that the use of pipelined communication on the mesh, like XnetP, has a cost proportional to the distance, but such a small constant (one clock cycle) that it can be used efficiently to simulate models of communication that violate this assumption for all computers in the MP-1 family.

address. All processor addresses are modulo N. We assume the processors, as well as the matrices, to have indices running from $0, \cdots, N-1$.

Standard Matrix Multiplication
on all processors:
Preskew A; Preskew B;
on all processors:
  $c \leftarrow ab$;
for $l = 1, N-1$
  on all processors (i,j):
    $a_{i,j} \Leftarrow a_{i,j+1}$;
    $b_{i,j} \Leftarrow b_{i+1,j}$;
    $c \leftarrow c + ab$;

This algorithm performs all the multiplications needed for (3) and accumulates them in c. The difference from the standard outer product update of C is that the index k in the term $a_{i,k}b_{k,j}$ takes different values (in fact, $k = (i+j+l) \pmod N$) on different processors in each step. Consequently, the updates take place in different order on the elements of C. In this algorithm we keep all the processors busy using only nearest neighbor communication (Xnet[1]).

In the preskewing all elements of A move along rows on the processor grid while B's elements are moving along columns. This is a very regular communication and consequently well suited for the Xnet. We tried two different implementations.

Linear Preskewing
/* A and B are initialized with
element (i,j) on processor (i,j) */
for $k = 1, N-1$
  on processors (i,j) where $i \geq k$:
    $a_{i,j} \Leftarrow a_{i,j+1}$;
  on processors (i,j) where $j \geq k$:
    $b_{i,j} \Leftarrow b_{i+1,j}$;

or alternatively,

Logarithmic Preskewing
/* A and B are initialized with
element (i,j) on processor (i,j) */
for $k = 0, \log N - 1$
  on all processors (i,j) where $i \pmod{2^k}$ is odd:
    $a_{i,j} \Leftarrow a_{i,j+2^k}$;
  on all processors (i,j) where $j \pmod{2^k}$ is odd:
    $b_{i,j} \Leftarrow b_{i+2^k,j}$; .

The total data transmission (words times distance) resulting from these two algorithms is in both cases $N^2(N-1)$, but their execution times on the MasPar MP-1 are different. With fewer iterations we reduce loop overhead and logical tests (with resulting changes in the active processor set), as well as the accumulated startup time for Xnet. Consequently, the logarithmic preskewing should perform better.

The router may also be used to preskew the matrices. The router views the processors as a linear array and each processor must compute the destination address

for its variable. The actual communication can then be viewed as taking place in parallel. We have a total of $2N(N-1)$ 64 bit words that must be moved. The communication rates in Table 1 refer to this and do not consider the distance of communication. If (i,j) is the coordinate of a processor, then $p = N*i+j$ is the router address. The individual i, j, and p are all predefined and available on each processor.

The router preskew then takes the following simple form.

Router Preskewing
/* A and B are initialized with
element (i,j) on processor $p = N*i+j$ */
on all processors p:
  $q \leftarrow p - i$;
on processors where $(j < i)$:
  $q \leftarrow q + N$;
$a_q \Leftarrow a_p$;
$q \leftarrow p - N*j$;
on processors where $(i < j)$:
  $q \leftarrow q + N^2$;
$b_q \Leftarrow b_p$;

While the speed of a preskew based on Xnet depends on the size of the computer, a router[2] preskew does not. Thus increasing the size of the machine makes the router more competitive relative to the Xnet.

TABLE 1
Mwords/s in preskewing.

Machine size	1024	2048	4096	8192
Matrix size N	32	64	64	128
Linear preskew	3.0	4.0	6.0	8.1
Log preskew	5.1	7.2	10.7	15.0
Router preskew	5.0	10.2	20.5	41.1

We present preskewing data for both square and rectangular machines in Table 1. The matrix size N will always be taken equal to the larger of the two sides if the processor mesh is nonsquare. In this case, the matrix is mapped to the processor array by having each processor store two matrix elements. We note that the router bandwidth increases proportionally with the machine size, resulting in a constant time for the preskewing, while the two algorithms using Xnet have a bandwidth increase proportionally to the square root of the machine size. This reflects the fact that the average communication distance grows as the square root of the number of processors. Also note how much faster the logarithmic preskew is compared with the linear; in fact, for the 1024 processor machine, this is the method of choice.

[2] Clearly this is only true for the current range of machines. In general one would expect the time to grow logarithmically with the number of processors since the number of stages in such a switch will increase with the number of processors.

3.2. Data flow for Winograd's algorithm. Winograd [25] proposed the following method for matrix multiplication: Let

$$d_{i,j} = \sum_{k=0}^{N/2-1} (a_{i,2k} + b_{2k+1,j})(a_{i,2k+1} + b_{2k,j}) \tag{4}$$

and

$$a_i^* = \sum_{k=0}^{N/2-1} a_{i,2k} a_{i,2k+1}, \tag{5}$$

$$b_j^* = \sum_{k=0}^{N/2-1} b_{2k+1,j} b_{2k,j}; \tag{6}$$

then the elements of C can be computed as

$$c_{i,j} = d_{i,j} - a_i^* - b_j^* \,. \tag{7}$$

The exact flop count for this algorithm is $2N^3 + 3N^2 - 2N$, which is slightly more than the standard product (3). However, the number of multiplications is one half at the expense of additions. Consequently, on the MP-1, there is a potential maximum speedup of 25 percent using Winograd's algorithm, if we are able to construct an efficient data flow scheme for the algorithm.

The numerical stability of this algorithm was analyzed by Brent [7]. He shows that scaling of the matrices A and B is essential. Define the norm $\|A\| = \max_{i,j} \mid a_{i,j} \mid$. If the crude but easy-to-implement scaling

$$A \leftarrow 2^p A, \qquad B \leftarrow 2^{-p} B, \tag{8}$$

where p is an integer such that

$$\frac{1}{2} \leq 2^{2p} \frac{\|A\|}{\|B\|} < 2 \,, \tag{9}$$

then (7) will compute $AB + E$ with $\|E\|$ bounded by

$$\|E\| \leq \frac{9}{8}(n^2 + 16n)u\|A\| \, \|B\|, \tag{10}$$

and with u being the unit roundoff of the machine. This compares well with the corresponding bound for the standard algorithm

$$\|E\| \leq n^2 u \|A\| \, \|B\| + O(u^2) \,, \tag{11}$$

although a generally stronger, componentwise bound exists for this algorithm [13]. In Table 2, we compare two different scaling algorithms. Both algorithms first find $\|A\|$ and $\|B\|$ in the two matrices. The scaling is then performed as outlined above. We scale by a power of two, implemented either as a shift of the exponent or by a straightforward multiplication. We report the performance in millions of 64 bit words scaled per second. This scaling takes advantage of the global or-tree for finding the maximum elements. We note that exponent shifting is much faster and also avoids extra rounding errors. The drawback is a more machine-dependent implementation.

TABLE 2
Mwords scaled/s in Winograd's algorithm.

Machine size	1024	2048	4096	8192
Matrix size	32	64	64	128
Multiplication	3	9	12	36
Exponent shift	9	22	35	86

The correction terms (5) and (6) are easily computed by a standard log-sum in parallel for all rows and columns. The choice of communication for this operation is XnetP. When found, the correction terms are broadcasted along rows or columns using XnetC. The parallel arithmetic complexity of (5) and (6) is $\log N$. In computing the log-sum there will be $\log N$ startups for the XnetP and a total transmission cost proportional to N. (Remember that on the MP-1, the transmission cost will be dominated by the $\log N$ term for all existing values of N.)

Keeping one element from each matrix on each processor, we need $(a_{i,2k}, b_{2k+1,j})$ on processor (i,j). Next, we need $(a_{i,2k+1}, b_{2k,j})$ followed by $(a_{i,2k+2}, b_{2k+3,j})$ and $(a_{i,2k+3}, b_{2k+2,j})$. This is the same regular data flow as in Cannon's algorithm, except that the elements of B are pairwise interchanged. The corresponding preskewing is shown in Fig. 2.

$$\begin{pmatrix} a_{00} & a_{01} & a_{02} & a_{03} \\ a_{11} & a_{12} & a_{13} & a_{10} \\ a_{22} & a_{23} & a_{20} & a_{21} \\ a_{33} & a_{30} & a_{31} & a_{32} \end{pmatrix} \begin{pmatrix} b_{10} & b_{01} & b_{32} & b_{23} \\ b_{00} & b_{31} & b_{22} & b_{13} \\ b_{30} & b_{21} & b_{12} & b_{03} \\ b_{20} & b_{11} & b_{02} & b_{33} \end{pmatrix}$$

FIG. 2. *Preskewing for Winograd* I.

With this initialization, we are able to obtain all the sums $a_{i,2k+1} + b_{2k,j}$ and $a_{i,2k} + b_{2k,j+1}$ on all processors. The difficulty is, however, that the two sums in (4) do not turn up at the same time on every processor. The processors are divided into two groups in a checkerboard pattern. On the "black" processors the second sum turns up one step later than on the "white." In order to do the multiplication and the update of $d_{i,j}$ simultaneously on all processors, we need to store the first sum in a register on the "white" processors until the second sum has been computed. This results in an extra interchange of the values of two variables. The algorithm can be described as follows.

Winograd I (Single Correction)
on all processors:
  Scale A; Scale B;
/* compute the correction terms */
on all processors:
  $a_i^* \leftarrow \text{logsum}(a_{i,1}a_{i,0} + a_{i,3}a_{i,2} + \cdots + a_{i,N-1}a_{i,N-2})$;
  $b_j^* \leftarrow \text{logsum}(b_{0,j}b_{1,j} + b_{2,j}b_{3,j} + \cdots + b_{N-2,j}b_{N-1,j})$;
  $c \leftarrow -a_i^* - b_j^*$;
/* preskew according to Fig. 2 */
on all processors:
  Preskew A; Preskew B;
/*multiplication phase*/

on all processors:
  $s0 \leftarrow a + b$;
for $l = 1, N/2$
  on all processors:
    $a_{i,j} \Leftarrow a_{i,j+1}$;
    $b_{i,j} \Leftarrow b_{i+1,j}$;
    $s1 \leftarrow a + b$;
    $a_{i,j} \Leftarrow a_{i,j+1}$;
    $b_{i,j} \Leftarrow b_{i+1,j}$;
    $s2 \leftarrow a + b$;
  on processors (i, j) where $(i + j)$ is even:
    $tmp \leftarrow s0$; $s0 \leftarrow s2$; $s2 \leftarrow tmp$;
  on all processors:
    $c \leftarrow c + s1 * s2$;

By unrolling the loop one level, the three assignments needed for swapping can be replaced by one.

Another strategy that allows us to group together pairs of elements of A and B where the index k differs by 1, is based on making copies of A and B that are shifted one column or row, respectively. If we keep the same simple data flow, but cyclically send one copy into the other, always sending the one which is a step ahead, we manage to move both copies two positions in only two steps, instead of four. However, we are now computing

$$d_{i,j} = \sum_{k=1}^{N/2} (a_{i,2k} + b_{2k-1,j})(a_{i,2k-1} + b_{2k,j}) \tag{12}$$

on half the processors. Note that the indices in (12) must be taken modulo N. Being different from (4), we need different correction terms on these processors. Using the two versions simultaneously, we now compute two sets of correction terms. For each processor we use the term that corresponds to the $d_{i,j}$ that is computed. The algorithm can be stated as follows.

Winograd II (Double Correction)
/* Scale the matrices as in (8) and (9) */
on all processors:
  Scale A; Scale B;
/* compute the correction terms */
on all processors:
  $a_i^* \leftarrow \text{logsum}(a_{i,1}a_{i,0} + a_{i,3}a_{i,2} + \cdots + a_{i,N-1}a_{i,N-2})$;
  $aa_i^* \leftarrow \text{logsum}(a_{i,0}a_{i,N-1} + a_{i,2}a_{i,1} + \cdots + a_{i,N-2}a_{i,N-3})$;
  $b_j^* \leftarrow \text{logsum}(b_{0,j}b_{1,j} + b_{2,j}b_{3,j} + \cdots + b_{N-2,j}b_{N-1,j})$;
  $bb_j^* \leftarrow \text{logsum}(b_{N-1,j}b_{0,j} + b_{1,j}b_{2,j} + \cdots + b_{N-3,j}b_{N-2,j})$;
on all processors where $(i + j)$ even:
  $c \leftarrow -a_i^* - b_j^*$;
on all processors where $(i + j)$ odd:
  $c \leftarrow -aa_i^* - bb_j^*$;
/* preskew as for Cannon's algorithm */
on all processors:
  Preskew A; Preskew B;

TABLE 3
SIMD *matrix multiplication kernels.*

Machine size	1024		2048		4096		8192	
Matrix size	32		64		64		128	
	Mflops	OH	Mflops	OH	Mflops	OH	Mflops	OH
Cannon	24	16%	54	8%	103	8%	226	4%
Winograd I	21	38%	47	24%	102	25%	220	13%
Winograd II	20	43%	57	27%	100	29%	258	16%

/* multiplication phase */
on all processors:
$\tilde{a}_{i,j} \Leftarrow a_{i,j+1}$;
$\tilde{b}_{i,j} \Leftarrow b_{i+1,j}$;
$c \leftarrow c + (a + \tilde{b}) * (\tilde{a} + b)$;
for $k = 1, N/2 - 1$
on all processors:
$a_{i,j} \Leftarrow \tilde{a}_{i,j+1}$;
$b_{i,j} \Leftarrow \tilde{b}_{i+1,j}$;
$\tilde{a}_{i,j} \Leftarrow a_{i,j+1}$;
$\tilde{b}_{i,j} \Leftarrow b_{i+1,j}$;
$c \leftarrow c + (a + \tilde{b}) * (\tilde{a} + b)$;

While computing the double set of log-sums we always have enough processors to do the arithmetic in parallel for the sums. However, when using XnetP for the communication all intermediate processors must be idle. The communication time for computing the correction terms is doubled, while the arithmetic has the same time complexity as in the single correction case.

3.3. Timing results. We have carefully timed the different routines. The results are presented in Tables 3 and 4. The Mflops[3] are based on flop counts for the standard method (3). We compare the three algorithms in Table 3, where all calculations are performed in 64 bit precision. We present Mflops figures and the percentage of the total time spent in "OverHead." The column labeled "OH" covers preskewing and, in the case of Winograd, scaling and computation of the correction terms.

The results require a few comments. When we compare the two variants of Winograd as stated in this paper, it seems that Winograd I should be slightly superior in terms of complexity. This advantage can be seen on the square machines (1024,4096). On the nonsquare machines, we need to store two matrix elements on each processor. This leads to a reduction from 4 to 3 in the nearest neighbor communication in the inner loop, but doubles the register requirements. In Winograd I there is the additional need to unroll the inner loop one level. The resulting code requires more registers than currently available. Winograd II is therefore considerably faster on the nonsquare machines. As predicted by the analysis, the overhead of all three algorithms is reduced as N increases. Cannon's algorithm is competitive on the smaller machines due to its lower overhead, but on the 8192 processor machine (and on larger

[3] Since Winograd's algorithm needs some additional operations for doing the correction terms ($O(n^2)$), the correct flop counts are actually somewhat higher here. But a fair comparison from a practical point of view requires the same flop counts for both algorithms. All Mflops figures in this paper refer to the standard method (3).

TABLE 4
Dependence on floating point format.

Precision	64 bit		32 bit	
Algorithm	Mflops	OH	Mflops	OH
Cannon	226	4%	461	7%
Winograd I	220	13%	384	16%
Winograd II	258	16%	452	21%

machines) we note that the 25 percent saving in arithmetic puts Winograd ahead in performance.

Since the relative speed of multiplication and addition depends on the length of the mantissa, we give results for both 32 bit and 64 bit precision floating point formats in Table 4. These formats have 23 and 52 bits in the mantissa, respectively. We observe that Cannon more than doubles in performance, while the speedup is about 75 percent for Winograd, consistent with the relative importance of multiplications in the two algorithms.

4. Block algorithms. In this section we discuss how to multiply matrices having more elements than the number of processors available. Again, assuming N^2 processors, we first deal with square matrices of size $n = kN$, where $k = 2, 3, \cdots$.

There are two common ways to partition the matrix. One can either divide it into k^2 blocks, each of size $N \times N$, and distribute one element of each submatrix to the corresponding processor. Alternatively, one can split the matrix into N^2 $k \times k$ blocks and distribute each block to an individual processor.

In the first case, one can simply do the matrix multiplication by a block version of the standard algorithm. This requires k^3 calls to a routine for doing the matrix multiplication of $N \times N$ matrices. The preskewing can be done once for each block, giving a total of k^2 calls to the preskewing routine. Similarly, for the Winograd kernel, both the scaling and the correction terms can be computed directly on the global matrix. This improves the parallel complexity to $O(k^2 + k \log N)$ for the correction terms and to $O(k^2)$ for the scaling. Thus, asymptotically, the arithmetic of the kernel loop will dominate the entire computation. This approach gives the same ratio between communication and arithmetic as for the $N \times N$ case considered in §3. In Table 5, we present data for this scheme with $N = 128$ and 8192 processors.

In the second case, it is straightforward to do a block version of Cannon's algorithm. In this case we get a block preskewing. In N steps each processor will do a matrix multiplication of $k \times k$ blocks and send the two blocks to its neighbors. Now we have $O(k^3 N)$ arithmetic operations, but only $O(k^2 N)$ communication. This advantage may, however, be offset by the more frequent access to memory of order $O(k^3 N)$. Using the relations (1) and (2) we obtain the inequality

$$(\gamma - \delta)k \geq \gamma + \hat{\delta}, \tag{13}$$

which must hold if this algorithm shall be faster than the first one considered. Here $\hat{\delta}$ corresponds to the memory access speed for the first blocking strategy. The relation shows that local memory access must be faster than nearest neighbor communication for the second blocking strategy to give a faster method. This is only true on the MP-1 if overlap between register loads and arithmetic can be achieved. The global $N \times N$ memory access (fetching one number to each processor) cannot easily be overlapped, and $\hat{\delta} \approx .5$ while the reading of local $k \times k$ blocks facilitates a δ of approximately .08

TABLE 5
Performance of block algorithms with $N \times N$ kernel blocks based on kernels from Table 3.

	64 bit precision				32 bit precision			
	Block Cannon		Block Winograd		Block Cannon		Block Winograd	
n	Time	Mflops	Time	Mflops	Time	Mflops	Time	Mflops
256	0.15	230	0.12	272	0.07	484	0.07	490
512	1.15	233	0.92	291	0.54	495	0.50	531
1024	9.14	235	7.14	301	4.30	500	3.89	551
2048	72.82	236	56.50	304	34.12	503	30.47	564

in our case. We conclude that for sufficiently large matrices, the second strategy will be most efficient. We employ Cannon's data flow, but have a choice between standard matrix multiplication at the block level or the use of Winograd's method.[4] We note that preskewing, scaling, and correction terms can be performed in the same way as above.

Finally, note that Winograd's algorithm cannot be applied to matrix blocks since (4)–(7) depends on commutativity. In addition to the two alternatives already mentioned, there is obviously a "virtual processor" Winograd method, where the data for each virtual processor is grouped locally and assigned to physical processors. The complexity of this method is similar to the first block method considered in this section, but the programming is more complex. In addition, this approach suffers from a more expensive and complicated preskewing.

4.1. Strassen's algorithm on an SIMD machine. Strassen first presented his algorithm for matrix multiplication in [23]. It is based on a recursive divide and conquer scheme. The algorithm is clearly presented in Chapter 1 of [13]. It is well known that the algorithm has a sequential complexity of $O(n^{2.807})$, as compared to $O(n^3)$ for ordinary matrix multiplication. Because of lower-order terms it is advisable to employ an ordinary matrix multiplication routine when the dimension of the blocks reaches some preset cutoff point $n_0 \geq 8$ [14]. Due to algorithm overhead that grows with the number of recursions, as well as efficient use of the hardware at hand, we chose to take $n_0 = 128$ for our 8192 processor machine. We can then use one of the computational kernels described in the previous section with $N = 128$.

Lately, there has been a renewed interest in Strassen's algorithm. Bailey [1] implemented it on a CRAY-2 and reported speedups up to 2.01 for $n = 2048$. The numerical properties of this algorithm are analyzed in [6] and more recently in [14]; see also Golub and Van Loan [13] for a discussion of problems where Strassen's algorithm should not be used. The algorithm satisfies the following error bound:

$$\|E\| \leq \left(\left(\frac{n}{n_0} \right)^{\log_2 12} (n_0^2 + 5n_0) - 5n \right) u \|A\| \, \|B\| + O(u^2) \,, \tag{14}$$

where $n_0 \leq n$ is the cutoff point mentioned above ($\log_2 12 \approx 3.6$). This should be compared with standard multiplication (11) and with Winograd's algorithm (10). The error bound is somewhat weaker, but may still be regarded as acceptable unless small, componentwise relative errors are required. Empirical results from both Bailey [1] and Higham [15] show that the error in Strassen is small enough to justify its use in applications where speed is crucial. Also note that our choice of $n_0 = 128$ improves

[4] In this case k should be even, or the code must simulate the algorithm for $k + 1$.

the bound for realistic values of n, compared to having a very small value. Both IBM and CRAY support routines for fast matrix multiplications using Strassen's algorithm.

In this section we restrict k to be a power of 2 (i.e., $k = 2^l$ $l = 1, 2, \cdots$) and we partition the matrix into k^2 blocks of size $N \times N$. With this layout of the matrix all additions and subtractions can be performed in parallel without any communication between the PEs. At each step of the algorithm, each processor views its data as being a local $n \times n$ matrix on which it is performing Strassen's algorithm. Once the cutoff point is reached, each processor will have one element that fits into one of the standard kernel matrix multiplication algorithms described earlier. Both Cannon's and Winograd's algorithms were tried as the kernel to perform the matrix multiplications. Note that in both cases we can perform the preskewing of the k^2 blocks in a preprocessing step. Also, the scaling step in Winograd's algorithm can be performed as part of the preprocessing. This reduces the "Overhead" in Table 3 significantly. The use of Winograd as a computational kernel in Strassen's algorithm also slightly changes the error bound (14) to

$$\|E\| \leq \left(\left(\frac{n}{n_0} \right)^{\log_2 12} \left(\frac{9}{8} n_0^2 + 23 n_0 \right) - 5n \right) u \|A\| \, \|B\| + O(u^2) \,. \tag{15}$$

Strassen's algorithm will have $l = \log(k)$ levels of recursion and require approximately $k^{2.8}$ (kernel) matrix multiplications each of size $N \times N$. Note that each processor therefore will do $2k^{2.8}N$ nearest neighbor communications compared with only $2k^2N$ for the block methods. Asymptotically, Strassen will always win due to the lower exponent in arithmetic complexity, but for practical problems we obtain the inequality

$$(1 + \gamma + \hat{\delta}/N)k^{2.8} \leq (1 + \delta)k^3 + (\gamma + \delta + \hat{\delta})k^2 \tag{16}$$

for values of k where Strassen's method will outperform the asymptotically best block algorithm. Here $\hat{\delta}$ again refers to memory access that cannot easily be overlapped with arithmetic. The last term on the right-hand side comes from the memory access when sending the blocks to neighbor processors. On the MP-1, this inequality is always satisfied. If we neglect the $\hat{\delta}/N$ term and the k^2 terms, then (16) simplifies to

$$k \geq \left(\frac{1 + \gamma}{1 + \delta} \right)^5 . \tag{17}$$

The value of k is therefore quite sensitive to an increase in γ. For example, if we assume that $\delta \ll 1$ and take the quite reasonable value of $\gamma = 1$, then $k \geq 32$, corresponding to five levels of recursion in Strassen. This implies that the matrix must be at least of dimension 4096, requiring more than 400 Mbytes of memory, perhaps exceeding the size of the machine.

The algorithm was tried on matrices of size kN, $N = 128$, $k = 2^l$, $l = 1, 2, 3, 4$. Tables 6 and 7 give the timings of the different cases. Comparing Table 5 with the left parts of Tables 6 and 7, we find, in agreement with the discussion, that the partitioning into $N \times N$ blocks is best for smaller matrices. The crossover point is around $n = 2048$, slightly higher than predicted. In 32 bit precision δ increases and the same effect is even more pronounced. As predicted by (16), Strassen's algorithm is faster than the block methods for any levels of recursion on the MP-1. We note that our block Winograd code is faster than the one processor CRAY-2 figures using

TABLE 6
Performance of block algorithms in 64 bit precision.

n	Block Cannon Time	Block Cannon Mflops	Block Winograd Time	Block Winograd Mflops	Strassen–Cannon Time	Strassen–Cannon Mflops	Strassen–Winograd Time	Strassen–Winograd Mflops
256	0.21	163	0.20	170	0.13	256	0.11	294
512	1.35	199	1.17	230	0.91	295	0.79	341
1024	9.62	223	7.82	275	6.34	339	5.47	392
2048	72.44	237	56.87	302	44.42	387	38.14	450

TABLE 7
Performance of block algorithms in 32 bit precision.

n	Block Cannon Time	Block Cannon Mflops	Block Winograd Time	Block Winograd Mflops	Strassen–Cannon Time	Strassen–Cannon Mflops	Strassen–Winograd Time	Strassen–Winograd Mflops
256	0.11	315	0.11	302	0.06	538	0.06	530
512	0.68	394	0.66	409	0.43	624	0.43	619
1024	4.80	447	4.40	488	3.00	716	3.01	714
2048	36.00	477	31.85	539	21.02	817	21.01	817

the CRAY MXM library, reported by Bailey [1]. Our results for Strassen's method are also quite comparable with his.

Another similar algorithm, due to Winograd [5], which uses only 15 additions and subtractions (as compared to 18 by Strassen), was also implemented. There was no significant improvement in execution time, since the block multiplication completely dominates the small saving in arithmetic.

We note that Manber [20] claims that Strassen's algorithm cannot be easily parallelized. Our results show a practical, parallel version, but depends on a very favorable, low value of the parameter γ.

5. Matrices of arbitrary size. Suppose we have two $n \times n$ matrices and N^2 processors. If n/N is an integer we can use any of the algorithms defined in §§3 or 4. If n/N is not an integer we may divide the matrices into $k \times k$ blocks, $k = \lceil n/N \rceil$, and place the K^2 blocks, $K = \lceil n/k \rceil$, in the upper left $K \times K$ part of processor array. With this mapping of the data there are at least two simple modifications of the standard algorithms from §4.

We may extend the matrix with zero blocks and run the algorithm as before. Considering only the multiplication part, the parallel complexity of this algorithm will be

$$2k^2 N\alpha(k+\gamma), \tag{18}$$

where α and γ are defined in (2). Alternatively, we only use the $K \times K$ processors. In this case the boundaries must be handled using two extra XnetP$[N-K]$ in the inner loop. For comparison we get:

$$2k^2 K\alpha(k+\gamma+\hat{\gamma}), \tag{19}$$

where $\hat{\gamma}$ is defined like γ, but using the time of XnetP$[N-K]$ instead of Xnet[1]. For the MP-1 one can assume that $\gamma < \hat{\gamma} < 2\gamma$ for interesting values of K. This shows that the last approach should be used if

$$K < \frac{k+\gamma}{k+2\gamma} N\,. \tag{20}$$

With $\gamma \approx 1/5$, this will almost always be the best choice.

Consider now the case where the matrix blocks in our partition are nonsquare. This is necessary when the matrices (or the processor array) are nonsquare. In Cannon's scheme the elements of two matrices move in every step. Cannon chose A and B to move, while the elements of C remain in place. We may as well move B and C or A and C. For the previously described preskewing, these alternative data flows force one of the matrices to move along diagonals. While the arithmetic work is independent of the data flow, the communication time is not. Assuming that the matrices are of different shape and partitioned as above, we will minimize the communication effort by always sending the two matrices with the smallest block sizes. This possibility is available when the interconnection network supports diagonal communication, as on the MP-1.

Finally, let us consider the case where the number of matrix elements is less then the number of processors. Alternative algorithms based on making copies of A and B to all processors exist. If this is done properly, up to n^3 processors can participate in the multiplication phase. Finally, the summation of all n^2 inner products must take at least $\log n$ steps. However, as proved by Gentleman [12], the communication complexity is still $O(n)$ for a two-dimensional mesh with nearest neighbor communication only. Typically, we want a binary tree network to support this kind of algorithm [10]. On the MP-1, one can do a rather efficient simulation of binary trees using XnetP. In particular, one can design efficient algorithms for matrices of dimension $n = 2^l$, $n < N$. Vajteršic has described such algorithms for the MP-1 in [24].

6. Conclusions. We have developed and analyzed data flow algorithms for Winograd's and Strassen's matrix multiplication algorithms and shown that they can be efficiently implemented on a state of the art massively parallel SIMD computer. The algorithms perform close to the theoretical maximum of the machine and provide a very cost-effective way of doing large scale matrix computations. Our algorithms can also be implemented on alternative SIMD machines like the AMT DAP and the CM-2. In order to predict the performance on these machines the parameters α, δ, and γ must be determined and major architectural differences (e.g., router performance and XnetP-type communication) must be taken into account. We note, in particular, that Strassen's algorithm depends on a very favorable communication speed γ. There will be a considerable challenge to maintain this property in future data parallel computing systems.

Acknowledgment. We thank Dr. Robert Schreiber for suggesting the data flow scheme employed in the second Winograd algorithm and for stimulating discussions. Also, thanks to Ken Jacobsen at MasPar for providing us with technical data and Erik Boman for coding the fast scaling algorithm used to stabilize the Winograd algorithm.

REFERENCES

[1] D. H. Bailey, *Extra high speed matrix multiplication on the Cray-2*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 603–607.

[2] K. Bell, B. Hatlestad, O. E. Hansteen, and P. O. Araldsen, NORSAM, *a programming system for the finite element method. User's manual, Part 1, General description*, The Technical University of Norway, Trondheim, 1973.

[3] C. H. Bishof, *Fundamental linear algebra computations on high-performance computers*, Tech. Report, Argonne National Laboratory, Argonne, IL, August 1990.

[4] T. Blank, *The MasPar* MP-1 *architecture*, in Proc. IEEE Compcon, Spring 1990, February 1990.

[5] G. BRASSARD AND P. BRATLEY, *Algorithms: Theory and Practice*, Prentice–Hall, Englewood Cliffs, NJ, 1988.

[6] R. P. BRENT, *Algorithms for matrix multiplication*, Tech. Report CS 157, Computer Science Department, Stanford University, Stanford, CA, 1970.

[7] ———, *Error analysis of algorithms for matrix multiplication and triangular decomposition using Winograd's identity*, Numer. Math., 16 (1970), pp. 145–156.

[8] L. E. CANNON, *A cellular computer to implement the Kalman filter algorithm*, Ph.D. thesis, Montana State University, Bozeman, MT, 1969.

[9] P. CHRISTY, *Software to support massively parallel computing on the MasPar* MP-1, in Proc. IEEE Compcon, Spring 1990.

[10] E. DEKEL, D. NASSIMI, AND S. SAHNI, *Parallel matrix and graphs algorithms*, SIAM J. Comput., 10 (1981), pp. 657–675.

[11] J. DONGARRA, J. DU CROZ, I. DUFF, AND S. HAMMARLING, *A set of level* 3 *basic linear algebra subprograms: Model implementation and test programs*, ACM Trans. Math. Software, 16 (1990), pp. 18–28.

[12] W. M. GENTLEMAN, *Some complexity results for matrix computations on parallel processors*, J. Assoc. Comput. Mach., 25 (1978), pp. 112–115.

[13] G. H. GOLUB AND C. F.VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[14] N. J. HIGHAM, *Exploiting fast matrix multiplication within the level* 3 BLAS, ACM Trans. Math. Software, 16 (1990), pp. 352–368.

[15] ———, *Stability of a method for multiplying complex matrices with three real matrix multiplications*, Tech. Report no. 181, Department of Mathematics, University of Manchester, Manchester, England, January 1990.

[16] INTEL COMPUTER CORPORATION, i860 64-*bit Microprocessor Programmer's Reference Manual*, 1990.

[17] H. J. JAGADISH AND T. KAILATH, *A family of new efficient arrays for matrix multiplication*, IEEE Trans. Comput., 38 (1989), pp. 149–155.

[18] B. W. KERNIGHAN AND D. M. RITCHIE, *The C programming language*, Prentice–Hall, Englewood Cliffs, NJ, 1978.

[19] H. T. KUNG, *Why systolic architecture?*, Comput. J., 15 (1982), pp. 37–46.

[20] U. MANBER, *Introduction to Algorithms*, Addison–Wesley, Reading, MA, 1989.

[21] M. METCALF AND J. REID, *Fortran* 90 *Explained*, Oxford Science Publications, Reading, MA, 1990.

[22] J. NICKOLLS, *The design of the MasPar* MP-1, *a cost effective massively parallel computer*, in Proc. IEEE Compcon, Spring 1990.

[23] V. STRASSEN, *Gaussian elimination is not optimal*, Numer. Math., 13 (1969), pp. 354–356.

[24] M. VAJTERŠIC, *Matrix multiplication algorithms for matrices of the size* $n \leq 128$ *on the MasPar parallel computer*, Tech. Report, Institutt for Informatikk, Universitetet i Bergen, Bergen, Norway, August 1990.

[25] S. WINOGRAD, *A new algorithm for inner product*, IEEE Trans. Comput., C-18 (1968), pp. 693–694.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 1, pp. 402–417, January 1992

026

GAUSS QUADRATURES: AN INVERSE PROBLEM*

JAROSLAV KAUTSKY†

Dedicated to Gene Golub on the occasion of his 60th birthday.

Abstract. The problem of finding the Gauss quadrature (i.e., the quadrature formula of the maximal polynomial order) for a given weight function is reversed: a weight function, for which a given quadrature formula with positive weights is the Gauss quadrature, is sought. Among all such weight functions, those minimizing, in the least square sense, the kth derivative ($k > 0$ given) are characterized. An algorithm for calculating the values of the minimizing weight functions is derived and applied to find positive weights for quadratures with equidistant knots. The concepts are further generalized to include Gauss–Radau and Gauss–Lobatto quadratures.

Key words. quadratures, Gauss quadratures, inverse problems, orthogonal polynomials, Jacobi matrices

AMS(MOS) subject classifications. primary 65F30; secondary, 65D07, 65D30, 65D32, 65F25

1. Introduction.

When approximating an integral $I_w(f) = \int_a^b f(x)w(x)dx$ by a quadrature formula $Q(f) = \sum_{j=1}^n w_j f(x_j)$, the Gauss quadrature has the advantages of having simple knots x_j inside the interval $[a, b]$, the maximal polynomial order of precision $2n$, and, importantly, positive weights w_j, for which it is sufficient to assume that the weight function w does not change sign in $[a, b]$. If, however, the knots are prescribed, the quadrature formula of the maximal polynomial order of precision (generally n) may have weights which change sign and attain large values, rendering the quadrature formula useless.

For any weight function w, positive in $[a, b]$, we can approximate the integral

$$(1.1)\quad I_1(f) = \int_a^b f(x)dx = \int_a^b \frac{f(x)}{w(x)}w(x)dx \simeq \sum_{j=1}^n w_j \frac{f(x_j)}{w(x_j)} = \sum_{j=1}^n \frac{w_j}{w(x_j)} f(x_j)$$

by the Gauss quadrature for w, leading to a quadrature for $I_1(f)$ with weights $w_j/w(x_j)$, which is exact for all f such that f/w is a polynomial of degree less than $2n$. We are therefore interested in the following problem.

PROBLEM 1.1. Given the knots x_j in $[a, b]$, is there a suitable weight function w for which these knots are the knots of the Gauss quadrature for w?

For w to be suitable we require two properties:

(a) It should be positive, not only at x_j, but everywhere in $[a, b]$.

(b) It should be smooth; particularly, f/w should be well approximated by polynomials.

We thus have a problem inverse to that of finding the Gauss quadrature for a given w, i.e., the knots and the interval are given and we want to describe all weight functions w on that interval for which the Gauss quadrature has the prescribed knots. Obviously, this is not possible (with a weight function not changing sign) if the knots are multiple (easy to avoid), or if one or both endpoints of the interval coincide with one of the knots. To include the latter situation, which is definitely of interest,

* Received by the editors February 8, 1991; accepted for publication (in revised form) August 2, 1991.

† School of Information Science and Technology, Flinders University, Bedford Park, South Australia, 5042 (j.kautsky@cc.flinders.edu.au).

we generalize the requirement on the weight function and introduce the following definitions.

DEFINITION 1.1. By $\mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ we denote the set of all functions w defined on $[a, b]$ such that the interpolatory quadrature for $I_w(f) = \int_a^b f(x)w(x)dx$ with the knots $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)^T$ has weights $\boldsymbol{w} = (w_1, w_2, \cdots, w_n)^T$ and polynomial order of precision at least M.

DEFINITION 1.2. By $\mathcal{W}(\boldsymbol{x}, a, b, M)$ we denote the union of $\mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ over all $\boldsymbol{w}$ such that $w_j > 0$ and $\|\boldsymbol{w}\|_1 = \sum_{j=1}^n w_j = b - a$.

We assume here that

$$a \le x_1 < x_2 < \cdots < x_n \le b \tag{1.2}$$

and that $M \le 2n$ ($M \le 2n-1$ or $M \le 2n-2$ if one or two, respectively, equalities hold in (1.2)). For the maximal M the interpolatory quadrature in Definition 1.1 will be the Gauss, Gauss–Radau, or Gauss–Lobatto quadrature. Since for any positive scalar λ, $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ implies $\lambda w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \lambda\boldsymbol{w})$, we normalize $\boldsymbol{w}$ in Definition 1.2 to avoid comparing pears with apples when we assess the suitability of a particular weight function w.

To describe (so that we find the suitable one mentioned above) *all* weight functions in $\mathcal{W}(\boldsymbol{x}, a, b, M)$ is not an easy task. However, we can divide and, hopefully, conquer the problem by considering, for fixed $\boldsymbol{w}$, the weight functions in $\mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$. Indeed, in §2 we characterize the unique weight function in $\mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ of given smoothness. We can then vary $\boldsymbol{w}$ to achieve the other desired property, positiveness of w. Such a search obviously covers all weight functions in $\mathcal{W}(\boldsymbol{x}, a, b, M)$.

In §3 we find the optimal weight functions for symmetric two- and three-point quadratures. The approach used there does not generalize easily to a larger number of knots. Therefore, in §4, we develop a numerical method for finding the smoothest weight function by representing it in terms of orthogonal polynomials. This leads to the problem of how to extend a given Jacobi matrix so that its spectrum lies in a given interval. In §5 we present results of numerical experiments involving equidistant knots and we conclude (§6) by a list of open problems.

Finding $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ resembles the (finite) Hausdorff moment problem (HMP, see [6] for overview) where one seeks a function having specified moments, but our situation is significantly different. We deal with the moments only implicitly—our input is the knots and, possibly, the weights of the Gauss quadrature. HMP is known to be ill posed [6]; intuitively, this is due to the bad condition of the map from the moments to the corresponding Gauss quadrature [3]. Thus this ill-posedness does not immediately apply to our problem. Furthermore, we do not seek to approximate the solution of the infinite HMP, but wish to identify a particular weight function whose finite number of moments also satisfy certain implicit conditions (prescribed Gauss knots, not weights). Consequently, the conditions on the existence of a solution are also simplified.

The work on this topic was initiated by a request from a theoretical physicist to find a good quadrature with given knots. However, the approach and techniques used here have been motivated by the author's joint work ([4],[5]) on orthogonal polynomials with Professor Gene H. Golub, to whom this work is gratefully dedicated.

2. Basic result. One way of achieving the smoothness of a function is to require the existence and reasonable smallness of its derivative. The aim of this section is to show that the weight function that minimizes $S_K(w) = \int_a^b (w^{(K)}(x))^2 dx$ over

all $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ is a polynomial of degree $M + 2K - 1$, satisfying certain boundary conditions. However, we first point out an obvious characterization of $\mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$.

LEMMA 2.1. *The weight function w is in $\mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ if and only if*

$$\int_a^b w(x)x^k dx = \sum_{j=1}^n w_j x_j^k, \qquad k = 0, 1, \cdots, M-1 \ .$$

The main result of this section resembles similar properties of natural interpolating splines.

THEOREM 2.2. *Let $M \geq K \geq 1$. The weight function minimizing $S_K(w)$ over all sufficiently smooth $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ is the unique polynomial q of degree $M + 2K - 1$ such that*

$$\int_a^b q(x)x^{j-1}dx = \mu_{j-1}, \qquad j = 1, 2, \cdots, M \ , \tag{2.1}$$

$$q^{(k)}(a) = q^{(k)}(b) = 0, \qquad k = K, K+1, \cdots, 2K-1 \ , \tag{2.2}$$

where

$$\mu_k = \sum_{j=1}^n w_j x_j^k, \qquad k = 0, 1, \cdots, M-1 \ . \tag{2.3}$$

Proof. Expressing q in standard powers expansion, the $M + 2K$ equations (2.1) and (2.2) are a linear system for the coefficients of q. We will show that if this system has a solution for some moments μ_j then it minimizes $S_K(w)$ and is unique (we can take the Gauss–Legendre quadrature as a special case for which $q = 1$ is such a unique solution). From that the existence follows for any moments μ_j. For any $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$, we have

$$S_K(w) = S_K(w - q + q) = S_K(w - q) + S_K(q) + 2X(w - q, q),$$

where the cross term X is

$$X(f, q) = \int_a^b f^{(K)}(x)q^{(K)}(x)dx \ .$$

Integrating by parts K times and taking into account the conditions (2.2) shows that

$$X(f, q) = \int_a^b f(x)q^{(2K)}(x)dx \ .$$

As $q^{(2K)}$ is a polynomial of degree $M - 1$, it now follows from Lemma 2.1 that $q \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$, and $X(w-q, q) = 0$ for all $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$. Thus $S_K(w) \geq S_K(q)$. If there are two solutions, then the Kth derivative and first M moments of their difference vanish. As $M \geq K$ they must be equal, which completes the proof. □

3. Two symmetric cases.

3.1. Two-point quadrature. The simplest symmetric case is a two-point quadrature on interval, say, $[-1, 1]$ with knots $\boldsymbol{x} = (-\xi, \xi)^T$ and weights $\boldsymbol{w} = (1, 1)^T$.

To preserve symmetry, this is the only choice of $\boldsymbol{w}$ to contribute to $\mathcal{W}(\boldsymbol{x}, a, b, 4) = \mathcal{W}(\boldsymbol{x}, a, b, 4, \boldsymbol{w})$ (here $M = 4$ for Gauss quadrature), so that the minimal $w_K \in \mathcal{W}(\boldsymbol{x}, a, b, 4, \boldsymbol{w})$, of minimal Kth derivative, depends only on ξ.

Our aim here is to specify the range of $\xi \in [0, 1]$ for which the minimal weight function w_K is positive, and to calculate the value $w_K(\xi)$. According to Theorem 2.2, $w_K(t)$ is a polynomial of order $4 + 2K$, which, due to symmetry, can be written in the form

$$w_K(t) = \sum_{j=0}^{K+1} a_j (t^2 - 1)^j \ .$$

The vanishing of the required derivatives is achieved by the restrictions, given in Table 3.1, on the coefficients a_j.

TABLE 3.1

K	Derivatives to vanish	Restrictions
1	1	$a_1 = 0$
2	2, 3	$a_1 = 8a_3, \quad a_2 = -2a_3$
> 2	> 2	$a_2 = a_3 = \cdots = 0$

Some work with MAPLE[1] shows that to match the moments these coefficients are as presented in Table 3.2.

TABLE 3.2

K	a_0	a_1	a_2	a_3
1	$\frac{3}{4}(7\xi^2 - 1)$	0	$\frac{105}{32}(1 - 3\xi^2)$	0
2	$\frac{1}{4}(27\xi^2 - 5)$	$8a_3$	$-2a_3$	$\frac{21}{64}(3\xi^2 - 1)$
> 2	$\frac{3}{2}(5\xi^2 - 1)$	$\frac{15}{4}(3\xi^2 - 1)$	0	0

Note that the Gauss knot $\xi = 1/\sqrt{3}$ leads to the constant weight function $w_K(t) = 1$ for all K. Further calculation shows that $w_K(t) > 0$ on $[-1, 1]$ if $\xi \in [\xi_{\min}, \xi_{\max}]$, where the limits and the weight function values are as given in Table 3.3.

TABLE 3.3

K	$\xi_{\min}$	$\xi_{\max}$	$w_K(\xi)$	$w_K(1/2)$
1	$\frac{1}{\sqrt{7}} \simeq 0.3780$	$\frac{9}{7\sqrt{3}} \simeq 0.7423$	$\frac{3}{32}(27 - 119\xi^2 - 105\xi^6 + 245\xi^4)$	$\frac{291}{256}$
2	$\sqrt{(\frac{5}{27})} \simeq 0.4303$	$\frac{\sqrt{151}}{\sqrt{261}} \simeq 0.7606$	$\frac{1}{64}(151 - 576\xi^2 + 1050\xi^4 - 336\xi^6 + 63\xi^8)$	$\frac{17311}{16384}$
> 2	$\frac{1}{\sqrt{5}} \simeq 0.4472$	$\frac{\sqrt{(3}}{\sqrt{5}} \simeq 0.7746$	$\frac{1}{4}(9 - 30\xi^2 + 45\xi^4)$	$\frac{69}{64}$

We observe that the midpoint quadrature formula ($\xi = \frac{1}{2}$) is in the range of positive w_K for all K, while the open quadrature formula ($\xi = \frac{1}{3}$) is not for any K.

[1] MAPLE is a registered trademark of Waterloo Maple Software.

3.2. Three-point quadrature. As a simplest case where the weights $\boldsymbol{w}$ are changing in $\mathcal{W}(\boldsymbol{x}, a, b, M)$, we will now consider a three-point quadrature, again on interval $[-1, 1]$, with knots $\boldsymbol{x} = (-\xi, 0, \xi)^T$ and weights $\boldsymbol{w} = 2(1, a, 1)^T/(2+a)$, where $a > 0$ is a free parameter. We restrict the consideration to minimizing S_1 only (i.e., $K = 1$), and to three choices of ξ; $\xi = \frac{1}{2}$ (with $M = 6$, open quadrature), $\xi = \frac{2}{3}$ (with $M = 6$, midpoint quadrature), and $\xi = 1$ (with $M = 4$, closed quadrature). Our aim is to find, in each of these three cases, the corresponding weights for the quadrature (1.1), and the value of a for which the weight function minimizing S_1 over $\mathcal{W}(\boldsymbol{x}, -1, 1, M)$ has maximal minimum which, hopefully, will be positive.

As above, according to Theorem 2.2, the minimizing weight function $w_1(t)$ is a polynomial of order 8 (6 for $\xi = 1$), which, due to symmetry, can be written in the form

$$w_{1,k}(t) = \sum_{j=0}^{k} a_j (t^2 - 1)^j, \qquad k = 3, 3 \quad \text{or } 2\ .$$

The vanishing of the first derivative at -1 and 1 is achieved by the restriction $a_1 = 0$.

Table 3.4 summarizes the values of the coefficients a_j and weights. All values should be multiplied by the normalizing factor $2/(2 + a)$.

TABLE 3.4

ξ	$w(0) = a_0$	a_1	a_2	a_3	$w(\xi)$
1/2	$\frac{5}{32}(1 + 2a)$	0	$\frac{315}{256}(17 - 6a)$	$\frac{1155}{256}(5 - 2a)$	$\frac{5}{2048}(1051 - 166a)$
2/3	$\frac{5}{48}(22 + 3a)$	0	$\frac{35}{128}(10 - 27a)$	$\frac{385}{128}(2 - 3a)$	$\frac{25}{432}(41 - 3a)$
1	$\frac{3}{8}(12 - a)$	0	$\frac{105}{64}(a - 4)$	0	$\frac{3}{8}(12 - a)$

As for the two-point quadrature with $K = 1$, $w_{1,2}(t)$ ($\xi = 1$) is minimal at $t = 0$ if $a_2 < 0$, and at $t = 1$ otherwise. A simple calculation shows that the maximum is reached (while scaling the above values by $2/(2 + a)$) at $a_2 = 0$, i.e., for $a = 4$, which reverts to constant $w(t)$, and thus the Gauss–Lobatto quadrature.

The other two cases are not so simple. Function $w_{1,3}(t)$ reaches its minimum in three different regions depending on the relations between a_0, a_2, and a_3. The ranges of a and the minima (without the scaling factor $2/(2 + a)$) are listed in Table 3.5.

TABLE 3.5

Region	A	B	C
Relations	$3a_3 > \max(3a_2, 2a_2)$	$a_2 > \max(a_3, 0)$	$3a_3 < 2a_2 < 0$
Minimum	$a_0 + a_2 - a_3$	a_0	$a_0 + \frac{4a_2^3}{27a_3^2}$
Open quadrature ($\xi = 1/2$)			
Region	$a < -1/2$	$-1/2 < a < 65/24$	$65/24 < a$
Minimum	$\frac{5}{64}(25a - 1)$	$\frac{5}{128}(8a - 23)$	$r_o(a)$
Midpoint quadrature ($\xi = 2/3$)			
Region	$a < 2/3$	$2/3 < a < 778/243$	$778/243 < a$
Minimum	$\frac{5}{1728}(675a - 482)$	$\frac{5}{432}(27a - 26)$	$r_m(a)$

Here

$$r_o(a) = \frac{5(4158551 - 4314744a + 1475904a^2 - 166400a^3)}{30976(8a - 17)^2}\ ,$$

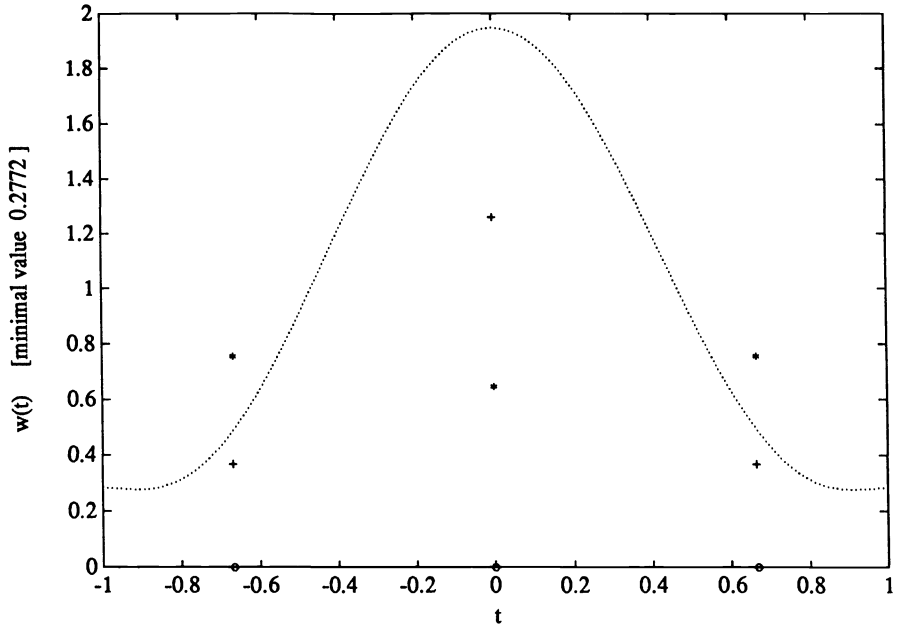


FIG. 3.1. *Weight function w and quadrature with three knots, $M = 6$, $K = 1$.* o: *knots,* + : *chosen weights,* $*$: *final weights.*

$$r_m(a) = \frac{5(2986281736 - 2540457108a + 688471974a^2 - 57572775a^3)}{940896(27a - 74)^2} .$$

The minima for regions A and B have positive derivatives, so the global minima are reached in region C. The rational functions $2r_{o,m}(a)/(2 + a)$ reach positive maxima listed in Table 3.6 (the optimal value for the closed quadrature is added for comparison).

TABLE 3.6

ξ	Optimal a	Minimal value of best w
$\frac{1}{2}$	3.1783566	0.010149
$\frac{2}{3}$	3.4153582	0.277238
1	4	1

In Figs. 3.1 and 3.2 we present the optimal weight functions for the midpoint and open three-point quadratures. For details of the layouts of these figures, see §5.

4. Numerical method. As the approach of the previous section is not likely to be successful for larger numbers of knots, we are interested in developing a numerical method to calculate the minimizing weight function w for given knots $\boldsymbol{x}$, weights $\boldsymbol{w}$, and interval $[a, b]$. The solution being a polynomial, we will represent it as a linear combination

$$w = \boldsymbol{p}_{M+2K}^T \boldsymbol{c}$$

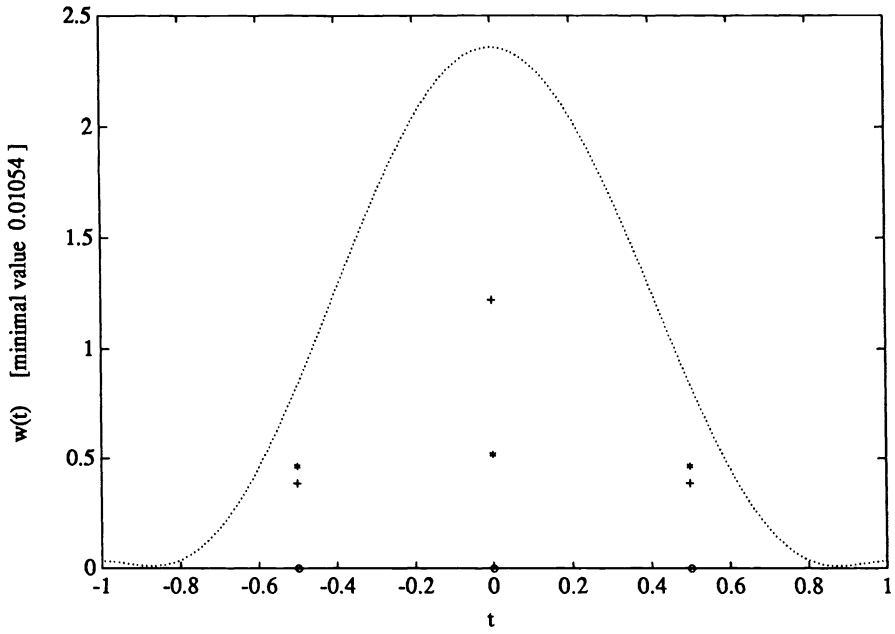


FIG. 3.2. *Weight function w and quadrature with three knots, $M = 6$, $K = 1$. o: knots, + : chosen weights, $*$: final weights.*

of suitably chosen orthogonal polynomials $\boldsymbol{p}_{M+2K} = (p_0, p_1, \cdots, p_{M+2K-1})^T$. Following, e.g., [2], we recall that orthogonal polynomials $\boldsymbol{p}_m$ satisfy the three-term relation

$$\beta_j p_j = (t - \alpha_j)p_{j-1} - (1 - \delta_{j1})\beta_{j-1}p_{j-2}, \qquad j = 1, \cdots, m,$$

which can be conveniently written in matrix notation

$$t\boldsymbol{p}_m = J_m\boldsymbol{p}_m + \beta_m p_m \boldsymbol{e}_m \tag{4.1}$$

where we have denoted t the identity function, $\boldsymbol{e}_m$ the mth column of the identity matrix, and J_m the Jacobi (symmetric, tridiagonal, unreduced) matrix

$$J_m = \begin{pmatrix} \alpha_1 & \beta_1 & 0 & \cdots & 0 \\ \beta_1 & \alpha_2 & \beta_2 & \cdots & 0 \\ 0 & \beta_2 & \alpha_3 & \ddots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \alpha_m \end{pmatrix}.$$

Our aim is to construct a system of linear equations implementing conditions (2.1) and (2.2) of Theorem 2.2 for the unknown coefficients $\boldsymbol{c}$. Our strategy is to choose the orthogonal polynomials (i.e., the matrix J_{M+2K}) in a way which will make these conditions, and those relating the moments of Lemma 2.1 with the given weights $\boldsymbol{w}$, simple.

We first establish two preliminary results. The first one can be seen by inspection of the Stieltjes algorithm [3] for constructing the elements of a Jacobi matrix.

LEMMA 4.1. *Let μ_j be moments of some weight function and α_j, β_j elements of the three-term relation (Jacobi matrix) for the corresponding orthogonal polynomials. For any m there is a one-to-one correspondence between the first m elements of the sequences $\{\mu_0, \mu_1, \mu_2, \cdots\}$ and $\{\mu_0, \alpha_1, \beta_1, \alpha_2, \beta_2, \cdots\}$.*

The second result, the proof of which is also straightforward, is concerned with the characterization of functions having the same moments.

LEMMA 4.2. *Let $\boldsymbol{p}_m$ be the polynomials orthogonal with respect to a positive weight function f on the interval (a, b) (which then contains the eigenvalues of the corresponding Jacobi matrix J_m). Then*

$$\int_a^b \boldsymbol{p}_m(t) f(t) dt = \mu_0^{1/2} \boldsymbol{e}_1 , \qquad \mu_0 = \int_a^b f(t) dt .$$

Furthermore, any function g satisfying

$$\int_a^b \boldsymbol{p}_m(t) g(t) dt = \mu_0^{1/2} \boldsymbol{e}_1$$

has the same first m moments as function f.

Given knots $\boldsymbol{x}$ and weights $\boldsymbol{w}$ of a Guass quadrature, we can construct (see [2]) a size n Jacobi matrix $\tilde{J}_n$ for which the corresponding polynomials are orthogonal with respect to any weight function with the same $2n$ moments, given by (2.3) with $M = 2n$. Applying Lemma 4.1 to these moments shows that, for $j \leq n$,

(a) the first $2j$ moments determine the diagonal element $\tilde{\alpha}_j$ and

(b) the first $2j - 1$ moments determine the subdiagonal element $\tilde{\beta}_{j-1}$.

This means that we have the following result.

THEOREM 4.3. *Let the matrix J_{M+2K} satisfy the following:*

(a) *the first $[M/2]$ diagonal elements equal those of $\tilde{J}_n$,*

(b) *the first $[(M-1)/2]$ subdiagonal elements equal those of $\tilde{J}_n$, and*

(c) *the eigenvalues of the order M principal submatrix of J_{M+2K} are in $[a, b]$.*

Denoting $\boldsymbol{p}_j$ the vector of the first j orthogonal polynomials given by J_{M+2K}, the weight function $w = \boldsymbol{p}_{M+2K}^T \mathbf{c}$ minimizing $S_K(w)$ over all $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ is determined by the solution $\mathbf{c}$ of the system

$$\int_a^b \boldsymbol{p}_M(t) \boldsymbol{p}_{M+2K}^T(t) dt \; \mathbf{c} = (b-a)^{1/2} \boldsymbol{e}_1 \tag{4.2}$$

$$\begin{aligned} \boldsymbol{p}_{M+2K}^{(k)}(a)^T \; \mathbf{c} &= 0 \\ \boldsymbol{p}_{M+2K}^{(k)}(b)^T \; \mathbf{c} &= 0, \qquad k = K, K+1, \cdots, 2K-1. \end{aligned} \tag{4.3}$$

Proof. The theorem follows from Lemma 2.1, Theorem 2.2, and the above considerations taking into account the normalization of the weights $\boldsymbol{w}$. The condition (c) is needed for (4.2) to enforce the moment preservation by Lemma 4.2. □

The main problem here is the construction of the matrix J_{M+2K}, that is, an extension of $\tilde{J}_n$, obtained from the data $\boldsymbol{x}$ and $\boldsymbol{w}$, in such a way that condition (c) of Theorem 4.3 holds. Our solution is based on the fact that a given Jacobi matrix can be (uniquely) escalated to assign to it two eigenvalues.

LEMMA 4.4. *Given Jacobi matrix J_m of size m and scalars λ_1, λ_2 such that*

$$\lambda_1 < \mathit{spectrum}(J_m) < \lambda_2,$$

the matrix

$$J_{m+1} = \begin{pmatrix} J_m & \beta \boldsymbol{e}_m \\ \beta \boldsymbol{e}_m^T & \alpha \end{pmatrix}$$

has eigenvalues λ_1, λ_2 *and*

$$\lambda_1 \leq \text{spectrum}(J_{m+1}) \leq \lambda_2$$

provided that

$$\begin{aligned} \alpha &= (\lambda_1 + \lambda_2 + \beta^2(\rho_1 + \rho_2))/2, \\ \beta^2 &= (\lambda_1 - \lambda_2)/(\rho_2 - \rho_1), \\ \rho_{1,2} &= \boldsymbol{e}_m^T (J_m - \lambda_{1,2} I)^{-1} \boldsymbol{e}_m \ . \end{aligned} \tag{4.4}$$

Proof. The matrix calculation is elementary. As the eigenvalues of J_m interlace those of J_{m+1}, the second largest eigenvalue of J_{m+1} must be smaller than the largest eigenvalue of J_m, and so on. □

For even $M = 2m$, the eigenvalues of the order $m \leq n$ principal submatrix $\tilde{J}_m$ of $\tilde{J}_n$ must lie inside $[a, b]$ (for $m = n$ by assumption), and we can choose a sequence of additional eigenvalues between the spectrum of $\tilde{J}_m$ and the endpoints, a and b, to extend the matrix to size M, by applying Lemma 4.4 repeatedly.

For odd $M = 2m - 1$, we can proceed as for even M, if the eigenvalues of $\tilde{J}_m$ lie inside $[a, b]$, but if $M + 1 = 2n$ and one of the prescribed knots coincides with an endpoint (Gauss–Radau quadrature), this is not possible. However, we can change the element $\tilde{\alpha}_{m,\text{new}} = \tilde{\alpha}_m + \alpha$ of $\tilde{J}_m$ without affecting the first $2m - 1 = M$ moments specified by the given quadrature. Our aim in this case is to choose the parameter α so that the spectrum of $\tilde{J}_m$ moves towards the centre of $[a, b]$ (we can do this for any odd M, and any knots to improve the further extension). Denoting $\tilde{J}_m = Q \text{diag}(\lambda_1, \cdots, \lambda_m) Q^T$, $\lambda_1 < \cdots < \lambda_m$, the ordered eigenvalue decomposition of $\tilde{J}_m$, the eigenvalues of $\tilde{J}_m + \alpha \boldsymbol{e}_m \boldsymbol{e}_m^T$ satisfy the secular equation

$$\omega(\lambda) \left(1 + \alpha \sum_{j=1}^{m} \frac{v_j^2}{\lambda_j - \lambda} \right) = 0$$

where $(v_1, \cdots, v_m) = \boldsymbol{e}_m^T Q$, and $\omega(\lambda) = \prod_{j=1}^{m} (\lambda_j - \lambda)$. Differentiating with respect to α, we find that

$$\lambda_k'(\alpha) = \frac{v_k^2}{1 + \alpha \sum_{j \neq k} (v_j^2 + v_k^2)/(\lambda_j - \lambda_k)} \ .$$

The new eigenvalues are $\lambda_{1,m}(0) + \alpha \lambda_{1,m}'(\rho\alpha)$ where, by the mean value theorem, $0 \leq \rho \leq 1$. To position them symmetrically in $[a, b]$ leads to a quadratic equation for α. We choose the root of smaller magnitude for the required correction. Numerical evidence shows that this correction is quite sensitive to the choice of parameter ρ. Using the Taylor expansion ($\rho = 0$) gave bad results but more implicit choices ($\frac{1}{2} \leq \rho \leq 1$) worked quite well.

The extension from size M to $M + 2K$ is arbitrary. Repeating the last elements α_M and β_{M-1} appears satisfactory.

Once the matrix J_{M+2K} is available, the implementation of Theorem 4.3, i.e., the construction of the matrix for the system of equations (4.2) and (4.3), is reasonably straightforward. For the first part we have

$$\int_a^b \boldsymbol{p}_M(t)\boldsymbol{p}_{M+2K}^T(t)dt \;=\; L_M\, L_{M+2K}^T,$$

where L_{M+2K} (L_M is its leading submatrix of order M) transforms Legendre orthogonal polynomials for $[a,b]$ into $\boldsymbol{p}_{M+2K}$. It can be constructed recurrently, row by row, from the relation

$$L_{M+2K}\,\hat{J} \;=\; J_{M+2K}\,L_{M+2K} + \boldsymbol{e}_{M+2K}\boldsymbol{z}^T\,,$$

for some $\boldsymbol{z}$ (not needed in the process), where $\hat{J}$ is the known Jacobi matrix for the Legendre weight function on $[a,b]$. The last $2K$ rows (4.3) of the system can be obtained by differentiating the identity (4.1) (without the last row but augmented at the top by $\boldsymbol{e}_1^T\boldsymbol{p}_m = (b-a)^{1/2}$ giving an explicit recurrence).

5. Numerical examples. Using the results of the previous sections, the aim of the numerical experiments was to find a positive weight function for which the Gauss knots are equidistant. More explicitly, if the distance between adjacent knots is constant, say h, and the distance between endpoints and extreme knots is ρh, $0 \le \rho \le 1$, we call the quadrature equidistant (closed for $\rho = 0$, midpoint for $\rho = \frac{1}{2}$). The strategy was to choose the weights $\boldsymbol{w}$ in such a way that, for a given K, the function minimizing $S_K(w)$ over all $w \in \mathcal{W}(\boldsymbol{x}, a, b, M, \boldsymbol{w})$ has a maximal minimum.

We have developed a set of MATLAB[2] programs implementing the numerical method of §4, together with the following steps:

(a) to choose $\boldsymbol{w}$ manually and to display the plot of the weight function and its minimum, and

(b) to search for the optimal $\boldsymbol{w}$ using MATLAB function *fmins* (Nelder–Meade simplex algorithm [1]).

The results of selected cases are in Figs. 5.1–5.7. Here, the "chosen weights" are the searched for optimal values of $\boldsymbol{w}$, and the "final weights" are the coefficients $w_j/w(x_j)$ of the quadrature in (1.1).

We first discuss the case of the midpoint quadrature with five knots. Due to symmetry and scaling there are only two free parameters in $\boldsymbol{w}$. In Fig. 5.1 we give, for $K = 1$, a typical starting guess with equal values in $\boldsymbol{w}$. We note that the weight function, a symmetric polynomial of degree 11, oscillates and turns negative near the endpoints. Although it is not obvious from the graph, the first derivative is zero at the endpoints (rechecked numerically from the coefficients of the solution). Figs. 5.2, 5.3, and 5.4, give the best (maximal minimum) weight functions for $K = 1, 2$, and 3, respectively. We observe that the oscillations have been smoothed out and the weight function has assumed a nice bell-like shape, possibly with gentle ripples near the endpoints (this pattern will prevail in examples with more knots). The case of minimizing the second derivative (Fig. 5.3, $K = 2$) is an exception, in that the ripples are significant. We suspected that we might have found a local, rather than global, extreme, but the presented results were confirmed by the following experiment. In the case $K = 1$ we found, by the above-mentioned manual search, good values of $\boldsymbol{w}$ giving a weight function with a shape similar to that of Fig. 5.3 (minimum 0.0226),

[2] MATLAB is a trademark of MathWorks, Inc.

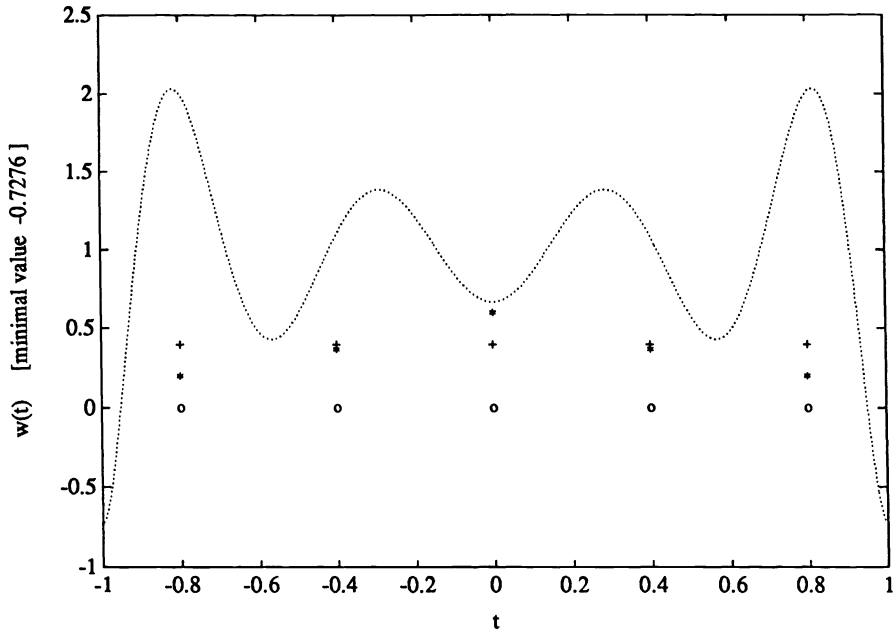


FIG. 5.1. *Weight function w and quadrature with five knots, M = 10, K = 1. o: knots, + : chosen weights, * : final weights.*

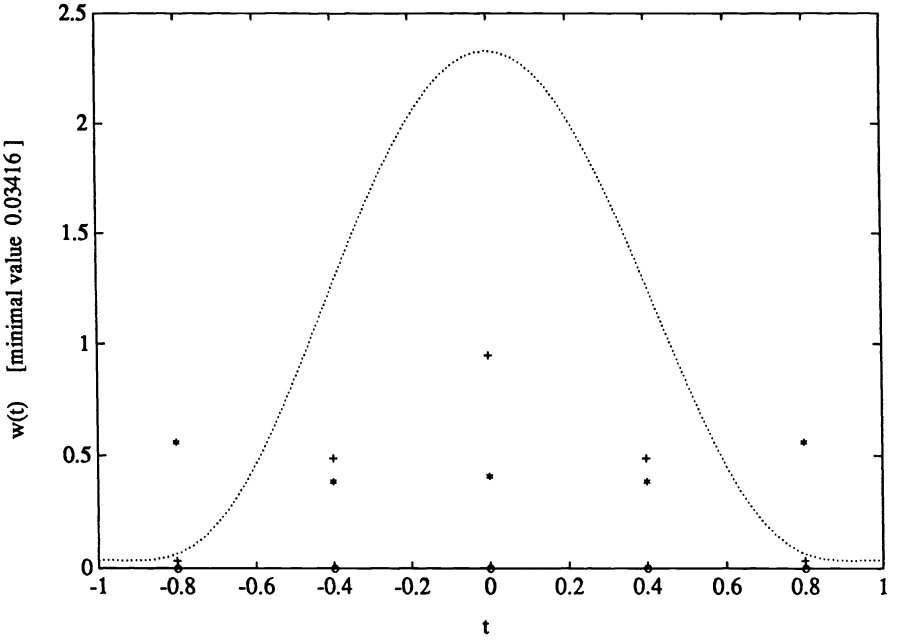


FIG. 5.2. *Weight function w and quadrature with five knots, M = 10, K = 1. o: knots, + : chosen weights, * : final weights.*

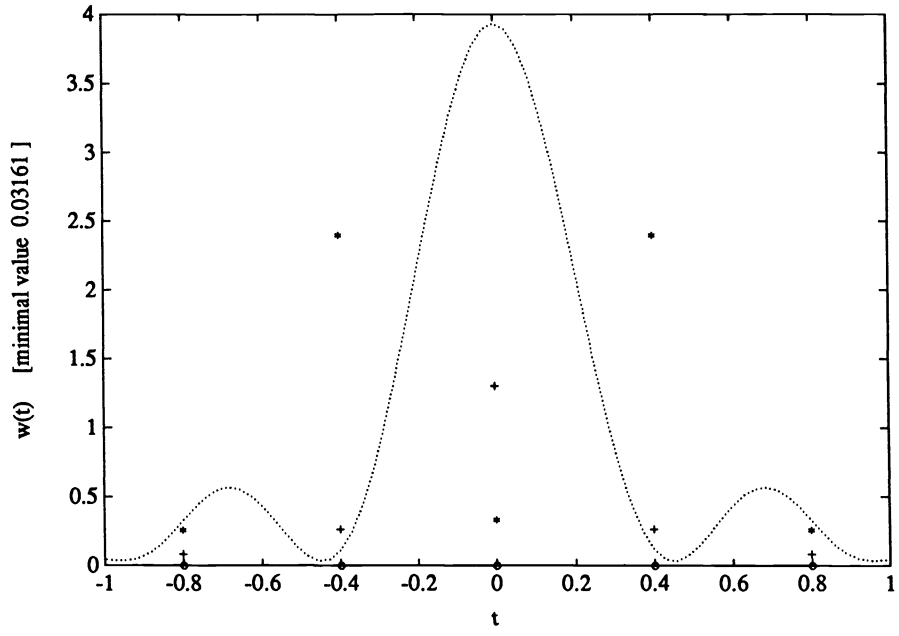


FIG. 5.3. *Weight function w and quadrature with five knots, M = 10, K = 2.* o: *knots,* + : *chosen weights,* ∗ : *final weights.*

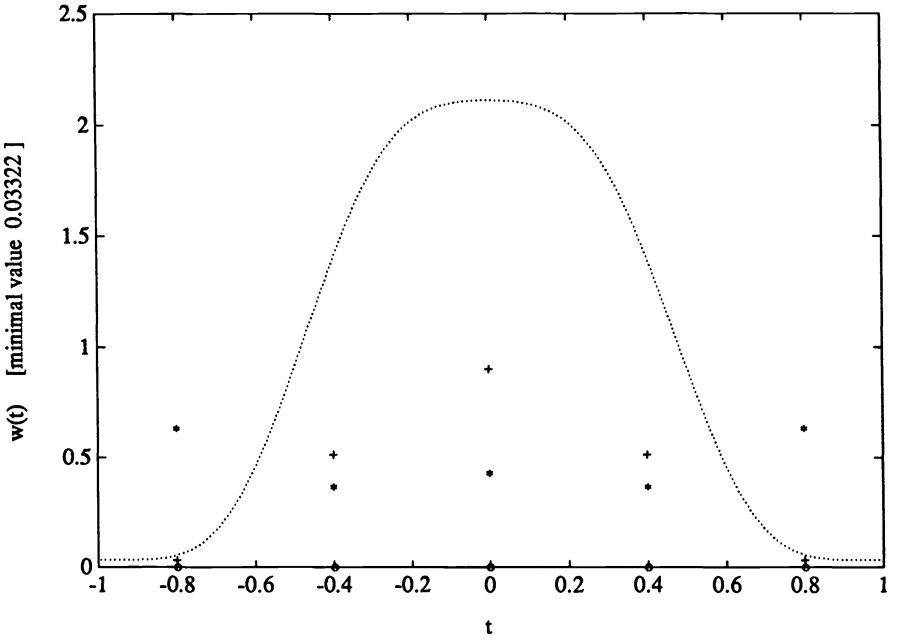


FIG. 5.4. *Weight function w and quadrature with five knots, M = 10, K = 3.* o: *knots,* + : *chosen weights,* ∗ : *final weights.*

TABLE 5.1

Maximal M equidistant quadratures with five points					
Weight function	Constant w	With maximal minimum			
$K,\ \rho$	1, 1/2	1, 1/2	2, 1/2	3, 1/2	2, 0
min $w(t)$	−0.725366	0.034059	0.029346	0.033209	0.185872
max $w(t)$/min $w(t)$	−2.80	68.36	134.07	63.59	14.45
w_1	0.4	0.035937	0.082406	0.033556	0.023541
w_2	0.4	0.488661	0.264247	0.514349	0.348803
w_3	0.4	0.950805	1.306695	0.904190	1.255312
$w_1/w(x_1)$	0.199430	0.561497	0.254805	0.632367	0.126089
$w_2/w(x_2)$	0.366431	0.385426	2.394967	0.366897	0.652959
$w_3/w(x_3)$	0.601283	0.408374	0.332116	0.428172	0.467222
Sum of weights	1.733007	2.302221	5.631660	2.426700	2.025318
System condition	$1.55\ 10^3$	$2.99\ 10^3$	$2.10\ 10^6$	$2.52\ 10^9$	$4.48\ 10^5$
	Coefficients in the power expansion ($\times 10^{-2}$) of $w(t)$				
1	0.006652	0.023283	0.039345	0.021117	0.026868
t^2	0.203032	−0.077503	−0.545283	−0.011004	−0.158159
t^4	−1.794566	0.065802	2.762596	−0.370964	0.387747
t^6	5.216279	0.039453	−6.388922	1.291962	−0.461128
t^8	−5.930627	−0.081007	7.454143	−1.965805	0.267394
t^{10}	2.291954	0.030313	−4.300143	1.583819	−0.060854
t^{12}			0.978696	−0.662723	
t^{14}				0.113930	
t^{16}					

and started the search from there. After exploring the neigbourhood of this, fairly different, value of $\boldsymbol{w}$, the algorithm again found the optimal solution given in Fig. 5.2. We have repeated this experiment for Fig. 5.3 ($K = 2$) in reverse, with the same result.

Figure 5.5 gives the best ($K = 2$) weight function with five equidistant Gauss–Lobatto knots. Table 5.1 summarizes some numerical values of the five cases in Figs. 5.1–5.5. Here $x_3 = 0$, $x_4 = -x_2 = 0.4$ or 0.5, and $x_5 = -x_1 = 0.8$ or 1 for midpoint ($\rho = \frac{1}{2}$) or closed ($\rho = 0$) quadrature, and we do not repeat the symmetric values of the weights. We do not present the coefficients of w in the orthogonal expansion as the base changes from case to case, but list the standard power expansion (not otherwise used in the calculations). We report that the orthogonal expansion coefficients for the best weight functions decrease in magnitude for higher powers more than those for the initial guesses. We also note the increasing magnitude of the condition of the system solved to obtain these coefficients, particularly for higher K (order of the minimized derivative). The sum of the total weights is not the length of the interval 2 as the quadrature integrates exactly products of w (itself a polynomial), and polynomials of degree 9 (7 in Fig. 5.5), which, of course, do not include a constant.

We now mention, briefly, a few results for midpoint quadratures with more than five knots. In Fig. 5.6 we have a positive weight function with eight equidistant Gauss knots minimizing S_1 (the result for $K = 2$ is very similar). The minimum is very small, so we have tried decreasing M to 14 (a "sub-Gauss" quadrature), and the result (Fig. 5.7) shows a different shape weight function, but still with a fairly small minimum. We have not found a positive weight function with 10 equidistant knots, with the maximal minimum reached for $\rho = \frac{1}{2}$ and $K = 2$ being −0.0003.

6. Conclusions and open problems. We were motivated by the task of finding a good quadrature with given knots, to attempt to solve an inverse problem for Gauss quadrature: "Given the Gauss quadrature, find a good weight function for it."

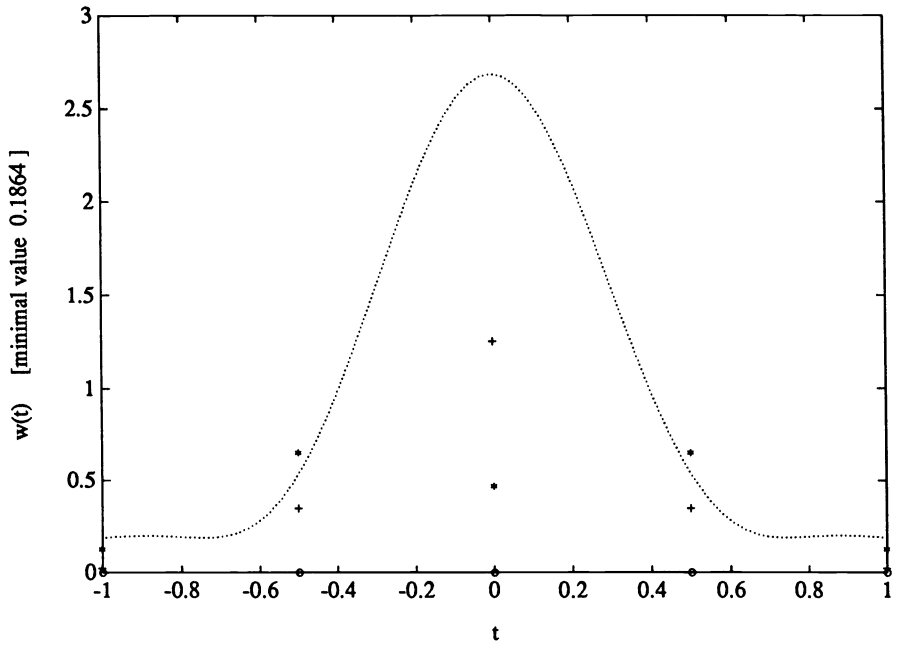


FIG. 5.5. *Weight function w and quadrature with five knots, $M = 8$, $K = 2$.* o: *knots,* + : *chosen weights,* $*$: *final weights.*

Although we have developed a numerical algorithm for solving the latter problem, the numerical solution appears quite difficult for a larger number of knots. In that sense we have not found easy new answers to the original problem, but have shown some theoretical and practical results for the inverse problem for Gauss quadratures.

In particular, we have searched, given n, for a smooth positive weight function, the n Gauss knots of which would be equidistant. We have observed that such a weight function tends to have a rather smooth bell shape (although it is a polynomial of high degree) with very small, near constant, values near the endpoints. This is not surprising as the Gauss knots for classical weight functions (Legendre, Gegenbauer, etc.) accumulate at endpoints, so to push them towards the center requires the weight function to emphasize the center of the interval. It is possibly of interest to know how small the weight function must be at the endpoints and, furthermore, if it exists for all n. Our results indicate that the cutoff value may be around $n = 10$.

There are a number of other open problems which remain:

(a) Another weight function can be introduced into the integral to be approximated

$$I_{\tilde{w}}(f) = \int_a^b f(x)\tilde{w}(x)dx = \int_a^b \frac{f(x)}{w(x)} w(x)\tilde{w}(x)dx \simeq \sum_{j=1}^{n} \frac{w_j}{w(x_j)} f(x_j).$$

This will be necessary if infinite intervals are to be considered. A similar result to that of Theorem 2.2 can be derived with the optimal weight function w being related to the antiderivatives of this weight function $\tilde{w}$, rather than a polynomial.

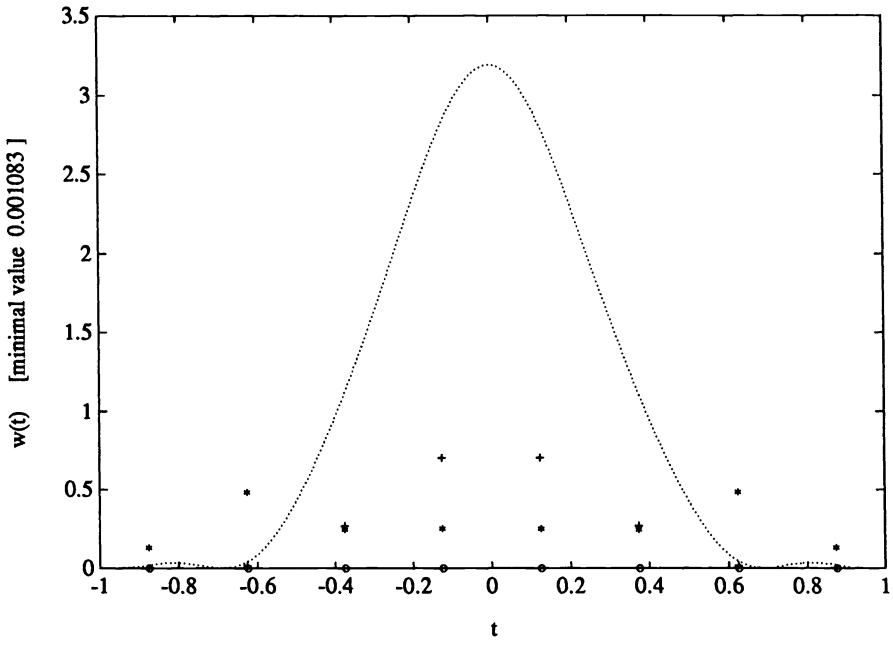


FIG. 5.6. *Weight function w and quadrature with eight knots, $M = 16$, $K = 1$. o: knots, + : chosen weights, * : final weights.*

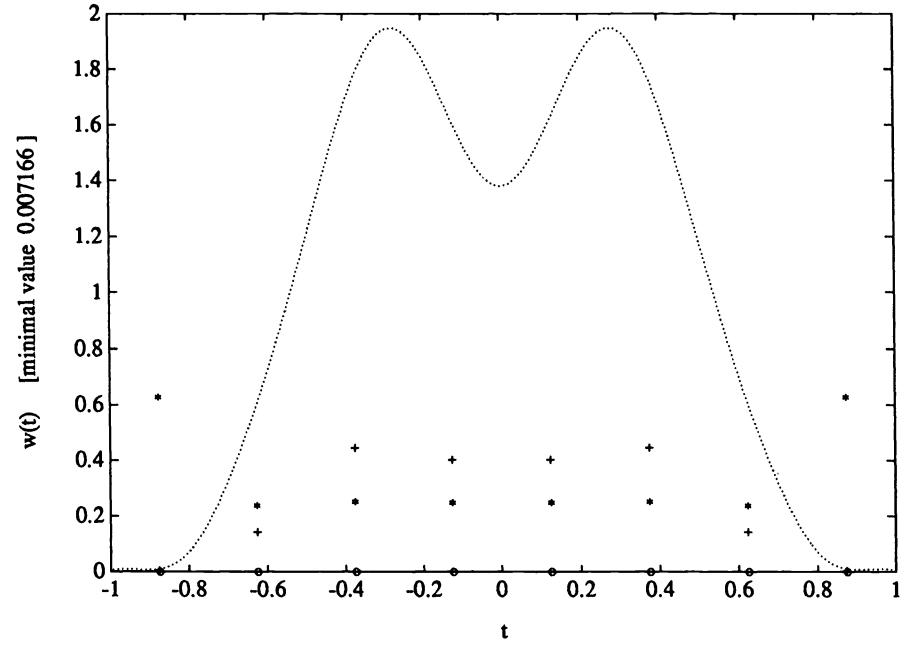


FIG. 5.7. *Weight function w and quadrature with eight knots, $M = 14$, $K = 2$. o: knots, + : chosen weights, * : final weights.*

(b) As for splines, a representation based on the Kth derivative of the solution to the minimization problem can be sought.
(c) The numerical conditioning of the algebraic system (4.2), (4.3) can be studied, particularly the dependence on the choice of eigenvalues when extending the Jacobi matrix to the desired size. We have used a sequence of equidistant values and have not experienced serious difficulties, although better strategies may exist.
(d) New theoretical properties of the optimal solution may lead to a more efficient global search for good positive weight functions.
(e) Minimizing other smoothness functionals, say, with a combination of derivatives, may lead to more interesting results.
(f) Some of the techniques derived in this paper may be applicable in solving the finite Hausdorff moment problem.

REFERENCES

[1] J. DENNIS AND D. WOODS, *New Computing Environments: Microcomputers in Large-Scale Computing*, A. Wouk, ed., Society for Industrial and Applied Mathemetics, Philadelphia, PA, 1987, pp. 116–122.
[2] S. ELHAY, G. GOLUB, AND J. KAUTSKY, *Updating and downdating of orthogonal polynomials with data fitting applications*, SIAM J. Matrix Anal. Appl., 12 (1990), pp. 327–353.
[3] W. GAUTSCHI, *On generating orthogonal polynomials*, SIAM J. Sci. Statist. Comput., 3 (1982), pp. 289–317.
[4] G. GOLUB AND J. KAUTSKY, *Calculation of Gauss quadratures with multiple free and fixed knots*, Numer. Math., 41 (1983), pp. 147–163.
[5] J. KAUTSKY AND G. GOLUB, *On the calculation of Jacobi matrices*, Linear Algebra Appl., 52/53 (1983), pp. 439–455.
[6] G. TALENTI, *Recovering a function from a finite number of moments*, Inverse Problems, 3 (1987), pp. 501–517.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 419–453, April 1992

A CHART OF NUMERICAL METHODS FOR STRUCTURED EIGENVALUE PROBLEMS*

ANGELIKA BUNSE-GERSTNER†, RALPH BYERS‡, AND VOLKER MEHRMANN†

Abstract. It is common in applied mathematics to encounter matrices that are symmetric, Hermitian, skew symmetric, skew Hermitian, symplectic, conjugate symplectic, J-symmetric, J-Hermitian, J-skew symmetric, or J-skew Hermitian. Eigenvalue algorithms for real and complex matrices that have at least *two* such algebraic structures are considered. In the complex case numerically stable algorithms were found that preserve and exploit both structures of 40 out of the 66 pairs studied. Of the remaining 26, algorithms were found that preserve part of the structure of 12 pairs. In the real case algorithms were found for all pairs studied. The algorithms are constructed from a small set of numerical tools, including orthogonal reduction to Hessenberg form, simultaneous diagonalization of commuting normal matrices, Francis's QR algorithm, the quaternion QR-algorithm, and structure revealing, symplectic, unitary similarity transformations.

Key words. eigenvalue, eigenvector, symmetric, Hermitian, skew symmetric, skew Hermitian, symplectic, conjugate symplectic, Hamiltonian matrices, quaternion matrices, QR-algorithm, Jacobi algorithm

AMS(MOS) subject classifications. 65F15, 65H10, 65H15, 15A18, 93C45, 93E25, 49A10

1. Introduction and notation.

This paper presents numerical methods for calculating all eigenvalues and eigenvectors of matrices that have more than one of the following special structures.

DEFINITION 1.1. A matrix $A \in \mathbb{C}^{m,m}$ is

—symmetric if $A^T = A$;
—Hermitian if $A^* = A$;
—orthogonal if $A^T A = I$;
—unitary if $A^* A = I$;
—J-symmetric if $m = 2n$ and $JA = (JA)^T$, where $J = \left[\begin{smallmatrix} 0 & I \\ -I & 0 \end{smallmatrix}\right]$ and I is the $n \times n$ identity matrix;
—J-Hermitian (or Hamiltonian) if $m = 2n$ and $JA = (JA)^*$;
—J-orthogonal (or symplectic) if $m = 2n$ and $A^T JA = J$;
—J-unitary (or conjugate symplectic) if $m = 2n$ and $A^* JA = J$;
—J-skew symmetric if $m = 2n$ and $JA = -(JA)^T$;
—J-skew Hermitian if $m = 2n$ and $JA = -(JA)^*$.

If $A \in \mathbb{R}^{m,m}$ some of these categories coalesce, e.g., Hermitian and symmetric, but other categories take extra structure. A real skew Hermitian matrix has eigenvalues that occur in $+/-$ pairs while a complex skew Hermitian matrix may not.

There are several reasons why we are interested in algorithms which are not only numerically stable but also preserve the algebraic structure of the problem, i.e., methods that are strongly stable in the sense defined by Bunch [B4]. Usually the algebraic structure forces the eigenvalues to lie in certain regions in the complex plane (e.g., on the unit circle or on the real axis) or to occur in different kinds of pairings. (See Table 2.4.) A treatment of the problem with an algorithm that does not reflect the algebraic structure produces (in the iterative process) intermediate eigenvalue problems, which are similar

* Received by the editors January 3, 1989; accepted for publication (in revised form) June 12, 1990.

† Fakultät für Mathematik, Universität Bielefeld, Postfach 8640, D-4800 Bielefeld 1, Germany. This research was partially supported by Forschungsschwerpunkt Mathematisierung, Universität Bielefeld (umatf112%dbiuni11.bitnet and mehrmann@dhdibm1).

‡ Department of Mathematics, University of Kansas, Lawrence, Kansas 66045. The research of this author was partially supported by National Science Foundation grant CCR-8820882 and University of Kansas General Research Allocations 3758-20-038 and 3693-20-038 (byers@ukanvax.bitnet).

or equivalent to the original problem but are without a structure that guarantees these properties. Rounding errors can cause eigenvalues to wander out of their required regions [V2]. This may be acceptable for eigenvalues, but it obscures the identity of the invariant subspaces. Structure preserving algorithms are often more efficient than general methods.

A classic example of the benefits of structure preserving algorithms is the symmetric eigenvalue problem. Subroutine **CG** from [S5] is perhaps the best available computer program for calculating all eigenvalues and eigenvectors of a general complex matrix. It has been shown both in theory and practice that in the presence of rounding errors, it computes the eigenvalues and eigenvectors of a rounding-error–small perturbation of the data. It has been tested on many different computers and many different problems.

Consider the real symmetric matrix $A = I - UU^T$ where $U = [1, 1, 1, 1, \cdots, 1]^T \in \mathbb{R}^n$. On an IBM/XT with machine precision approximately 10^{-17}, rounding errors caused **CG** to compute eigenvalues with imaginary parts as large as 10^{-16} and eigenvectors with imaginary parts as large as 7/10. Real symmetric matrices have real eigenvalues and admit a system of real orthonormal eigenvectors. The eigenvalues calculated by **CG** are not realistic, because they are not eigenvalues of a symmetric matrix.

Subroutine **RS**, also from [S5], is designed specifically for the real symmetric eigenvalue problem. When the same problem was solved by **RS**, the computed eigenvalues have zero imaginary part and the computed eigenvectors are real and orthonormal up to the precision of the arithmetic. Furthermore, **RS** uses only one-fourth of the storage and does less than one-fourth of the arithmetic that **CG** does. For the real symmetric eigenvalue problem, **RS** is superior to **CG** in every way.

One of the goals of this paper is to extend the advantages of structure preserving algorithms to problems with other symmetry structures. We present a few basic numerical techniques that combine to form structure preserving algorithms for doubly structured eigenvalue problems.

Structured eigenvalue problems appear in many scientific and engineering applications. We briefly discuss a few of them below.

The real symmetric eigenvalue problem arises in almost every area of science and engineering. It is the most extensively studied of the structured eigenvalue problems. Efficient, numerically stable, structure preserving algorithms are well established [P5]. The excellent efficiency and accuracy of these methods result directly from their ability to exploit and preserve symmetry. This area is still quite active, with much effort directed toward supercomputer algorithms, e.g., [C2], [D3], [H2], [S3], [W1].

The J-symmetric or Hamiltonian eigenvalue problem occurs when solving continuous time linear-quadratic optimal control problems and algebraic Riccati equations [A6], [B16], [L1], [L2], [M2], [P1], [S1], [V1]. Consider, for example, the well-known control problem of selecting a control function $u(t) \in \mathbb{R}^m$ to minimize

$$J(x,u) = \int_0^\infty x(t)^T Q x(t) + u(t)^T R u(t)\, dt \tag{1.2}$$

subject to

$$\dot{x} = Ax + Bu, \qquad x(0) = x_0, \tag{1.3}$$

where $Q = Q^T \in \mathbb{R}^{n,n}$ is positive semidefinite, $R = R^T \in \mathbb{R}^{m,m}$ is positive definite, $A \in \mathbb{R}^{n,n}$, $B \in \mathbb{R}^{n,m}$, and $x(t) \in \mathbb{R}^n$. Under mild assumptions [A6], there is a unique optimal solution given by a linear feedback law

$$u(t) = -R^{-1} B^T X x. \tag{1.4}$$

In (1.4) $X \in \mathbb{R}^{n,n}$ is the symmetric positive semidefinite solution of the algebraic Riccati equation

$$0 = Q + A^T X + XA - XBR^{-1}B^T X. \tag{1.5}$$

Solutions to (1.5) are given by $X = -ZY^{-1}$ where the columns of $\begin{bmatrix} Y \\ Z \end{bmatrix}$ span the invariant subspace of the real J-symmetric matrix

$$M = \begin{bmatrix} A & BR^{-1}B^T \\ Q & -A^T \end{bmatrix} \tag{1.6}$$

corresponding to eigenvalues with negative real part [A3], [A4], [B15], [B16], [L1]–[L3], [L5], [M2], [P1], [P7], [V1]. Complex control problems of the same type that lead to eigenvalue problems for J-Hermitian matrices occur in the control of rotor systems [M3], [M4]. Finding an efficient numerically stable algorithm that also preserves the J-symmetric structure is still an open problem. Progress has been made on the special case of J-symmetric matrices stemming from single input or single output control problems [B15], [B16]. At this writing, in the general case, we must either drop the preservation of structure and treat the problem as a general unsymmetric eigenvalue problem [L1], [V1], or give up possible gains in efficiency by refining solutions produced by conditionally stable methods [B1], [B12], [B13], [B17], [G2], [M2], [R1].

A Jacobi-type method for this problem has been studied in [B18], but is still not completely satisfactory.

Symplectic or conjugate symplectic eigenvalue problems come from the solution of discrete-time linear-quadratic optimal control problems and discrete algebraic Riccati equations analogous to the continuous case described above [L2], [M1], [M2], [P3], [P6], [S1], [S4]. As in the J-symmetric case, progress has been made in the single input or single output case [M1], but in the general case no entirely satisfactory method is known. Some algorithms treat this problem as a general nonsymmetric eigenvalue problem [A3]–[A5], [L5], [P3], [V1], but this sacrifices the efficiency and numerical stability that a structure preserving algorithm might offer. Other algorithms maintain the structure only at the risk of numerical instability [B2], [G2].

For symplectic, J-symmetric, or J-skew symmetric problems, entirely satisfactory algorithms are known only in case the problem has an additional structure [D2], [B16], [M1]. This paper systematically extends the list of these special cases. In particular we exhibit numerically stable, structure preserving algorithms for most cases in which a J-symmetry structure is combined with another special structure. Usually the algorithm is a variant QR or Jacobi iteration that preserves both special structures. Hopefully, this is another step towards understanding what algorithms should be like for the general case of a J-symmetric structure. Some of these methods (e.g., the quaternion QR algorithm [B9]) are block-QR iterations. Block-QR iterations are of interest for parallel and vector computations and a study of simple special cases might help to find block-QR algorithms for more general problems.

Applications of the orthogonal and unitary eigenvalue problem include calculation of Gaussian quadrature formulae on the unit circle and Pisarenko frequency estimates [C3]. Numerical methods for orthogonal and unitary eigenvalue problems have recently been proposed in [A2], [G5], and [G6].

The skew symmetric or skew Hermitian eigenvalue problems occur in mechanical and quantum mechanical problems. They can be treated efficiently, structure preserving and numerically stable, by QR and Jacobi methods, e.g., [G4], [P4], [P5].

J-skew symmetric or J-skew Hermitian eigenvalue problems which are furthermore Hermitian arise in quantum mechanical problems with time reversal symmetry [D2].

The numerical solution of these problems is simpler than the numerical solution of the nonskew problems. Zeros in these structures can be exploited to split the problem into smaller ones.

Near-best uniform rational approximation [A1], [G7] requires the computation of the symmetric singular value decomposition [B11] which leads to a J-symmetric and symmetric eigenvalue problem. Other problems of this type occur in linear response theory, Hartree–Fock wave functions [O1], and some mechanical problems [L1].

In [L1], Example 6 (with $r = s = 1$) is J-symmetric and orthogonal. Solutions of linear-quadratic control problems with the matrix sign function method [A3], [A5], [B1], [B2], [B17], [G2], [R1] transform a real J-symmetric invariant subspace problem into a J-symmetric and symplectic invariant subspace problem.

The paper is divided into six sections. Section 2 reviews the algebraic structures under consideration, discusses elementary structure preserving similarity transformations, and lists some structured factorizations. Section 3 summarizes some basic numerical tools including reduction to Hessenberg-like forms, QR-like methods, and simultaneous diagonalization. Section 3 also comments on the rounding error analysis and numerical stability of these numerical tools. Section 4 gives a chart of structure preserving algorithms for complex matrices. Each position in the chart corresponds to an algebraic eigenvalue problem with two special structures. Numerically stable methods are outlined for most cases. Section 5 gives a chart of structure preserving algorithms for real matrices. Section 6 is a list of conclusions and open problems.

2. Algebraic structures. In this section we briefly review the basic algebraic structures and properties of the matrix classes, for which we will describe numerically stable, structure preserving methods. Basically we have a Lie algebra or Lie group structure (cf. [B3]).

DEFINITION 2.1. Let A be an $m \times m$ matrix and let $(1/\lambda) \in \mathbb{C}$ be not in the spectrum of A. Then $\mathscr{C}(A) = (I + \lambda A)(I - \lambda A)^{-1}$ is called the *Cayley transformation* of A and $\mathscr{C}^*(A) = (I + \lambda A)(I - \bar{\lambda} A)^{-1}$ is called the *conjugate Cayley transformation* of A [G1].

We have the following well-known relations [G1].

THEOREM 2.2.

—*If* $A \in \mathbb{R}^{m,m}$ *is* J*-symmetric, then* $\mathscr{C}(A)$ *is symplectic.*

—*If* $A \in \mathbb{R}^{m,m}$ *is skew symmetric, then* $\mathscr{C}(A)$ *is orthogonal.*

—*If* $A \in \mathbb{C}^{m,m}$ *is* J*-Hermitian, then* $\mathscr{C}^*(A)$ *is conjugate symplectic.*

—*If* $A \in \mathbb{C}^{m,m}$ *is skew Hermitian, then* $\mathscr{C}^*(A)$ *is unitary.*

Thus, we have the Lie Algebra, Lie group table (Table 2.3).

Many of these structures impose special structures on the eigenvalues and eigenvectors. For example, if λ is an eigenvalue of a J-Hermitian matrix H, then $-\bar{\lambda}$ is also

TABLE 2.3

Theorem 2.2 relates the algebras in the left-hand column with the corresponding groups in the right-hand column.

Lie algebra, product $[A, B] = AB - BA$	Lie group, matrix multiplication
J-symmetric	symplectic
skew symmetric	orthogonal
J-Hermitian (Hamiltonian)	conjugate symplectic
skew Hermitian	unitary

The Hermitian and J-skew Hermitian matrices are obtained from the skew Hermitian and J-Hermitian classes by multiplication with $i = \sqrt{-1}$.

an eigenvalue. If H is diagonalizable, then it admits a symplectic diagonalizing similarity transform [B8], [B14], [L4]. Other eigenstructures are summarized in Table 2.4.

Some of our numerical tools are based on the elementary observation that similarity transformations of Lie algebra elements (and their relatives by a factor i) by members of the corresponding Lie group are algebra endomorphisms and thus the algebra structure is preserved. Similarity transformations of Lie group elements by other Lie group elements are group endomorphisms and thus the group structure is preserved (cf. [B3]).

To promote numerical stability we consider only unitary similarity transformations, or in the real case, orthogonal similarity transforms. Almost all the transformations that will be used are unitary symplectic, real orthogonal symplectic matrices, or unitary conjugate symplectic matrices. The last two classes are contained in the *centralizer* of J,

$$\mathscr{C}_{2n}(J) = \{A \in \mathbb{C}^{2n,2n} \mid AJ = JA\} = \left\{ \begin{bmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{bmatrix} \middle| A_1, A_2 \in \mathbb{C}^{n,n} \right\},$$

while the first is in

$$\hat{\mathscr{C}}_{2n}(J) = \{A \in \mathbb{C}^{2n,2n} \mid AJ = J\bar{A}\}.$$

TABLE 2.4

Class of matrices	Eigenvalues	Eigenvectors
(1) J-symmetric	pairs $\lambda, -\lambda$	$Ax = \lambda x \Rightarrow (Jx)^T A = -\lambda (Jx)^T$
(2a) real skew symmetric	purely imaginary	$Ax = \lambda x \Rightarrow x^T A = -\lambda x^T$
(2b) complex skew symmetric	pairs $\lambda, -\lambda$	$Ax = \lambda x \Rightarrow x^T A = -\lambda x^T$
(3) J-skew symmetric	double eigenvalues	$Ax = \lambda x \Rightarrow (Jx)^T A = \lambda (Jx)^T$
(4a) real symmetric	real eigenvalues	$Ax = \lambda x \Rightarrow x^T A = \lambda x^T$
(4b) complex symmetric	essentially arbitrary	$Ax = \lambda x \Rightarrow x^T A = \lambda x^T$
(5) symplectic	pairs $\lambda, \frac{1}{\lambda}$	$Ax = \lambda x \Rightarrow (Jx)^T A = \frac{1}{\lambda} (Jx)^T$
(6a) real orthogonal	$\lvert\lambda\rvert = 1$	$Ax = \lambda x \Rightarrow x^T A = \frac{1}{\lambda} x^T$
(6b) complex orthogonal	pairs $\lambda, \frac{1}{\lambda}$	$Ax = \lambda x \Rightarrow x^T A = \frac{1}{\lambda} x^T$
(7) J-Hermitian	pairs $\lambda, -\bar{\lambda}$	$Ax = \lambda x \Rightarrow (Jx)^* A = -\bar{\lambda} (Jx)^*$
(8) skew Hermitian	purely imaginary	$Ax = \lambda x \Rightarrow x^* A = -\bar{\lambda} x^*$
(9) J-skew Hermitian	pairs $\lambda, \bar{\lambda}$	$Ax = \lambda x \Rightarrow (Jx)^* A = \bar{\lambda} (Jx)^*$
(10) Hermitian	real eigenvalues	$Ax = \lambda x \Rightarrow x^* A = \lambda x^*$
(11) conjugate symplectic	pairs $\lambda, \frac{1}{\bar{\lambda}}$	$Ax = \lambda x \Rightarrow (Jx)^* A = \frac{1}{\bar{\lambda}} (Jx)^*$
(12) unitary	$\lvert\lambda\rvert = 1$	$Ax = \lambda x \Rightarrow x^* A = \frac{1}{\bar{\lambda}} x^*$

The algebraic structure of the centralizer and the anticentralizer (defined below) is clarified by a similarity transformation with the unitary matrix

$$X := \frac{1}{\sqrt{2}} \begin{bmatrix} I & I \\ iI & -iI \end{bmatrix},$$

which diagonalizes J.

LEMMA 2.5. *Let X be the matrix defined above. If*

$$A = \begin{bmatrix} A_1 & -A_2 \\ A_2 & A_1 \end{bmatrix} \in \mathbb{C}_{2n}(J),$$

where $A_1, A_2 \in \mathbb{C}^{n,n}$, then

$$X^*AX = \begin{bmatrix} A_1 + iA_2 & 0 \\ 0 & A_1 - iA_2 \end{bmatrix}.$$

If A is in the anticentralizer of J,

$$\mathcal{C}^a_{2n}(J) = \{ A \in \mathbb{C}^{2n,2n} \mid AJ = -JA \} = \left\{ \begin{bmatrix} A_1 & A_2 \\ A_2 & -A_1 \end{bmatrix} \middle| A_1, A_2 \in \mathbb{C}^{n,n} \right\},$$

then

$$X^*AX = \begin{bmatrix} 0 & A_1 - iA_2 \\ A_1 + iA_2 & 0 \end{bmatrix}. \qquad \square$$

Many of the similarity transformations we use are generated as products of easily constructed elementary, unitary symplectic matrices. These are Householder symplectic and Givens symplectic matrices [P1]. Householder symplectic matrices are of the form

$$U = \begin{bmatrix} U_1 & O \\ O & \overline{U_1} \end{bmatrix}, \tag{2.6}$$

where for some $w \in \mathbb{C}^n$,

$$U_1 = I - \frac{ww^*}{2w^*w}. \tag{2.7}$$

Givens simplectic matrices are complex rotations in planes $(k, n + k)$, i.e.,

$$U = \begin{bmatrix} C & \bar{S} \\ -S & \bar{C} \end{bmatrix}, \tag{2.8}$$

where

$$C = I + (c - 1)e_k e_k^T, \qquad S = s e_k e_k^T, \tag{2.9}$$

$c, s \in \mathbb{C}$ with $|c|^2 + |s|^2 = 1$ and for $k \in \{1, \cdots, n\}$ e_k is the kth unit vector.

Unitary conjugate symplectic matrices are products of Householder conjugate symplectic matrices and Givens conjugate symplectic matrices. Householder conjugate symplectic matrices are of the form

$$U = \begin{bmatrix} U_1 & 0 \\ 0 & U_1 \end{bmatrix}, \tag{2.10}$$

with U_1 as in (2.7) and Givens conjugate symplectic matrices are of the form

$$U=\begin{bmatrix} C & S \\ -S & C \end{bmatrix} \tag{2.11}$$

with c, s as in (2.9) and furthermore, $\bar{c}s \in \mathbb{R}$ [P1].

The real symmetric, Hermitian, the real skew symmetric, skew Hermitian, and the real orthogonal and unitary matrices are normal and hence diagonalizable.

The J-symmetric, J-Hermitian, symplectic, J-symplectic, conjugate symplectic, J-skew symmetric, and J-skew Hermitian matrices are not necessarily diagonalizable, but have specially structured Schur-like and Jordan-like forms [B14], [L4], [P1]. We use only the Schur-like forms described in the following table.

THEOREM 2.12 (Table of Special Schur Forms). *If $A \in \mathbb{C}^{2n,2n}$ has the structure in column* 1 *and the restriction in column* 2, *then there is a unitary matrix $U \in \mathbb{C}^{2n,2n}$ with the structure in column* 3 *such that $T = U^{-1}AU$ has the structure in column* 4.

TABLE OF SPECIAL SCHUR FORMS

Structure of A	Restriction on A	Structure of U	Schur-like form
(i) J-Hermitian	no eigenvalues on the imaginary axis	unitary conjugate symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^* \end{bmatrix}$, T_1 upper triangular, T_2 Hermitian
(ii) conjugate symplectic	no eigenvalues on the unit circle	unitary conjugate symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & T_1^{-*} \end{bmatrix}$, T_1 upper triangular
(iii) J-skew Hermitian	no eigenvalues on the real axis	unitary conjugate symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & T_1^* \end{bmatrix}$, T_1 upper triangular, T_2 skew Hermitian
(iv) J-symmetric	no eigenvalues on the imaginary axis	unitary symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^T \end{bmatrix}$, T_1 upper triangular, T_2 symmetric
(v) symplectic	no eigenvalues on the unit circle	unitary symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^{-T} \end{bmatrix}$, T_1 upper triangular
(vi) J-skew symmetric	no real eigenvalues	unitary symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & T_1^T \end{bmatrix}$, T_1 upper triangular, T_2 skewsymmetric

Proof. (i), (iv), (v), and (vi) follow from minor modifications of the proof given in [P1].

(ii) Let $\check{\mathscr{C}}^*$ be the inverse transformation of the conjugate Cayley transformation $\mathscr{C}^*$, then $B = \check{\mathscr{C}}^*(A)$ is J-Hermitian. By (i) there exists $U \in \mathbb{C}^{2n,2n}$ unitary conjugate symplectic such that

$$U^*BU = \begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^* \end{bmatrix}.$$

It follows that if $1/\lambda$ is not an eigenvalue of A, then

$$\mathscr{C}^*(U^*BU) = U^*\mathscr{C}^*(B)U = U^*AU = \mathscr{C}^*\left(\begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^* \end{bmatrix}\right)$$

$$= \begin{bmatrix} (I+\lambda T_1)(I-\bar{\lambda}T_1)^{-1} & \tilde{T}_2 \\ 0 & (I-\lambda T_1^*)(I+\bar{\lambda}T_1^*)^{-1} \end{bmatrix}.$$

(iii) Let $B = iA$, then B is J-Hermitian and, by (i), there exists $U \in \mathbb{C}^{2n,2n}$ unitary and conjugate symplectic such that

$$U^*BU = \begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^* \end{bmatrix}.$$

Clearly then,

$$U^*AU = \begin{bmatrix} -iT_1 & -iT_2 \\ 0 & iT_1^* \end{bmatrix} = \begin{bmatrix} -iT_1 & \tilde{T}_2 \\ 0 & (-iT_1)^* \end{bmatrix}. \qquad \square$$

In the real case, we use real orthogonal matrices to transform to real Schur-like forms. In the following table, a *quasi-triangular* matrix is a block upper triangular matrix with size 1×1 or 2×2 diagonal blocks.

THEOREM 2.13 (Table of Special Real Schur-Like Forms). *If $A \in \mathbb{R}^{2n,2n}$ has the structure in column* 1 *and the restriction in column* 2, *then there is an orthogonal matrix $U \in \mathbb{R}^{2n,2n}$ with the structure in column* 3 *such that $T = U^TAU$ has the structure of column* 4.

TABLE OF SPECIAL REAL SCHUR-LIKE FORMS

Structure of A	Restriction on A	Structure of U	Real Schur-like form
(i) J-symmetric	no nonzero eigenvalues on imaginary axis	orthogonal and symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^T \end{bmatrix}$, T_1 upper quasi-triangular, T_2 symmetric
(ii) symplectic	no eigenvalues on the unit circle except possibly $\lambda = 1$	orthogonal and symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & T_1^{-1} \end{bmatrix}$, T_1 upper quasi-triangular
(iii) J-skew symmetric	no nonzero eigenvalues on the real axis	orthogonal and symplectic	$\begin{bmatrix} T_1 & T_2 \\ 0 & T_1^T \end{bmatrix}$, T_1 upper quasi-triangular, T_2 skew symmetric

Proof. The proof follows from minor modifications of the proof given in [P1]. □

Another special Schur form is the *Murnaghan canonical form* [M5] of real skew symmetric matrices.

THEOREM 2.14. *Let $A \in \mathbb{R}^{n,n}$ be a rank $2k$ skew symmetric matrix; then there exists an orthogonal $U \in \mathbb{R}^{n,n}$ such that*

$$U^T A U = \begin{bmatrix} \overbrace{0}^{k} & \overbrace{D_1}^{k} & \overbrace{0}^{n-2k} \\ -D_1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{matrix} \} k \\ \} k \\ \} n-2k \end{matrix},$$

where $D_1 \in \mathbb{R}^{k,k}$ is diagonal and nonsingular.

Proof. See, e.g., [M5] and [P4]. □

The QR-like algorithms described in this paper transform a matrix to a special Schur-like form. They need initial reductions to the special Hessenberg-like forms of the following theorems.

THEOREM 2.15. *Let $M \in \mathbb{C}^{2n,2n}$. Let $s \in \mathbb{C}^{2n}$ be such that $s^*s = 1$, $s^*Js = 0$.*

(i) *There exists $S \in \mathbb{C}^{2n2,n}$ unitary conjugate symplectic with first column vector s such that*

$$S^*MS = H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} = \begin{bmatrix} \diagdown & \square \\ \diagdown & \square \end{bmatrix},$$

where $H_{11}, H_{21} \in \mathbb{C}^{n,n}$ are upper Hessenberg matrices and $H_{12}, H_{22} \in \mathbb{C}^{n,n}$. The elements on the subdiagonal of H_{11} are real and nonnegative. The elements on the subdiagonal of H_{21} lie on the imaginary axis and the kth subdiagonal element of H_{11} is not smaller than the modulus of the kth subdiagonal element of H_{21}. S, H can be computed with a finite number of operations.

(ii) *If for all $k \in \{1, \cdots, n-1\}$ the kth subdiagonal element of H_{11} is strictly greater than the modulus of the kth subdiagonal element of H_{21}, then this reduction is uniquely determined for any choice of s.*

(iii) *If M and s are real, then S and H can be chosen to be real and H_{21} will then be upper triangular, i.e.,*

$$S^T M S = \begin{bmatrix} \diagdown & \square \\ \diagup & \square \end{bmatrix}.$$

Proof. See [B6], [B7], or [B8, p. 97]. □

As a corollary we obtain a generalization of results of [P1] and [V2].

COROLLARY 2.16. *Let $M \in \mathbb{C}^{2n,2n}$ and let*

$$S^* M S = H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

be the reduction from Theorem 2.15(i) with first column s of S.

(i) *If M is J-Hermitian, then $H_{22} = -H_{11}^*$, H_{12} is Hermitian and H_{21} is an Hermitian tridiagonal matrix with purely imaginary subdiagonal entries. If M, S are real, then S, H can be chosen real. Then H_{21} is diagonal.*

(ii) *If M is J-skew Hermitian, then $H_{22} = H_{11}^*$, H_{12} is skew Hermitian and H_{21} is a purely imaginary skew Hermitian tridiagonal matrix. If M, S are real, then S, H can be chosen real. Then, H_{21} vanishes.*

Replacing conjugate symplectic by symplectic, Theorem 2.15 becomes Theorem 2.17.

THEOREM 2.17. (i) *Let* $M \in \mathbb{C}^{2n,2n}$ *and let* $s \in \mathbb{C}^{2n}$ *such that* $s^*s = 1$. *There exists a unitary symplectic* $S \in \mathbb{C}^{2n,2n}$ *such that*

$$H = S^*MS = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix} = \begin{bmatrix} \diagdown & \square \\ \diagdown & \square \end{bmatrix},$$

where H_{11} *is an* $n \times n$ *upper Hessenberg matrix with real nonnegative subdiagonal entries,* H_{21} *is upper triangular, and* H_{12}, $H_{12} \in \mathbb{C}^{n,n}$.

(ii) *If none of the diagonal elements of* H_{11} *vanishes, then this reduction is uniquely determined for any first column* $s \in \mathbb{C}^{2n}$.

Proof. See [B6], [B7], or [B8, p. 139]. □

COROLLARY 2.18. *Let* $M \in \mathbb{C}^{2n,2n}$ *and let*

$$S^*MS = H = \begin{bmatrix} H_{11} & H_{12} \\ H_{21} & H_{22} \end{bmatrix}$$

be the reduction from Theorem 2.17(i) *with first column* s *of* S.

(i) *If* M *is J-symmetric, then*

$$H_{22} = -H_{11}^T, \qquad H_{21}^T = H_{21}, \qquad H_{12}^T = H_{12}.$$

(ii) *If* M *is J-skew symmetric, then* $H_{22} = H_{11}^T$, $H_{21} = 0$, $H_{12}^T = -H_{12}$. *If, in addition,* M *is Hermitian, then* $H_{21} = H_{12} = 0$ *and* $H_{11} = H_{22}$ *is real symmetric tridiagonal.*

We close this section with the singular value decomposition of a matrix and the Takagi singular value decomposition of a complex symmetric matrix, which are important for our algorithms.

THEOREM 2.19. (i) *If* $A \in \mathbb{C}^{n,n}$, *then there exists unitary matrices* U, $V \in \mathbb{C}^{n,n}$, *such that* $UAV = \Sigma$ *is diagonal, and if* $\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_n)$, *then* $0 \leqq \sigma_n \leqq \sigma_{n-1} \leqq \cdots \leqq \sigma_1$.

(ii) *If* $A \in \mathbb{C}^{n,n}$ *is symmetric, then there exists a unitary matrix* Q *such that* $A = Q\Sigma Q^T$, *where* $\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_n)$ *with nonnegative* $\sigma_1 \geqq \sigma_2 \geqq \cdots \geqq \sigma_n \geqq 0$.

Proof. The proof of (i) is well known (see, e.g., [G4]).

(ii) See [T1], [T2], [S2], and [H4] for the proof. □

3. Basic algorithms. In this section we describe the basic algorithms from which special structured methods are constructed. We assume knowledge of the Francis QR iteration [F2], [F3], the Jacobi method, and Householder and Givens matrices [G4]. In the following, the construction, storage, and matrix multiplication by rotations and reflections are assumed to be carried out as in [G4, Chap. 3].

3.1. QR-like methods. We follow the outline of the QR iteration to construct different algorithms for the computation of Schur-like forms for matrices with special structure.

We call a matrix $V \in \mathbb{C}^{m,m}$ *structure preserving* if for any $A \in \mathbb{C}^{n,n}$ with a special structure, $V^{-1}AV$ also has that structure.

ALGORITHM 3.1 Framework of QR-like algorithm.

Input: $A \in \mathbb{K}^{m,m}$ ($\mathbb{K} = \mathbb{R}$ or $\mathbb{K} = \mathbb{C}$) with one of the structures in Definition 1.1.

Output: A unitary matrix U such that UAU^* has the Schur form of Theorem 2.12 or 2.13.

Step 1: Perform an initial reduction with a unitary (orthogonal) transformation U that preserves the structure, $A_0 = UAU^*$, such that A_0 has a suitable, Hessenberg-like condensed form.

Step 2:

FOR $k = 1, 2, \cdots$

Select a holomorphic shift function f (to be described later).

Set $a = f(A_{k-1})e_1$. Let V_k be unitary (orthogonal) and structure preserving, such that $V_k a = \alpha e_1$, $\alpha \in \mathbb{K}$, $\|a\| = \alpha$.

Set $\tilde{A}_k = V_k A_k V_k^*$, $U := V_k U$.

Find W_k, $W_k^* W_k = I$, such that similarity transformation with W_k preserves the structure of $\tilde{A}_k$, $W_k e_1 = e_1$ and $W_k A_k W_k^*$ is again in the Hessenberg-like condensed form.

Let $A_k = W_k \tilde{A}_k W_k^*$, $U := W_k U$.

END FOR

QR-like algorithms essentially differ in the choice of the shift function f and the structure preserving transformations.

The usual shift strategies in the QR algorithm are to choose the single shift

$$f(A) = A - \lambda I \tag{3.2}$$

for λ an approximation to an eigenvalue or the double shift

$$f(A) = (A - \lambda I)(A - \mu I) \tag{3.3}$$

for λ, μ approximations to a pair of eigenvalues. In the real case, if $\lambda \notin \mathbb{R}$, then we choose $\mu = \bar{\lambda}$ to avoid complex arithmetic. For the Hamiltonian QR algorithm in [B16] and the symplectic QR algorithms of [M1], the so-called "Cayley shifts" are used, i.e.,

$$f(A) = (A - \lambda I)(A + \bar{\lambda} I)^{-1} \tag{3.4}$$

or again in the real case to avoid complex arithmetic

$$f(A) = (A - \lambda I)(A - \bar{\lambda} I)(A + \lambda I)^{-1}(A + \bar{\lambda} I)^{-1}. \tag{3.5}$$

For complex symmetric matrices, the computation of the Takagi singular value decomposition of Theorem 2.19 suggested in [B11] uses essentially the framework of Algorithm 3.1, with the modification that the transformations are $A_0 = UAU^T$, $\tilde{A}_k = V_k A_k V_k^T$, etc., instead of similarity transformations. The initial reduction is to complex symmetric tridiagonal form and the shift function is

$$f(A) = A^2 - \lambda I \tag{3.6}$$

for some suitably chosen real λ.

The quaternion QR algorithm [B9] also fits the framework of Algorithm 3.1. It works with matrices of the form

$$M = \begin{bmatrix} A & -\bar{B} \\ B & \bar{A} \end{bmatrix},$$

where A, $B \in \mathbb{C}^{n,n}$. These are complex representations of $n \times n$ quaternion matrices [B9], [R2]. It uses the shift function

$$f\left(\begin{bmatrix} A & -\bar{B} \\ B & \bar{A} \end{bmatrix}\right) = \left(\begin{bmatrix} A & -\bar{B} \\ B & \bar{A} \end{bmatrix} - \lambda \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}\right)\left(\begin{bmatrix} A & -\bar{B} \\ B & \bar{A} \end{bmatrix} - \bar{\lambda} \begin{bmatrix} I & 0 \\ 0 & I \end{bmatrix}\right). \tag{3.7}$$

An antiquaternion matrix is simply i times a quaternion matrix, so one algorithm serves for both.

The reduction to the condensed Hessenberg-like form of Theorems 2.15 and 2.17 can be performed by the following algorithm due to [P1] and [V2].

ALGORITHM 3.8 [B7], [B8]. Reduction to condensed form
Input: $M_0 \in \mathbb{C}^{2n,2n}$.
Output: A unitary, symplectic matrix $U \in \mathbb{C}^{2n,2n}$ such that

$$U^*M_0U = H \triangleq \begin{bmatrix} \ddots & \square \\ & \square \end{bmatrix}.$$

Set $U = I$.
FOR $k = 1, \cdots, n-1$
Set $m = \begin{bmatrix} m_1 \\ m_2 \end{bmatrix} = M_{k-1}e_k$, where $m_1, m_2 \in \mathbb{C}^n$.
Construct an $n \times n$ complex Householder transformation Q_1 [G4, p. 40], such that $\bar{Q}_1 m_2$ has zeros in positions $k+2, \cdots, n$.
Let

$$\tilde{m} = \begin{bmatrix} Q_1 & 0 \\ 0 & \bar{Q}_1 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \end{bmatrix}.$$

Construct a Givens symplectic matrix $P = \begin{bmatrix} C & \bar{S} \\ -S & \bar{C} \end{bmatrix}$, where $c = (\tilde{m}_{k+1})/w$, $s = \tilde{m}_{n+k+1}/w$, and $w = \sqrt{(\tilde{m}_{k+1})^2 + (\tilde{m}_{n+k+1})^2}$.
(If $\tilde{m}_{k+1} = \tilde{m}_{n+k+1} = 0$, then set $c = 1$, $s = 0$.) In $\hat{m} = \begin{bmatrix} \hat{m}_1 \\ \hat{m}_2 \end{bmatrix} := P\tilde{m}$ the $(n+k+1)$st element is eliminated.
Construct a complex $n \times n$ Householder transformation Q_2 that eliminates the entries in positions $k+2, \cdots, n$ of $\hat{m}_1$.
Let

$$M_k = \begin{bmatrix} Q_2 & 0 \\ 0 & \bar{Q}_2 \end{bmatrix} \begin{bmatrix} C & \bar{S} \\ -S & \bar{C} \end{bmatrix} \begin{bmatrix} Q_1 & 0 \\ 0 & \bar{Q}_1 \end{bmatrix} \cdot M_{k-1} \begin{bmatrix} Q_1^* & 0 \\ 0 & \bar{Q}_1^* \end{bmatrix} \begin{bmatrix} \bar{C} & -\bar{S} \\ S & C \end{bmatrix} \begin{bmatrix} Q_2^* & 0 \\ 0 & \bar{Q}_2^* \end{bmatrix}$$

and

$$U = U \begin{bmatrix} Q_1^* & 0 \\ 0 & \bar{Q}_1^* \end{bmatrix} \begin{bmatrix} \bar{C} & -\bar{S} \\ S & C \end{bmatrix} \begin{bmatrix} Q_2^* & 0 \\ 0 & \bar{Q}_2^* \end{bmatrix}$$

END FOR

Any practical implementation of Algorithm 3.8 should store and apply the rotations and reflections in the efficient manner described in [G4], [P1], [P5], and [W2].

For the special case that the (2, 1) block of M_0 has rank 1, Algorithm 3.8 can be modified to yield even more zeros in the condensed form [B16], [M1].

Remark 3.9. There is a similar algorithm for the reduction of M to the form

$$\begin{bmatrix} \ddots & \square \\ \ddots & \square \end{bmatrix}$$

using unitary conjugate symplectic transformations [B7], [B8]. If $M_0 \in \mathbb{R}^{2n,2n}$, then U and H are real. If M_0 has further structure, then the form of H will often be more condensed.

For matrices which have two of the properties in Table 2.4 we give the corresponding condensed forms in Tables 3.10(a)–(c) below. (See also Table 4.1.)

TABLE 3.10(a)
Condensed form under unitary conjugate symplectic similarity transformation (Remark 3.9).

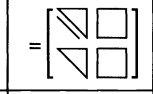
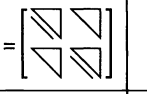
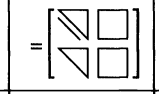
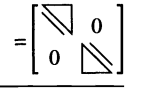
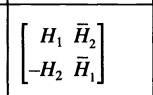
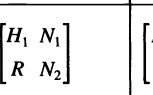
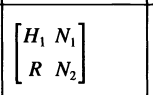
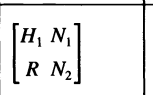
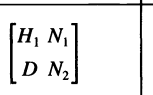
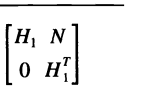
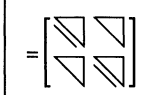
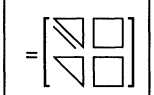
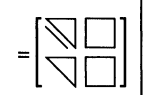
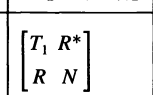
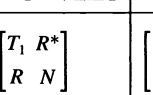
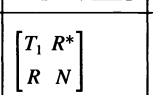
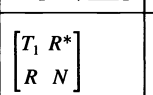
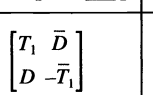
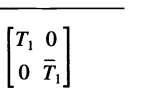
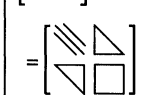
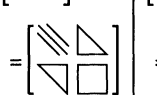
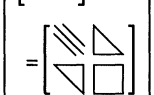
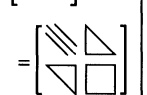
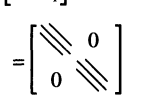
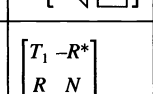
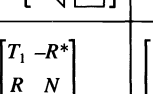
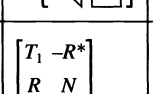
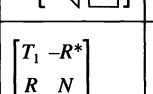
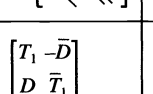
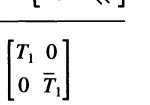
	$QQ^* = I$	$QJQ^* = J$	$Q = Q^*$	$Q = -Q^*$	$JQ = (JQ)^*$	$JQ = -(JQ)^*$
$QQ^* = I$	$\begin{bmatrix} H_1 & N_1 \\ H_2 & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & H_2 \\ -H_2 & H_1 \end{bmatrix}$	$\begin{bmatrix} T_1 & H_2^* \\ H_2 & N \end{bmatrix}$	$\begin{bmatrix} T_1 & -H_2^* \\ H_2 & N \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ T_1 & -H_1^* \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ T_1 & H_1^* \end{bmatrix}$
$QJQ^* = J$		$\begin{bmatrix} H_1 & N_1 \\ H_2 & N_2 \end{bmatrix}$	$\begin{bmatrix} T_1 & H_2^* \\ H_2 & N \end{bmatrix}$	$\begin{bmatrix} T_1 & H_2^* \\ -H_2 & N \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ T_1 & -H_1^* \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ T_1 & H_1^* \end{bmatrix}$
$Q = Q^*$			$\begin{bmatrix} T_1 & H_2^* \\ H_2 & N \end{bmatrix}$	0	$\begin{bmatrix} T_1 & T_2 \\ T_2 & -T_1 \end{bmatrix}$	$\begin{bmatrix} T_1 & T_2 \\ -T_2 & T_1 \end{bmatrix}$
$Q = -Q^*$				$\begin{bmatrix} T_1 & -H_2^* \\ -H_2 & N \end{bmatrix}$	$\begin{bmatrix} T_1 & T_2 \\ -T_2 & T_1 \end{bmatrix}$	$\begin{bmatrix} T_1 & T_2 \\ T_2 & -T_1 \end{bmatrix}$
$JQ = (JQ)^*$					$\begin{bmatrix} H_1 & N_1 \\ T_1 & -H_1^* \end{bmatrix}$	0
$JQ = -(JQ)^*$						$\begin{bmatrix} H_1 & N_1 \\ T_1 & -H_1^* \end{bmatrix}$

Let $M \in \mathbb{C}^{2n,2n}$. If M has the structure at the head of column i and the structure at the head of row j in the tables below, then U^*MU has the condensed form in position (i, j). A zero indicates that the zero matrix is the only matrix with both structures.

In the table N, N_1, N_2, H_1, $H_2 \in \mathbb{C}^{n,n}$; H_1, H_2 are upper Hessenberg matrices, N is Hermitian, skew Hermitian, symmetric, or skew symmetric; T_1, $T_2 \in \mathbb{C}^{n,n}$ are tridiagonal matrices; $D \in \mathbb{C}^{n,n}$ is a diagonal matrix; and $R \in \mathbb{C}^{n,n}$ is an upper triangular matrix.

In Table 3.10(a) the transformation U is unitary and conjugate symplectic. The extra zeros result from symmetry or unitarity or because the matrix is in the centralizer of J, $\mathscr{C}_{2n}(J)$, or in the anticentralizer of J, $\mathscr{C}^a_{2n}(J)$. Note that Tables 3.10(a) and 3.10(c) are symmetric, but Table 3.10(b) is not.

TABLE 3.10(b)
Condensed form under unitary symplectic similarity (Algorithm 3.8).

	$Q^TQ = I$	$Q^TJQ = J$	$Q = Q^T$	$Q = -Q^T$	$JQ = (JQ)^T$	$JQ = -(JQ)^T$
$QQ^* = I$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & \bar{H}_2 \\ -H_2 & \bar{H}_1 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix}$ 	$\begin{bmatrix} H_1 & 0 \\ 0 & -H_1^T \end{bmatrix}$ 
$QJQ^* = J$	$\begin{bmatrix} H_1 & \bar{H}_2 \\ -H_2 & \bar{H}_1 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ D & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix}$ 
$Q = Q^*$	$\begin{bmatrix} T_1 & R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & \bar{D} \\ D & -\bar{T}_1 \end{bmatrix}$ 	$\begin{bmatrix} T_1 & 0 \\ 0 & \bar{T}_1 \end{bmatrix}$ 
$Q = -Q^*$	$\begin{bmatrix} T_1 & -R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & -R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & -R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & -R^* \\ R & N \end{bmatrix}$ 	$\begin{bmatrix} T_1 & -\bar{D} \\ D & \bar{T}_1 \end{bmatrix}$ 	$\begin{bmatrix} T_1 & 0 \\ 0 & \bar{T}_1 \end{bmatrix}$ 
$JQ = (JQ)^*$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & \bar{R} \\ R & -\bar{H}_1 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & \bar{R} \\ -R & \bar{H}_1 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix}$ 
$JQ = -(JQ)^*$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & \bar{R} \\ -R & \bar{H}_1 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & \bar{R} \\ R & -\bar{H}_1 \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix}$ 	$\begin{bmatrix} H_1 & N \\ 0 & H_2^T \end{bmatrix}$ 

In Table 3.10(b) the transformation U is unitary and symplectic. The extra zeros in the condensed forms result from symmetry or unitarity or because the matrix is an $n \times n$ quaternion or antiquaternion matrix.

In Table 3.10(c) the transformation U is unitary and symplectic with the exception of positions (1, 2) and (4, 5), where U is unitary and conjugate symplectic. In these two cases the matrices are in $\mathscr{C}_{2n}(J)$.

Other special reductions include the well-known tridiagonalization procedures for Hermitian and skew Hermitian matrices [P5], [G4] and the corresponding real versions.

3.2. Jacobi methods. Jacobi-like methods are methods which are variations or generalizations of the Jacobi method for symmetric or Hermitian matrices [J1], [B5], [G4],

TABLE 3.10(c)

Condensed form under unitary symplectic or unitary conjugate symplectic similarity (Algorithm 3.8, Remark 3.9).

	$Q^TQ=I$	$Q^TJQ=J$	$Q=Q^T$	$Q=-Q^T$	$JQ=(JQ)^T$	$JQ=-(JQ)^T$
$Q^TQ=I$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N_1 \\ R & H_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \diagdown & \blacksquare \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ 0 & \blacksquare \end{bmatrix}$
$Q^TJQ=J$		$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \diagdown & \blacksquare \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ 0 & \blacksquare \end{bmatrix}$
$Q=Q^T$			$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	0	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \diagdown & \blacksquare \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ 0 & \blacksquare \end{bmatrix}$
$Q=-Q^T$				$\begin{bmatrix} H_1 & N_1 \\ R & N_2 \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \blacksquare & \square \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \diagdown & \blacksquare \end{bmatrix}$	$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ 0 & \blacksquare \end{bmatrix}$
$JQ=(JQ)^*$					$\begin{bmatrix} H_1 & N \\ D & -H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ \diagdown & \blacksquare \end{bmatrix}$	0
$JQ=-(JQ)^T$						$\begin{bmatrix} H_1 & N \\ 0 & H_1^T \end{bmatrix} = \begin{bmatrix} \blacksquare & \square \\ 0 & \blacksquare \end{bmatrix}$

[S6]. They have gained importance in recent years because they have inherent parallelism. Apart from the Hermitian/symmetric Jacobi method we will use the Jacobi method for real skew-symmetric matrices from [P4].

For our purposes Jacobi-like methods for $2n \times 2n$ Hermitian or skew Hermitian matrices can be performed using the four basic types of transformation matrices: Two double rotations of the form

$$\tilde{U}_H = \begin{bmatrix} U_1 & 0 \\ 0 & \bar{U}_1 \end{bmatrix}, \qquad U_H = \begin{bmatrix} U_1 & 0 \\ 0 & U_1 \end{bmatrix},$$

where $U_1 \in \mathbb{R}^{n\times n}$ is a Givens rotation and two matrices of the form

$$\tilde{U}_G = \begin{bmatrix} C & \bar{S} \\ -S & \bar{C} \end{bmatrix}, \qquad U_G = \begin{bmatrix} C & S \\ -S & C \end{bmatrix},$$

with $C = I + (c-1)e_j e_j^T$, $S = s e_j e_j^T$, $|c|^2 + |s|^2 = 1$, and for U_G the further requirement $\bar{c}s \in \mathbb{R}$.

So, $\tilde{U}_H$, $\tilde{U}_G$ are unitary symplectic, and U_H, U_G are unitary conjugate symplectic. Similarity transformations with U_H and U_G preserve J-Hermitian, J-skew Hermitian, and conjugate symplectic structures. Similarity transformations with $\tilde{U}_H$ and $\tilde{U}_G$ preserve J-symmetric, J-skew symmetric, symplectic, and quaternion structure. Jacobi-like algorithms for eigenvalue problems with symmetry and a symplectic structure are currently developed by the authors. To distinguish these methods from the usual Jacobi methods, we call them *symplectic Jacobi* methods.

Other Jacobi-like methods include the Jacobi method for normal matrices [G3], the Jacobi methods for nonsymmetric matrices of [E1] and [E2] or [S7], the Hamiltonian Jacobi method [B18] for real J-symmetric or J-Hermitian matrices, and the Kogbetliantz algorithm [K2], [P2] for computing singular value decompositions.

3.3. Simultaneous diagonalization. Many of the algorithms outlined below require the simultaneous diagonalization of commuting pairs of normal matrices. Usually there is a special structure in addition to normality. Examples include commuting pairs of real symmetric matrices, a skew Hermitian matrix and an Hermitian matrix, and a pair of real skew symmetric matrices. In this section we point out some of the difficulties associated with simultaneous diagonalization and propose a family of Jacobi-like algorithms for simultaneously diagonalizing commuting pairs of normal matrices.

For ease of explication, we will use the special case of a commuting pair of symmetric matrices $A = A^T \in \mathbb{R}^{n,n}$ and $B = B^T \in \mathbb{R}^{n,n}$ to illustrate the problems and algorithms. Afterwards, we will summarize the modifications required by other kinds of commuting pairs. A more detailed study of simultaneous diagonalization will appear elsewhere [B10].

To be viable, a simultaneous diagonalization algorithm must work with both members of the pair simultaneously. To see how simultaneous diagonalization algorithms that violate this principle can fail, consider the family of *diagonalize-one-then-diagonalize-the-other* methods. These are the methods suggested by the classic proof that commuting pairs of diagonalizable matrices have a common system of eigenvectors [N1]. They are characterized by using a conventional algorithm to diagonalize A alone, then performing the same similarity transformation to B. So, for example, we might use the eigensystem routine **RS** from [S5], to find an orthogonal matrix $U \in \mathbb{R}^{n,n}$ and a diagonal matrix $D \in \mathbb{R}^{n,n}$ such that $A = U^T D U$. Although **RS** does not produce the diagonal entries of D in any particular order, it is easy to order the eigenvalues in decreasing algebraic order (say) along the diagonal of D. Then $E := UBU^T$ is block diagonal with the jth diagonal block of order equal to the multiplicity of the jth eigenvalue of A. In particular, if A has distinct eigenvalues, then E is diagonal and the simultaneous diagonalization is complete. In any case, a subsequent, block diagonal similarity transformation can diagonalize E without disturbing D.

Rounding errors destroy this elegant approach. Suppose, for example, that rounding error perturbs the commuting pair A, B to the nearly commuting pair

$$\tilde{A} = \begin{bmatrix} 1 & \varepsilon & 0 & 0 \\ \varepsilon & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}$$

and

$$\tilde{B} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & \varepsilon \\ 0 & 0 & \varepsilon & 1 \end{bmatrix},$$

where ε is a small quantity caused by rounding error. If $\varepsilon \neq 0$, then $\tilde{A}$ and $\tilde{B}$ do not commute. Even if no other rounding errors occur, the *diagonalize-one-then-diagonalize-the-other* procedure increases the magnitude of some off-diagonal entries to 1. The results do not improve if the procedure is repeated. It never discovers that the trivial similarity transformation $U = I$ "reduces" the off-diagonal entries of $\tilde{D} = U\tilde{A}U^T$ and $\tilde{E} = U\tilde{B}U^T$ to ε's and zeros.

The same kind of failure dooms any simultaneous diagonalization algorithm that does a good job of diagonalizing one member of a commuting pair without regard to what the similarity transformation does to the other member.

Goldstine and Horwitz [G3] present a procedure for diagonalizing normal matrices by simultaneously diagonalizing the Hermitian and skew Hermitian parts. Their algorithm chooses similarity transformations taking both A and B into account. The simultaneous diagonalization algorithms presented below are adaptations of [G3] to matrices with special structure.

Following the classic approach of Jacobi [J1] we will measure progress toward simultaneous diagonalization by the quantity

$$\mathit{off}(A,B) = \sum_{i<j} a_{ij}^2 + \sum_{i<j} b_{ij}^2. \tag{3.11}$$

It seems to be best to scale A and B to have approximately equal norm so that contributions from A and B are of equal "importance."

In broad outline, the algorithm consists of a sequence of similarity transformations by plane rotations. We favor using plane rotations in the simple row serial order of eliminating pivots elements in positions $(1, 2), (1, 3), \cdots, (1, n), (2, 3), (2, 3), \cdots, (2, n), (3, 4), \cdots, (n-1, n)$. The angle of each plane rotation is chosen to minimize $\mathit{off}(A, B)$ as described below. We observe quadratic convergence [B10] similar to Jacobi's method for the symmetric eigenvalue problem [P5], [F1], [R3].

Let $R = R(\theta, i, j) \in \mathbb{R}^{n,n}$ denote a plane rotation through angle θ in the (i, j)th plane, i.e., R agrees with the identity matrix except that

$$\begin{bmatrix} r_{ii} & r_{ij} \\ r_{ji} & r_{jj} \end{bmatrix} = \begin{bmatrix} c & s \\ -s & c \end{bmatrix} \tag{3.12}$$

where $c = \cos(\theta)$ and $s = \sin(\theta)$. An easy calculation shows that

$$\begin{aligned} \mathit{off}(RAR^T, RBR^T) = \mathit{off}(A,B) - a_{ij}^2 - b_{ij}^2 + ((c^2-s^2)a_{ji} - sc(a_{jj}-a_{ii}))^2 \\ + ((c^2-s^2)b_{ji} - sc(b_{jj}-b_{ii}))^2. \end{aligned} \tag{3.13}$$

In general it is not possible to choose $c = \cos(\theta)$ and $s = \sin(\theta)$ so that both the last two terms are zero. However, it is possible to choose θ to minimize (3.13).

The angle θ must minimize

$$\left\| \begin{bmatrix} a_{ji} & (a_{jj}-a_{ii})/2 \\ b_{ji} & (b_{jj}-b_{ii})/2 \end{bmatrix} \begin{bmatrix} \cos(2\theta) \\ \sin(2\theta) \end{bmatrix} \right\|_2, \tag{3.14}$$

i.e., $[\cos(2\theta)\ \sin(2\theta)]^T$ must be a left singular vector of

$$\begin{bmatrix} a_{ji} & (a_{jj}-a_{ii})/2 \\ b_{ji} & (b_{jj}-b_{ii})/2 \end{bmatrix}, \tag{3.15}$$

that corresponds to the smaller singular value. There are four roots $\theta \in [-\pi, \pi]$. To avoid cycling, we choose the root $\theta \in (-\pi/4, \pi/4)$. Although the formulae given in [G3] are more complicated, this is the same choice of c and s used there to diagonalize the normal matrix $A + iB$. The following algorithm is known to enjoy local quadratic convergence [B10], [R3]. Variations have been shown to converge globally [G3].

ALGORITHM 3.16. Simultaneous diagonalization of commuting real symmetric matrices.
Input: $A = A^T \in \mathbb{R}^{n,n}$ and $B = B^T \in \mathbb{R}^{n,n}$ such that $AB = BA$; a convergence tolerance $\tau > 0$
Output: $Q \in \mathbb{R}^{n,n}$, such that $Q^TQ = I$; $D \in \mathbb{R}^{n,n}$; and $E \in \mathbb{R}^{n,n}$ such that $AQ = QD$, $BQ = QE$, and $off(D, E) \leqq \tau(\|A\| + \|B\|)$.
Set $Q := I$; $D := A$; $E := B$.
WHILE $off(D, E) > \tau(\|A\| + \|B\|)$ DO
FOR $i = 1, 2, 3, \cdots, n$
FOR $j = i + 1, \cdots, n$
Let $\theta \in (-\pi/4, \pi/4)$ be chosen such that $[\cos(2\theta)\ \sin(2\theta)]^T$ is the left singular vector corresponding to the smaller singular value of (3.15).
Overwrite $D := RDR^T$; $E := RER^T$; $Q := QR$.
END FOR
END FOR
END WHILE

If rotations are stored and applied in the efficient manner outlined in [G4] or § 6.4 of [P5], then the above algorithm uses $4n^3$ flops per iteration of the outermost loop. Rarely, in our experience, does $\tau = 10^{-14}$ make the algorithm run more than six iterations. Since only the upper triangles of A and B need be stored, the algorithm needs only $2n^2$ storage.

As mentioned above, rounding errors often corrupt the commuting property of a pair of would-be commuting matrices. As they arise in the next section, a computed pair of "commuting" matrices A and B actually satisfies

$$(A+\Delta A)(B+\Delta B)=(B+\Delta B)(A+\Delta A) \tag{3.17}$$

for perturbations ΔA and ΔB satisfying

$$\|\Delta A\| + \|\Delta B\| \leqq O(n^3)\varepsilon(\|A\| + \|B\|) \tag{3.18}$$

where ε is the precision of the arithmetic. Thus, using orthogonal similarity transformations, it is possible to simultaneously diagonalize A and B except for a rounding-error small perturbation bounded by the right-hand-side of (3.18).

Note that it is easy to find examples satisfying the weaker assumption $AB - BA = E$ where $\|E\| \leqq O(n^3)\varepsilon(\|A\| + \|B\|)$ which do not satisfy (3.17) and (3.18).

A simultaneous diagonalization algorithm must ultimately set small off-diagonal entries to zero and produce a genuinely commuting pair of diagonal matrices. Assumptions (3.17) and (3.18) are a necessary condition for such an algorithm to be backward stable. Two consequences of the norm preserving property of unitary similarity transformations are

(i) Assumptions (3.17) and (3.18) are preserved despite rounding errors, and
(ii) Setting rounding-error small off-diagonal entries to zero is equivalent to a

rounding-error small perturbation of the original A and B. Hence, under assumptions (3.17) and (3.18), the Jacobi-like algorithm described above is backward stable.

Two kinds of modifications of the above basic simultaneous diagonalization algorithm are needed to suit it to structures other than symmetric matrices. Complex similarity transformations are required to diagonalize a commuting pair of complex normal matrices. The real rotations of the above algorithm become complex rotations. The analogue of (3.13) for complex rotations is a two real variable minimization. For the case of a commuting Hermitian and skew Hermitian pair, explicit expressions for the minimizing parameters appear in [G3].

A commuting pair of real matrices that contains a real skew symmetric matrix or a real orthogonal matrix cannot be diagonalized by real similarity transformations alone. Nevertheless, complex arithmetic can be avoided by permitting one or both members of the pair to converge to block diagonal form with one-by-one and two-by-two blocks. The real rotations of the above algorithm become the 4-by-4 orthogonal "Jacobi annihilators" of [P4]. Jacobi annihilators have a natural parametrization in terms of three real parameters. The minimization problem grows to involve 16 entries of A and B in place of the three that appear in (3.13). We have no explicit formulae for this more complicated minimizer.

3.4. The "X-trick." The "X-trick" is a simple, structure revealing, symplectic, unitary similarity transformation. If $M = \left[\begin{smallmatrix} F & G \\ H & K \end{smallmatrix}\right] \in \mathbb{K}^{2n,2n}$, ($\mathbb{K} = \mathbb{C}$ or $\mathbb{K} = \mathbb{R}$) and $X = (1/\sqrt{2})\left[\begin{smallmatrix} I & I \\ iI & -iI \end{smallmatrix}\right]$,

$$X^* M X = \frac{1}{2}\begin{bmatrix} F+K+i(G-H) & F-K-i(G+H) \\ F-K+i(G+H) & F+K-i(G-H) \end{bmatrix}.$$

Immediate consequences of Lemma 2.5 are the block diagonal matrices in Table 3.19.

In the block diagonal cases, we solve one or two $n \times n$ eigenvalue problems. In the block antidiagonal case we compute one or two $n \times n$ singular value decompositions instead. (Details appear in § 4.)

In other cases the "X-trick" leads to the problem of simultaneous diagonalization of a pair of commuting normal matrices. Algorithms for these problems appear in [G3] and [B10].

3.5. Error analysis and numerical stability. We finish the basic algorithm section with a remark on the numerical stability of the above-mentioned algorithms.

The algorithms use a sequence of elementary unitary similarity transformations to reduce a matrix A to a Schur-like form T. Well-known rounding-error analysis for standard methods apply here [W2]. At every step the result of the similarity transformation $\tilde{T}$ satisfies

$$(A+E)U = U\tilde{T}$$

where U is a product of unitary matrices and $\|E\|$ is bounded by a small constant multiple of the precision of the arithmetic. The algorithms are designed to drive T to one of the Schur-like forms of Theorem 2.12. For the structure preserving methods E has the same structure as A. So the algorithms produce eigenvalues of a nearby *structured* matrix with appropriate eigenvalue pairing and structured matrix of eigenvectors. Thus, these methods are strongly stable in the sense of [B4].

4. The complex chart. In this section we now discuss all combinations of two of the algebraic properties in Table 2.4. We list them in the form of charts.

TABLE 3.19

Block structures of matrices with twofold symmetry. Let $M = \left[\begin{smallmatrix} F & G \\ H & K \end{smallmatrix}\right] \in \mathbb{K}^{2n,2n}$($\mathbb{K} = \mathbb{R}$ *or* $\mathbb{K} = \mathbb{C}$). *If* M *has the structures at the heads of column* i *and row* j, *then* X^*MX *has the block structure in position* (i, j).

	$QJQ^* = J$	$JQ = (JQ)^*$	$JQ = -(JQ)^*$
$QQ^* = I$	$\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix}$		
$Q = Q^*$		$\begin{bmatrix} 0 & F-iG \\ F+iG & 0 \end{bmatrix}$, $F = F^*, G = G^*$	$\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix}$, $F = F^*, G = -G^*$
$Q = -Q^*$		$\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix}$, $F = -F^*, G = G^*$	$\begin{bmatrix} 0 & F-iG \\ F+iG & 0 \end{bmatrix}$, $F = -F^*, G = -G^*$
$QQ^T = I$	$\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix}$,		
$Q = Q^T$		$\begin{bmatrix} 0 & F-iG \\ F+iG & 0 \end{bmatrix}$, $F = F^T, G = G^T$	$\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix}$, $F = F^T, G = -G^T$
$Q = -Q^T$		$\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix}$, $F = -F^T, G = G^T$	$\begin{bmatrix} 0 & F-iG \\ F+iG & 0 \end{bmatrix}$, $F = -F^T, G = -G^T$

The first chart (Table 4.1) covers the complex case and lists unitary, structure preserving methods that may be applied to solve the corresponding eigenvalue problem. Question marks indicate cases for which structure preserving algorithms are not known yet. Entries marked with "$\frac{1}{2}$" correspond to algorithms that preserve part but not all of the structure. A "T" corresponds to cases which are trivial either because there are no matrices with the prescribed pair of structures or because the eigenvalues are entirely determined by these structures. For example, only the zero matrix is both Hermitian and skew Hermitian. An "X" indicates that the X-trick zeros the diagonal blocks or zeros the off-diagonal blocks, deflating the eigenvalue problem to one or two n-by-n problems. An "S" indicates a method based on simultaneous diagonalization of commuting normal matrices. A "Q" indicates that the quaternion QR algorithm is satisfactory. A "V" indicates that there exists some other satisfactory method already. "Real" indicates that the corresponding class contains only real matrices and/or matrices with zero real part. Clearly the chart is symmetric. For real problems we give a separate chart in the next section.

In the following we assume that $Q \in \mathbb{C}^{m,m}$, where if necessary, $m = 2n$. In the latter case, we set $Q = \left[\begin{smallmatrix} F & G \\ H & K \end{smallmatrix}\right]$, with $F, G, H, K \in \mathbb{C}^{n,n}$.

(a) *The Northwest corner of the chart.*

Position (1, 1): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$. This is the class of unitary matrices. Structure exploiting unitary QR-algorithms are given in [A2] and [G6].

TABLE 4.1
Summary of structure preserving methods.

	$QQ^* = I$	$QJQ^* = J$	$Q = Q^*$	$Q = -Q^*$	$JQ = (JQ)^*$	$JQ = -(JQ)^*$	$QQ^T = I$	$QJQ^T = J$	$Q = Q^T$	$Q = -Q^T$	$JQ = (JQ)^T$	$JQ = -(JQ)^T$
$QQ^* = I$	V	X	T	T	$\frac{1}{2}$	S	Real	Q	S	S	S	V
$QJQ^* = J$		?	$\frac{1}{2}$	S	T	T	$\frac{1}{2}$	Real	?	?	?	$\frac{1}{2}$
$Q = Q^*$			V	T	X	X	S	V	Real	Real	Q	V
$Q = -Q^*$				V	X	X	S	V	Real	Real	Q	V
$JQ = (JQ)^*$					?	T	?	?	$\frac{1}{2}$	$\frac{1}{2}$	Real	Real
$JQ = -(JQ)^*$						?	?	?	$\frac{1}{2}$	$\frac{1}{2}$	Real	Real
$QQ^T = I$							?	X	T	T	?	$\frac{1}{2}$
$QJQ^T = J$								?	?	?	T	T
$Q = Q^T$									?	T	?	$\frac{1}{2}$
$Q = -Q^T$										?	$\frac{1}{2}$	$\frac{1}{2}$
$JQ = (JQ)^T$											?	T
$JQ = -(JQ)^T$												V

V: satisfactory algorithm not listed below.
Q: quaternion QR algorithm preserves all structure.
?: no structure preserving method known yet.
$\frac{1}{2}$: algorithm preserving some but not all structure.
Real: class of matrices contains only real matrices and/or matrices with zero real part.
T: trivial case.
X: X-trick block diagonalizes or block off-diagonalizes.
S: simultaneous diagonalization of commuting normal matrices.

Position (1, 2): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, and $QJQ^* = J$. Then $Q \in \mathcal{C}_{2n}(J)$, the centralizer of J. Applying the X-trick and Lemma 2.5 we obtain

$$X^*QX = \frac{1}{2}\begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix},$$

where $F + iG$, $F - iG$ are unitary. So this eigenproblem can be solved by applying methods from position (1, 1) to $F + iG$ and $F - iG$.

Position (1, 3): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, and $Q^* = Q$. This implies that all the eigenvalues of Q are $+1$ and -1, so there is essentially nothing to do. The invariant subspace for the eigenvalue $+1$ is Range $(Q + I)$. All the matrices Q with this property are of the form PDP^*, where D is diagonal with ± 1 on the diagonal and P is unitary.

Position (1, 4): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, and $Q^* = -Q$. A matrix Q is in this class if and only if iQ is a matrix of position (1, 3).

Position (1, 5): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $JQ = (JQ)^* = -Q^*J$. Then $Q = \left[\begin{smallmatrix} F & G \\ H & -F^* \end{smallmatrix}\right]$, where $G = G^*$, $H = H^*$. Applying the X-trick, we obtain

$$\begin{aligned} W = X^*QX &= \frac{1}{2}\begin{bmatrix} F-F^*+i(G-H) & F+F^*-i(G+H) \\ F+F^*+i(G+H) & F-F^*-i(G-H) \end{bmatrix} \\ &=: \begin{bmatrix} A+iB & C-iD \\ C+iD & A-iB \end{bmatrix}, \end{aligned}$$

where $A = -A^*$, $B = -B^*$, $C = C^*$, $D = D^*$.

Let

$$W_1 = \frac{1}{2}(W - W^{-1}) = \frac{1}{2}(W - W^*) = \begin{bmatrix} A+iB & 0 \\ 0 & A-iB \end{bmatrix},$$

and let

$$W_2 = \frac{1}{2}(W + W^{-1}) = \frac{1}{2}(W + W^*) = \begin{bmatrix} 0 & C-iD \\ C+iD & 0 \end{bmatrix}.$$

Observe that W_1 and W_2 commute. Thus we can simultaneously diagonalize W_1 and W_2. W_1 is skew Hermitian, its eigenvalues are the imaginary parts of the eigenvalues of Q; W_2 is Hermitian, its eigenvalues are the real parts of the eigenvalues of Q. Finding the eigenvalues of W_1 reduces to computing the eigenvalues of two n-by-n Hermitian matrices. Finding the eigenvalues of W_2 reduces to an n-by-n singular value decomposition.

The simultaneous diagonalization can be performed according to [G3] or, taking advantage of the special structure of W_1' and W_2', according to [B10].

The eigenvalue problem for the Hermitian matrix W_2 is essentially a singular value decomposition for $(C + iD)$. A head start on the simultaneous diagonalization of W_1 and W_2 is then easily obtained from the singular vectors of $(C + iD)$.

Position (1, 6): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$ and $JQ = -(JQ)^* = Q^*J$. Then $\hat{Q} := iQ$ is a member of the class in position (1, 5).

Position (2, 2): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$. For conjugate symplectic matrices, the only known unitary, structure preserving QR-like method is the symplectic QR algorithm in [M1] for the case in which the (2, 1) block of $Q \in \mathbb{C}^{2n,2n}$ is of rank 1.

Another nonunitary structure preserving method for this problem is the sign function method [B2], [G2], and Newton's method for discrete algebraic Riccati equations (see, e.g., [A3], [A4], [H1], [H3], [K1]) also works in many cases. For an overview of the methods and a comparison, see [M2].

Position (2, 3): $Q \in \mathbb{C}^{n,n}$, $Q^*JQ = J$, $Q = Q^*$. Then $Q = \left[\begin{smallmatrix} F & G \\ G^* & K \end{smallmatrix}\right]$ where $F = F^*$, $K = K^*$. The X-trick gives

$$W = X^*QX = \frac{1}{2}\begin{bmatrix} F+K+i(G-G^*) & F-K-i(G+G^*) \\ F-K+i(G+G^*) & F+K-i(G-G^*) \end{bmatrix}$$
$$=: \begin{bmatrix} A+iB & C-iD \\ C+iD & A-iB \end{bmatrix},$$

where $A = A^*$, $B = -B^*$, $C = C^*$, $D = D^*$. Now $Q^*JQ = J$; thus

$$W^{-1} = -\begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} W \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} = -\begin{bmatrix} A+iB & -(C+iD) \\ -(C+iD) & A-iB \end{bmatrix}.$$

Thus we get

$$W_1 = \frac{1}{2}(W - W^{-1}) = \begin{bmatrix} A+iB & 0 \\ 0 & A-iB \end{bmatrix}$$

and

$$W_2 = \frac{1}{2}(W + W^{-1}) = \begin{bmatrix} 0 & C-iD \\ C+iD & 0 \end{bmatrix}$$

where W_1, W_2 are Hermitian.
So, this is a simultaneous diagonalization problem, with head start as in positions (1, 5) and (1, 6).

Position (2, 4): $Q \in \mathbb{C}^{n,n}$, $Q^*JQ = J$, $Q = -Q^*$. Then $\hat{Q} = iQ$ is a member of the class in position (2, 3).

Position (2, 5): $Q \in \mathbb{C}^{n,n}$, $Q^*JQ = J$, $JQ = (JQ)^* = -Q^*J$. It follows that $Q^2 = -I$, so the eigenvalues are $\pm i$. Matrices in this class are of the form $Q = PDP^{-1}$ where D is a matrix of $\pm i$'s.

Position (2, 6): $Q \in \mathbb{C}^{n,n}$, $Q^*JQ = J$, $JQ = -(JQ)^* = Q^*J$. Then $\hat{Q} = iQ$ is a member of the class in position (2, 5).

Position (3, 3): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$. This is the most extensively studied special structure [P5].

Position (3, 4): $Q \in \mathbb{C}^{n,n}$, $Q^* = Q = -Q^*$ implies $Q = 0$.

Position (3, 5): $Q \in \mathbb{C}^{n,n}$, $Q^* = Q$ and $JQ = (JQ)^* = -Q^*J$. Then $Q = \left[\begin{smallmatrix} F & G \\ G^* & -F^* \end{smallmatrix}\right]$, with $F = F^*$, $G = G^*$. The X-trick gives

$$W = X^*QX = \begin{bmatrix} 0 & F-iG \\ F+iG & 0 \end{bmatrix}.$$

Let

$$U\Sigma V = F - iG$$

be the singular value decomposition with U, $V \in \mathbb{C}^{n\times n}$ unitary and $\Sigma = \operatorname{diag} \times (\sigma_1, \sigma_2, \sigma_3, \cdots, \sigma_n) \in \mathbb{C}^{n\times n}$. Then

$$\begin{bmatrix} U^* & 0 \\ 0 & V^* \end{bmatrix} \begin{bmatrix} 0 & F-iG \\ F+iG & 0 \end{bmatrix} \begin{bmatrix} U & 0 \\ 0 & V \end{bmatrix} = \begin{bmatrix} 0 & \Sigma \\ \Sigma & 0 \end{bmatrix},$$

so Q has eigenvalues $\pm\sigma_i$.

Position (3, 6): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, and $JQ = -(JQ)^* = Q^*J$. Then

$$Q = \begin{bmatrix} F & G \\ G^* & F^* \end{bmatrix}$$

with $F^* = F$ and $G^* = -G$ is in $\mathscr{C}_{2n}(J)$, and consequently

$$W = X^*QX = \begin{bmatrix} F+iG & 0 \\ 0 & F-iG \end{bmatrix},$$

which consists of two $n \times n$ Hermitian eigenvalue problems.

Position (4, 4): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^*$. For the skew Hermitian eigenvalue problem we may apply the Hermitian methods to iQ.

Position (4, 5): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^*$, and $JQ = (JQ)^* = -Q^*J$. $\hat{Q} = iQ$ is a member of the class in position (3, 6).

Position (4, 6): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^*$, and $JQ = -(JQ)^* = Q^*J$. $\hat{Q} = iQ$ is a member of the class in position (3, 5).

Position (5, 5): For J-Hermitian matrices the only known unitary, structure preserving QR-like method is in the Hamiltonian QR algorithm [B16] for the case where the (2, 1) block of $Q \in \mathbb{C}^{2n,2n}$ is of rank 1. A structure preserving Jacobi-like method is given in [B18]. Other nonunitary methods for this problem are the sign function method [B1], [G2], [B17] and also Newton's method for the continuous algebraic Riccati equation [A3], [A4], [H1], [K1]. For an overview and a comparison of these methods, see [M2].

Position (5, 6): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^*$, and $JQ = -(JQ)^*$ implies $Q = 0$.

Position (6, 6): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^*$. Then $\hat{Q} = iQ$ is a member of the class in position (5, 5). In the real case we have further simplifications discussed later.

In summary we see that except for a few diagonal positions, the Northwest corner of the chart can be treated with unitary structure preserving methods.

(b) *The Southeast corner of the chart.*

Position (7, 7): $Q \in \mathbb{C}^{n,n}$, $QQ^T = I$. At this writing there is no known unitary, structure preserving method for this problem.

Position (7, 8): $Q \in \mathbb{C}^{n,n}$, $Q^TQ = I$, and $Q^TJQ = J$. Then $Q \in \mathscr{C}_{2n}(J)$ and $X^*QX = \left[\begin{smallmatrix} F+iG & 0 \\ 0 & F-iG \end{smallmatrix}\right]$, where $(F + iG)^T = (F - iG)^{-1}$. The QR algorithm computes the Schur form of $F + iG$, $V^*(F + iG)V = R$, where V is unitary and R upper triangular. Then $V^T(F - iG)\bar{V} = R^{-T}$.

Position (7, 9): $Q \in \mathbb{C}^{n,n}$, $Q^TQ = I$, and $Q = Q^T$ implies that Q is diagonalizable with eigenvalues ± 1. Every matrix with this property can be written as $Q = UDU^{-1}$, where for some integer s,

$$D = \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} \begin{matrix} \} & s \\ \} & 2n-s \end{matrix}$$

and $U^{-1} = U^T$. This is proved as follows. Clearly since $Q^2 = I$, $Q = VDV^{-1}$ for some diagonal matrix D with ± 1's on the diagonal. It follows that $VDV^{-1} = V^{-T}DV^T$ and thus $V^TVD = DV^TV$ and consequently

$$V^TV = \begin{bmatrix} V_1 & 0 \\ 0 & V_2 \end{bmatrix}$$

with partitioning conformal to D.

Now V_1, V_2 are complex symmetric, thus they have a Takagi singular value decomposition (Theorem 2.21). $V_i = U_i \Sigma_i U_i^T$, $i = 1, 2$ with U_i unitary. Let $\tilde{U}_i = U_i \Sigma^{1/2}$, $i = 1, 2$; then it follows that

$$\begin{bmatrix} \tilde{U}_1^{-1} & 0 \\ 0 & \tilde{U}_2 \end{bmatrix} V^T V \begin{bmatrix} \tilde{U}_1^{-T} & 0 \\ 0 & \tilde{U}_2^{-T} \end{bmatrix} = I.$$

Taking

$$U = V \begin{bmatrix} \tilde{U}_1^{-T} & 0 \\ 0 & \tilde{U}_2^{-T} \end{bmatrix}$$

it follows that $U^{-1} = U^T$ and $Q = UDU^T = UDU^{-1}$.

Position (7, 10): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^T$, and $QQ^T = I$ implies that $Q^2 = -I$, i.e., all eigenvalues are $\pm i$. As in Position (7, 9) we can show that every matrix with these properties can be written as $Q = UDU^{-1}$, where $U^{-1} = U^T$ and D is a matrix of $\pm i$'s and we have the same number of $+i$'s and $-i$'s.

Position (7, 11): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^T$, and $QQ^T = I$. We observe the same difficulties as in positions (8, 9), (8, 10) below.

Position (7, 12): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^T$, $QQ^T = I$. According to Table 3.10(c) we have a unitary symplectic U such that

$$U^* Q U = \begin{bmatrix} F_1 & G_1 \\ 0 & F_1^T \end{bmatrix},$$

where $G_1 = -G_1^T$. We might then apply an unstructured eigenvalue method to F_1. But the algorithm in [V2] easily generalizes to the complex case, such that we can work on Q directly, preserving the property $(JQ)^T = -(JQ)$. Unfortunately $QQ^T = I$ is not used and not preserved by the transformation.

Position (8, 8): $Q \in \mathbb{C}^{n,n}$, $Q^T J Q = J$. At this writing there is no known unitary, structure preserving method.

Position (8, 9): $Q \in \mathbb{C}^{n,n}$, $Q = Q^T$, and $QJQ^T = J$. Then

$$Q = \begin{bmatrix} F & G \\ G^T & K \end{bmatrix}$$

with $F^T = F$, $K = K^T$.

$$W = X^* Q X = \frac{1}{2} \begin{bmatrix} F+K+i(G-G^T) & F-K-i(G+G^T) \\ F-K+i(G+G^T) & F+K-i(G-G^T) \end{bmatrix}$$

$$=: \begin{bmatrix} A+iB & C-iD \\ C+iD & A-iB \end{bmatrix}$$

with $A = A^T$, $B = -B^T$, $C = C^T$, $D = D^T$.

Now

$$W^{-1} = W^*(JQ^T J^T)X = -\begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} W \begin{bmatrix} I & 0 \\ 0 & -I \end{bmatrix} = -\begin{bmatrix} A+iB & -C+iD \\ -C-iD & A-iB \end{bmatrix}.$$

Thus

$$W_1 = \frac{W + W^{-1}}{2} = \begin{bmatrix} 0 & C-iD \\ C+iD & 0 \end{bmatrix}, \qquad W_2 = \frac{W - W^{-1}}{2} = \begin{bmatrix} A+iB & 0 \\ 0 & A-iB \end{bmatrix}.$$

The problem is now a simultaneous diagonalization problem for a *nonnormal* commuting pair. An entirely satisfactory algorithm for this problem is unknown.

Position (8, 10): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^T$, and $QJQ^T = J$. We observe the same difficulties as in positions (8, 9) and (7, 11).

Position (8, 11): $Q \in \mathbb{C}^{n,n}$, $Q^TJQ = J$, and $JQ = (JQ)^T = -Q^TJ$. We then have $J = Q^TJQ = -JQQ$; thus $Q^2 = -I$, i.e., all eigenvalues are $\pm i$ and Q is diagonalizable. Any Q with this property can be expressed in the form $Q = PDP^{-1}$ with $P^TJP = J$, and D is a diagonal matrix of $\pm i$'s, with an equal number of $+i$'s and $-i$'s.

Position (8, 12): $Q \in \mathbb{C}^{n,n}$, $QJ = -(QJ)^T = JQ^T$, $QJQ^T = J$ implies $Q^2 = I$; thus all eigenvalues are $+1$ and -1 and every matrix with this property is of the form $Q = PDP^{-1}$, and D is a diagonal matrix of ± 1's.

Position (9, 9): $Q \in \mathbb{C}^{n,n}$, $Q = Q^T$. Every complex m-by-m matrix is similar to a complex symmetric matrix. At this writing, there is no known unitary, structure preserving method for complex symmetric matrices. A nonunitary structure preserving method for this problem is the QL algorithm in [C1].

Position (9, 10): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^T$, and $Q = Q^T$ implies $Q = 0$.

Position (9, 11): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^T = -JQ$, and $Q = Q^T$. Then $Q = \left[\begin{smallmatrix} F & G \\ G & -F \end{smallmatrix}\right]$ with $G = G^T$, $F = F^T$, and $W = X^*QX = \left[\begin{smallmatrix} 0 & F - iG \\ F + iG & 0 \end{smallmatrix}\right]$. This block structure is tantalizing; however, we do not have a satisfactory eigenvalue algorithm.

Position (9, 12): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^T = Q^TJ$, and $Q = Q^T$. Then $Q = \left[\begin{smallmatrix} F & G \\ -G & F \end{smallmatrix}\right]$ with $F^T = F$, $G^T = -G$. As in position (7, 12), there is a unitary symplectic U such that

$$U^*QU = \begin{bmatrix} F_1 & G_1 \\ 0 & F_1^T \end{bmatrix}.$$

A complex version of the algorithm in [V2] can be applied to Q directly, preserving the property $JQ = -(JQ)^T$.

Position (10, 10): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^T$. Complex skew symmetric matrices have pairs of eigenvalues λ, $-\lambda$ (Table 2.4), but at this writing, there is no known unitary, structure preserving method for this problem.

Position (10, 11): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^T = -Q^TJ$ and $-Q = Q^T$. This implies that $JQ = QJ$, so $Q = \left[\begin{smallmatrix} F & G \\ -G & F \end{smallmatrix}\right]$ and thus $W = X^*QX = \left[\begin{smallmatrix} F+iG & 0 \\ 0 & F-iG \end{smallmatrix}\right]$, with $F = -F^T$, $G = -G^T$, and $F + iG$ and $F - iG$ complex skew symmetric. Although we do not have a satisfactory algorithm for the $n \times n$ complex skew symmetric problem, some of the structure is preserved by treating $F + iG$ and $F - iG$ as unstructured problems.

Position (10, 12): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^T = Q^TJ$ and $Q = -Q^T$. Then $Q = \left[\begin{smallmatrix} F & G \\ G & -F \end{smallmatrix}\right]$ where $F^T = -F$, $G = -G^T$. As in position (7, 12), some of the structure is preserved by applying the algorithm of [V2].

Position (11, 11): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^T$. We do not know a unitary structure preserving method for this case.

Position (11, 12): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^T = JQ^T$ implies $Q = 0$.

Position (12, 12): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^T$. All of the structure is preserved by the algorithm in [V2]. This is one of the few diagonal entries which admit an entirely structure preserving algorithm.

We have seen that it is more difficult to preserve the structure in this quarter of the chart. This is because few unitary matrices preserve these structures.

(c) *The Northeast corner of the chart.*

Position (1, 7): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $QQ^T = I$ implies that $Q^T = Q^TQQ^* = \bar{Q}^T$. So Q is real orthogonal. We discuss this problem in context with the real chart.

Position (1, 8): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $QJQ^T = J$ implies that $Q^TJ = JQ^*$ or $J\bar{Q} = QJ$. Thus $Q = \left[\begin{smallmatrix} F & G \\ -\bar{G} & \bar{F} \end{smallmatrix}\right]$, which is a unitary quaternion matrix and thus may be treated by the quaternion QR algorithm [B9], which preserves the structure.

Position (1, 9): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $Q = Q^T$. Then,

$$Q = \begin{bmatrix} F & G \\ G^T & K \end{bmatrix}$$

with $F = F^T$, $K = K^T$. Set

$$W_1 = \frac{Q+Q^*}{2} = \frac{Q+\bar{Q}}{2} = \operatorname{Re}(Q),$$

$$W_2 = \frac{Q-Q^*}{2} = \frac{Q-\bar{Q}}{2} = i \operatorname{Im}(Q).$$

So W_1 and iW_2 form a commuting pair of real symmetric matrices which may be simultaneously diagonalized [B10], [G3].

Position (1, 10): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $Q = -Q^T$. Set

$$W_1 = \frac{Q+Q^*}{2} = \frac{Q-\bar{Q}}{2} = i \operatorname{Im}(Q),$$

$$W_2 = \frac{Q-Q^*}{2} = \frac{Q+\bar{Q}}{2} = \operatorname{Re}(Q).$$

So, iW_1 and W_2 form a commuting pair of real skew symmetric matrices which may be simultaneously diagonalized [B10].

Position (1, 11): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $JQ = (JQ)^T = -Q^TJ$. Then

$$Q = \begin{bmatrix} F & G \\ H & -F^T \end{bmatrix},$$

$G = G^T$, $H = H^T$,

$$W_1 = \frac{Q+Q^*}{2} = \frac{1}{2}\begin{bmatrix} F+F^* & G+H^* \\ H+G^* & -F^T-\bar{F} \end{bmatrix} =: \begin{bmatrix} A & \bar{B} \\ B & -\bar{A} \end{bmatrix}$$

is an Hermitian antiquaternion matrix [B9], and

$$W_2 = \frac{Q-Q^*}{2} = \frac{1}{2}\begin{bmatrix} F-F^* & G^*-H \\ H-G^* & -F^T+\bar{F} \end{bmatrix} =: \begin{bmatrix} C & -\bar{D} \\ D & \bar{C} \end{bmatrix}$$

is a skew Hermitian quaternion matrix. We may apply the simultaneous diagonalization procedure based on a quaternion Jacobi method [B10].

Position (1, 12): $Q \in \mathbb{C}^{n,n}$, $QQ^* = I$, $JQ = -(JQ)^T = Q^TJ$. Algorithm 3.8 gives

$$W := U^*QU =: \begin{bmatrix} A & 0 \\ 0 & A^T \end{bmatrix},$$

where U is symplectic and unitary. The unitary QR algorithm applied to A finishes the diagonalization. A simultaneous diagonalization method as in position (1, 11) may also be applied here.

Position (2, 7): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$, $QQ^T = I$ implies that $QJ = J\bar{Q}$, i.e., Q is a quaternion matrix $\left[\begin{smallmatrix} F & -\bar{H} \\ H & \bar{F} \end{smallmatrix}\right]$, to which the quaternion QR algorithm applies. Note that we

have no additional property for the quaternion matrix as in Position (1, 8). Thus, part of the structure is lost.

Position (2, 8): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$, $QJQ^T = J$ implies $Q = \bar{Q}$, i.e., the problem is real and is thus discussed in the real chart.

Position (2, 9): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$, $Q = Q^T$. At this writing, there is no known unitary structure preserving method for this case.

Position (2, 10): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$, $Q = -Q^T$. At this writing, there is no known unitary structure preserving method for this case.

Position (2, 11): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$, $JQ = (JQ)^T = -Q^T J$ implies $Q\bar{Q} = -I$. At this writing, there is no known unitary structure preserving method for this case.

Position (2, 12): $Q \in \mathbb{C}^{n,n}$, $QJQ^* = J$, $JQ = -(JQ)^T = Q^T J$. Algorithm 3.8 reduces Q to block upper triangular form with n-by-n blocks. The (1, 1) block is the transpose of the (2, 2) block. This preserves the $JQ = -(JQ)^T$ structure, but not the $QJQ^* = J$ structure. A Schur-like form can be obtained from the QR algorithm applied to the (1, 1) block.

Position (3, 7): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, $QQ^T = I$ implies that $Q^T = Q^{-1} = \bar{Q}$. So if $Q = A + iB$ with A, B real, then

$$I = (A+iB)(\overline{A+iB}) = (A+iB)(A-iB) = A^2 + B^2 + i(BA - AB).$$

Thus A and B commute, A is real symmetric, and B is real skew symmetric. So the Jacobi-like algorithm for simultaneous diagonalization of a real symmetric and a real skew symmetric matrix applies [B10].

Position (3, 8): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, $QJQ^T = J$. A symplectic Jacobi-like algorithm for Hermitian symplectic matrices can be constructed by using the unitary symplectic similarity transformations discussed in § 3.

Position (3, 9): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, $Q = Q^T$ implies that Q is real. See § 5.

Position (3, 10): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, $Q = -Q^T$ implies that $Q = -\bar{Q}$, i.e., $Q = i\tilde{Q}$ with $\tilde{Q}$ real and $\tilde{Q} = -\tilde{Q}^T$. This is essentially a real problem. See § 5.

Position (3, 11): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, $JQ = (JQ)^T = -Q^T J$ implies that $QJ = -J\bar{Q}$, thus Q is an antiquaternion matrix [B9] of the form $Q = \left[\begin{smallmatrix} A & \bar{B} \\ B & -\bar{A} \end{smallmatrix}\right]$, with $A = A^*$, $B = B^T$. We may use the quaternion QR method [B9] (in the Hermitian version) or a quaternion Jacobi method [B10], both of which preserve all the structure.

Position (3, 12): $Q \in \mathbb{C}^{n,n}$, $Q = Q^*$, $JQ = -(JQ)^T = Q^T J$. So, Q is a Hermitian quaternion matrix. A numerical method for this problem appears in [D2]. It uses Algorithm 3.8 to split Q into the direct sum of two copies of a real symmetric, tridiagonal matrix.

Position (4, 7): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^*$, $QQ^T = I$ implies that $Q^T = Q^{-1} = -\bar{Q}$. So if $Q = A + iB$, with A, B real, then A is skew symmetric, B is symmetric, and

$$I = -(A+iB)(\overline{A+iB}) = (-A-iB)(A-iB) = -A^2 - B^2 - i(BA - AB).$$

So A and B commute and the Jacobi-like algorithm [B10] for simultaneous diagonalization of a real symmetric and a real skew symmetric matrix applies.

Position (4, 8): $Q \in \mathbb{C}^{n,n}$, $Q = -Q^*$, $QJQ^T = J$.

$$Q = \begin{bmatrix} F & G \\ -G^* & K \end{bmatrix}, \qquad F = -F^*, \qquad K = -K^*.$$

A symplectic Jacobi-like algorithm for skew Hermitian symplectic matrices can be constructed by using the unitary symplectic transformations discussed in § 3.

Positions (4, 9), (4, 10), (4, 11), (4, 12): Each of these classes consist of matrices which are scalar multiples by i of matrices in the corresponding classes in row 3.

Position (5, 7): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^*$, $QQ^T = I$. At this writing, there is no known unitary structure preserving method for this case.

Position (5, 8): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^*$, $QJQ^T = J$ implies $Q\bar{Q} = -I$. We do not know a method for this structure.

Position (5, 9): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^*$, $Q = Q^T$ implies that $JQ = -Q^*J = -\bar{Q}J$ is a symmetric antiquaternion matrix $Q = \left[\begin{smallmatrix} F & G \\ \bar{G} & -\bar{F} \end{smallmatrix}\right]$, with $F = F^T$, $G = G^*$. We may apply the quaternion QR algorithm [B9], but this does not preserve the symmetry of Q, so part of the structure is lost.

Position (5, 10): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^*$, $Q = -Q^T$ implies that $JQ = \bar{Q}J$, i.e., Q is a skew symmetric quaternion matrix. The quaternion QR may again be applied but the skew symmetry of Q is not preserved, so part of the structure is lost.

Position (5, 11): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^*$, $JQ = (JQ)^T$ implies that the problem is real. See § 5.

Position (5, 12): $Q \in \mathbb{C}^{n,n}$, $JQ = (JQ)^* = -(JQ)^T$ implies that $Q = -\bar{Q}$, so the problem is purely imaginary. Let $Q = i\tilde{Q}$; then we have a problem for $\tilde{Q}$ satisfying $J\tilde{Q} = -(J\tilde{Q})^T$ with $\tilde{Q}$ real. See § 5.

Position (6, 7): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^*$, $Q^TQ = I$. At this writing, there is no known unitary structure preserving method for this case.

Position (6, 8): $Q \in \mathbb{C}^{n,n}$, $JQ = -(JQ)^*$, $Q^TJQ = J$. It follows that $Q^TQ^* = I$ or $Q\bar{Q} = I$. We do not know a method for this case.

Positions (6, 9), (6, 10), (6, 11), (6, 12): Each of these classes consist of matrices which are scalar multiples by i of matrices in the corresponding classes in row 5.

In summary, in the Northeast corner, we have problems that are either reducible to the real case or quaternion problems or are trivial. Often, we can preserve only part of the structure.

If, in one of the above cases, there is both a Jacobi-like and a QR-like method, then in a conventional serial computing environment, the QR-like method is less expensive in work and storage. However, Jacobi methods may be able to take better advantage of parallelism.

5. The real chart. In this section we discuss orthogonal structure preserving methods for real, multiple structured matrices. In Table 5.1 we assume $Q \in \mathbb{R}^{m,m}$, where if necessary, $m = 2n$ and then $Q = \left[\begin{smallmatrix} F & G \\ H & K \end{smallmatrix}\right]$ with $F, G, H, K \in \mathbb{R}^{n,n}$.

Position (1, 1): $Q \in \mathbb{R}^{n,n}$, $QQ^T = I$. Probably the best available method in terms of storage and preservation of structure is the orthogonal QR method in [A2] and [G5]. The Francis double shift implicit QR method and the Goldstine/Horwitz Jacobi method [G3] also preserve this structure.

Position (1, 2): $Q \in \mathbb{R}^{n,n}$, $QQ^T = I$, $Q^TJQ = J$ implies that $JQ = QJ$, $Q = \left[\begin{smallmatrix} F & G \\ -G & F \end{smallmatrix}\right]$ is in $C_{2n}(J)$, the centralizer of J. So, Q is the $2n$-by-$2n$ real representation of the n-by-n complex matrix $F + iG$. Observe that $F + iG$ is unitary, so all eigenvalue eigenvector information for Q can be obtained from the unitary QR algorithm of [A2] and [G5].

Position (1, 3): $Q \in \mathbb{R}^{n,n}$, $QQ^T = I$, $Q = Q^T$. As in position (1, 3) of the complex chart, we have that all eigenvalues are $+1$, -1 and any matrix with these properties may be written as $Q = PDP^T$, where D is diagonal with elements ± 1 and P is real orthogonal.

Position (1, 4): $Q \in \mathbb{R}^{n,n}$, $QQ^T = I$, $Q = -Q^T$. As in position (1, 4) of the complex chart, we have that any matrix with this property may be written as $Q = PDP^T$, where D is diagonal with an equal number of $+i$ and $-i$ diagonal entries.

TABLE 5.1
Summary of structure preserving algorithms for real matrices.

	$QQ^T = I$	$QJQ^T = J$	$Q = Q^T$	$Q = -Q^T$	$JQ = (JQ)^T$	$JQ = -(JQ)^T$
$QQ^T = I$	V	V	T	T	S	V
$QJQ^T = J$		?	S	S	T	T
$Q = Q^T$			V	T	Q	V
$Q = -Q^T$				V	V	V
$JQ = (JQ)^T$					?	T
$JQ = -(JQ)^T$						V

V: satisfactory algorithm not listed below.
Q: quaternion algorithm preserves all structure.
?: no structure preserving method known yet.
T: trivial case.
S: simultaneous diagonalization of commuting normal matrices.

Position (1, 5): $QQ^T = I$, $QJ = (QJ)^T = -JQ^T$. So

$$Q = \begin{bmatrix} F & G \\ H & -F^T \end{bmatrix},$$

with $G = G^T$, $H = H^T$. Set

$$W_1 = \frac{Q+Q^{-1}}{2} = \frac{Q+Q^T}{2} = \frac{1}{2}\begin{bmatrix} F+F^T & G+H^T \\ H+G^T & -F^T-F \end{bmatrix} = \begin{bmatrix} A & B \\ B & -A \end{bmatrix},$$

with $A = A^T$ and $B = B^T$. So W_1 is an antiquaternion matrix. Set

$$W_2 = \frac{Q-Q^{-1}}{2} = \frac{Q-Q^T}{2} = \frac{1}{2}\begin{bmatrix} F-F^T & G-H^T \\ H-G^T & -F^T+F \end{bmatrix} = \begin{bmatrix} C & D \\ -D & C \end{bmatrix},$$

with $C = -C^T$ and $D = D^T$. This is a skew symmetric quaternion matrix. We thus may use a variation of the simultaneous diagonalization procedure based upon the quaternion Jacobi method [B10].

Position (1, 6): $Q \in \mathbb{R}^{n,n}$, $QQ^T = I$, $JQ = -(JQ)^T$. Observe that

$$Q = \begin{bmatrix} F & G \\ H & F^T \end{bmatrix},$$

with $G = -G^T$, $H = -H^T$. Algorithm 3.8 preserves both structures and reduces Q to the form

$$\begin{bmatrix} \ddots & 0 \\ 0 & \ddots \end{bmatrix}.$$

The real orthogonal QR method of [A2] and [G5], applied to each block separately, is a good choice to finish diagonalizing A.

Position (2, 2): Except for the case of a rank 1 matrix H or G [M1], we do not know any structure preserving numerically stable method for this case.

Position (2, 3): $Q \in \mathbb{R}^{n,n}$, $Q^T J Q = J$, $Q = Q^T$. Set

$$W_1 = \frac{Q+Q^{-1}}{2} = \frac{1}{2}(Q - JQ^T J) = \frac{1}{2}\begin{bmatrix} F+K^T & G+G^T \\ G^T+G & K-F^T \end{bmatrix} =: \begin{bmatrix} A & B \\ B & -A \end{bmatrix},$$

$$B = B^T, \qquad A = A^T$$

and set

$$W_2 = \frac{Q-Q^{-1}}{2} = \frac{1}{2}(Q + JQ^T J) = \frac{1}{2}\begin{bmatrix} F+K^T & G-G^T \\ G^T-G & F^T+K \end{bmatrix} =: \begin{bmatrix} C & D \\ -D & C \end{bmatrix},$$

$$C = C^T, \qquad D = -D^T.$$

W_1 is a symmetric antiquaternion matrix. W_2 is a symmetric quaternion matrix. These may be simultaneously diagonalized by a symmetric quaternion Jacobi-like algorithm [B10].

Position (2, 4): $Q \in \mathbb{R}^{n,n}$, $Q^T J Q = J$, $Q = -Q^T$.

$$Q = \begin{bmatrix} F & G \\ -G^T & K \end{bmatrix},$$

with $F = -F^T$, $K = -K^T$. Set

$$W_1 = \frac{Q+Q^{-1}}{2} = \begin{bmatrix} F-K^T & G+G^T \\ -G^T-G & K-F^T \end{bmatrix} =: \begin{bmatrix} A & B \\ -B & A \end{bmatrix},$$

$A = -A^T$, $B =: B^T$, which is a skew symmetric quaternion and set

$$W_2 = \frac{Q-Q^{-1}}{2} = \begin{bmatrix} F+K^T & G-G^T \\ -G^T+G & K+F^T \end{bmatrix} =: \begin{bmatrix} C & D \\ D & -C \end{bmatrix},$$

which is a skew symmetric antiquaternion. These may be simultaneously diagonalized by a skew symmetric quaternion Jacobi method [B10].

Position (2, 5): $Q \in \mathbb{R}^{n,n}$, $Q^T J Q = J$, $JQ = (JQ)^T$ implies that $Q^2 = -I$. Matrices in this class are of the form PJP^{-1}, where P is symplectic.

Position (2, 6): $Q^T J Q = J$, $JQ = -(JQ)^T$ implies $Q^2 = I$; thus all eigenvalues are $+1$, -1. These matrices are of the form $P\left[\begin{smallmatrix} I & 0 \\ 0 & -I \end{smallmatrix}\right]P^{-1}$ for an orthogonal symplectic matrix P.

Position (3, 3): $Q \in \mathbb{R}^{n,n}$, $Q = Q^T$. The real symmetric case has been extensively studied [P5].

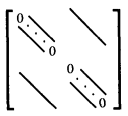
Position (3, 4): $Q \in \mathbb{R}^{n,n}$, $Q = Q^T = -Q^T$ implies $Q = 0$.

Position (3, 5): $Q \in \mathbb{R}^{n,n}$, $Q = Q^T$, $JQ = (JQ)^T$ implies that $JQ = -QJ$; thus Q is in $\mathscr{C}^a_{2n}(J)$, the anticentralizer of J, i.e., $Q = \left[\begin{smallmatrix} F & G \\ G & -F \end{smallmatrix}\right]$ with $F = F^T$, $G = G^T$, which is also antiquaternion. So we may apply the symmetric quaternion QR algorithm [B9].

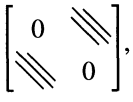
Position (3, 6): $Q \in \mathbb{R}^{n,n}$, $Q = Q^T$, $JQ = -(JQ)^T$ implies that $JQ = QJ$; thus Q is in $\mathscr{C}_{2n}(J)$, the centralizer of J, i.e., $Q = \left[\begin{smallmatrix} F & G \\ -G & F \end{smallmatrix}\right]$ with $F = F^T$, $G = -G^T$, which is via the X-trick equivalent to the Hermitian eigenproblem for $F + iG \in \mathbb{C}^{n,n}$ [P5].

Position (4, 4): $Q \in \mathbb{R}^{n,n}$, $Q = -Q^T$. Pardekooper's skew symmetric Jacobi method [P4] preserves the structure. So does the QR algorithm.

Position (4, 5): $Q \in \mathbb{R}^{n,n}$, $Q = -Q^T$, $JQ = (JQ)^T$ implies that $JQ = QJ$ and then $Q = \left[\begin{smallmatrix} F & G \\ -G & F \end{smallmatrix}\right]$ with $G = G^T$, $F = -F^T$. Applying Algorithm 3.8 we obtain a form



which is permutationally similar to the form



i.e., after a simple permutation Q splits into $\left[\begin{smallmatrix} 0 & \tilde{G} \\ -\tilde{G} & 0 \end{smallmatrix}\right]$ with a real symmetric tridiagonal G. Diagonalizing G by the symmetric QR or Jacobi method followed by an appropriate permutation, then yields the diagonal form for Q.

Position (4, 6): $Q \in \mathbb{R}^{n,n}$, $Q = -Q^T$, $JQ = -(JQ)^T$ implies $JQ = -QJ$, and then $Q = \left[\begin{smallmatrix} F & G \\ G & -F \end{smallmatrix}\right]$ with $G = -G^T$, $F = -F^T$. Applying Algorithm 3.8 we obtain a splitting $\left[\begin{smallmatrix} \tilde{F} & 0 \\ 0 & -\tilde{F} \end{smallmatrix}\right]$ and we can treat $\tilde{F}$ with the skew symmetric QR or Pardekooper's Jacobi method [P4].

Position (5, 5): Only in the case that for

$$Q = \begin{bmatrix} F & G \\ H & -F^T \end{bmatrix},$$

H or G are rank 1, is a structure preserving numerically stable QR-like method known [B16]. A Jacobi-like method is given in [B18].

Position (5, 6): $Q \in \mathbb{R}^{n,n}$, $QJ = (QJ)^T = -(QJ)^T$ implies $Q = 0$.

Position (6, 6): $Q \in \mathbb{R}^{n,n}$, $QJ = -(QJ)^T = JQ^T$. If we apply Algorithm 3.8, we obtain an orthogonal symplectic P such that

$$P^T Q P = \begin{bmatrix} \tilde{F} & \tilde{G} \\ 0 & \tilde{F}^T \end{bmatrix},$$

where $\tilde{F}$ is upper Hessenberg. A Schur-like form is readily available from the Schur decomposition of $\tilde{F}$.

6. Conclusion. We have shown that for most of our specially structured eigenvalue problems a few basic ingredients combine to form strongly numerically stable, structure preserving methods. The basic methods include the Francis QR algorithm, the quaternion QR algorithm, Jacobi methods, simultaneous diagonalization, and Algorithm 3.8. There are still some specially structured problems where we do not know structure preserving, unitary methods. The main reason is that the class of structure preserving unitary transformation is too small.

There are other special structures of interest that are not considered here. For example, a J-symmetric and $\left[\begin{smallmatrix} 0 & I \\ I & 0 \end{smallmatrix}\right]$-symmetric twofold structure is studied in [O1].

Some of the discussed problems (e.g., the symplectic eigenvalue problems) originate from specially structured matrix pencils. We speculate that QZ-type methods for matrices with very special structure can be obtained by generalizing the described algorithms to the pencil case.

Acknowledgments. We wish to thank Professor Greg Ammar for helpful conversations and Professor George Cybenko and an anonymous referee for suggestions, criticism, and corrections.

REFERENCES

[A1] V. M. ADAMJAN, D. Z. AROV, AND M. G. KREIN, *Analytic properties of Schmidt pairs for Hankel operators and the generalized Schur-Takagi problem*, Math. USSR Sb., 15 (1971), pp. 31–73.

[A2] G. S. AMMAR, W. B. GRAGG, AND L. REICHEL, *On the eigenproblem for orthogonal matrices*, in Proc. 25th Annual IEEE Conference on Decision and Control, Athens, GA, pp. 1963–1966.

[A3] W. F. ARNOLD III, *Numerical solution of algebraic Riccati equations*, Tech. Report, NWCTP 652, Naval Weapons Research Center, China Lake, CA, 1984.

[A4] ———, *Numerical solution of algebraic Riccati equations*, Ph.D. thesis, Department of Electrical Engineering, University of Southern California, Los Angeles, CA, 1983.

[A5] W. F. ARNOLD III AND A. J. LAUB, *Generalized eigenproblem algorithms and software for algebraic Riccati equations*, Proc. IEEE, 72 (1984), pp. 1748–1754.

[A6] M. ATHANS AND P. L. FALB, *Optimal Control*, MacGraw-Hill, New York, 1966.

[B1] A. Y. BARRAUD, *Investigation autour de la fonction signe d'une matrice, application à l'equation de Riccati*, RAIRO Automat., 13 (1979), pp. 335–368.

[B2] ———, *Produit étoile et fonction signe de matrice application à l'equation de Riccati dans le cas discret*, RAIRO Automat., 14 (1980), pp. 55–85.

[B3] J. G. F. BELINFANTE AND B. KOLMAN, *A Survey of Lie Groups and Lie Algebras with Applications and Computational Methods*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1972.

[B4] J. R. BUNCH, *The weak and strong stability of algorithms in numerical algebra*, Linear Algebra Appl., 88 (1987), pp. 49–66.

[B5] W. BUNSE AND A. BUNSE-GERSTNER, *Numerische Lineare Algebra*, Teubner-Verlag, Stuttgart, FRG, 1985.

[B6] A. BUNSE-GERSTNER, *Matrix factorization for symplectic* QR*-like methods*, Linear Algebra Appl., 83 (1986), pp. 49–77.

[B7] ———, *Eigenvalue algorithms for matrices with special structure*, Colloquia Mathematica Societatis Janos Bolyai 50, Numerical Methods, Janos Bolyai Society, Miskolc, Hungary, 1986, pp. 141–163.

[B8] ———, QR *like algorithms*, Habilitationsschrift, Fakultät für Mathematik, Universität Bielefeld, D-4800 Bielefeld 1, FRG, 1986.

[B9] A. BUNSE-GERSTNER, R. BYERS, AND V. MEHRMANN, *The quaternion* QR *algorithm*, Numer. Math., 55 (1989), pp. 83–95.

[B10] ———, *Numerical methods for simultaneous diagonalization*, preprint 90-009, Sonderforschungsbereich Diskrete Strukturen in der Mathematik, Universität Bielefeld, D-4800 Bielefeld 1, FRG, 1990; SIAM J. Matrix Anal. Appl., submitted.

[B11] A. BUNSE-GERSTNER AND W. B. GRAGG, *Singular value decompositions of complex symmetric matrices*, J. Comp. Appl. Math., 21 (1988), pp. 41–54.

[B12] A. BUNSE-GERSTNER AND V. MEHRMANN, *A symplectic* QR*-like algorithm for the solution of the real algebraic Riccati equation*, IEEE Trans. Automat. Control, 31 (1986), pp. 1104–1113.

[B13] A. BUNSE-GERSTNER, V. MEHRMANN, AND D. WATKINS, *An* SR *algorithm for Hamiltonian matrices, based on Gaussian elimination*, Methods Oper. Res., 58 (1989), pp. 339–358.

[B14] N. BURGOYNE AND R. CUSHMAN, *Normal forms for real linear Hamiltonian systems*, in The 1976 Ames Research Center (NASA) Conference on Geometric Control Theory, C. Martin, R. Herman, eds., Mathematical Science Press, Brookline, MA, 1976, pp. 483–528.

[B15] R. BYERS, *Hamiltonian and symplectic algorithms for the algebraic Riccati equation*, Ph.D. thesis. Center for Applied Mathematics, Cornell University, Ithaca, NY, 1983.

[B16] ———, *A Hamiltonian* QR*-algorithm*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 212–229.

[B17] ———, *Solving the algebraic Riccati equation with the matrix sign function*, Linear Algebra Appl., 85 (1987), pp. 267–279.

[B18] ———, *A Hamiltonian-Jacobi algorithm*, IEEE Trans. Automat. Control, to appear.

[C1] J. K. CULLUM AND R. A. WILLOUGHBY, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol.* I, *Theory*, Birkhäuser, Boston, 1985.

[C2] J. J. M. CUPPEN, *A divide and conquer method for the symmetric tridiagonal eigenproblem*, Numer. Math., 36 (1981), pp. 177–195.

[C3] G. CYBENKO, *Computing Pisarenko frequency estimates*, in Proc. Princeton Conference on Information Science and Systems, Department of Electrical Engineering, Princeton University, Princeton, NJ, 1985, pp. 587–591.

[D1] J. DONGARRA, J. R. BUNCH, C. B. MOLER, AND G. W. STEWART, *Linpack Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[D2] J. DONGARRA, J. R. GABRIEL, D. D. KÖLLING, AND J. H. WILKINSON, *The eigenvalue problem for Hermitian matrices with time reversal symmetry*, Linear Algebra Appl., 60 (1984), pp. 27–42.

[D3] J. J. DONGARRA AND D. C. SORENSEN, *A fully parallel algorithm for the symmetric eigenvalue problem*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 139–154.

[E1] P. J. EBERLEIN, *A Jacobi like method for the automatic computation of eigenvalues and eigenvectors of an arbitrary matrix*, J. Soc. Indust. Appl. Math., 10 (1962), pp. 74–88.

[E2] A. ERDMANN, *Über Jacobi-ähnliche Verfahren zur Lösung des Eigenwertproblems nicht-normaler komplexer Matrizen*, Dissertation, Fachbereich Mathematik, Fernuniversität–Gesamthochschule Hagen, FRG, 1984.

[F1] G. E. FORSYTHE AND P. HENRICI, *The cyclic Jacobi method for computing the principal values of a complex matrix*, Trans. Amer. Math. Soc., 94 (1960), pp. 1–23.

[F2] J. FRANCIS, *The* QR*-transformation, Part* I, Comput. J., 4 (1961), pp. 265–271.

[F3] ———, *The* QR*-transformation, Part* II, Comput. J., 5 (1962), pp. 332–345.

[G1] F. R. GANTMACHER, *Theory of Matrices*, I, II, Chelsea, New York, 1959.

[G2] J. D. GARDINER AND A. J. LAUB, *A generalization of the matrix sign function solution for algebraic Riccati equations*, Internat. J. Control, 44 (1986), pp. 823–832.

[G3] H. H. GOLDSTINE AND L. P. HORWITZ, *A procedure for the diagonalization of normal matrices*, J. Assoc. Comput. Mach., 6 (1959), pp. 176–195.

[G4] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

[G5] W. B. GRAGG, *The* QR *algorithm for unitary Hessenberg matrices*, J. Comput. Appl. Math., 16 (1986), pp. 1–8.

[G6] W. B. GRAGG AND L. REICHEL, *A divide and conquer algorithm for the unitary eigenvalue problem*, in Hypercube Multiprocessors, M. T. Heath, ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1987, pp. 639–647.

[G7] M. GUTKNECHT, *Rational Carathéodory–Fejér approximation on a disk, a circle and an interval*, J. Approx. Theory, 41 (1984), pp. 257–278.

[H1] S. J. HAMMARLING, *Newton's method for solving the algebraic Riccati equation*, NPL Report, DITC 12/82, National Physical Laboratory, Teddington, Middlesex, U.K., 1982.

[H2] V. HARI, *On cyclic Jacobi methods for the positive definite generalized problem*, Dissertation, Fachbereich Mathematik, Fernuniversität–Gesamthochschule Hagen, Hagen, Federal Republic of Germany, 1984.

[H3] G. A. HEWER, *An iterative technique for the computation of the steady state gains for the discrete optimal regulator*, IEEE Trans. Automat. Control, 16 (1971), pp. 382–384.

[H4] R. HORN AND C. R. JOHNSON, *Matrix Analysis*, Cambridge University Press, Cambridge, 1985.

[J1] C. G. J. JACOBI, *Über ein leichtes Verfahren, die in der Theorei der Säkularstörungen workommenden Gleichungen numerisch aufzulösen*, Crelle's J., 30 (1846), pp. 51–94.

[K1] D. L. KLEINMAN, *On an iterative technique for Riccati equation computations*, IEEE Trans. Automat. Control, 13 (1968), pp. 114–115.

[K2] E. G. KOGBETLIANTZ, *Solution of linear equations by diagonalization of coefficient matrix*, Quart. Appl. Math., 13 (1955), pp. 123–132.

[L1] A. J. LAUB, *A Schur method for solving algebraic Riccati equations*, IEEE Trans. Autom. Control, 24 (1979), pp. 913–921.

[L2] ———, *Algebraic aspects of generalized eigenvalue problems for solving Riccati equations*, Computational and Combinatorial Methods in Systems Theory, C. I. Byrnes and A. Lindquist, eds., Elsevier, North Holland, 1986, pp. 213–227.

[L3] ———, *Numerical linear algebra aspects of control design computations*, IEEE Trans. Automat. Control, 30 (1985), pp. 97–108.

[L4] A. J. LAUB AND K. MEYER, *Canonical forms for symplectic and Hamiltonian matrices*, Celest. Mechanics, 9 (1974), pp. 213–238.

[L5] K. H. LEE, *Generalized eigenproblem structures and solution methods for Riccati equations*, Ph.D. thesis, University of Southern California, Los Angeles, CA, 1983.

[M1] V. MEHRMANN, *A symplectic orthogonal method for single input or single output discrete time optimal linear quadratic control problems*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 221–248.

[M2] ———, *The linear quadratic control problem: Theory and numerical algorithms*, Habilitationsschrift, Fakultät für Mathematik, Universität Bielefeld, Bielefeld 1, Federal Republic of Germany, 1987.

[M3] P. C. MÜLLER, *Eigenverhalten rotationssymmetrischer Rotorsysteme*, Z. Angew. Math. Mech., 60 (1980), pp. 165–167.

[M4] ———, *Allgemeine lineare Theorie für Rotorsysteme ohne oder mit kleinen Unsymmetricien*, Ingenieur Archiv, 51 (1981), pp. 61–74.

[M5] F. D. MURNAGHAN AND A. WINTNER, *A canonical form for real matrices under orthogonal transformations*, Proc. Nat. Acad. Sci. U.S.A., 17 (1931), pp. 417–420.

[N1] B. NOBLE AND J. W. DANIEL, *Applied Linear Algebra*, Second Edition, Prentice-Hall, Englewood Cliffs, NJ, 1977.

[O1] J. OLSON, H. J. A. JENSEN, AND P. JØRGENSEN, *Solution of large matrix equations which occur in response theory*, J. Comput. Phys., 74 (1988), pp. 265–282.

[P1] C. PAIGE AND C. F. VAN LOAN, *A Schur decomposition for Hamiltonian matrices*, Linear Algebra Appl., 41 (1981), pp. 11–32.

[P2] C. PAIGE AND P. VAN DOOREN, *On the quadratic convergence of Kogbetliantz's algorithm for computing the singular value decomposition*, Linear Algebra Appl., 77 (1986), pp. 301–313.

[P3] T. PAPPAS, A. J. LAUB, AND N. R. SANDELL, *On the numerical solution of the discrete-time algebraic Riccati equation*, IEEE Trans. Automat. Control, 25 (1980), pp. 631–641.

[P4] M. H. C. PARDEKOOPER, *An eigenvalue algorithm for skew symmetric matrices*, Numer. Math., 17 (1971), pp. 189–202.

[P5] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[P6] H. J. PAYNE AND L. M. SILVERMAN, *On the discrete time algebraic Riccati equation*, IEEE Trans. Automat. Control, 18 (1973), pp. 226–234.

[P7] J. POTTER, *Matrix quadratic solutions*, SIAM J. Appl. Math., 14 (1966), pp. 496–501.

[R1] J. ROBERTS, *Linear model reduction and solution of the algebraic Riccati equation by the use of the sign function*, Internat. J. Control, 32 (1980), pp. 667–687.

[R2] N. RÖSCH, *Time-reversal symmetry, Kramer's degeneracy and the algebraic eigenvalue problem*, Chemical Phys., 80 (1983), pp. 1–5.

[R3] A. RUHE, *On the convergence of the Jacobi method for normal matrices*, BIT, 7 (1967), pp. 305–313.

[S1] A. P. SAGE, *Optimum Systems Control*, Prentice-Hall, Englewood Cliffs, NJ, 1966.

[S2] I. SCHUR, *Ein Satz über quadratische Formen mit komplexen Koeffizienten*, Amer. J. Math., 67 (1945), pp. 472–480.

[S3] D. S. SCOTT, M. T. HEATH, AND R. C. WARD, *Parallel block Jacobi eigenvalue algorithms using systolic arrays*, Linear Algebra Appl., 77 (1986), pp. 345–355.

[S4] L. M. SILVERMAN, *Discrete Riccati equations: Alternative algorithms, asymptotic properties and system theory interpretations*, in Advances in Control System, C. Leondes, ed., Academic Press, NY, 1976, pp. 313–386.

[S5] B. T. SMITH, J. M. BOYLE, J. J. DONGARRA, B. S. GARBOW, Y. IKEBE, V. C. KLEMA, AND C. B. MOLER, *Matrix Eigensystem Routines*—EISPACK *Guide*, Lecture Notes in Computer Science 6, Springer-Verlag, New York, 1976.

[S6] G. W. STEWART, *Introduction to Matrix Computations*, Academic Press, New York, 1973.

[S7] ———, *A Jacobi-like algorithm for computing the Schur decomposition of a nonhermitian matrix*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 853–864.

[T1] T. TAKAGI, *Remarks on an algebraic problem*, Japan J. Math., 2 (1925), pp. 13–17.

[T2] ———, *On an algebraic problem related to an analytic theorem of Landau*, Japan J. Math., 1 (1924), pp. 82–93.

[V1] P. VAN DOOREN, *A generalized eigenvalue approach for solving Riccati equations*, SIAM J. Sci. Statist. Comput., 2 (1981), pp. 121–135.

[V2] C. VAN LOAN, *A symplectic method for approximating all the eigenvalues of a Hamiltonian matrix*, Linear Algebra Appl., 61 (1984), pp. 233–251.

[W1] R. A. WHITESIDE, N. S. OSTLUND, AND P. G. HIBBARD, *A parallel Jacobi diagonalization algorithm for a loop multiple processor system*, IEEE Trans. Comput., 33 (1984), pp. 409–413.

[W2] H. J. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 454–458, April 1992

002

A SHARP BOUND ON POSITIVE SOLUTIONS OF LINEAR DIOPHANTINE EQUATIONS*

I. BOROSH† AND L. B. TREYBIG†

Abstract. Let $Ax = b$ be an $m \times n$ system of linear equations with rank $m < n$ and integer coefficients. Denote by Y the maximum of the absolute values of the $m \times m$ minors of the augmented matrix (A, b). It is proved that if the system has an integral solution $x = (x_i)$ with each $x_i \geqq 0$, and either $Ax = 0$ has no such solution which is nontrivial or there is an $m \times (m+1)$ submatrix A' of A with rank m such that $A'y = 0$ has a solution with positive integer components, then $Ax = b$ has an integral solution with each $0 \leqq x_i \leqq Y$. The bound is sharp.

Key words. linear Diophantine equations, positive integral solutions, minors, rank, bound, Smith's normal form, group knapsack

AMS(MOS) subject classifications. primary 15A36; secondary 11D04, 90C10

1. Introduction. It has been recently proved [5] that if a system $\mathbf{Ax} = \mathbf{b}$ of linear equations with integer coefficients and rank m has a solution in integers, it has a solution bounded by the maximum Y of the absolute values of the $m \times m$ minors of the augmented matrix $[\mathbf{A}, \mathbf{b}]$. This bound is sharp. However, other bounds in terms of the system's coefficients have been obtained for the case of a homogeneous system. A notable example is the classical Siegel's Lemma [6], which has been generalized by Bombieri and Vaaler [1]. Their result also guarantees the existence of a basis of small solutions. It was conjectured in [3] that a similar result to the one in [5], with the same bound, holds for nonnegative solutions of linear Diophantine equations.

Several papers by the authors and others have dealt with this conjecture, and partial results were obtained. On one hand, the conjecture was proved in special cases, such as the case $n = m + 1$, one equation, the case where $\mathbf{Ax} = \mathbf{0}$ has no nonnegative nontrivial solution [3], and the homogeneous case [2]. On the other hand, larger bounds have been obtained in the general case [3], [4], [7], and [11], the lowest one being $(n - m + 1)Y$ [4]. It has also been noted that the conjecture would give a sharp bound.

In this paper the conjecture will be proved for a large class of matrices $\mathbf{A}$, a class which includes those for which all $m \times m$ minors are nonzero.

As a byproduct of the proof, it turns out that the "small" positive solution obtained does not have too many nonzero entries, generalizing the well-known linear programming result that the existence of a feasible solution implies the existence of a basic feasible solution. Here, since we have an integer programming problem, we get only an "almost" basic solution.

2. Notation and main results. Let $\mathbf{A}$, $\mathbf{b}$ be, respectively, $m \times n$ and $m \times 1$ integer matrices with $m < n$, $\mathbf{b} \neq 0$, and rank $(\mathbf{A}) = m$. We will consider the system:

$$\mathbf{Ax} = \mathbf{b}. \tag{1}$$

Denote by $D(i_1, i_2, \cdots, i_m)$ the $m \times m$ minor of $[\mathbf{A}, \mathbf{b}]$ built on the columns $i_1 \cdots i_m$ of $[\mathbf{A}, \mathbf{b}]$, where $\mathbf{b}$ is considered as the $(m+1)$st column of $[\mathbf{A}, \mathbf{b}]$. Let

$$Y = \max_{1 \leqq i_1 < i_2 < \cdots < i_m \leqq m+1} |D(i_1, i_2, \cdots, i_m)|.$$

* Received by the editors December 26, 1989; accepted for publication October 4, 1990.

† Department of Mathematics, Texas A&M University, College Station, Texas 77843 (MathDept@VENUS.TAMU.edu).

It was proved in [3] that if $\mathbf{Ax} = \mathbf{0}$ has no nontrivial nonnegative solution, then every nonnegative solution of (1) is bounded by Y. We will therefore consider only the case where the homogeneous system has a nontrivial nonnegative solution. In fact, we will assume the following somewhat stronger condition:

(C) *There exists an* $m \times (m+1)$ *submatrix* $\mathbf{A}'$ *of* $\mathbf{A}$ *of rank* m *such that* $\mathbf{A}'\mathbf{x} = \mathbf{0}$ *has a nontrivial solution with all components positive.*

We will denote by $g \geqq 1$ the greatest common denominator (g.c.d.) of all $m \times m$ minors of $\mathbf{A}'$. Our main results are presented in the following theorem.

THEOREM 1. *If* $\mathbf{A}$ *satisfies condition* (C) *and* (1) *has a solution in nonnegative integers, then* (1) *has such a solution within the bound* Y. *Moreover, the number of nonzero entries in this solution is at most* $m + 1 + \log_2 g$.

COROLLARY 2. *If* $\mathbf{A}x = \mathbf{0}$ *has a nonnegative nontrivial integer solution and if none of the* $m \times m$ *minors of* $\mathbf{A}$ *is* 0 *and* (1) *has a solution in nonnegative integers, then it has such a solution within the bound* Y.

3. Proof of the main results. The proof of Theorem 1 follows a technique introduced by Gomory [9]. The system (1) is transformed into an abelian group knapsack problem using Smith's normal form of matrices. The next lemma appears as an exercise in [8, Ex. 12, p. 245]. We present a proof, since it is not long.

LEMMA. *Let* G *be a finite abelian group and let* $g_1, \cdots, g_k, g^*$ *be elements of* G. *Assume that*

$$\sum_{i=1}^{k} t_i g_i = g^* \tag{2}$$

has a solution in nonnegative integers $t_1, \cdots, t_k$. *Then* (2) *has a solution in nonnegative integers* $\bar{t}_1, \cdots, \bar{t}_k$ *such that*

$$\prod_{i=1}^{k} (1+\bar{t}_i) \leqq |G| \tag{3}$$

and

$$\sum_{i=1}^{k} \bar{t}_i \leqq |G| - 1, \tag{4}$$

where $|G|$ *denotes the order of* G.

Proof of the Lemma. It is enough to prove (3) since (4) is a trivial consequence. Let $\bar{t} = (\bar{t}_1, \cdots, \bar{t}_k)$ be the solution of (1) in nonnegative integers, such that $\sum_{i=1}^{k} \bar{t}_i$ is minimal, and if there is more than one such solution, let $\bar{t}$ be the smallest one in lexicographic order. Assume that (3) does not hold and let $T = \{\mathbf{s} = (s_1, \cdots, s_k) |$ each s_i is an integer and $0 \leqq s_i \leqq \bar{t}_i$ for $i = 1, \cdots, k\}$. Since

$$|T| = \prod_{i=1}^{k} (1+\bar{t}_i) \geqq |G| + 1,$$

there exists $\mathbf{s}^{(1)}, \mathbf{s}^{(2)}$ in T such that $\mathbf{s}^{(1)} \neq \mathbf{s}^{(2)}$ and

$$\sum_{i=1}^{k} s_i^{(1)} g_i = \sum_{i=1}^{k} s_i^{(2)} g_i.$$

Assume that $\mathbf{s}^{(1)}$ precedes $\mathbf{s}^{(2)}$ in lexicographic order. Clearly, $\bar{t} + (\mathbf{s}^{(1)} - \mathbf{s}^{(2)})$ and $\bar{\mathbf{t}} + (\mathbf{s}^{(2)} - \mathbf{s}^{(1)})$ are both solutions of (2) and have nonnegative entries. We must have

$$\sum_{i=1}^{k} s_i^{(1)} = \sum_{i=1}^{k} s_i^{(2)},$$

otherwise the sum of the entries of $\bar{\mathbf{t}} + (\mathbf{s}^{(1)} - \mathbf{s}^{(2)})$ or $\bar{\mathbf{t}} + (\mathbf{s}^{(2)} - \mathbf{s}^{(1)})$ would be smaller than $\sum_{i=1}^{k} \bar{t}_i$. Therefore,

$$\sum_{i=1}^{k} \bar{t}_i + (s_i^{(1)} - s_i^{(2)}) = \sum_{i=1}^{k} \bar{t}_i,$$

and since $\mathbf{s}^{(1)}$ precedes $\mathbf{s}^{(2)}$ in the lexicographic order, $\bar{\mathbf{t}} + \mathbf{s}^{(1)} - \mathbf{s}^{(2)}$ precedes $\bar{\mathbf{t}}$, contradicting the minimality of $\bar{\mathbf{t}}$. □

Proof of Theorem 1. Assume, without loss of generality, that the matrix $\mathbf{A}'$ in condition (C) consists of the first $(m + 1)$ columns of $\mathbf{A}$. Let $\mathbf{S}$ be Smith's normal form of $\mathbf{A}'$ [10, p. 25]. There exist unimodular matrices $\mathbf{P}$, $\mathbf{Q}$ such that

$$\mathbf{PA'Q} = \mathbf{S} = \begin{pmatrix} s_1 & 0 \cdots 0 & 0 \\ 0 & s_2 \cdots 0 & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots s_m & 0 \end{pmatrix}, \tag{5}$$

where $s_1 = d_1(A')$ and

$$s_i = \frac{d_i(\mathbf{A}')}{d_{i-1}(\mathbf{A}')} \quad \text{for } i = 2, \cdots, m,$$

and $d_i(\mathbf{A}')$ is the g.c.d. of all $i \times i$ minors of $\mathbf{A}'$. In particular

$$\prod_{i=1}^{m} s_i = d_m(\mathbf{A}') = g.$$

Partition $\mathbf{A}$ as $\mathbf{A} = (\mathbf{A}'|\mathbf{N})$ and partition $\mathbf{x}$ accordingly into $\mathbf{x}^T = [\mathbf{x}'|\mathbf{x}'']^T$, so that (1) becomes:

$$\mathbf{A'x'} + \mathbf{Nx''} = \mathbf{b}. \tag{6}$$

Define $\mathbf{y}$ with $\mathbf{y}^T = (y_1, \cdots, y_{m+1})$ by

$$\mathbf{x'} = \mathbf{Qy}, \tag{7}$$

and get from (5) and (6):

$$\mathbf{Sy} + \mathbf{PNx''} = \mathbf{Pb}. \tag{8}$$

Let $\mathbf{g}^{(1)}, \cdots, \mathbf{g}^{(k)}$ be the columns of $\mathbf{PN}$ with $k = n - m - 1$ and $g^* = \mathbf{Pb}$. Then (8) can be written as a system of congruences:

$$x_{m+2}\mathbf{g}^{(1)} + \cdots + x_n\mathbf{g}^{(k)} \equiv \mathbf{g}^* \bmod \begin{pmatrix} s_1 \\ s_2 \\ \vdots \\ s_m \end{pmatrix}. \tag{9}$$

Any solution to (9) provides a nonnegative integer vector $\mathbf{x}''$ and an integer vector $\mathbf{y}$ satisfying (8) and using (7), $\mathbf{x}'$ is computed (an arbitrary integer value can be given for y_{m+1}) so that $\mathbf{x}'$ and $\mathbf{x}''$ satisfy (6). The process can be reversed since $\mathbf{Q}$ is unimodular.

Equation (9) can be considered as a group knapsack problem, the group being $G = Z_{s_1} \oplus Z_{s_2} \oplus \cdots \oplus Z_{s_m}$ with

$$|G| = \prod_1^m s_i = g.$$

According to the lemma, (9) has a solution in nonnegative integers $\bar{x}_{m+2}, \cdots, \bar{x}_n$ such that

$$\prod_{i=m+2}^{n} (1+\bar{x}_i) \leqq g,$$

and therefore

$$\sum_{i=m+2}^{n} \bar{x}_i \leqq g-1.$$

We use now the process described earlier to complete $(\bar{x}_{m+2}, \cdots, \bar{x}_n)$ to an integer solution $\bar{\mathbf{x}}$ of (1). Let $\mathbf{h}$ be a solution to $\mathbf{A}'\mathbf{x} = 0$, which by condition (C), has positive entries. We can always assume that

$$\mathbf{h} = (h_1, \cdots, h_{m+1})$$

$$= \frac{1}{g}(|D(2,3,\cdots,m+1)|, |D(1,3,\cdots,m+1)|, \cdots, |D(1,2,\cdots,m)|).$$

For every integer u, $\mathbf{z} = \bar{\mathbf{x}} + u\mathbf{h}$ is an integer solution to (1), and $z_i \geqq 0$ for $i = m+2, \cdots, n$. Let u be the smallest integer such that $z_i \geqq 0$ for all $i = 1, \cdots, n$. Then there exists i_0, $1 \leqq i_0 \leqq m+1$ such that $z_{i_0} < h_{i_0}$, and we can assume, without loss of generality, that $i_0 = m+1$. Then solving (1) for $z_1, \cdots, z_m$, we get by Cramer's formula

$$Dz_i = d_{i,m+1}z_{m+1} + \sum_{l=m+2}^{n} d_{i,l}z_l + e_i, \qquad i = 1, \cdots, m, \tag{10}$$

where $D = D(1,2,\cdots,m)$ and $d_{i,j}$ and e_i are all $m \times m$ minors of $[\mathbf{A}, \mathbf{b}]$. By construction we have

$$z \geqq 0 \quad \text{for } i = 1, \cdots, n,$$

$$z_{m+1} < h_{m+1} = \frac{|D|}{g},$$

and

$$\sum_{l=m+2}^{n} z_l = \sum_{l=m+2}^{n} \bar{x}_l \leqq g-1.$$

For $1 \leqq i \leqq m$, we get from (10):

$$0 \leqq z_i \leqq \frac{|d_{i,m+1}|}{|D|} z_{m+1} + \sum_{l=m+2}^{n} \frac{|d_{i,l}|}{|D|} z_l + \frac{|e_i|}{|D|}$$

$$\leqq \frac{Y}{|D|}\left(\frac{|D|}{g} - 1\right) + \frac{Y}{|D|}(g-1) + \frac{Y}{|D|}$$

$$\leqq Y\left(\frac{1}{g} + \frac{g-1}{|D|}\right) = Y\left[1 - \left(1-\frac{1}{g}\right)\left(1-\frac{g}{|D|}\right)\right] \leqq Y.$$

In fact, we get a strict inequality unless $g = 1$ or $|D| = g$.

To prove the second claim of the theorem, if we let r be the number of the nonzero z_i for $i = m + 2, \cdots, n$, then

$$g \geqq \prod_{i=m+2}^{n} (1 + z_i) \geqq 2^r. \qquad \square$$

Proof of Corollary 2. Let $\mathbf{A}$ satisfy the conditions in Corollary 2 and let $\mathbf{A}'$ be an $m \times k$ submatrix of $\mathbf{A}$ such that $\mathbf{A}'\mathbf{x} = 0$ has a positive solution and k is minimal. We can assume, without loss of generality, that $\mathbf{A}'$ consists of the first k columns of $\mathbf{A}$. Clearly, $k \geqq m + 1$, otherwise some m columns of $\mathbf{A}$ would be linearly dependent, and since k is minimal, we have $x_i > 0$ for $i = 1, \cdots, k$. To show that $\mathbf{A}$ satisfies condition (C), we need only show that $k = m + 1$. If $k > m + 1$, there exists a solution $y \in Z^m$ of $\mathbf{A}'\mathbf{y} = 0$, independent of $\mathbf{x}$ over the rationals Q. Let

$$\lambda = \max \left\{ -\frac{x_i}{y_i} \middle| y_i > 0 \right\} = -\frac{x_{i_0}}{y_{i_0}}$$

and consider the solution $\mathbf{w} = \mathbf{x} + \lambda\mathbf{y}$ to $\mathbf{A}x = 0$. Then $\mathbf{w} \geqq 0$ and $w_{i_0} = 0$, contradicting the minimality of k. $\square$

Remark. It does not seem that condition (C) is necessary for the existence of a solution to (1) within the bound Y. In fact, we were able to prove it for the case where $\mathbf{A}$ is a $2 \times n$ matrix by noticing that if a minor of $\mathbf{A}$ is 0, the corresponding columns are integer multiples of the same column, and this enables the reduction of the problem to the $2 \times (n - 1)$ case.

REFERENCES

[1] E. BOMBIERI AND J. VAALER, *On Siegel's Lemma*, Invent. Math., 73 (1983), pp. 11–32.
[2] I. BOROSH, *A sharp bound for positive solutions of homogeneous linear diophantine equations*, Proc. Amer. Math. Soc., 60 (1976), pp. 19–21.
[3] I. BOROSH AND L. B. TREYBIG, *Bounds on positive integral solutions of linear diophantine equations*, Proc. Amer. Math. Soc., 55 (1976), pp. 299–304.
[4] I. BOROSH, M. FLAHIVE, AND L. B. TREYBIG, *Small solutions of linear diophantine equations*, Discrete Math., 58 (1986), pp. 215–220.
[5] I. BOROSH, M. FLAHIVE, D. RUBIN, AND L. B. TREYBIG, *A sharp bound for solutions of linear diophantine equations*, Proc. Amer. Math. Soc., 105 (1989), pp. 844–846.
[6] J. W. S. CASSELS, *An introduction to Diophantine approximations*, in Cambridge Tracts in Math. Phys. No. 45, Cambridge University Press, New York, 1957.
[7] J. VON ZUR GATHEN AND M. SIEVEKING, *A bound on solutions of linear equalities and inequalities*, Proc. Amer. Math. Soc., 72 (1978), pp. 155–158.
[8] R. S. GARFINKEL AND G. L. NEMHAUSER, *Integer Programming*, John Wiley and Sons, New York, 1972.
[9] R. E. GOMORY, *Some polyhedra related to combinatorial problems*, Linear Algebra Appl., 2 (1969), pp. 451–558.
[10] M. NEWMAN, *Integral Matrices*, Academic Press, New York, London, 1972.
[11] C. H. PAPADIMITRIOU, *On the complexity of integer programming*, J. Assoc. Comput. Mach., 28 (1981), pp. 765–768.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 459–473, April 1992

003

OPTIMAL AND SUPEROPTIMAL CIRCULANT PRECONDITIONERS*

EVGENIJ E. TYRTYSHNIKOV†

Abstract. While applying iterative methods to solve a linear algebraic system with matrix A, one often uses some preconditioner C. Two kinds of preconditioners are investigated: the "optimal" one, which minimizes $\|C-A\|_F$, and the "superoptimal" one, which minimizes $\|I-C^{-1}A\|_F$. It is proved that both inherit nonsingularity and positive-definiteness from A. Fast algorithms for finding superoptimal preconditioners are suggested that take $O(n^2 \log_2 n)$ operations in case of arbitrary A of order n, and only $O(n \log_2 n)$ operations in case of Toeplitz or doubly Toeplitz A.

Key words. Toeplitz systems, circulants, iterative methods, preconditioners

AMS(MOS) subject classification. 65

1. Introduction.

Consider the system of linear algebraic equations

$$A_n u_n = f_n, \tag{1.1}$$

where A is a nonsingular real dense matrix of order n, and suppose that direct methods (such as Gaussian elimination) require more computer resources than we have at our disposal. It is expedient to apply iterative methods in this situation.

Fast convergence of iterations is usually connected with choosing the splitting

$$A_n = C_n - K_n, \tag{1.2}$$

where C_n is a nonsingular "easily invertible" matrix considered as a good approximation to A_n. When choosing C_n from some set M_n, it seems worthwhile to consider the following optimization problem:

$$\|I - C_n^{-1}A_n\| \to \min, \qquad C_n \in M_n, \tag{1.3}$$

or, in the simplified form, the following one:

$$\|C_n - A_n\| \to \min, \qquad C_n \in M_n. \tag{1.4}$$

We shall call C_n an optimal preconditioner if it satisfies (1.4) and a superoptimal preconditioner if it satisfies (1.3).

This paper discusses optimal and superoptimal preconditioners in the cases when M_n is the set of circulant matrices, doubly circulant matrices, or, in the general case, multilevel circulant matrices. Assume that the Frobenius norm is used in (1.3) and (1.4).

We recall that $C_n = [c_{ij}]_{ij=0}^{n-1}$ is called a circulant matrix if

$$c_{ij} = c_{i-j(\bmod n)}, \qquad 0 \leqq i, j \leqq n-1; \tag{1.5}$$

$T_n = [t_{ij}]_{ij=0}^{n-1}$ is called a Toeplitz matrix if

$$t_{ij} = t_{i-j}, \qquad 0 \leqq i, j \leqq n-1. \tag{1.6}$$

For $n = 4$ we have:

$$C = \begin{bmatrix} c_0 & c_3 & c_2 & c_1 \\ c_1 & c_0 & c_3 & c_2 \\ c_2 & c_1 & c_0 & c_3 \\ c_3 & c_2 & c_1 & c_0 \end{bmatrix}, \qquad T = \begin{bmatrix} t_0 & t_{-1} & t_{-2} & t_{-3} \\ t_1 & t_0 & t_{-1} & t_{-2} \\ t_2 & t_1 & t_0 & t_{-1} \\ t_3 & t_2 & t_1 & t_0 \end{bmatrix}.$$

* Received by the editors January 22, 1990; accepted for publication (in revised form) July 31, 1990.

† Department of Numerical Mathematics, Union of Soviet Socialist Republics (USSR) Academy of Sciences, Leninskij Prosp. 32-A, Moscow 117334, Union of Soviet Socialist Republics.

Further, there are many cases in which some matrix A of order n may be considered as a multilevel matrix. This means that each of its elements

$$a_{ij}, \quad 0 \leqq i, j \leqq n-1,$$

can be pointed to with the aid of s pairs of indices

$$i_l, \quad j_l, \quad 0 \leqq i_l, j_l \leqq n_l - 1 \qquad (n = n_1 \cdots n_s):$$

$$\begin{aligned} a_{ij} &= a_{i_1 j_1; \cdots; i_s j_s}, \\ i &= i_1 n_2 n_3 \cdots n_s + i_2 n_3 \cdots n_s + \cdots + i_{s-1} n_s + i_s, \\ j &= j_1 n_2 n_3 \cdots n_s + j_2 n_3 \cdots n_s + \cdots + j_{s-1} n_s + j_s. \end{aligned} \tag{1.7}$$

The number s is called the number of levels of A; $n_1, \cdots, n_s$ are the orders of these levels.

Let all the pairs of indices be fixed, save one pair corresponding, for example, to the level l. Then one may write $a_{ij} = \tilde{a}_{i_l j_l}$. The level l of A is called circulant if

$$\tilde{a}_{i_l j_l} = \tilde{a}_{i_l - j_l (\mathrm{mod}\, n_l)}, \qquad 0 \leqq i_l, j_l \leqq n_l - 1,$$

and is called Toeplitz if

$$\tilde{a}_{i_l j_l} = \tilde{a}_{i_l - j_l}, \qquad 0 \leqq i_l, j_l \leqq n_l - 1.$$

Thus, a doubly Toeplitz matrix is a matrix with two levels, i.e., $s = 2$ and every level is circulant. For $n = 6$, $n_1 = 3$, $n_2 = 2$, a doubly circulant matrix looks like this:

$$C = \begin{bmatrix} c_{0;0}\, c_{0;1} & c_{2;0}\, c_{2;1} & c_{1;0}\, c_{1;1} \\ c_{0;1}\, c_{0;0} & c_{2;1}\, c_{2;0} & c_{1;1}\, c_{1;0} \\ c_{1;0}\, c_{1;1} & c_{0;0}\, c_{0;1} & c_{2;0}\, c_{2;1} \\ c_{1;1}\, c_{1;0} & c_{0;1}\, c_{0;0} & c_{2;1}\, c_{2;0} \\ c_{2;0}\, c_{2;1} & c_{1;0}\, c_{1;1} & c_{0;0}\, c_{0;1} \\ c_{2;1}\, c_{2;0} & c_{1;1}\, c_{1;0} & c_{0;1}\, c_{0;0} \end{bmatrix}.$$

It is well known that circulant and multilevel circulant matrices are easily invertible by means of Fast Fourier Transforms (FFTs). One step of inverse iteration with such a matrix takes only $O(n \log n)$ operations. Therefore, one requirement on the preconditioner—that it must be easily invertible—is fulfilled in this case.

Note, by the way, that for the multilevel circulant matrix the inverse keeps the same structural property. Hence, for this class of matrices there is no difference between the application of implicit or explicit preconditioners.

Circulant and doubly circulant preconditioners were earlier discussed in [1]–[3], [6], [8], [9], and [12]. There are many concrete examples of their efficiency. The theoretical background, however, is often lacking. Recently Strang and Chan [3], [12] made a great stride in this direction. They proved that if A_n is a symmetric Toeplitz matrix generated by the Fourier–Laurent series of a real positive function and C_n is the symmetric circulant matrix which copies the middle diagonals of A_n, then the eigenvalues of $C_n^{-1} A_n$ cluster around 1 (except for a few of them). This result is formulated asymptotically (with $n \to \infty$), but numerical examples often show the clustering of eigenvalues even if n is not large.

The effect of Strang's preconditioner seems amazing, for its construction does not involve as much as half the information about A_n. Apparently, using the information in a more complete way, we may expect a better result. We hope this can be done by means of the solutions of the above problems, (1.3) and (1.4).

Recently, Chan [2] has considered the optimal preconditioner $C_n = [c_{i-j}]$, which corresponds to (1.4) when $A_n = [a_{p-q}]$ is a Toeplitz matrix and M_n is the set of all circulant matrices of order n. It can be easily verified that

$$c_{i-j} = \frac{1}{n} \sum_{p-q = i-j (\mathrm{mod}\ n)} a_{p-q}. \tag{1.8}$$

Chan had a series of numerical tests with various symmetric A_n. In all cases, the optimal preconditioner caused the spectrum of $C_n^{-1} A_n$ to be located in a more narrow interval than Strang's preconditioner did. But the clustering of the spectrum was not improved.

The real advantage of the optimal preconditioner is that it is guaranteed to be positive definite if the original matrix is. This fact is one of our results. Note that Strang's preconditioner could be singular or indefinite.

When exploiting circulant preconditioners, we should not confine ourselves only to Toeplitz matrices given as A_n. We expect circulant preconditioners to be helpful for a wide class of linear algebraic systems tied with the solution of integral equations. We base this statement on the experience of solving some 2-dimensional and 3-dimensional problems in subsonic aerodynamics. Doubly circulant preconditioners are natural for the special difficulties arising in 3-dimensional problems. In particular, doubly circulant preconditioners allow the fast solution of doubly Toeplitz systems in many cases.

In § 2 we shall formulate the auxiliary statements for further references. Section 3 is devoted to optimal preconditioners, and § 4 deals with superoptimal ones. We shall prove that both inherit the property of positive-definiteness from A_n.

In § 5 we propose a fast algorithm for the calculation of superoptimal preconditioners. It has $O(n^2 \log_2 n)$ complexity if A_n is an arbitrary matrix of order n. A special algorithm is derived for A_n being Toeplitz or doubly Toeplitz, which requires only $O(n \log_2 n)$ operations. Some numerical results are presented in § 6.

2. Auxiliary statements.

THEOREM 2.1. *Let the set of indices*

$$N_n = \{(i,j): i,j = 1, \cdots, n\}$$

be written as a union

$$N_n = \bigcup_{l=0}^{k} N_n^{(l)}$$

of nonintersecting subsets $N_n^{(l)}$, the lth one with n_l elements, and suppose that M_n comprises matrices $C_n = [c_{ij}]$ characterized by the following conditions:

(1) *if* $(i,j) \in N_n^{(0)}$, *then* $c_{ij} = 0$;

(2) *if* $(i_1, j_1), (i_2, j_2) \in N_n^{(l)}$ *for some* $1 \leqq l \leqq k$, *then*

$$c_{i_1 j_1} = c_{i_2 j_2}.$$

Then the optimal preconditioner for $A_n = [a_{pq}]$ is defined as follows:

$$\begin{aligned} c_{ij} &= 0, && (i,j) \in N_n^{(0)}; \\ c_{ij} &= \sum_{(p,q) \in N_n^{(l)}} a_{pq}/n_l, && (i,j) \in N_n^{(l)}, \quad 1 \leqq l \leqq k. \end{aligned} \tag{2.1}$$

We omit the proof because it is trivial. Note that (1.8) is a simple corollary of Theorem 2.1 (in this case $N_0^{(0)} = \varnothing$). There is no difficulty in getting optimal precon-

ditioners in all the situations this theorem deals with. In the case $n_1 = \cdots = n_k = 1$, M_n consists of matrices with some prescribed sparse pattern. One could, if needed, easily get an analogous statement involving a generalized Frobenius norm with weights.

Below, we shall use the well-known spectral theorem for circulant and multilevel circulant matrices.

THEOREM 2.2. *Let A be a multilevel circulant matrix of order n, which has s circulant levels of orders $n_1, \cdots, n_s$ $(n = n_1 \cdots n_s)$. Then*

$$A = n^{-1} F_n^* \operatorname{diag}(F_n a) F_n, \tag{2.2}$$

where

$$F_n = F_{n_1} \times \cdots \times F_{n_s} \quad \text{and} \quad F_{n_l} = \left[\exp\left(i \frac{2\pi km}{n_l} \right) \right]$$

is the Fourier transform matrix (i is the imaginary unit); a is the first column of A; and $\operatorname{diag}(F_n a)$ *is the diagonal matrix composed of the components of the vector $F_n a$.*

Equation (2.2) demonstrates that a multilevel circulant is unitarily diagonalizable, its eigenvalues are the components of the vector $F_n a$, and its eigenvectors are the columns of F_n^*. Since F_n^* differs from F_n by column permutations, we conclude that the eigenvectors of A comprise the columns of F_n.

3. Optimal preconditioners. Let $A = [a_{ij}]$ be a real nonsingular matrix of order n and let $C = [c_{i-j}]$ be a real circulant matrix of the same order. Assume that C is the solution of the optimization problem (1.4):

$$\|C - A\|_F \to \min, \qquad C \in \mathscr{T}_n, \tag{3.1}$$

where $\mathscr{T}_n$ is the set of all real circulants of order n. Then applying Theorem 2.1 we get

$$c_k = \frac{1}{n} \sum_{p - q = k (\operatorname{mod} n)} a_{pq}, \qquad k = 0, 1, \cdots, n-1. \tag{3.2}$$

THEOREM 3.1. *Let A be a real matrix of order n, such that $(Ax, x) > 0$ for $x \neq 0$, and suppose that C is defined by* (3.2). *Then C is nonsingular and, moreover, $(Cx, x) > 0$ for $x \neq 0$.*

If $A = A^T$, then also $C = C^T$. Setting the eigenvalues of A and C in increasing order, we have the following inequalities:

$$\begin{aligned} &\lambda_1(A) + \cdots + \lambda_j(A) \geqq \lambda_1(C) + \cdots + \lambda_j(C), \\ &\lambda_n(A) + \cdots + \lambda_{n-j+1}(A) \leqq \lambda_n(C) + \cdots + \lambda_{n-j+1}(C), \qquad j = 1, \cdots, n. \end{aligned} \tag{3.3}$$

Proof. Consider the orthonormal system of vectors

$$u_k = \frac{1}{\sqrt{n}} [\varepsilon^{k \cdot 0} \varepsilon^{k \cdot 1} \cdots \varepsilon^{k \cdot (n-1)}]^T, \qquad k = 0, 1, \cdots, n-1, \tag{3.4}$$

where

$$\varepsilon = \exp\left(i \frac{2\pi}{n} \right).$$

By Theorem 2.2, u_k is the eigenvector of the circulant matrix C which corresponds to

the eigenvalue (Cu_k, u_k). From (3.2) we find

$$(Cu_k, u_k) = \frac{1}{n} \sum_{p=0}^{n-1} \varepsilon^{-kp} \sum_{q=0}^{n-1} c_{p-q} \varepsilon^{kq} = \sum_{l=0}^{n-1} c_l \varepsilon^{-kl}$$

$$= \sum_{l=0}^{n-1} \varepsilon^{-kl} \frac{1}{n} \sum_{p-q=l(\mathrm{mod}\, n)} a_{pq} = \frac{1}{n} \sum_{p=0}^{n-1} \varepsilon^{-kp} \sum_{q=0}^{n-1} a_{pq} \varepsilon^{kq} = (Au_k, u_k),$$

that is,

$$(Cu_k, u_k) = (Au_k, u_k), \qquad k = 0, 1, \cdots, n-1. \tag{3.5}$$

We assumed that $(Au_k, u_k) > 0$ for all k and, therefore, that all eigenvalues of C differ from zero. Hence, C is nonsingular.

Further, take a real nonzero vector

$$x = \sum_{k=0}^{n-1} \alpha_k u_k \tag{3.6}$$

and write

$$(Cx, x) = \left(\sum_{k=0}^{n-1} \alpha_k Cu_k, \sum_{l=0}^{n-1} \alpha_l u_l \right) = \sum_{k=0}^{n-1} |\alpha_k|^2 (Cu_k, u_k) > 0. \tag{3.7}$$

Thus, $(Cx, x) > 0$ for $x \neq 0$.

If $A = A^T$, then

$$\sum_{p-q=k(\mathrm{mod}\, n)} a_{pq} = \sum_{p-q=n-k(\mathrm{mod}\, n)} a_{pq}, \qquad k = 1, \cdots, n-1. \tag{3.8}$$

From (3.2), we obtain

$$c_k = c_{n-k}, \qquad k = 1, \cdots, n-1, \tag{3.9}$$

and this is equivalent to the symmetry of C.

Let us reorder vectors $u_0, \cdots, u_{n-1}$ to get vectors $v_1, \cdots, v_n$ so that

$$(Cv_1, v_1) \geqq (Cv_2, v_2) \geqq \cdots \geqq (Cv_n, v_n).$$

Hence,

$$\lambda_j(C) = (Cv_j, v_j).$$

For an orthonormal system $z_1, \cdots, z_j$, the inequalities

$$\sum_{l=1}^{i} \lambda_{n-l+1}(A) \leqq \sum_{l=1}^{j} (Az_j, z_j) \leqq \sum_{l=1}^{j} \lambda_l(A)$$

are fulfilled (see [7]). Allowing for (3.4) we conclude that (3.3) holds. This completes the proof. □

To proceed further, let us consider problem (3.1) with the assumption that $\mathscr{T}_n$ is the set of all real s-level matrices which have circulant levels of orders $n_1, \cdots, n_s$. Any matrix $C \in \mathscr{T}_n$ is defined by its first column. In accordance with (1.7), we write

$$c_{i_1; \cdots; i_s} \equiv c_{i_1 0; \cdots; i_s 0}. \tag{3.10}$$

By Theorem 2.1 we have

$$c_{i_1;\cdots;i_s} = \frac{1}{n} \sum_{\substack{p_1 - q_1 = i_1 (\mathrm{mod}\, n_1) \\ \cdots \\ p_s - q_s = i_s (\mathrm{mod}\, n_s)}} a_{p_1 q_1;\cdots;p_s q_s}. \tag{3.11}$$

THEOREM 3.2. *Let A be a real matrix of order $n = n_1 \cdots n_s$ and C be the s-level matrix with circulant levels of orders $n_1, \cdots, n_s$, defined by* (3.11). *Suppose that $(Ax, x) > 0$ for $x \neq 0$. Then C is nonsingular and, moreover, $(Cx, x) > 0$ for $x \neq 0$.*

If $A = A^T$, then also $C = C^T$. With this, the eigenvalues of A and C, taken in increasing order, satisfy (3.3).

Proof. Denote by ε_l any n_lth degree root of unity; set

$$u_l \equiv \frac{1}{\sqrt{n_l}} [\varepsilon_l^0 \varepsilon_l^1 \cdots \varepsilon_l^{n_l - 1}]^T, \qquad l = 1, \cdots, s, \tag{3.12}$$

and consider the vector

$$u = u_1 \times \cdots \times u_s.$$

According to Theorem 2.2, u is a normalized eigenvector of C, which is coupled with the eigenvalue (Cu, u). Choosing various roots ε_l and forming corresponding vectors u we can get an orthonormal system of eigenvectors of C. Pursuing the analogy with (3.5) we find

$$(Cu, u) = (Au, u)$$

for vectors u mentioned above. This implies that C is nonsingular and that $(Cx, x) > 0$ with $x \neq 0$.

If $A = A^T$, then

$$\sum_{\substack{p_1 - q_1 = k_1 (\mathrm{mod}\, n_1) \\ \cdots \\ p_s - q_s = k_s (\mathrm{mod}\, n_s)}} a_{p_1 q_1;\cdots;p_s q_s} = \sum_{\substack{p_1 - q_1 = n_1 - k_1 (\mathrm{mod}\, n_1) \\ \cdots \\ p_s - q_s = n_s - k_s (\mathrm{mod}\, n_s)}} a_{p_1 q_1;\cdots;p_s q_s}, \tag{3.13}$$

$$0 \leqq k_1 \leqq n_1 - 1, \cdots, 0 \leqq k_s \leqq n_s - 1.$$

This means that C is symmetric. The relations for the eigenvalues of A and C are derived by the same reasoning we have used in the proof of Theorem 3.1. Actually, Theorem 3.1 is simply a corollary of Theorem 3.2 corresponding to the case $s = 1$. □

4. Superoptimal preconditioners. Suppose that C is the solution of the optimization problem of the form (1.3):

$$\| I - C^{-1} A \|_F \to \min, \qquad C \in \mathring{\mathscr{T}}_n, \tag{4.1}$$

where $\mathring{\mathscr{T}}_n$ is the set of real nonsingular circulants of order n. Let

$$\hat{C} = C^{-1} = \sum_{j=0}^{n-1} \hat{c}_j Q^j \tag{4.2}$$

where

$$Q = \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 & 1 \\ 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & \cdots & 0 & 0 \\ \cdots & \cdots & & \cdots & & \cdots \\ 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}. \tag{4.3}$$

Instead of (4.1), consider the following problem:

$$\|I-\hat{C}A\|_F \to \min, \qquad \hat{C}\in\mathscr{T}_n, \tag{4.4}$$

where $\mathscr{T}_n$ is the set of all real circulants of order n.

In order to solve problem (4.4), let us investigate the functional

$$\begin{aligned}\mathscr{F}(\hat{c}_0,\hat{c}_1,\cdots,\hat{c}_{n-1}) &= \left\| I-\sum_{j=0}^{n-1}\hat{c}_jQ^jA\right\|_F^2 \\ &= \operatorname{tr}\left(\left(I-\sum_{i=0}^{n-1}\hat{c}_iA^TQ^{-i}\right)\left(I-\sum_{j=0}^{n-1}\hat{c}_jQ^jA\right)\right) \\ &= n-\sum_{i=0}^{n-1}\hat{c}_if_i+\sum_{i=0}^{n-1}\sum_{j=0}^{n-1}\hat{c}_i\hat{c}_ju_{ij}\end{aligned} \tag{4.5}$$

where

$$f_i \equiv \operatorname{tr}(A^TQ^{-i}+Q^iA), \tag{4.6}$$

$$u_{ij} \equiv \operatorname{tr}(A^TQ^{j-i}A). \tag{4.7}$$

Compute the partial derivatives of $\mathscr{F}$ and require them to be zero:

$$\frac{\partial\mathscr{F}}{\partial\hat{c}_i} = -f_i+\sum_{j=0}^{n-1}\hat{c}_j(u_{ij}+u_{ji}) = 0, \tag{4.8}$$

or, in a matrix-vector notation,

$$(U+U^T)\begin{bmatrix}\hat{c}_0\\ \cdots\\ \hat{c}_{n-1}\end{bmatrix} = \begin{bmatrix}f_0\\ \cdots\\ f_{n-1}\end{bmatrix}, \qquad U=[u_{ij}]_{ij=0}^{n-1}. \tag{4.9}$$

The solution of this linear algebraic system will give the minimum point of $\mathscr{F}$.

THEOREM 4.1. *Let A be a real nonsingular matrix of order n. Then the linear algebraic system* (4.9) *has a unique solution. If $(Ax,x)>0$ with $x\neq 0$, then the circulant matrix $\hat{C}$ with the first column*

$$[\hat{c}_0\cdots\hat{c}_{n-1}]^T$$

is nonsingular and, moreover, $(\hat{C}x,x)>0$ for $x\neq 0$. If $A=A^T$, then $\hat{C}=\hat{C}^T$.

Proof. By (4.7) and (4.3), the matrix U is circulant. Taking into account well-known properties of traces of matrices, we have

$$u_{ij} = \operatorname{tr}(A^TQ^{j-i}A) = \operatorname{tr}(Q^{j-i}AA^T). \tag{4.10}$$

From (3.2) and (4.3) we have that the matrix

$$\tilde{U} = \frac{1}{n}U \tag{4.11}$$

is nothing other than the solution of the optimization problem

$$\|\tilde{U}-AA^T\|_F \to \min, \qquad \tilde{U}\in\mathscr{T}_n. \tag{4.12}$$

From Theorem 3.1 the matrix $\tilde{U}$ is symmetric positive definite, because AA^T is (A is nonsingular by the condition of the theorem). As a consequence of (4.11), the matrix

$$U+U^T = 2U \tag{4.13}$$

is symmetric positive definite and so we see that the system (4.9) has a unique solution.

Denote by F the circulant matrix with the first column $[f_0/2 \cdots f_{n-1}/2]^T$. Since a product of circulants is circulant, from (4.9) and (4.13), we have

$$U\hat{C} = F. \tag{4.14}$$

Then

$$F = [\operatorname{tr}(Q^{i-j}A)]_{ij=0}^{n-1}, \tag{4.15}$$

so the circulant matrix

$$\tilde{F} = \frac{1}{n}F \tag{4.16}$$

is the solution of the optimization problem

$$\|\tilde{F} - A\|_F \to \min, \qquad \tilde{F} \in \mathscr{T}_n, \tag{4.17}$$

and by Theorem 3.1, if $(Ax, x) > 0$ with $x \neq 0$, then $(\tilde{F}x, x) > 0$ with $x \neq 0$. Therefore,

$$(Fx, x) > 0 \quad \text{with } x \neq 0. \tag{4.18}$$

By (4.14) $\hat{C}$ is nonsingular.

Further, since U is symmetric and positive definite,

$$(\hat{C}x, x) = (U^{-1}Fx, x) = (U^{-1/2}Fx, U^{-1/2}x) = ((U^{-1/2}FU^{1/2})U^{-1/2}x, U^{-1/2}x). \tag{4.19}$$

Equations (3.6) and (3.7) show that a (nonsymmetric) circulant with all eigenvalues being positive is positive definite. It is clear that all eigenvalues of circulants $U^{-1/2}$, F, $U^{1/2}$ are positive and, therefore, all eigenvalues of $U^{-1/2}FU^{1/2}$ are positive as well. By (4.19) we have

$$(\hat{C}x, x) > 0 \quad \text{with } x \neq 0. \tag{4.20}$$

If $A = A^T$, then $F = F^T$ by Theorem 3.1. The relationship (4.14) implies $\hat{C} = \hat{C}^T$, since the product of symmetric circulants (and an inverse of a nonsingular circulant) remains symmetric circulant. This completes the proof. □

THEOREM 4.2. *Let A be a real nonsingular matrix of order $n = n_1 \cdots n_s$ and C be the solution of the optimization problem* (4.1), *assuming that $\mathscr{T}_n$ is the set of real nonsingular s-level matrices with circulant levels of orders $n_1, \cdots, n_s$. Then* (4.1) *has a unique solution. If $(Ax, x) > 0$ with $x \neq 0$, then $(Cx, x) > 0$ with $x \neq 0$. If $A = A^T$ then $C = C^T$.*

Let Q_l denote the matrix of the form (4.3) but of order n_l. Then for $\hat{C} = C^{-1}$ we have

$$\hat{C} = \sum_{j_1=0}^{n_1-1} \cdots \sum_{j_s=0}^{n_s-1} \hat{c}_{j_1;\cdots;j_s} Q_1^{j_1} \times \cdots \times Q_s^{j_s} \tag{4.21}$$

where $\hat{c}_{j_1;\cdots;j_s}$ are the components of the first column $\hat{c}$ of the matrix $\hat{C}$, arranged in the order that corresponds to (1.7).

Consider the functional

$$\begin{aligned} \mathscr{F} = \|I - \hat{C}A\|_F^2 = n - & \sum_{\substack{0 \le i_1 \le n_1 - 1 \\ \cdots \\ 0 \le i_s \le n_s - 1}} \hat{c}_{i_1;\cdots;i_s} \\ & + \sum_{\substack{0 \le i_1, j_1 \le n_1 - 1 \\ \cdots \\ 0 \le i_s, j_s \le n_s - 1}} \hat{c}_{i_1;\cdots;i_s} \hat{c}_{j_1;\cdots;j_s} u_{i_1 j_1;\cdots;i_s j_s} \end{aligned} \tag{4.22}$$

where

$$f_{i_1;\cdots;i_s} = \operatorname{tr}(A^T Q_1^{-i_1} \times \cdots \times Q_s^{-i_s} + Q_1^{i_1} \times \cdots \times Q^{i_s} A), \tag{4.23}$$

$$u_{i_1 j_1;\cdots;i_s j_s} = \operatorname{tr}(A^T Q^{j_1 - i_1} \times \cdots \times Q^{j_s - i_s} A). \tag{4.24}$$

In order to compute $\hat{c}$ we get the system of linear algebraic equations

$$(U + U^T)\hat{c} = f, \tag{4.25}$$

where U is the s-level circulant matrix with components $u_{i_1 j_1;\cdots;i_s j_s}$ and f is the vector with components $f_{i_1;\cdots;i_s}$ taken in such order that i_s is the first increasing, then i_{s-1} is, and so on. System (4.25) can be analyzed in the same manner as system (4.9), studied above.

5. Fast algorithms for finding superoptimal preconditioners. When using superoptimal preconditioners in practice, one obviously should be assured that there is a way to compute them sufficiently quickly. We have already known that components

$$\hat{c}_0, \hat{c}_1, \cdots, \hat{c}_{n-1}$$

are determined as the solution of the linear algebraic system (4.9) with a symmetric positive-definite circulant matrix of coefficients. Such a system can be solved with $O(n \log_2 n)$ complexity applying Theorem 2.2 and the FFT. But the main work lies in the computation of components u_{ij}, f_i.

Suppose first that A is a matrix with no special structure. Then the vector $[f_0 \cdots f_{n-1}]^T$—the right-hand side of (4.9)—can be calculated at a cost of $O(n^2)$ operations.

Further, U is entirely defined by its first row; the components of this row are of the form

$$u_{0j} = \operatorname{tr}(A^T Q^j A) = \sum_{r=0}^{n-1} a_r^T Q^j a_r, \tag{5.1}$$

where $a_0, a_1, \cdots, a_{n-1}$ are the columns of A. Let r be fixed and first find the values

$$\alpha_{rj} \equiv a_r^T Q^j a_r, \tag{5.2}$$

using

$$[\alpha_{r0} \cdots \alpha_{rn-1}] = a_r^T C_r, \qquad C_2 = [Q^0 a_r, \cdots, Q^{n-1} a_r]. \tag{5.3}$$

Note that C_r is circulant. For every r the multiplication of a row by a circulant can be performed at a cost of $O(n \log_2 n)$ operations. Therefore, all the values

$$\alpha_{rj}, \qquad 0 \leqq r, j \leqq n-1,$$

can be had with $O(n^2 \log_2 n)$ complexity. We need another $O(n^2)$ additions to calculate u_{0j}, $0 \leqq j \leqq n-1$, from the values α_{rj}. Thus, $O(n^2 \log_2 n)$ operations are sufficient to compute the superoptimal circulant preconditioner $\hat{C}$ when A is a general matrix.

In what follows, we study how a special structure of A can be used to accelerate the computing of the u_{0j}. Note that $u_{0j} = \operatorname{tr}(A^T Q^j A) = \operatorname{tr}(Q^j A A^T)$. Hence, we shall reach our goal if we consider a somewhat general problem. Namely, let a matrix

$$M = [m_{ij}]_{ij=0}^{n-1} \tag{5.4}$$

be given and let our goal be the computation of values

$$s_k \equiv s_k(M) \equiv \sum_{i-j=k(\bmod n)} m_{ij}, \qquad k = 0, 1, \cdots, n-1. \tag{5.5}$$

Obviously, $u_{0j} = s_j$.

THEOREM 5.1. *Let* $M = LR$, *where*

$$L = \begin{bmatrix} l_0 & & & & \\ l_1 & l_0 & & 0 & \\ l_2 & l_1 & l_0 & & \\ \cdots & \cdots & \cdots & & \\ l_{n-1} & l_{n-2} & l_{n-3} & \cdots & l_0 \end{bmatrix}, \qquad R = \begin{bmatrix} r_0 & r_1 & r_2 & \cdots & r_{n-1} \\ & r_0 & r_1 & \cdots & r_{n-2} \\ & & r_0 & \cdots & r_{n-3} \\ 0 & & & \cdots & \cdots \\ & & & & r_0 \end{bmatrix}. \tag{5.6}$$

Then

$$\begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ \cdots \\ s_{n-1} \end{bmatrix} = P \begin{bmatrix} r_{n-1} \\ r_{n-2} \\ r_{n-3} \\ \cdots \\ r_0 \end{bmatrix} + Q \begin{bmatrix} 0 \\ r_{n-1} \\ 2r_{n-2} \\ \cdots \\ (n-1)r_1 \end{bmatrix} \tag{5.7}$$

where

$$P = \begin{bmatrix} l_{n-1} & 2l_{n-2} & 3l_{n-3} & \cdots & nl_0 \\ & l_{n-1} & 2l_{n-2} & \cdots & (n-1)l_1 \\ & & l_{n-1} & \cdots & (n-2)l_2 \\ 0 & & & \cdots & \cdots \\ & & & & l_{n-1} \end{bmatrix},$$

$$Q = \begin{bmatrix} l_0 & & & & \\ l_1 & l_0 & & & 0 \\ l_2 & l_1 & l_0 & & \\ \cdots & \cdots & \cdots & \cdots & \\ l_{n-1} & l_{n-2} & l_{n-3} & \cdots & l_0 \end{bmatrix}. \tag{5.8}$$

Proof. The relations

$$m_{ij} = m_{i-1,j-1} + l_i r_j, \qquad 1 \leqq i, j \leqq n-1, \tag{5.9}$$

are true, since the matrices L and R are triangular. Therefore,

$$\begin{aligned} s_k &= \sum_{j=0}^{n-k-1} m_{k+j,j} + \sum_{j=0}^{k-1} m_{j,n-k+j} \\ &= \sum_{j=0}^{n-k-1} (n-k-j) l_{k+j} r_j + \sum_{j=0}^{k-1} (k-j) l_j r_{n-k+j}. \end{aligned} \tag{5.10}$$

The last two terms agree with two terms in (5.7). The proof is over. □

COROLLARY 1. *The values* s_k, $0 \leqq k \leqq n-1$, *can be computed with* $O(n \log_2 n)$ *complexity.*

Indeed, Theorem 5.1 yields that the computation we are considering is reducible to matrix by vector multiplications with matrices P and Q being Toeplitz, i.e., we need to apply the so-called aperiodic convolution, which takes $O(n \log_2 n)$ operations, as is well known [11].

COROLLARY 2. *Suppose the conditions of Theorem* 5.1 *are fulfilled; then the values* $s_k(RL)$, $0 \leqq k \leqq n-1$, *can be computed with* $O(n \log_2 n)$ *complexity.*

It is not difficult to check that the relations

$$s_k(M) = s_{n-k(\bmod n)}(JMJ), \qquad 0 \leqq k \leqq n-1, \tag{5.11}$$

are valid, where

$$J=\begin{bmatrix} 0 & & 1 \\ & \iddots & \\ 1 & & 0 \end{bmatrix}. \tag{5.12}$$

So relations

$$s_k(RL) = s_{n-k(\bmod n)}((JRJ)(JLJ)) \tag{5.13}$$

hold and it remains to be noted that matrices JRJ and JLJ are Toeplitz lower and upper triangular, respectively. Hence, we may directly use Corollary 1.

COROLLARY 3. *For arbitrary Toeplitz matrices A and B of order n, the values $s_k(AB)$, $0 \leqq k \leqq n-1$, can be computed with $O(n \log_2 n)$ complexity.*

Write

$$A = L_A + R_A, \qquad B = L_B + R_B,$$

where L_A, L_B are Toeplitz lower triangular and R_A, R_B are Toeplitz upper triangular. Then $AB = T + L_A R_B + R_A L_B$, where the matrix $T = L_A L_B + R_A R_B$ is Toeplitz, because the product of Toeplitz lower (upper) triangular matrices remains Toeplitz lower (upper) triangular and the sum of Toeplitz matrices is also Toeplitz. The first row and column of T can be obtained with $O(n \log_2 n)$ operations (by an aperiodic convolution). Further,

$$s_k(AB) = s_k(T) + s_k(L_A R_B) + s_k(R_A L_B),$$

and we need to apply Corollaries 1 and 2.

COROLLARY 4. *For any nonsingular Toeplitz matrix A of order n the superoptimal circulant preconditioner can be computed with $O(n \log_2 n)$ operations.*

COROLLARY 5. *Let A be a nonsingular real Toeplitz symmetric matrix of order $n = 2^L$. Then $O(n)$ operations are sufficient to reduce the computation of the superoptimal circulant preconditioner $\hat{C}$ to the performance of four complex* FFT*s of order n and three complex* FFT*s of order $n/4$.*

Writing $A = L + L^T$ where L is a Toeplitz lower triangular matrix, we get $AA^T = A^2 = L^2 + (L^2)^T + LL^T + L^TL$. The first column of L^2 can be calculated using two FFTs of order n and, after that, $O(n)$ operations are needed to find $s_k(L^2 + (L^2)^T)$ for all k. To calculate $s_k(LL^T)$ we use (5.7) and Theorem 2.2; this can be done by two FFTs of order n if the result of one of the previous transforms has been saved. Note that the FFTs operate on the vectors which are real, or conjugate, and, hence, are reducible to FFTs of complex vectors of less than half the order (see [5], [14]). Three FFTs of order n are needed to determine $\hat{C}$ from (4.14); these FFTs work with real symmetric vectors of even order and, therefore, are reducible to FFTs of complex vectors of order $n/4$ [5].

Thus, using split-algorithms of FFT [10] we compute a superoptimal circulant preconditioner for a real symmetric Toeplitz matrix with $1.55\, n \log_2 n$ complex multiplications and $4.75\, n \log_2 n$ complex additions-subtractions ($n = 2^L$) plus lower order terms. The operation count can be a bit improved if the relations $s_k = s_{n-k(\bmod n)}$, $0 \leqq k \leqq n-1$, are used.

COROLLARY 6. *Suppose that a nonsingular matrix A is written in the form*

$$A = \sum_{j=0}^{t} L_j R_j, \tag{5.14}$$

where L_j and R_j are Toeplitz lower and upper triangular, respectively. Then the optimal and the superoptimal circulant preconditioners for A can be computed with $O(tn \log_2 n)$ operations.

For superoptimal preconditioners, AA^T must be representable in a form like (5.14), but with the number of terms less than or equal to $2t + 1$ (see [4], [13]).

Now we shall switch to the case of M being composed of blocks m_{ij}, each of order p. It is easy to show that Theorem 5.1 still holds in this case. But now s_k is a block of order p, i.e.,

$$s_k = [s_{ij}^{(k)}]_{ij=0}^{p-1}.$$

Assume that we want to compute the values

$$s_{kl} = \sum_{i-j=l(\mathrm{mod}\ p)} s_{ij}^{(k)}, \qquad 0 \leqq l \leqq p-1, \qquad 0 \leqq k \leqq n-1. \tag{5.15}$$

If l were fixed, then the computation of s_{kl}, $0 \leqq k \leqq n-1$, would be reduced to the following problem: let two block circulants

$$V = [v_{i-j}], \qquad W = [w_{i-j}]$$

of block order $N \geqq 2n-1$ be given and the values

$$s_l(z_{i-j}), \quad \text{where } Z \equiv [z_{i-j}] = VW,$$

are to be computed.

From Theorem 2.2 we set

$$\begin{bmatrix} \hat{v}_0 \\ \cdots \\ \hat{v}_{N-1} \end{bmatrix} = (F_N \times I_p) \begin{bmatrix} v_0 \\ \cdots \\ v_{N-1} \end{bmatrix}, \qquad \begin{bmatrix} \hat{w}_0 \\ \cdots \\ \hat{w}_{N-1} \end{bmatrix} = (F_N \times I_p) \begin{bmatrix} w_0 \\ \cdots \\ w_{N-1} \end{bmatrix}; \tag{5.16}$$

then

$$\begin{bmatrix} z_0 \\ \cdots \\ z_{N-1} \end{bmatrix} = \frac{1}{N}(F_N^* \times I_p) \begin{bmatrix} \hat{v}_0 \hat{w}_0 \\ \cdots \\ \hat{v}_{N-1} \hat{w}_{N-1} \end{bmatrix}. \tag{5.17}$$

Hence,

$$\begin{bmatrix} s_l(z_0) \\ \cdots \\ s_l(z_{N-1}) \end{bmatrix} = \frac{1}{N} F_N^* \begin{bmatrix} s_l(\hat{v}_0 \hat{w}_0) \\ \cdots \\ s_l(\hat{v}_{N-1} \hat{w}_{N-1}) \end{bmatrix}. \tag{5.18}$$

We have described a way to implement (5.7) using the property that a (block) Toeplitz matrix by vector multiplication is reducible to a (block) circulant by (block) circulant multiplication, and the order of the (block) circulants may be equal to $N \geqq 2n-1$ where n is the order of the given (block) Toeplitz matrix. While exploiting the FFT and choosing $N = 2^L$, we always can guarantee that $N \leqq 4n$.

Equations (5.16)–(5.18) permit us to build up superoptimal doubly circulant preconditioners for doubly Toeplitz matrices at a cost of $O(n \log_2 n)$ operations.

6. Numerical results and some remarks. We have mentioned above that in comparison with Strang's preconditioners, optimal and superoptimal circulant preconditioners inherit nonsingularity and positive-definiteness from the original matrix. For example, if we take

$$A = \begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & 0 \\ & -1 & 2 & -1 & & \\ & & \cdots & & & \\ 0 & & & & -1 & 2 \end{bmatrix}_{n \times n},$$

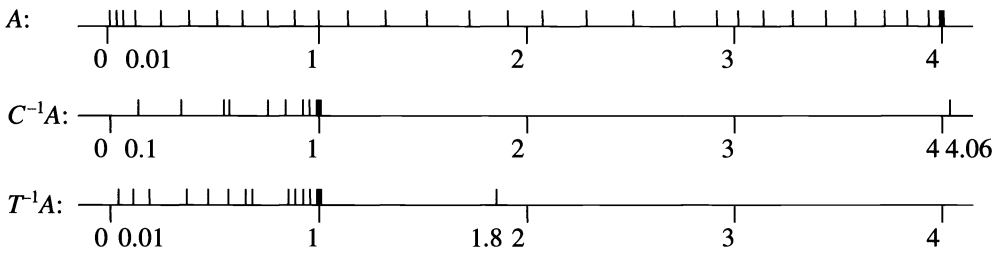
then Strang's preconditioner is of the form

$$S=\begin{bmatrix} 2 & -1 & & & & -1 \\ -1 & 2 & -1 & & 0 & \\ & -1 & 2 & -1 & & \\ & 0 & & \cdots & & \\ -1 & & & & -1 & 2 \end{bmatrix},$$

and so it is singular. Instead, the optimal preconditioner is of the form

$$C=\begin{bmatrix} 2 & -\frac{n-1}{n} & & & -\frac{n-1}{n} \\ -\frac{n-1}{n} & 2 & -\frac{n-1}{n} & & 0 \\ & 0 & \cdots & & \\ -\frac{n-1}{n} & & & -\frac{n-1}{n} & 2 \end{bmatrix}$$

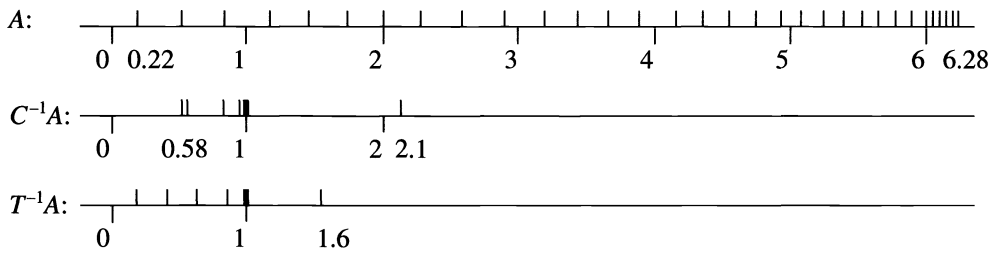
and we see that $C = C^T > 0$. The distributions of eigenvalues of A, $C^{-1}A$, and $T^{-1}A$ (T is a superoptimal preconditioner) are shown in the following chart ($n = 32$):



Further, for the Toeplitz matrix

$$A=\left[\frac{1}{0.25-(i-j)^2}\right]_{n\times n}$$

we have the following picture ($n = 32$):



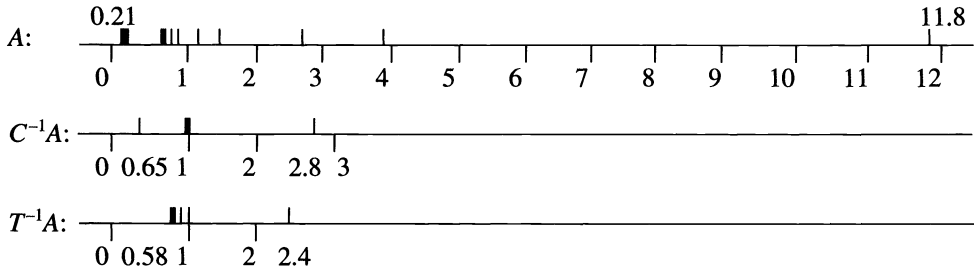
Note that

$$\|I-C^{-1}A\|_F^2 \approx 1.7, \qquad \|I-T^{-1}A\|_F^2 \approx 1.4.$$

Our next example will give some new information about properties of two preconditioners. Let

$$A = \left[\frac{1}{\sqrt{1+|i-j|}} \right]_{n \times n}.$$

The eigenvalues are distributed as shown below ($n = 32$):



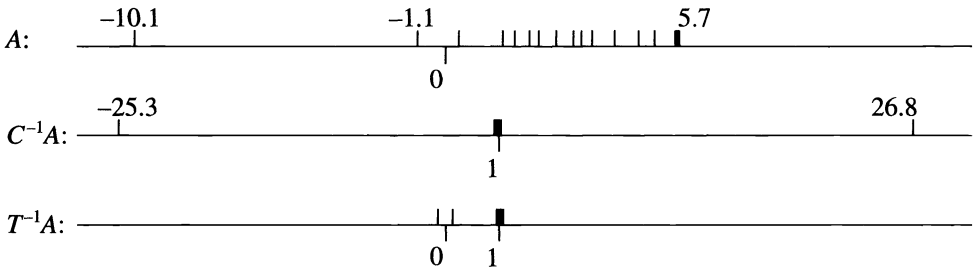
Curiously, the eigenvalues of $T^{-1}A$ cluster on the interval [0.73, 0.83], but the eigenvalues of $C^{-1}A$ go on clustering around 1, as they do in all observations. Here we have

$$\|I - C^{-1}A\|_F^2 \approx 3.62, \qquad \|I - T^{-1}A\|_F^2 \approx 3.57.$$

The last example will be the Toeplitz symmetric-indefinite matrix

$$A = \left[\frac{1}{0.25 - \sqrt{|i-j|}} \right]_{n \times n}.$$

The real parts of eigenvalues are distributed as in the following chart:



Note that $T^{-1}A$ has one pair of complex eigenvalues but all eigenvalues of $C^{-1}A$ are real. We have

$$\|I - C^{-1}A\|_F^2 \approx 1361.2, \qquad \|I - T^{-1}A\|_F^2 \approx 2.3.$$

Acknowledgments. The author is grateful to Professor Gene Golub for useful remarks and for his invitation to write this paper. He is also thankful to Professor James Bunch for his corrections, which helped to improve the style of the presentation.

REFERENCES

[1] L. BERG, *Solution of large linear systems with help of circulant matrices*, Z. Angew. Math. Mech., 55 (1975), pp. 439–441.

[2] T. F. CHAN, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 766–771.

[3] R. H. CHAN AND G. STRANG, *The asymptotic Toeplitz-circulant eigenvalue problem*, Numerical Analysis Report 87-5, Applied Mathematics Department, Massachusetts Institute of Technology, Cambridge, MA, May 1987.

[4] J. CHEN, T. KAILATH, AND H. LEV-ARI, *Fast parallel algorithms for QR and triangular factorizations*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 899–913.

[5] J. W. COOLEY, P. A. W. LEWIS, AND P. D. WELCH, *The Fast Fourier Transform Algorithm: Programming considerations in the calculation of sine, cosine and Laplace transforms*, J. Sound Vibration, 12 (1970), pp. 315–337.

[6] M. P. EKSTROM, *An iterative-improvement approach to the numerical solution of vector Toeplitz systems*, IEEE Trans. Comput., 33 (1974), pp. 320–325.

[7] M. MARCUS AND H. MINC, *Matrix Theory and Matrix Inequalities*, Allyn and Bacon, Boston, MA, 1964.

[8] C. L. RINO, *The inversion of covariance matrices by finite Fourier transforms*, IEEE Trans. Inform Theory, 16 (1970), pp. 230–232.

[9] P. J. SHERMAN, *Circulant approximations of the inverses of Toeplitz matrices and related quantities with applications of stationary random processes*, IEEE Trans. Acoust. Speech Signal Process., 33 (1985), pp. 1630–1632.

[10] H. V. SORENSEN, D. J. JONES, AND C. S. BURGUS, *On computing the split-radix FFT*, IEEE Trans. Acoust. Speech Signal Process, 34 (1986), pp. 152–156.

[11] T. G. STOCKHAM, *High-speed convolution and correlation*, in Proc. AFIPS 28 Spring Joint Computer Conference, Washington, D.C., 1966, pp. 229–233.

[12] G. STRANG, *A proposal for Toeplitz matrix calculations*, Stud. Appl. Math., 74 (1986), pp. 171–176.

[13] E. E. TYRTYSHNIKOV, *Fast algorithms for block Toeplitz matrices*, Soviet J. Numer. Anal. Math. Modelling, 1 (1986), pp. 121–139.

[14] V. V. VOEVODIN AND E. E. TYRTYSHNIKOV, *Computer Processes with Toeplitz Matrices*, Nauka, Moscow, 1987. (In Russian.)

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 474–489, April 1992

004

PERTURBATION OF THE EIGENVALUES OF QUADRATIC MATRIX POLYNOMIALS*

H. LANGER†, B. NAJMAN‡, AND K. VESELIĆ§

Abstract. Perturbation properties of a quadratic matrix pencil containing a "small" parameter are considered. Main results concern the splitting properties of multiple eigenvalues and the corresponding Puiseux expansions. For hermitian pencils it is proved that the Puiseux expansion generates groups of eigenvalues ordered according to the partial algebraic multiplicities of the unperturbed eigenvalue.

Key words. eigenvalues, quadratic matrix pencils, perturbation theory

AMS(MOS) subject classification. 15A18

Introduction. Let C and K be $n \times n$ matrices. We study the eigenvalues[1] $\lambda(\varepsilon)$ of the matrix polynomial

$$T(\lambda,\varepsilon)=\lambda^2 I+\lambda(1+\varepsilon)C+K \tag{0.1}$$

for small ε, considering $T(\lambda, \varepsilon)$ as a perturbation of the matrix polynomial

$$T(\lambda,0)=\lambda^2 I+\lambda C+K=:A(\lambda) \tag{0.2}$$

by the function $B(\lambda, \varepsilon) := \lambda\varepsilon C$. These eigenvalues are of interest as they represent the oscillation frequencies of a vibrating system described by the differential equation

$$\frac{d^2y}{dt^2}+(1+\varepsilon)C\frac{dy}{dt}+Ky=0.$$

Here C corresponds to the damping forces of the system. So the functions $\varepsilon \mapsto \lambda(\varepsilon)$, for small ε, give information on how these oscillation frequencies change if the damping is changed by εC.

As in [6] and [8] (see also [7] and [9]) our main tool for studying the behavior of $\lambda(\varepsilon)$ near $\varepsilon = 0$ is the Newton diagram of the function $\det T(\lambda, \varepsilon)$.

In the following, λ_0 always denotes an eigenvalue of the unperturbed matrix polynomial $A(\lambda)$ of algebraic multiplicity a. Recall that according to the general results of analytic perturbation theory for small ε the polynomial $T(\lambda, \varepsilon)$ has a eigenvalues $\lambda(\varepsilon)$ near λ_0 such that $\lambda(0) = \lambda_0$; these eigenvalues appear in groups and have Puiseux expansions at $\varepsilon = 0$ (see [1], [5]). In § 1 of this note we show that if the matrix C in (0.1) is diagonal (C and K need not be hermitian) and if $\lambda_0 \neq 0$, then the number of groups of eigenvalues $\lambda(\varepsilon)$ near λ_0 of the perturbed polynomial $T(\lambda, \varepsilon)$ can be estimated not only by the algebraic multiplicity a of λ_0 as an eigenvalue of $A(\lambda)$, but also by the algebraic multiplicity b of zero as an eigenvalue of the matrix $A(\lambda_0)$, and that in the case where $a > b$, there are eigenvalues $\lambda(\varepsilon)$ such that $\lambda(\varepsilon) - \lambda_0 = \gamma\varepsilon^\beta + o(|\varepsilon|^\beta)$ as

* Received by the editors June 18, 1990; accepted for publication (in revised form) August 15, 1990.

† Fachbereich Mathematik, Universität Regensburg, D-8400 Regensburg, Federal Republic of Germany (langer@vax1.rz.uni-regensburg.dbp.de).

‡ Department of Mathematics, University of Zagreb, P.O. Box 187, YU-41001 Zagreb, Yugoslavia. The research of this author was supported by Samoupravna Interesna Zajednica of Socijalistička Republika Hrvatiska.

§ Fachbereich Mathematik, Fernuniversität Hagen, Postfach 940, D-5800 Hagen 1, Federal Republic of Germany (ma704%dhafeu11.bitnet).

[1] For the definition of eigenvalues, eigenvectors, and associated vectors of matrix polynomials, we refer to [3].

$\varepsilon \to 0$ with some $\beta \leqq b/a$, $\gamma \neq 0$, that is, b/a is an upper bound for the smallest (worst) exponent of ε. A similar result holds if $\lambda_0 = 0$.

In § 2 we consider the case of hermitian matrices C and K. Denote the partial multiplicities of the eigenvalue λ_0 of $A(\lambda)$ by $m_1, \cdots, m_g$. We assume that there are k groups of mutually equal m_j, the jth group containing n_j elements:

$$
\begin{aligned}
(0.3) \qquad & m_1 = \cdots = m_{n_1} < m_{n_1+1} = \cdots \\
& = m_{n_1+n_2} < \cdots < m_{n_1+\cdots+n_{k-1}+1} = \cdots = m_{n_1+\cdots+n_k}
\end{aligned}
$$

and define

$$
\tilde{n}_j := n_1 + \cdots + n_j, \qquad \tilde{m}_j := m_{\tilde{n}_j}, \qquad j = 1, \cdots, k.
$$

In [6] we introduced a sufficient condition (relation (6)) which assured that among the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 for each $j \in \{1, \cdots, k\}$, there are $n_j \tilde{m}_j$ eigenvalues $\lambda_{j,\nu\sigma}(\varepsilon)$, $\nu = 1, \cdots, n_j$; $\sigma = 1, \cdots, \tilde{m}_j$, satisfying for $\varepsilon \to 0$ the asymptotic relations

$$
(0.4) \qquad \lambda_{j,\nu\sigma}(\varepsilon) = \lambda_0 + \gamma_{j\nu} \Theta_{\tilde{m}_j,\sigma} \varepsilon^{1/\tilde{m}_j} + o(|\varepsilon|^{1/\tilde{m}_j})
$$

with numbers $\gamma_{j\nu} \neq 0$ and $\Theta_{m,\sigma} := \exp(2\pi i(\sigma - 1)/m)$, $\sigma = 1, 2, \cdots, m$. If these relations hold, we shall say that $T(\lambda, \varepsilon)$ has the *completely regular splitting property* at λ_0.[2]

In other words, $T(\lambda, \varepsilon)$ has the completely regular splitting property at λ_0 if for each partial multiplicity m_i ($i = 1, \cdots, g$) from λ_0 there emerge m_i eigenvalues of the form

$$
(0.5) \qquad \lambda_{i\rho}(\varepsilon) = \lambda_0 + \gamma_i \Theta_{m_i,\rho} \varepsilon^{1/m_i} + o(|\varepsilon|^{1/m_i}) \qquad (\varepsilon \to 0),
$$

$\rho = 1, \cdots, m_i$, with $\gamma_i \neq 0$. We shall say that $T(\lambda, \varepsilon)$ has the *regular splitting property* at λ_0 if for each partial multiplicity m_i of the eigenvalue λ_0 from λ_0 there emerge m_i eigenvalues $\lambda_{i\rho}(\varepsilon)$ of $T(\lambda, \varepsilon)$ of the form (0.5), where $\gamma_i \neq 0$ for those i for which $m_i > 1$. That is, all the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 can still be written in the form (0.4), however for the real differentiable eigenvalues[3] the derivative at $\varepsilon = 0$ may be zero; for all the other eigenvalues the first nonzero (i.e., the leading) term in its Puiseux expansion at $\varepsilon = 0$ is $\gamma\varepsilon^{1/\tilde{m}_j}$, $\gamma \neq 0$. A sufficient condition for the regular splitting property was given in [7]. Applying this sufficient condition we show that for hermitian matrices C and K the matrix function $T(\lambda, \varepsilon)$ has the regular splitting property at every real nonzero eigenvalue of $A(\lambda)$.

Finally, in § 3 we prove that the perturbed eigenvalues have additional properties if one of the matrices C or K is definite.

Our interest in the particular matrix polynomial $T(\lambda, \varepsilon)$ was largely motivated by [4]. We mention also the paper [2], where a more general matrix polynomial $T(\lambda, \varepsilon)$ was considered.

1. Results for a general diagonable matrix C. Let C and K be $n \times n$ matrices and $A(\lambda)$, $T(\lambda, \varepsilon)$ be given by (0.2) and (0.1), respectively. In this section we often assume that C is diagonable, that is, that there exists a basis in $\mathbf{C}^n$ such that

$$
(1.1) \qquad C = \operatorname{diag}(c_1, \cdots, c_n), \qquad c_i \in \mathbf{C}, \qquad i = 1, \cdots, n.
$$

Let λ_0 be an eigenvalue of $A(\lambda)$ of geometric multiplicity g and algebraic multiplicity

---

[2] Some physical examples exhibiting this behavior can be found in [10].

[3] The eigenvalue $\lambda(\varepsilon)$ is called real differentiable at $\varepsilon = 0$ if for $\varepsilon \to 0$, ε real, there exists $\lim \varepsilon^{-1}(\lambda(\varepsilon) - \lambda(0))$.

a. Then zero is an eigenvalue of the matrix $A(\lambda_0)$ of geometric multiplicity g; denote by b its algebraic multiplicity. Evidently, $b \geqq g$, and all three cases $b > a$, $b = a$, and $b < a$ are possible. In fact, if $A(\lambda_0)$ is hermitian, then $b = g$; as a can be either equal to g or larger than g, it remains to show that b can be larger than a.

Example. Let

$$C = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \qquad K = \begin{bmatrix} -3 & 1 \\ -1 & 0 \end{bmatrix}, \qquad \lambda_0 = 1.$$

Then $t(\lambda) := \det(\lambda^2 I + \lambda C + K) = \lambda^4 + \lambda^3 - 3\lambda^2 + 1$ and $t(\lambda_0) = 0$, $(dt/d\lambda)(\lambda_0) \neq 0$; hence $a = 1$. As

$$A(1) = I + C + K = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix},$$

we find $\det(\mu I - A(1)) = \mu^2$ and $b = 2$.

Let $\alpha_1, \cdots, \alpha_n$ be the eigenvalues of $A(\lambda_0)$, counted according to their algebraic multiplicities, and denote by s_j, $j = 0, \cdots, n-1$, the elementary symmetric function of order $n - j$ of $\alpha_1, \cdots, \alpha_n$:

$$s_0 = \alpha_1 \cdots \alpha_n, \qquad s_1 = \sum_{k=1}^{n} \prod_{j:j\neq k} \alpha_j, \cdots, \qquad s_{n-1} = \sum_{k=1}^{n} \alpha_k.$$

As exactly b of the α_j's are zero, it follows that $s_0 = \cdots = s_{b-1} = 0$, $s_b \neq 0$.

Furthermore, we put

$$t(\lambda, \varepsilon) := \det T(\lambda, \varepsilon),$$

and for $k = 1, \cdots, n-1$ and mutually different $\nu_1, \cdots, \nu_k \in \{1, \cdots, n\}$,

$$t(\lambda, \varepsilon; \nu_1, \cdots, \nu_k) = \det \check{T}(\lambda, \varepsilon; \nu_1, \cdots, \nu_k),$$

where $\check{T}(\lambda, \varepsilon; \nu_1, \cdots, \nu_k)$ is the matrix obtained from $T(\lambda, \varepsilon)$ by deleting the rows and columns with indices $\nu_1, \cdots, \nu_k$. Introducing

$$a(\nu_1, \cdots, \nu_k) := t(\lambda_0, 0; \nu_1, \cdots, \nu_k),$$

the following identities hold:

$$s_0 = \det A(\lambda_0) = t(\lambda_0, 0), \qquad s_j = \sum_{1 \leqq \nu_1 < \cdots < \nu_j \leqq n} a(\nu_1, \cdots, \nu_j), \qquad j = 1, \cdots, n-1.$$

If f is a polynomial in $\lambda - \lambda_0$ and ε, and $f(\lambda_0, 0) = 0$, we denote by $N(f)$ the falling part of the corresponding Newton diagram of f and by $E(f)$ the set of its extremal points (for the definition of the Newton diagram we refer to [1] and [11]).

PROPOSITION 1.1. *Let a be defined as above.*

(i) *The point $(a, 0)$ belongs to $E(t)$.*

(ii) *If C is diagonable and $\lambda_0 \neq 0$ is an eigenvalue of $A(\lambda_0)$ of algebraic multiplicity b, then there exists a point $(x_0, y_0) \in E(t)$ such that*

$$y_0 \leqq b - x_0. \tag{1.2}$$

Proof. We put

$$t_{ij} := \frac{\partial^{i+j} t}{\partial \lambda^i \partial \varepsilon^j}(\lambda_0, 0).$$

Since λ_0 is an eigenvalue of $A(\lambda)(= T(\lambda, 0))$ of algebraic multiplicity a, we have

$$t_{00} = \cdots = t_{a-1,0} = 0, \qquad t_{a,0} \neq 0,$$

and this is equivalent to (i).

In order to prove (ii) we can assume that C is of the form (1.1). Then

$$T(\lambda, \varepsilon) = \begin{bmatrix} \lambda^2 + \lambda(1+\varepsilon)c_1 + k_{11} & k_{12} & \cdots & k_{1n} \\ k_{21} & \lambda^2 + \lambda(1+\varepsilon)c_2 + k_{22} & \cdots & k_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ k_{n1} & k_{n2} & \cdots & \lambda^2 + \lambda(1+\varepsilon)c_n + k_{nn} \end{bmatrix}.$$

Applying the formula for the derivative of a determinant it follows that

$$\frac{\partial t(\lambda, \varepsilon)}{\partial \lambda} = \sum_{\nu=1}^{n} (2\lambda + (1+\varepsilon)c_\nu) t(\lambda, \varepsilon; \nu), \tag{1.3}$$

$$\frac{\partial t(\lambda, \varepsilon)}{\partial \varepsilon} = \sum_{\nu=1}^{n} \lambda c_\nu t(\lambda, \varepsilon; \nu), \tag{1.4}$$

and for $k = 1, \cdots, n-1$

$$\frac{\partial t(\lambda, \varepsilon; \nu_1, \cdots, \nu_k)}{\partial \lambda} = \sum_{\substack{\nu=1 \\ \nu \neq \nu_1, \cdots, \nu_k}}^{n} (2\lambda + (1+\varepsilon)c_\nu) t(\lambda, \varepsilon : \nu_1, \cdots, \nu_k, \nu), \tag{1.5}$$

$$\frac{\partial t(\lambda, \varepsilon; \nu_1, \cdots, \nu_k)}{\partial \varepsilon} = \sum_{\substack{\nu=1 \\ \nu \neq \nu_1, \cdots, \nu_k}}^{n} \lambda c_\nu t(\lambda, \varepsilon; \nu_1, \cdots, \nu_k, \nu); \tag{1.6}$$

here we put $t(\lambda, \varepsilon; 1, \cdots, n) = 1$. The identities (1.3)–(1.6) imply

$$t_{01} = \lambda_0 \sum_{\nu=1}^{n} c_\nu a(\nu), \qquad t_{10} = \sum_{\nu=1}^{n} (2\lambda_0 + c_\nu) a(\nu),$$

$$t_{02} = \lambda_0^2 \sum_{\nu_1, \nu_2} c_{\nu_1} c_{\nu_2} a(\nu_1, \nu_2),$$

$$t_{11} = \sum_{\nu=1}^{n} c_\nu a(\nu) + \lambda_0 \sum_{\nu_1, \nu_2} c_{\nu_1} (2\lambda_0 + c_{\nu_2}) a(\nu_1, \nu_2),$$

$$t_{20} = 2 \sum_{\nu=1}^{n} a(\nu) + \sum_{\nu_1, \nu_2} (2\lambda_0 + c_{\nu_1})(2\lambda_0 + c_{\nu_2}) a(\nu_1, \nu_2),$$

where the sum $\sum_{\nu_1, \nu_2}$ runs over all ordered pairs of different $\nu_1, \nu_2 \in \{1, \cdots, n\}$. In order to simplify the notation, we define for $k \leqq l$

$$D_{kl} := \sum c_{\nu_1} \cdots c_{\nu_k} a(\nu_1, \cdots, \nu_k, \cdots, \nu_l),$$

where the sum runs again over all mutually different $\nu_1, \cdots, \nu_l$ (i.e., over all variations of l elements of the integers $1, \cdots, n$).

Now the t_{ij}, $i+j \leqq 2$, can be written as

$$t_{01} = \lambda_0 D_{11}, \qquad t_{10} = 2\lambda_0 D_{01} + D_{11}, \qquad t_{02} = \lambda_0^2 D_{22},$$

$$t_{11} = 2\lambda_0^2 D_{12} + D_{11} + \lambda_0 D_{22}, \tag{1.7}$$

$$t_{20} = 4\lambda_0^2 D_{02} + 2D_{01} + 4\lambda_0 D_{12} + D_{22}.$$

If $l > 2$ we obtain in a similar way

$$t_{0l} = \lambda_0^l D_{ll},$$

$$t_{1,l-1} = 2\lambda_0^l D_{l-1,l} + (l-1)\lambda_0^{l-2} D_{l-1,l-1} + \lambda_0^{l-1} D_{ll},$$

$$t_{i,l-i} = 2^i \lambda_0^l D_{l-i,l} + \cdots + \lambda_0^{l-i} D_{ll}, \qquad i = 2, \cdots, l,$$

where $\cdots$ represent the terms containing D_{jl} with $j \neq l-i, l$ and D_{jk} with $k < l$. Evidently, (1.7) implies that these relations hold also for $l = 1$ and 2.

Now suppose that $\lambda_0 \neq 0$. Then we have

$$D_{ll} = \lambda_0^{-l} t_{0l},$$

$$D_{l-1,l} = (2\lambda_0^l)^{-1}(t_{1,l-1} - \lambda_0^{l-1} D_{ll}) + (2\lambda_0^l)^{-1}(l-1)\lambda_0^{l-2} D_{l-1,l-1}$$

$$= (2\lambda_0^l)^{-1}(t_{1,l-1} - \lambda_0^{-1} t_{0l}) + \tfrac{1}{2}\lambda_0^{-2}(l-1) D_{l-1,l-1},$$

and for general $i = l, l-1, \cdots, 0$:

$$D_{il} = \sum_{k=0}^{l-i} \gamma_k^{(i,l)} t_{k,l-k} + R_{il}, \tag{1.8}$$

where R_{il} denotes a sum of terms containing D_{jk} with $k \leqq l-1$, and $\gamma_k^{(i,l)}$ are complex numbers, $\gamma_{l-i}^{(i,l)} \neq 0$. By induction with respect to l, it follows that D_{jl} is a linear expression in t_{ik} where $i + k \leqq l$. As R_{0l} in (1.8) contains only D_{jk} with $k \leqq l-1$, it does not contain the term t_{l0}. Therefore it follows from (1.8) that

$$t_{l0} = (\gamma_l^{(0,l)})^{-1} D_{0l} + \sum \rho_{jk}^{(l)} t_{jk}, \tag{1.9}$$

where the sum extends over all (j, k) with $j + k \leqq l$, $(j, k) \neq (l, 0)$, $l = 1, \cdots, n$.

After these preparations, the statement (ii) follows easily. Indeed, if $a \leqq b$, then (ii) is a consequence of (i). If $a > b$ we consider (1.9) for $l = b$, and assume that no point (x, y) with $y \leqq b - x$ belongs to $E(t)$. Then no such point belongs to $N(t)$ and from (1.9) it follows that $D_{0b} = 0$. On the other hand, we have $D_{0b} = s_b \cdot b! \neq 0$, a contradiction.

THEOREM 1.2. *Assume that C is diagonable, that $\lambda_0 \neq 0$ is an eigenvalue of $A(\lambda)$ of algebraic multiplicity a, and that zero is an eigenvalue of the matrix $A(\lambda_0)$ of algebraic multiplicity b. Then the following statements hold*:

(i) *The number of groups of eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 is less than or equal to* $\min(a, b)$.

(ii) *If $a > b$, then at least one eigenvalue $\lambda(\varepsilon)$ of $T(\lambda, \varepsilon)$ near λ_0 satisfies the asymptotic relation*

$$\lambda(\varepsilon) = \lambda_0 + \gamma\varepsilon^\beta + o(|\varepsilon|^\beta) \qquad (\varepsilon \to 0)$$

with some $\beta \leqq b/a$, $\gamma \neq 0$.

Proof. If $a \leqq b$, then the statement (i) is trivial as $T(\lambda, \varepsilon)$ has only a eigenvalues near λ_0. Suppose, therefore, that $a > b$ and consider the point $P_0 = (x_0, y_0) \in E(t)$ with the property (1.2), which exists according to Proposition 1.1. Then there are at most x_0 groups of eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 which correspond to the part of $N(t)$ to the left

of P_0 (as there are only x_0 eigenvalues corresponding to that part of $N(t)$), and at most y_0 groups which correspond to the part of $N(t)$ to the right of P_0 (as $N(t)$ can have at most y_0 segments there). As $x_0 + y_0 \leqq b$ the statement (i) follows. In order to prove (ii) we observe that $E(t)$ contains a point (x_0, y_0) below or on the line $y = -x + b$. It follows from $a > b$ that the modulus of the slope of the segment of $N(t)$ which ends at $(a, 0)$ is less than or equal to b/a.

In the following corollaries of Theorem 1.2 we always assume that $\lambda_0 \neq 0$ and that C is diagonable.

COROLLARY 1.3. *If $b = 1$, then the eigenvalues $\lambda_j(\varepsilon), j = 1, \cdots, a$, of $T(\lambda, \varepsilon)$ near λ_0 form one group; they have the Puiseux expansions*

$$\lambda_j(\varepsilon) = \lambda_0 + \sum_{k=1}^{\infty} \alpha_k \theta_{a,j}^k \varepsilon^{k/a}, \qquad j = 1, \cdots, a,$$

with $\alpha_1 \neq 0$.

The assumption $b = 1$ in Corollary 1.3 implies that the geometric multiplicity g (of both the eigenvalue λ_0 of $A(\lambda)$ and the eigenvalue zero of $A(\lambda_0)$) is one. On the other hand, if $g = 1$ and $A(\lambda_0)$ is normal, then also $b = 1$.

COROLLARY 1.4. *Assume $b = 2$.*

(i) *If $a \leqq 2$, then all the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 are real differentiable at $\varepsilon = 0$.*

(ii) *If $a > 2$, then the following alternative holds:*

(ii$_1$) *The eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 form two groups, one of them consisting of $a - 1$ eigenvalues having the Puiseux expansion*

$$\lambda_j(\varepsilon) = \lambda_0 + \sum_{k=1}^{\infty} \alpha_k \theta_{a-1,j}^k \varepsilon^{k/(a-1)}, \qquad j = 1, \cdots, a-1,$$

with $\alpha_1 \neq 0$, the other group consisting of one analytic eigenvalue $\lambda_a(\varepsilon)$.

(ii$_2$) *No eigenvalue of $T(\lambda, \varepsilon)$ near λ_0 is differentiable at $\varepsilon = 0$. These eigenvalues form either one group:*

$$\lambda_j(\varepsilon) = \lambda_0 + \alpha_2 \theta_{a,j}^2 \varepsilon^{2/a} + \sum_{k=3}^{\infty} \alpha_k \theta_{a,j}^k \varepsilon^{k/a}, \qquad j = 1, \cdots, a,$$

with $a_2 \neq 0$, or two groups, each of them containing at least two eigenvalues.

Proof. First we observe that $b = 2$ implies $s_1 = 0$; therefore from (1.7) it follows that $\lambda_0 t_{10} = t_{01}$.

(i) If $a = 1$, then the analyticity of the eigenvalue of $T(\lambda, \varepsilon)$ near λ_0 is well known (and follows from the fact that $N(t)$ consists of one segment with slope $-k$, k a natural number). If $a = 2$, then $t_{10} = t_{01} = 0$, that is, $(0, 1) \notin N(t)$. As $(2, 0) \in N(t)$, the moduli of the slopes of (the one or two segments of) $N(t)$ are greater than or equal to one.

(ii) If $a > 2$, then we have $t_{10} = t_{01} = t_{20} = 0$. Moreover, it follows from (1.7) that

$$0 = 8\lambda_0^4 s_2 + 2\lambda_0 t_{11} - t_{02}.$$

Therefore, as $s_2 \neq 0$, at least one of the points $P_1 = (1, 1)$, $P_2 = (0, 2)$ belongs to $E(t)$. If $P_1 \in E(t)$, then we are in case (ii$_1$) and the claim follows from the fact that $E(t)$ contains P_1 and $P_0 = (a, 0)$. If $P_1 \notin E(t)$, then $P_0, P_2 \in E(t)$; $N(t)$ may also contain a third point $(m, 1)$ with $2 \leqq m \leqq [a/2]$. Again, all the claims in (ii$_2$) follow from the form of the Newton diagram of t.

The next two examples illustrate the various possibilities in Corollary 1.4.

Example 1. Let

$$C=\begin{bmatrix} c_1 & 0 \\ 0 & c_2 \end{bmatrix}, \qquad K=\begin{bmatrix} -1-c_1 & 1 \\ 0 & -1-c_2 \end{bmatrix},$$

and denote $c_1 + c_2 = \gamma_1$, $c_1c_2 = \gamma_2$. Let $\lambda_0 = 1$. Since $A(\lambda_0) = \left[\begin{smallmatrix} 0 & 1 \\ 0 & 0 \end{smallmatrix}\right]$, it follows that $b = 2$. A calculation shows that

$$t(\lambda)=\det A(\lambda)=\lambda^4+\gamma_1\lambda^3+(\gamma_2-\gamma_1-2)\lambda^2+(-2\gamma_2-\gamma_1)\lambda+\gamma_2+\gamma_1+1,$$

hence $t(\lambda_0) = 0$, $t'(\lambda_0) = 0$, $t''(\lambda_0) = 2(\gamma_2 + 2\gamma_1 + 4)$, $t'''(\lambda_0) = 24 + 6\gamma_1$. If $\gamma_2 + 2\gamma_1 \neq -4$, then $a = 2$ and we are in case (i).

If $\gamma_2 + 2\gamma_1 = -4$, then $a \geqq 3$ and we are in case (ii); moreover, $a = 4$ only if, additionally, $\gamma_1 = -4$, implying $c_1 = c_2 = -2$. Another computation shows that

$$t(\lambda,\varepsilon)=\lambda^4+\gamma_1(1+\varepsilon)\lambda^3+((1+\varepsilon)^2\gamma_2-\gamma_1-2)\lambda^2-(1+\varepsilon)(\gamma_1+2\gamma_2)\lambda+\gamma_1+\gamma_2+1,$$

hence $t_{11} = (\partial^2 t/\partial\lambda\partial\varepsilon)(1, 0) = 2\gamma_1 + 2\gamma_2$, $t_{02} = (\partial^2 t/\partial\varepsilon^2)(1, 0) = 2\gamma_2$. It follows that if $\gamma_1 + \gamma_2 \neq 0$ then we are in case (ii$_1$); this is the case, for example, if $c_1 = 0$, $c_2 = -2$.

If $\gamma_1 + \gamma_2 = 0$, then we are in case (ii$_2$); together with $\gamma_2 + 2\gamma_1 = -4$, this implies $\gamma_1 = -4$, $\gamma_2 = 4$, hence $c_1 = c_2 = -2$. In this case $N(t)$ consists of the points (4, 0), (2, 1), and (0, 2).

Example 2. Let

$$C=\begin{bmatrix} c_1 & 0 & 0 \\ 0 & c_2 & 0 \\ 0 & 0 & c_3 \end{bmatrix}, \qquad K=\begin{bmatrix} -c_1 & 1 & 1 \\ 1 & -c_2 & 1 \\ 1 & 1 & -c_3 \end{bmatrix}, \qquad \lambda_0=1.$$

Since

$$A(\lambda_0)=\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix},$$

it follows that $b = g = 2$. Set $\gamma_1 = c_1 + c_2 + c_3$, $\gamma_2 = c_1c_2 + c_1c_3 + c_2c_3$, $\gamma_3 = c_1c_2c_3$. A calculation yields

$$\begin{aligned} t(\lambda)=\det A(\lambda)&=\lambda^6+\gamma_1\lambda^5+(\gamma_2-\gamma_1)\lambda^4+(\gamma_3-2\gamma_2)\lambda^3 \\ &\quad+(\gamma_2-3\gamma_3-3)\lambda^2+(3\gamma_3-\gamma_1)\lambda+\gamma_1-\gamma_3+2, \\ t(\lambda_0)=t'(\lambda_0)&=0, \qquad t''(\lambda_0)=2(12+4\gamma_1+\gamma_2). \end{aligned}$$

Therefore the alternative (ii) prevails if and only if $4\gamma_1 + \gamma_2 = -12$. Another calculation gives

$$t_{11}=\frac{\partial^2}{\partial\lambda\partial\varepsilon}\det T(\lambda,\varepsilon)\Big|_{\substack{\lambda=1\\ \varepsilon=0}}=4\gamma_1+2\gamma_2.$$

We see that (ii$_2$) holds if and only if $\gamma_1 = -6$, $\gamma_2 = 12$.

If λ_0 is an eigenvalue of $A(\lambda)$, it is possible that $t(\lambda_0, \varepsilon) = 0$ for all ε, that is, $\lambda(\varepsilon) = \lambda_0$ is an eigenvalue of $T(\lambda, \varepsilon)$. In this case we call λ_0 a *trivial eigenvalue* of $T(\lambda, \varepsilon)$. The exponent $d \geqq 1$ such that

$$t(\lambda,\varepsilon)=(\lambda-\lambda_0)^d\tilde{t}(\lambda,\varepsilon),$$

where $\tilde{t}(\lambda, \varepsilon)$ is holomorphic in λ, ε and $\tilde{t}(\lambda_0, \cdot)$ is not identically zero, is called the

algebraic multiplicity of the trivial eigenvalue λ_0 of $T(\lambda, \varepsilon)$. Evidently, λ_0 is a trivial eigenvalue of $T(\lambda, \varepsilon)$ of algebraic multiplicity $d > 0$ if and only if the left endpoint of $N(t)$ is (d, m) for some $m \geqq 0$.

Finally, we consider the unperturbed eigenvalue $\lambda_0 = 0$. Evidently, $\lambda_0 = 0$ is an eigenvalue of $A(\lambda)$ if and only if it is an eigenvalue of $T(\lambda, \varepsilon)$ and if and only if it is an eigenvalue of $A(0) = K$, and all three geometric multiplicities coincide. Hence if $\lambda_0 = 0$ is an eigenvalue of $A(\lambda)$, it is a trivial eigenvalue of $T(\lambda, \varepsilon)$. Moreover, if there exists a Jordan chain of length 2 to the eigenvalue $\lambda_0 = 0$ of $A(\lambda)$, that is, if there exist elements $x_0 \neq 0$ and x_1 such that

$$Kx_0 = 0, \qquad Cx_0 + Kx_1 = 0,$$

then with $\hat{x}_0 := x_0$, $\hat{x}_1(\varepsilon) := (1 + \varepsilon)x_1$, we have

$$K\hat{x}_0 = 0, \qquad (1+\varepsilon)C\hat{x}_0 + K\hat{x}_1(\varepsilon) = 0,$$

that is, a chain of length two is associated to the trivial eigenvalue $\lambda(\varepsilon) = 0$ of $T(\lambda, \varepsilon)$. Observe that although the eigenvalue λ_0 and the eigenvector x_0 do not depend on ε, the first associated vector $\hat{x}_1(\varepsilon)$ does.

PROPOSITION 1.5. *Assume that C is diagonable and that $\lambda_0 = 0$ is an eigenvalue of $A(\lambda)$. If (x, y) belongs to $N(t)$, then $x > y$.*

Proof. For $l \in \{1, \cdots, n-1\}$, $k \leqq l$ define

$$g_{kl}(\lambda, \varepsilon) = \sum (2\lambda + c_{\nu_1}(1+\varepsilon)) \cdots (2\lambda + c_{\nu_k}(1+\varepsilon)) t(\lambda, \varepsilon; \nu_1, \cdots, \nu_l),$$

where the sum runs over all possible choices of mutually different $\nu_1, \cdots, \nu_l$; note that with D_{kl} defined in the proof of Proposition 1.1, we have $D_{kl} = g_{kl}(0, 0)$.

From (1.3) we find

$$\frac{\partial g_{kl}}{\partial \lambda}(\lambda, \varepsilon) = 2k g_{k-1,l}(\lambda, \varepsilon) + g_{k+1,l+1}(\lambda, \varepsilon). \tag{1.10}$$

We prove by induction that for every k, $1 \leqq k \leqq n$, there exist positive numbers α_{kj} $(0 \leqq 2j \leqq k)$ such that

$$\frac{\partial^k t}{\partial \lambda^k}(\lambda, \varepsilon) = \sum_{0 \leqq 2j \leqq k} \alpha_{kj} g_{k-2j,k-j}(\lambda, \varepsilon). \tag{1.11}$$

From (1.3) we see that (1.11) holds for $k = 1$; if it holds for some k, then we find from (1.10)

$$\begin{aligned} \frac{\partial^{k+1} t}{\partial \lambda^{k+1}}(\lambda, \varepsilon) &= \sum_{0 \leqq 2j \leqq k} \alpha_{kj} \frac{\partial g_{k-2j,k-j}}{\partial \lambda}(\lambda, \varepsilon) \\ &= \sum_{0 \leqq 2j \leqq k} \alpha_{kj} [2(k-2j) g_{k-2j-1,k-j}(\lambda, \varepsilon) + g_{k-2j+1,k-j+1}(\lambda, \varepsilon)], \end{aligned}$$

and this is of the form (1.11), with k replaced by $k + 1$. From the definition of g_{kl} and from (1.6) it follows that

$$\frac{\partial^r g_{kl}}{\partial \varepsilon^r}(0,0) = \frac{k!}{(k-r)!} D_{kl} \quad \text{if } r \leqq k \leqq l \tag{1.12}$$

and

$$\frac{\partial^r g_{kl}}{\partial \varepsilon^r}(0,0) = 0 \quad \text{if } k \leqq l, k < r. \tag{1.13}$$

From (1.11) we conclude that if $r > k$, then $t_{kr} = 0$; hence $(k, r) \in N(t)$ implies $k \geqq r$. To exclude $k = r$, note that by (1.11)–(1.13) we have

$$t_{kk} = \frac{\partial^{2k} t}{\partial \varepsilon^k \partial \lambda^k}(0,0) = \alpha_{k0} k! D_{kk},$$

$$t_{k,k-1} = \frac{\partial^{2k-1} t}{\partial \varepsilon^{k-1} \partial \lambda^k}(0,0) = \alpha_{k0} \frac{\partial^{k-1} g_{kk}}{\partial \varepsilon^{k-1}}(0,0) = \alpha_{k0} k! D_{kk},$$

hence

$$t_{kk} = t_{k,k-1}.$$

If $t_{kk} \neq 0$, then also $t_{k,k-1} \neq 0$. According to the definition of $N(t)$ the point (k, k) does not belong to $N(t)$.

From Proposition 1.5 we obtain a result similar to Theorem 1.2.

PROPOSITION 1.6. *Assume that C is diagonable. Let $\lambda_0 = 0$ be an eigenvalue of $A(\lambda)$ of algebraic multiplicity a and let d be the algebraic multiplicity of $\lambda_0 = 0$ as a trivial eigenvalue of $T(\lambda, \varepsilon)$. Then the following statements hold*:

(i) *The nontrivial eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 form at most $d - 1$ groups.*

(ii) *At least one eigenvalue $\lambda(\varepsilon)$ of $T(\lambda, \varepsilon)$ near λ_0 satisfies the asymptotic relation*

$$\lambda(\varepsilon) = \gamma \varepsilon^\beta + o(|\varepsilon|^\beta) \qquad (\varepsilon \to 0)$$

with some $\beta \leqq (d-1)/(a-d)$, $\gamma \neq 0$.

Proof. The left endpoint of $N(t)$ is (d, m) for some $m \leqq d - 1$ by Proposition 1.5; the right endpoint of $N(t)$ is $(a, 0)$. Both statements of the proposition follow from this.

Remark. In the generic case of [4] the algebraic multiplicity a is less than or equal to two.

Hence if $\lambda_0 = 0$, then $a = d = 2$ and the only eigenvalue $\lambda(\varepsilon)$ of $T(\lambda, \varepsilon)$ near $\lambda_0 = 0$ is the trivial eigenvalue. If $\lambda_0 \neq 0$, then either $a = 1$ and $\lambda(\varepsilon)$ is analytic or $a = 2$. In the latter case Corollaries 1.3 or 1.4(i) apply if $b = 1$ or $b = 2$, respectively. There remains the case $a = 2$, $b > 2$ where Theorem 1.2 does not provide any additional information. This case does not occur if C and K are hermitian and λ_0 is real.

2. The regular splitting property in the hermitian case. In this section we apply the results of [6] and [7] to the quadratic polynomial

$$T(\lambda, \varepsilon) = \lambda^2 I + \lambda(1+\varepsilon)C + K,$$

where C and K are hermitian $n \times n$ matrices. Again, let

$$A(\lambda) = T(\lambda, 0) = \lambda^2 I + \lambda C + K$$

be the corresponding unperturbed polynomial and suppose first that λ_0 is a real eigenvalue of $A(\lambda)$ with partial multiplicities $m_1, \cdots, m_g$, ordered as in (0.3). If $n > g$ we put $m_{g+1} = \cdots = m_n = 0$. According to a theorem of Rellich there exist an analytic $n \times n$ matrix function $U(\lambda)$ such that $U(\bar{\lambda})^* = U(\lambda)^{-1}$, real analytic functions $\phi_j(\lambda)$, and numbers $\sigma_j \in \{-1, 1\}$, $j = 1, \cdots, n$, such that $\varphi_j(\lambda_0) > 0$ and

$$A(\lambda) = U(\lambda) \operatorname{diag}(\mu_1(\lambda), \cdots, \mu_n(\lambda)) U(\lambda)^{-1}, \tag{2.1}$$

where $\mu_j(\lambda) := \sigma_j(\lambda - \lambda_0)^{m_j} \varphi_j(\lambda)^2$, $j = 1, \cdots, n$. The number σ_j is the sign characteristic corresponding to the partial multiplicity m_j, $j = 1, \cdots, g$. Defining the matrices

$$\Phi(\lambda) := \operatorname{diag}(\varphi_1(\lambda), \cdots, \varphi_n(\lambda)),$$

$$D(\lambda) := \operatorname{diag}((\lambda - \lambda_0)^{m_1}, \cdots, (\lambda - \lambda_0)^{m_n}), \qquad S := \operatorname{diag}(\sigma_1, \cdots, \sigma_n),$$

it follows that

$$A(\lambda) = U(\lambda)\Phi(\lambda)SD(\lambda)\Phi(\lambda)U(\lambda)^{-1}.$$

If we denote by R_1 the left upper block of size $g \times g$ of the $n \times n$ matrix R, then the matrix H, defined by

$$H = S_1^{-1}\Phi_1(\lambda_0)^{-1}\left[U(\lambda_0)^* \frac{\partial B}{\partial \varepsilon}(\lambda_0, 0)U(\lambda_0)\right]_1 \Phi_1(\lambda_0)^{-1}, \tag{2.2}$$

where $B(\lambda, \varepsilon) := T(\lambda, \varepsilon) - T(\lambda, 0) = \lambda\varepsilon C$, determines the leading terms of the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 (see [6], [7]). The first g columns x_i, $i = 1, \cdots, g$, of the unitary matrix $U(\lambda_0)$ form an orthogonal basis of the kernel of $A(\lambda_0)$. Thus

$$\tilde{H} := \lambda_0[U(\lambda_0)^* CU(\lambda_0)]_1 = \lambda_0((Cx_j \mid x_i))_{i,j=1,\cdots,g}, \tag{2.3}$$

and if we put

$$\tilde{\Delta}_j := \det \tilde{H}(\tilde{n}_{j-1} + 1, \tilde{n}_{j-1} + 2, \cdots, g), \qquad j = 1, \cdots, k,$$

where $\tilde{H}(\tilde{n}_{j-1} + 1, \cdots, g)$ denotes the $(g - \tilde{n}_{j-1}) \times (g - \tilde{n}_{j-1})$ submatrix of $\tilde{H}$ in the right lower corner, then the numbers Δ_j from relation (5) of [6] and $\tilde{\Delta}_j$ differ only by a nonzero factor which originates from the diagonal matrices $S_1^{-1}\Phi_1(\lambda_0)^{-1}$ and $\Phi_1(\lambda_0)^{-1}$ in (2.2).

Below, we also use the following simple fact about the eigenvalues of a (not necessarily hermitian) quadratic polynomial $A(\lambda) = \lambda^2 I + \lambda C + K$. If x_0 is an eigenvector of $A(\lambda)$, corresponding to the eigenvalue λ_0, and if for x_0 there exists an associated vector y_0 of $A(\lambda)$:

$$A(\lambda_0)x_0 = 0, \qquad A(\lambda_0)y_0 + (2\lambda_0 + C)x_0 = 0, \tag{2.4}$$

then for each vector x_0^* from the kernel of $A(\lambda_0)^*$ we have

$$2\lambda_0(x_0 \mid x_0^*) + (Cx_0 \mid x_0^*) = 0. \tag{2.5}$$

In particular, if $A(\lambda)$ is hermitian, that is $C = C^*$, $K = K^*$ (or, equivalently, $A(\lambda)^* = A(\bar{\lambda})$), this x_0^* is an eigenvector of the polynomial $A(\lambda)$ corresponding to the eigenvalue $\bar{\lambda}_0$. In this case, if λ_0 is nonzero and real, we can choose $x_0^* = x_0$ and obtain

$$(Cx_0 \mid x_0) = -2\lambda_0 \|x_0\|^2 \neq 0.$$

If, however, λ_0 is nonreal, it may happen that $(Cx_0 \mid x_0^*) = 0$ for eigenvalues x_0, x_0^* of $A(\lambda)$ at λ_0 and $\bar{\lambda}_0$, respectively (see the example at the end of this section).

The main result of this section is the following theorem.

THEOREM 2.1. *Let C and K be hermitian $n \times n$ matrices and let $\lambda_0 \neq 0$ be a real eigenvalue of the matrix polynomial $A(\lambda) = \lambda^2 I + \lambda C + K$. Then the matrix function $T(\lambda, \varepsilon) = \lambda^2 I + \lambda(1 + \varepsilon)C + K$ has the regular splitting property at λ_0.*

Proof. We consider the matrix $\tilde{H}$ from (2.3). If $i \in \{1, \cdots, g\}$ is such that $m_i > 1$, then the identity (2.5) implies

$$(Cx_i \mid x_l) = -2\lambda_0(x_i \mid x_l) = -2\lambda_0\delta_{il}, \qquad l = 1, \cdots, g.$$

Therefore $\tilde{H}$ has the following form:

$$\tilde{H} = \lambda_0\left[\begin{array}{c|c} \cdots & 0 \\ \hline 0 & -2\lambda_0 I_{g-n_1} \end{array}\right] \quad \text{if } m_1 = \cdots = m_{n_1} = 1, m_{n_1+1} > 1, \tag{2.6}$$

where the left upper corner is an $n_1 \times n_1$ matrix which cannot, in general, be specified, and

$$\tilde{H} = -2\lambda_0^2 I_g \quad \text{if } m_1 > 1.$$

According to the remarks following (2.3), we see that for the determinants Δ_j from [6, eq. (5)] it holds that

$$\Delta_2 \cdots \Delta_k \neq 0.$$

Moreover, if $m_1 > 1$, then we have

$$\Delta_1 \cdots \Delta_k \neq 0.$$

The claim of Theorem 2.1 now follows from Corollary 1 of [7].

Remark. If $m_1 > 1$, these considerations and Theorem 1 of [6] imply that $T(\lambda, \varepsilon)$ has the completely regular splitting property at λ_0.

It is a well-known fact that if the algebraic multiplicity a of the eigenvalue λ_0 of the matrix polynomial $A(\lambda)$ is one, then the polynomial $T(\lambda, \varepsilon)$ has only one eigenvalue $\lambda(\varepsilon)$ near λ_0 and this eigenvalue is a holomorphic function of ε. In the following two theorems, we consider the special situations where $a > 1$ and the geometric multiplicity g of λ_0 equals either 1 or a. Evidently, if $g = 1$, then the matrix $\tilde{H}$ reduces to a number.

THEOREM 2.2. *Let C and K be hermitian $n \times n$ matrices, and let $\lambda_0 \in \mathbf{C}$ be an eigenvalue of the matrix polynomial $A(\lambda) = \lambda^2 I + \lambda C + K$ of geometric multiplicity $g = 1$ and algebraic multiplicity $a > 1$. Let x_0 and x_0^* be eigenvectors of $A(\lambda)$ at λ_0 and $\bar{\lambda}_0$, respectively. Then*

(i) *If λ_0 is nonreal and $(x_0 | x_0^*) \neq 0$, or if λ_0 is real and nonzero, then $T(\lambda, \varepsilon)$ has the regular splitting property at λ_0; the eigenvalues $\lambda_j(\varepsilon)$, $j = 1, \cdots, a$, of $T(\lambda, \varepsilon)$ near λ_0 form one group with Puiseux expansions*

$$\lambda_j(\varepsilon) = \lambda_0 + \sum_{k=1}^{\infty} \alpha_k \theta_{a,j}^k \varepsilon^{k/a}, \qquad j = 1, \cdots, a,$$

where $\alpha_1 \neq 0$.

(ii) *If $\lambda_0 = 0$, then $\lambda(\varepsilon) = 0$ is for $\varepsilon \neq 0$ an eigenvalue of $T(\lambda, \varepsilon)$ of geometric multiplicity one and algebraic multiplicity two; in the case where $a > 2$ the remaining $a - 2$ eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 belong to one group having Puiseux expansions*

$$\lambda_j(\varepsilon) = \sum_{k=1}^{\infty} \alpha_k \theta_{\alpha-2,j}^k \varepsilon^{k/(a-2)}, \qquad j = 1, \cdots, a-2,$$

with $\alpha_1 \neq 0$.

Proof. (i) If λ_0 is real, then the statement is an immediate consequence of Theorem 2.1 or the following considerations in the nonreal case, as we can take $x_0^* = x_0$, hence $(x_0 | x_0^*) \neq 0$. Therefore suppose $\lambda_0 \neq \bar{\lambda}_0$.

We linearize the polynomial $A(\lambda)$ by introducing the $2n \times 2n$ matrices

$$A = \begin{bmatrix} -C & -K \\ I & 0 \end{bmatrix}, \qquad G = \begin{bmatrix} 0 & I \\ I & C \end{bmatrix}, \qquad B = \begin{bmatrix} -C & 0 \\ 0 & 0 \end{bmatrix}.$$

Then the eigenvalue problems for $A(\lambda)$ and $T(\lambda, \varepsilon)$ are equivalent to the eigenvalue problems for $A - \lambda I$, $A + \varepsilon B - \lambda I$, respectively. Noting that the matrix A is G-hermitian, we can apply the remarks of [6], $n^0 3$, in particular, the relation (19). If x_0 (x_0^*, respec-

tively) is a eigenvector of $A(\lambda)$ at λ_0 ($\bar{\lambda}_0$, respectively), then

$$\begin{bmatrix} \lambda_0 x_0 \\ x_0 \end{bmatrix} \quad \left(\begin{bmatrix} \bar{\lambda}_0 x_0^* \\ x_0^* \end{bmatrix}, \text{ respectively} \right)$$

is an eigenvector of A at λ_0 (at $\bar{\lambda}_0$, respectively). Since the inner product $[\cdot, \cdot]$ in [6, eq. (19)] is defined by the matrix G above, it follows that

$$h_{11} = -\lambda_0 (Cx_0 | x_0^*),$$

which according to (2.5) equals $2\lambda_0^2 (x_0 | x_0^*)$. Now Theorem 1 of [6] can be applied and the statement follows.

(ii) We have shown before Proposition 1.5 that if $\lambda_0 = 0$ is an eigenvalue of $A(\lambda)$ of geometric multiplicity 1 and algebraic multiplicity greater than or equal to 2, then $\lambda(\varepsilon) = 0$ is an eigenvalue of $T(\lambda, \varepsilon)$ of algebraic multiplicity greater than or equal to 2. On the other hand, zero is an eigenvalue of $A(0)$ of geometric and algebraic multiplicity 1, which implies $D_{01} = s_1 \neq 0$. Using the identities (1.3)–(1.6) it is not hard to see that $t_{20} = \frac{1}{2} t_{21} + 2D_{01}$. If $a = 2$ there is nothing to prove. If $a > 2$, then $t_{20} = 0$, hence $t_{21} \neq 0$. Therefore $N(t)$ consists of the points (2.1) and $(a, 0)$ and the statement follows.

If $a > 1$ and $g = a$, then it follows from Corollary 5 of [7] that all the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 are real differentiable. If, additionally, $\Delta_1 = \det H$ is nonzero (the matrix H is defined by (2.2)), then it follows from Theorem 1 of [6] that all derivatives of the eigenvalues at $\varepsilon = 0$ are nonzero. In a special case we can prove that the eigenvalues form more than one group.

THEOREM 2.3. *Suppose that, with the notation of Theorem* 2.1, *for the real eigenvalue* $\lambda_0 \neq 0$ *of* $A(\lambda)$, *we have*

$$g = a > 1.$$

If all the σ_j, $j = 1, \cdots, g$, *are equal, then there are at least two groups of eigenvalues of* $T(\lambda, \varepsilon)$ *near* λ_0.

Proof. According to Corollary 7 of [7] (applied to $\tilde{A}(\lambda) = A(\lambda_0 - \lambda)$), if all the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 belong to one group, then there exists an $\alpha \in \mathbf{R}$ such that $H = \alpha I$. We consider again the Rellich representation (2.1) of $A(\lambda)$. If x_i is the ith column of $U(\lambda_0)$, it follows from (2.2) that

$$\lambda_0 (Cx_i | x_j) = \sigma_i \varphi_i(\lambda_0) \varphi_j(\lambda_0) h_{ij}, \qquad i, j = 1, \cdots, g.$$

As $\mu_i(\lambda) = \sigma_i (\lambda - \lambda_0) \varphi_i(\lambda)^2$ and $h_{ij} = \alpha \delta_{ij}$, we conclude that

$$\lambda_0 (Cx_i | x_j) = \alpha \frac{d\mu_i}{d\lambda}(\lambda_0) \delta_{ij}. \tag{2.7}$$

Differentiating the identity $\mu_i(\lambda) = (A(\lambda) U(\lambda) e_i | U(\lambda) e_i)$ at $\lambda = \lambda_0$, we find

$$\frac{d\mu_i}{d\lambda}(\lambda_0) = \left(\frac{dA}{d\lambda}(\lambda_0) x_i | x_i \right) = ((C + 2\lambda_0) x_i | x_i).$$

Using the identity (2.7) it follows that

$$\lambda_0 (Cx_i | x_j) = \alpha ((C + 2\lambda_0) x_i | x_j) \delta_{ij},$$

hence $\lambda_0 \neq \alpha$ and

$$(Cx_i | x_j) = \frac{2\alpha\lambda_0}{\lambda_0 - \alpha} \delta_{ij}, \qquad i, j = 1, \cdots, g.$$

Let $X_0 := \ker A(\lambda_0)$, $X_1 := X_0^{\perp}$. With respect to the decomposition $\mathbf{C}^n = X_0 \oplus X_1$ the matrix C has the block structure

$$C = \begin{bmatrix} dI & C_2^* \\ C_2 & C_3 \end{bmatrix},$$

where $d := 2\alpha\lambda_0/(\lambda_0 - \alpha)$ and C_3 is hermitian. Therefore K and $T(\lambda, \varepsilon)$ decompose as follows:

$$K = \begin{bmatrix} -(\lambda_0^2 + \lambda_0 d)I & -\lambda_0 C_2^* \\ -\lambda_0 C_2 & R - \lambda_0 C_3 - \lambda_0^2 I \end{bmatrix},$$

$$T(\lambda, \varepsilon) = \begin{bmatrix} h(\lambda, \varepsilon)I & (\lambda + \lambda\varepsilon - \lambda_0)C_2^* \\ (\lambda + \lambda\varepsilon - \lambda_0)C_2 & R(\lambda, \varepsilon) \end{bmatrix}$$

with

$$h(\lambda, \varepsilon) := \lambda^2 - \lambda_0^2 + (\lambda + \lambda\varepsilon - \lambda_0)d,$$

$$R(\lambda, \varepsilon) := (\lambda^2 - \lambda_0^2)I + (\lambda + \lambda\varepsilon - \lambda_0)C_3 + R,$$

where $\det R \neq 0$. If

$$T = \begin{bmatrix} T_{11} & T_{12} \\ T_{21} & T_{22} \end{bmatrix}$$

is a block matrix and $\det T_{11} \neq 0$, then $\det T = \det T_{11} \det (T_{22} - T_{21}T_{11}^{-1}T_{12})$. Hence we find

$$t(\lambda, \varepsilon) := \det T(\lambda, \varepsilon) = h(\lambda, \varepsilon)^g \det \left[R(\lambda, \varepsilon) - \frac{(\lambda + \lambda\varepsilon - \lambda_0)^2}{h(\lambda, \varepsilon)} C_2 C_2^* \right]. \tag{2.8}$$

Since $R = R(\lambda_0, 0)$ is invertible, it follows that $R(\lambda, \varepsilon)$ is invertible for (λ, ε) near $(\lambda_0, 0)$. For such values of λ, ε define

$$Q(\lambda, \varepsilon) := C_2^* R(\lambda, \varepsilon)^{-1} C_2.$$

Applying the identity $\det (I - AB) = \det (I - BA)$ to (2.8), it follows that

$$t(\lambda, \varepsilon) = \det R(\lambda, \varepsilon) \det (h(\lambda, \varepsilon)I - (\lambda + \lambda\varepsilon - \lambda_0)^2 Q(\lambda, \varepsilon)). \tag{2.9}$$

Let $q_j(\lambda, \varepsilon)$, $j = 1, \cdots, g$, be the eigenvalues of the matrix $Q(\lambda, \varepsilon)$. As $Q(\lambda, \varepsilon)$ is hermitian for real λ and ε, the eigenvalues $q_j(\lambda, \varepsilon)$ are also real analytic in λ and ε. Furthermore, let

$$f_j(\lambda, \varepsilon) := h(\lambda, \varepsilon) + (\lambda + \lambda\varepsilon - \lambda_0)^2 q_j(\lambda, \varepsilon).$$

It follows from (2.9) that the eigenvalue equation $t(\lambda, \varepsilon) = 0$ factors into g equations

$$f_j(\lambda, \varepsilon) = 0, \qquad j = 1, \cdots, g. \tag{2.10}$$

The analyticity of f_j near $(\lambda_0, 0)$ and the relations

$$\frac{\partial f_j}{\partial \lambda}(\lambda_0, 0) = \frac{\partial h}{\partial \lambda}(\lambda_0, 0) = 2\lambda_0 + d = \frac{1}{1 - \alpha} \neq 0,$$

imply by the implicit function theorem that each equation in (2.10) gives a real analytic

function $\lambda_j(\varepsilon)$ near $\varepsilon = 0$, that is, the eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 form g "groups," a contradiction. The theorem is proved.

The following example shows that the regular splitting property does not always hold for nonreal λ_0. Let

$$C = \begin{bmatrix} 2 & 0 & 0 \\ 0 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix}, \qquad K = \begin{bmatrix} 2 & 0 & -1 \\ 0 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}.$$

Then the matrix polynomial $A(\lambda) = \lambda^2 I + \lambda C + K$ has the eigenvalue $\lambda_0 = -1 + i$ with geometric multiplicity $g = 1$ and algebraic multiplicity $a = 2$. The algebraic multiplicity b of the eigenvalue zero of $A(\lambda_0)$ is three. A calculation gives

$$t_{00} = t_{10} = t_{01} = 0, \qquad t_{20} = 8, \qquad t_{11} = 8i, \qquad t_{02} = -16,$$

and therefore the essential part $t_0(\lambda, \varepsilon)$ of $t(\lambda, \varepsilon)$ near $(\lambda_0, 0)$ is

$$t_0(\lambda, \varepsilon) = 4(\lambda - \lambda_0)^2 + 8i\varepsilon(\lambda - \lambda_0) - 8\varepsilon^2$$

and the two analytic eigenvalues of $T(\lambda, \varepsilon)$ are

$$\lambda_1(\varepsilon) = 1 - i + (1 - i)\varepsilon + O(|\varepsilon|^2),$$

$$\lambda_2(\varepsilon) = 1 - i - (1 + i)\varepsilon + O(|\varepsilon|^2).$$

Note that in this example the eigenvector of $A(\lambda)$ corresponding to λ_0 is

$$x_0 = \begin{bmatrix} 1 \\ i \\ 0 \end{bmatrix},$$

the eigenvalue corresponding to $\bar{\lambda}_0$ is

$$x_0^* = \begin{bmatrix} i \\ 1 \\ 0 \end{bmatrix},$$

hence $(x_0 | x_0^*) = 0$. The matrices C and K are positive definite.

Finally, let the (hermitian) matrix C be of rank one. If $\lambda_0 \neq 0$ is a real eigenvalue of $A(\lambda)$ such that $-\lambda_0^2$ is not an eigenvalue of K, then

$$\operatorname{codim} \mathscr{R}(A(\lambda_0)) = 1.$$

Therefore $g = 1$, and it follows from Theorem 2.2 that the regular splitting property always holds in this case.

3. Some results for definite matrices C and K. In Theorem 2.1 we have shown that for a real eigenvalue $\lambda_0 \neq 0$ of $A(\lambda)$ the number of real differentiable eigenvalues of $T(\lambda, \varepsilon)$ near λ_0 coincides with the number of partial multiplicities m_j which are equal to one. Here we prove that if C is semidefinite and $\lambda(\varepsilon)$ is a real differentiable eigenvalue of $T(\lambda, \varepsilon)$, then we have either $\lambda'(0) \neq 0$ or $\lambda(\varepsilon) = \lambda_0$ for all ε in a neighbourhood of $\varepsilon = 0$.

THEOREM 3.1. *Let $C, K, A(\lambda), T(\lambda, \varepsilon)$, and λ_0 be as in Theorem 2.1, and suppose additionally that C is semidefinite. If $\lambda(\varepsilon)$ is a real differentiable eigenvalue of $T(\lambda, \varepsilon)$ near λ_0 with $(d\lambda/d\varepsilon)(0) = 0$, then $\lambda(\varepsilon) = \lambda_0$ for all ε in a neighbourhood of $\varepsilon = 0$.*

Proof. (1) We first reduce the general case to special ones. Let L_0 be the subspace of $\mathbf{C}^n$ spanned by the common eigenvectors of C and K and let L_1 be the subspace of

vectors from L_0 which belong to the kernel of C. Finally, define L_2 to be the orthogonal complement of L_1 in L_0 and L_3 to be the orthogonal complement of L_0 in $\mathbf{C}^n$. Then we have

$$\mathbf{C}^n = L_1 \oplus L_2 \oplus L_3$$

and, with respect to this decomposition,

$$T(\lambda,\varepsilon)=\begin{bmatrix} T_1(\lambda,\varepsilon) & 0 & 0 \\ 0 & T_2(\lambda,\varepsilon) & 0 \\ 0 & 0 & T_3(\lambda,\varepsilon) \end{bmatrix},$$

$$A(\lambda)=\begin{bmatrix} A_1(\lambda) & 0 & 0 \\ 0 & A_2(\lambda) & 0 \\ 0 & 0 & A_3(\lambda) \end{bmatrix},$$

where $T_1(\lambda, \varepsilon) = A_1(\lambda) = \lambda^2 + K$, $T_2(\lambda, \varepsilon) = \operatorname{diag}(\lambda^2 + \lambda(1 + \varepsilon)c_j + k_j)$ if the common eigenvectors of C and K are chosen as a basis of L_2 and c_j, k_j denote the corresponding eigenvalues of C and K, respectively, $c_j \neq 0$, and $T_3(\lambda, \varepsilon) = \lambda^2 I + \lambda(1 + \varepsilon)C_3 + K_3$, where C_3 and K_3 have no common eigenvector.

(2) Without loss of generality, we can suppose that C is nonnegative definite (otherwise we replace C by $-C$ and λ by $-\lambda$). Let $\lambda(\varepsilon)$ be a real differentiable eigenvalue of $T(\lambda, \varepsilon)$ near λ_0 with $\lambda'(0) = 0$. Then $\lambda(\varepsilon)$ is an eigenvalue of a component $T_j(\lambda, \varepsilon)$, $j = 1, 2, 3$. If it is an eigenvalue of $T_1(\lambda, \varepsilon)$, it is evidently independent of ε. If it is an eigenvalue of $T_2(\lambda, \varepsilon)$, then it must be a solution of an equation

$$\lambda^2 + \lambda(1+\varepsilon)c_j + k_j = 0$$

with $c_j \neq 0$. Implicit differentiation yields $2\lambda_0\lambda'(0) + \lambda'(0)c_j + \lambda_0 c_j = 0$, hence $\lambda'(0) = 0$ would imply $\lambda_0 c_j = 0$, which is impossible. It remains to consider the case that $\lambda(\varepsilon)$ is an eigenvalue of $T_3(\lambda, \varepsilon)$.

The semidefiniteness of C implies $(C_3 y | y) > 0$ for all $y \in L_3 \cap \ker T_3(\lambda_0, 0)$, $y \neq 0$. Indeed $(C_3 y | y) = 0$ yields $C_3 y = 0$, hence $(\lambda_0^2 + K_3)y = 0$, which is impossible as C_3 and K_3 have no common eigenvector. Therefore the left upper corner in the matrix $\tilde{H}$ in (2.6) associated with $T_3(\lambda, \varepsilon)$ is positive definite and consequently the corresponding determinant Δ_1 is nonzero. Now the claim follows from Theorem 1 of [6].

Remark. It follows from the proof of Theorem 3.1 that the semidefiniteness of C can be replaced by the semidefiniteness of C on $\ker A(\lambda_0)$ (or of C_3 on $\ker A_3(\lambda_0)$).

Finally, we prove two results about the eigenvalues $\lambda(\eta)$ of the matrix polynomial $S(\lambda, \eta) := \lambda^2 I + \lambda\eta C + K$, which is considered as a perturbation of $P(\lambda) := \lambda^2 + K$ by $\lambda\eta C$, if K is positive definite.

THEOREM 3.2. *Let C be a hermitian and K a positive-definite hermitian matrix, and let λ_0 be a (purely imaginary) eigenvalue of $P(\lambda) = \lambda^2 + K$ of (geometric and algebraic) multiplicity g. Then all g eigenvalues $\lambda(\eta)$ of $S(\lambda, \eta) = \lambda^2 I + \lambda\eta C + K$ near λ_0 are analytic functions at $\eta = 0$ and $(d\lambda/d\eta)(0)$ is real.*

Proof. Let $K^{1/2}$ be the positive-definite square root of K. Define

$$B:=\begin{bmatrix} 0 & K^{1/2} \\ -K^{1/2} & 0 \end{bmatrix}, \qquad C:=\begin{bmatrix} 0 & 0 \\ 0 & -C \end{bmatrix}.$$

Then the eigenvalue problem for $P(\lambda)$ and $S(\lambda, \eta)$ is equivalent to the eigenvalue problem for $B - \lambda I$ and $B + \eta C - \lambda I$, respectively. Making use of the observation [12] that $iB - \eta C$ is a hermitian matrix for real η, it follows that its eigenvalues $\mu(\eta)$ are analytic

functions at $\eta = 0$ with real coefficients. This proves the theorem as $\lambda(\eta) = -i\mu(-i\eta)$ for some $\mu(\eta)$.

THEOREM 3.3. *Let C be a hermitian and K a positive-definite hermitian matrix. Then there are at most finitely many $\eta \in \mathbf{C}$ such that some eigenvalue $\lambda(\eta)$ of $S(\lambda, \eta) = \lambda^2 I + \lambda\eta C + K$ is not semisimple (i.e., that not all of its partial multiplicities are one).*

Proof. According to the general matrix perturbation theory ([1] or [5]) the eigenvalues, the eigenprojections, and the eigennilpotents of $S(\eta) = B + \eta C$ are algebraic functions of η with an (at most) finite set of singularities. Therefore, for any η which does not belong to this finite set of exceptional points, we find a connected open set $\mathscr{D}$ which contains η, intersects the imaginary axis, and contains no singular point. Since $S(\eta)$ is antihermitian if η is on the imaginary axis, the eigennilpotents vanish there and consequently they vanish on $\mathscr{D}$.

REFERENCES

[1] H. BAUMGÄRTEL, *Analytic Perturbation Theory for Matrices and Operators*, Akademie-Verlag, Berlin, 1984.

[2] A. S. DEIF AND P. HAGEDORN, *Matrix polynomials subjected to small perturbations*, Z. Angew. Math. Mech., 66 (1986), pp. 403–412.

[3] I. GOHBERG, P. LANCASTER, AND L. RODMAN, *Matrix Polynomials*, Academic Press, New York, 1982.

[4] ———, *Quadratic matrix polynomials with a parameter*, Adv. in Appl. Math., 7 (1986), pp. 253–281.

[5] T. KATO, *Perturbation Theory for Linear Operators*, Springer-Verlag, Berlin, Heidelberg, New York, 1966.

[6] H. LANGER AND B. NAJMAN, *Remarks on the perturbation of analytic matrix functions* II, Integral Equations Operator Theory, 12 (1989), pp. 392–407.

[7] ———, *Remarks on the perturbation of analytic matrix functions* III, Integral Equations Operator Theory, Submitted.

[8] B. NAJMAN, *Remarks on the perturbation of analytic matrix functions*, Integral Equations Operator Theory, 9 (1986), pp. 593–599.

[9] ———, *The Newton diagram method and the perturbation of eigenvalues*, Ber. Math.-Statist. Sekt. Forschungsgesellsch. Joanneum, 304 (1989), pp. 101–110.

[10] K. VESELIĆ, *On linear vibrational systems with one dimensional damping* II, preprint, Fernuniversität Hagen, Hagen, FRG, 1989; Integral Equations Operator Theory, to appear.

[11] M. M. WAINBERG AND W. A. TRENOGIN, *Theorie der Lösungsverzweigung bei nichtlinearen Gleichungen*, Akademie-Verlag, Berlin, 1973.

[12] A. WIEGNER, personal communication, 1988.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 490–506, April 1992

005

A LOOK-AHEAD LEVINSON ALGORITHM FOR INDEFINITE TOEPLITZ SYSTEMS*

TONY F. CHAN† AND PER CHRISTIAN HANSEN‡

Abstract. An extension of Levinson's algorithm for solving linear systems with symmetric indefinite Toeplitz matrices is presented. This new algorithm is able to "look ahead" and, if necessary, use block Gaussian elimination to skip all ill-conditioned leading principal submatrices encountered during the recursive processing. This makes the new algorithm numerically stable for a broader class of symmetric Toeplitz matrices than the standard Levinson algorithm. In addition, a reliable condition number estimate is produced. The overhead is typically small.

Key words. symmetric indefinite Toeplitz matrices, Levinson's algorithm, Durbin's algorithm, numerical stability, condition estimation

AMS(MOS) subject classification. 65F30

1. Introduction. Many matrix problems arising in signal processing, as well as in several other applications, involve Toeplitz matrices (see, e.g., [2], [5], [12]–[14]). In some of these applications the Toeplitz matrices are guaranteed to be positive definite; but in other applications such as *eigenfilter problems*, *harmonic retrieval*, and *linear prediction*, the matrices may also be indefinite [7]. Since these matrices are often large and dense, the Toeplitz structure must be exploited when solving the system of equations with the Toeplitz matrix, but preferably without sacrificing the numerical stability of the algorithm. Our new algorithm is a first step in devising such a general algorithm.

In this presentation, for clarity we restrict our discussion to *real symmetric* Toeplitz matrices:

$$T_n = \begin{bmatrix} \rho_0 & \rho_1 & \rho_1 & \cdots & \rho_{n-1} \\ \rho_1 & \rho_0 & \rho_2 & \cdots & \rho_{n-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \rho_{n-1} & \rho_{n-2} & \rho_{n-3} & \cdots & \rho_0 \end{bmatrix} \in \mathbb{R}^{n\times n}. \tag{1.1}$$

The extension to the nonsymmetric case is considered in forthcoming papers [4] and [10]. The classical algorithm for solving Toeplitz systems $T_n x = b = [\beta_1, \cdots, \beta_n]^T$ is the *Levinson algorithm* [1], [14], which requires $3n^2$ multiplications for general matrices and $2n^2$ multiplications for symmetric matrices. From a linear algebraic point of view, the classical Levinson algorithm for solving the problem $T_n x = b$, as well as the classical Durbin algorithm [8, § 4.7.2] for solving the Yule–Walker problem $T_n y = -r = -[\rho_1, \cdots, \rho_n]^T$, are basically recursive sequences of updating the solutions to increasingly larger systems of equations involving all the leading submatrices T_k, $k = 1, \cdots, n-1$ of T_n in their natural order. Hence, the accuracy of the computed solutions x and y implicitly depends on the condition of all these submatrices. In the general case, when the matrix is indefinite, one or more of the leading submatrices T_k may be arbitrarily close to being singular, even if T_n is well conditioned. Only if the matrix T_n is positive definite and well conditioned can we guarantee that all the leading submatrices are well

* Received by the editors August 1, 1989; accepted for publication (in revised form) September 6, 1990. This work was supported by National Science Foundation contract NSF-DMS87-14612.

† Department of Mathematics, University of California, 405 Hilgard Ave., Los Angeles, California 90024 (chan@math.ucla.edu).

‡ UNI•C (Danish Computing Center for Research and Education), Building 305, Technical University of Denmark, DK-2800 Lyngby, Denmark (unipch@wuli.uni-c.dk).

conditioned, and both algorithms are therefore numerically stable only for this class of matrices [2]. The same is true for all the "fast" $O(n \log^2 n)$ algorithms (cf. the discussion in [2]). The $O(n^2)$ QR-factorization due to Sweet [17] may also break down for indefinite matrices [15], and the complexity is $25n^2$, i.e., much larger than that of the Levinson algorithm. Therefore, it is important to develop an efficient algorithm for general Toeplitz matrices.

Our approach is essentially a block method using block Gaussian elimination steps, and in each step the block size is adjusted adaptively to ensure optimal conditioning of the submatrix involved. This idea is based on the fact that, during the Levinson algorithm, it is possible to compute inexpensive estimates of the condition number of the leading submatrices. Hence, our algorithm can "look ahead" and always choose locally the best-conditioned leading submatrix for the block Gaussian elimination step. As a result, the new algorithm is numerically stable for a broader class of matrices than the classical Levinson algorithm, namely those matrices that have at most $p - 1$ consecutive ill-conditioned leading submatrices. The overhead is difficult to predict, because it depends on the number of ill-conditioned submatrices, as well as on the maximum number of steps p that the user is willing to "look ahead." In the worst case (which is very unlikely to occur in practice) the complexity of the extended algorithm is about $\frac{1}{2}(p^3/6 + (5/2)p^2 + (10/3)p + 3)n^2$ multiplications. For positive-definite matrices the relative overhead decreases with the order n of the matrix, and for $n = 64$ it is less than 10 percent. For indefinite matrices with one ill-conditioned leading submatrix, which typically appear in Toeplitz eigenvalue problems [6], [19], the overhead is about 25 percent. An indefinite matrix of order $n = 120$ with 39 distinct singular leading submatrices required an overhead of 44 percent. Note that much of this overhead is due to the necessary condition estimation in each step.

The idea of skipping certain leading submatrices is a natural one and, in fact, has been used before by several authors [7], [9], [18], [20]. In all these algorithms, the decision for taking a block step is based solely on the size of the so-called "prediction error," which appears as a denominator in the algorithms. The prediction error is zero if and only if the corresponding leading submatrix is singular. In [7], [9], and [20], a block step is taken only if a true zero prediction error is encountered. Moreover, these algorithms make explicit use of the fact that the leading submatrix is singular, and therefore need further development before they can be used in finite-precision arithmetic. Sweet [18] uses a simple threshold-type test wherein he takes a block step when the prediction error is less than a preset threshold value. However, an ill-conditioned leading submatrix is not guaranteed to be revealed by a small prediction error, and his algorithm is therefore not guaranteed to detect the need for a block step. Our algorithm is based on a much more reliable—and still efficient—test for ill-conditioned submatrices.

The paper is organized as follows. In § 2, we review the classical Levinson algorithm, while the new extended Levinson algorithm is presented in § 3. In § 4, we briefly discuss its numerical stability, and in § 5 we give some important details concerning the practical implementation of the extended algorithm. Finally, in § 6 we present some numerical results.

2. The classical Levinson algorithm. Since our new algorithm is basically an extension of the classical Levinson algorithm, let us first summarize this algorithm following the formulation in [8, § 4.7.3]. At the kth stage, we have recursively computed the solution $x_k \in \mathbb{R}^k$ to the order-k problem

$$T_k x_k = b_k = [\beta_1, \cdots, \beta_k]^T \tag{2.1}$$

and, simultaneously, the solution $y_k \in \mathbb{R}^k$ to the Yule–Walker problem of order k,

$$T_k y_k = -r_k = -[\rho_1, \cdots, \rho_k]^T. \tag{2.2}$$

The next step is then to solve the problems $T_{k+1}x_{k+1} = b_{k+1}$ and $T_{k+1}y_{k+1} = -r_{k+1}$. First, we factorize the matrix T_{k+1} as follows:

$$T_{k+1} = \begin{bmatrix} T_k & E_k r_k \\ r_k^T E_k & \rho_0 \end{bmatrix} = \begin{bmatrix} I_k & 0 \\ r_k^T E_k T_k^{-1} & 1 \end{bmatrix} \begin{bmatrix} T_k & E_k r_k \\ 0^T & \gamma^{(k)} \end{bmatrix}. \tag{2.3}$$

Here, and throughout the paper, E_k denotes the exchange matrix antidiag $(1, \cdots, 1) \in \mathbb{R}^{k\times k}$. Note that $E_k^2 = I_k$ and that a Toeplitz matrix as well as its inverse are persymmetric, i.e., $T_k E_k = E_k T_k^T$ and $T_k^{-1}E_k = E_k(T_k^{-1})^T$. Since T_k is symmetric, this means that the Schur complement $\gamma^{(k)}$ of T_k in T_{k+1} is given by

$$\gamma^{(k)} \equiv \rho_0 - r_k^T E_k T_k^{-1} E_k r_k = \rho_0 - r_k^T T_k^{-1} r_k = \rho_0 + r_k^T y_k. \tag{2.4}$$

Now, write

$$x_{k+1} = \begin{pmatrix} \nu_k \\ \mu^{(k)} \end{pmatrix} \quad \text{and} \quad y_{k+1} = \begin{pmatrix} u_k \\ \alpha^{(k)} \end{pmatrix}.$$

Using the factorization in (2.3), it is easy to see that x_{k+1} and y_{k+1} are the solutions to the systems

$$\begin{bmatrix} T_k & E_k r_k \\ 0^T & \gamma^{(k)} \end{bmatrix} \begin{pmatrix} \nu_k \\ \mu^{(k)} \end{pmatrix} = \begin{pmatrix} b_k \\ \beta_{k+1} - r_k^T E_k x_k \end{pmatrix}, \tag{2.5a}$$

$$\begin{bmatrix} T_k & E_k r_k \\ 0^T & \gamma^{(k)} \end{bmatrix} \begin{pmatrix} u_k \\ \alpha^{(k)} \end{pmatrix} = \begin{pmatrix} -r_k \\ -\rho_{k+1} - r_k^T E_k y_k \end{pmatrix}. \tag{2.5b}$$

Solving these systems by backsubstitution, we see that the new vectors x_{k+1} and y_{k+1} are given by the following updates to x_k and y_k:

$$x_{k+1} = \begin{pmatrix} x_k \\ 0 \end{pmatrix} + \begin{pmatrix} E_k y_k \\ 1 \end{pmatrix} \mu^{(k)}, \qquad y_{k+1} = \begin{pmatrix} y_k \\ 0 \end{pmatrix} + \begin{pmatrix} E_k y_k \\ 1 \end{pmatrix} \alpha^{(k)} \tag{2.6}$$

where the scalars $\mu^{(k)}$ and $\alpha^{(k)}$ are computed by

$$\mu^{(k)} = \frac{\beta_{k+1} - r_k^T E_k x_k}{\gamma^{(k)}}, \qquad \alpha^{(k)} = \frac{-\rho_{k+1} - r_k^T E_k y_k}{\gamma^{(k)}}. \tag{2.7}$$

The Durbin algorithm for solving the Yule–Walker problem is incorporated in the Levinson algorithm because the vector y_k is needed in the updating (2.6) of x_k.

For efficiency, $\gamma^{(k)}$ should not be computed via its definition (2.4). Instead, it is computed recursively along with the x_k and y_k by the following recursion (starting with $\gamma^{(0)} = \rho_0$ and $\alpha^{(0)} = -\rho_1/\rho_0$):

$$\gamma^{(k+1)} = \gamma^{(k)} - (-\rho_{k+1} - r_k^T E_k y_k)\alpha^{(k)} = (1 - (\alpha^{(k)})^2)\gamma^{(k)}. \tag{2.8}$$

It is easy to show that the classical Levinson algorithm for symmetric Toeplitz matrices requires $2n^2 - 2$ multiplications and divisions. Because of the division by $\gamma^{(k)}$ in (2.7), we immediately see that instability will occur if one or several successive $\gamma^{(k)}$ encountered during the recursive process become numerically small. This cannot occur for well-conditioned symmetric positive-definite matrices, for which there is a monotonic decrease of the $\gamma^{(k)}$ so that they cannot be smaller than the final value, but it is possible for

indefinite matrices. To avoid a small $|\gamma^{(k)}|$ we could incorporate pivoting, but this would destroy the Toeplitz structure. This motivates us to perform a block step instead, as described in the next section.

3. The extended Levinson algorithm. We are now ready to describe our extension of the Levinson algorithm. Assume, at stage k, that we have encountered an ill-conditioned T_{k+1}. We therefore wish to perform a p-step, i.e., to skip ahead from T_k to a well-conditioned T_{k+p}. Assume for the moment that p is given; in § 5 we address the question of how to actually choose p. Let us define the *shifted* vectors $r_{k,i} \equiv [\rho_{1+i}, \rho_{2+i}, \cdots, \rho_{k+i}]^T$ (in particular, $r_{k,0} = r_k$) and the $k \times p$ matrix $R_p \equiv [r_{k,0}, \cdots, r_{k,p-1}]$. Then we can write

$$T_{k+p} = \begin{bmatrix} T_k & E_k R_p \\ R_p^T E_k & T_p \end{bmatrix}. \tag{3.1}$$

In order to update the x_k and y_k to x_{k+p} and y_{k+p}, we proceed in a similar fashion as in the classical Levinson algorithm and use block Gaussian elimination to solve the two $(k+p)$-by-$(k+p)$ systems $T_{k+p}x_{k+p} = b_{k+p}$ and $T_{k+p}y_{k+p} = -r_{k+p}$. First, factorize T_{k+p} as

$$T_{k+p} = \begin{bmatrix} I_k & 0 \\ R_p^T E_k T_k^{-1} & I_p \end{bmatrix} \begin{bmatrix} T_k & E_k R_p \\ 0 & \Gamma_p^{(k)} \end{bmatrix}, \tag{3.2}$$

where the symmetric Schur complement $\Gamma_p^{(k)} \in \mathbb{R}^{p \times p}$ of T_k is given by

$$\Gamma_p^{(k)} = T_p - R_p^T E_k T_k^{-1} E_k R_p = T_p - R_p^T T_k^{-1} R_p = T_p + R_p^T Y_p \tag{3.3}$$

and where the matrix Y_p is defined as the solution to the problem

$$T_k Y_p = -R_p. \tag{3.4}$$

Note that the first column of Y_p is simply y_k. Now, write

$$x_{k+p} = \begin{pmatrix} v_k \\ w_p^{(k)} \end{pmatrix} \quad \text{and} \quad y_{k+p} = \begin{pmatrix} u_k \\ z_p^{(k)} \end{pmatrix}$$

and define

$$b_{k+p} = \begin{pmatrix} b_k \\ b_p^{(k)} \end{pmatrix} \quad \text{and} \quad r_{k+p} = \begin{pmatrix} r_k \\ r_p^{(k)} \end{pmatrix},$$

where $b_p^{(k)} = [\beta_{k+1}, \cdots, \beta_{k+p}]^T$ and $r_p^{(k)} = [\rho_{k+1}, \cdots, \rho_{k+p}]^T$. Then we use the factorization in (3.2) to obtain the systems

$$\begin{bmatrix} T_k & E_k R_p \\ 0 & \Gamma_p^{(k)} \end{bmatrix} \begin{pmatrix} v_k \\ w_p^{(k)} \end{pmatrix} = \begin{pmatrix} b_k \\ b_p^{(k)} - R_p^T E_k x_k \end{pmatrix}, \tag{3.5a}$$

$$\begin{bmatrix} T_k & E_k R_p \\ 0 & \Gamma_p^{(k)} \end{bmatrix} \begin{pmatrix} u_k \\ z_p^{(k)} \end{pmatrix} = \begin{pmatrix} -r_k \\ -r_p^{(k)} - R_p^T E_k y_k \end{pmatrix}. \tag{3.5b}$$

If we solve these systems by backsubstitution, we see that the new vectors x_{k+p} and y_{k+p} are given by the following updates to x_k and y_k:

$$x_{k+p} = \begin{pmatrix} x_k \\ 0 \end{pmatrix} + \begin{pmatrix} E_k Y_p \\ I_p \end{pmatrix} w_p^{(k)}, \qquad y_{k+p} = \begin{pmatrix} y_k \\ 0 \end{pmatrix} + \begin{pmatrix} E_k Y_p \\ I_p \end{pmatrix} z_p^{(k)}, \tag{3.6}$$

where $w_p^{(k)}$ and $z_p^{(k)}$ are the solutions to the symmetric indefinite systems

$$\Gamma_p^{(k)} w_p^{(k)} = b_p^{(k)} - R_p^T E_k x_k, \qquad \Gamma_p^{(k)} z_p^{(k)} = -r_p^{(k)} - R_p^T E_k y_k. \tag{3.7}$$

It is easy to see that this will handle the instability problem as long as p is chosen properly. It would be expensive if we were to solve for the extraneous $p-1$ vectors in Y_p (3.4) naively, say using Levinson's algorithm, because we would have to choose an a priori number $p_{\max}$ such that $p < p_{\max}$ and then carry along $p_{\max} - 1$ extra systems, thus increasing the overhead of the algorithm by $(p_{\max} - 1)n^2$ multiplications. Instead, we present an updating procedure which is cheaper, because we only compute Y_p when necessary. The idea is to write the columns of Y_p in terms of updates to already computed quantities. This is possible due to the following theorem.

THEOREM 1. *Let the p columns of Y_p be denoted by $y_{k,i}$, $i = 1, \cdots, p-1$, with each column satisfying $T_k y_{k,i} = -r_{k,i}$. Also, let $(y_{k,i})_1$ denote the first component of the vector $y_{k,i}$, and define the "upshift" matrix*

$$\Delta_k \equiv \begin{pmatrix} 0 & 1 & & & 0 \\ 0 & 0 & 1 & & \\ & 0 & 0 & \ddots & \\ & & \ddots & \ddots & 1 \\ 0 & & & 0 & 0 \end{pmatrix} \in \mathbb{R}^{k \times k}. \tag{3.8}$$

Then all the vectors $y_{k,i}$ can be computed recursively from $y_{k,0} = y_k$ by means of the relation

$$y_{k,i} = \Delta_k y_{k,i-1} - (y_{k,i-1})_1 y_k + c_i s_k, \qquad i = 1, \cdots, p-1 \tag{3.9}$$

where we have defined the vector

$$s_k \equiv \frac{1}{\gamma^{(k-1)}} \begin{pmatrix} E_{k-1} y_{k-1} \\ 1 \end{pmatrix} \tag{3.10}$$

and where c_i is the ith component of the right-hand side of the second equation in (3.7)

$$c = -r_p^{(k)} - R_p^T E_k y_k. \tag{3.11}$$

Proof. Consider the following matrix-vector product of order $k+1$:

$$T_{k+1} \begin{pmatrix} y_{k,i-1} \\ 0 \end{pmatrix} = \begin{pmatrix} \rho_0 & r_k^T \\ r_k & T_k \end{pmatrix} \begin{pmatrix} (y_{k,i-1})_1 \\ \Delta_k y_{k,i-1} \end{pmatrix} = \begin{pmatrix} \rho_0 (y_{k,i-1})_1 + r_k^T \Delta_k y_{k,i-1} \\ (y_{k,i-1})_1 r_k + T_k \Delta_k y_{k,i-1} \end{pmatrix}. \tag{3.12}$$

Using $T_k y_{k,i} = -r_{k,i}$ and (3.4) it follows that this vector is also given by

$$\begin{aligned} T_{k+1} \begin{pmatrix} y_{k,i-1} \\ 0 \end{pmatrix} &= \begin{pmatrix} T_k & E_k r_k \\ r_k^T E_k & \rho_0 \end{pmatrix} \begin{pmatrix} y_{k,i-1} \\ 0 \end{pmatrix} = \begin{pmatrix} -r_{k,i-1} \\ r_k^T E_k y_{k,i-1} \end{pmatrix} \\ &= \begin{pmatrix} -\rho_i \\ -r_{k,i} \end{pmatrix} - \begin{pmatrix} 0 \\ -\rho_{k+i} - r_k^T E_k y_{k,i-1} \end{pmatrix} \\ &= \begin{pmatrix} -\rho_i \\ -r_{k,i} \end{pmatrix} - \begin{pmatrix} 0 \\ -\rho_{k+i} - y_k^T E_k r_{k,i-1} \end{pmatrix}. \end{aligned} \tag{3.13}$$

In the last step, we have used $r_k^T E_k y_{k,i-1} = -y_k^T T_k E_k y_{k,i-1} = -y_k^T E_k T_k y_{k,i-1} = y_k^T E_k r_{k,i-1}$. Equating the last k elements of the vectors in (3.12) and (3.13), we obtain

$$T_k \Delta_k y_{k,i-1} = -r_{k,i} - (y_{k,i-1})_1 r_k - \begin{pmatrix} 0 \\ c_i \end{pmatrix}, \qquad i = 1, \cdots, p-1.$$

By multiplying with T_k^{-1} and using $T_k y_{k,i} = -r_{k,i}$ again, we get

$$\Delta_k y_{k,i-1} = y_{k,i} + (y_{k,i-1})_1 y_k - T_k^{-1}\begin{pmatrix} 0 \\ c_i \end{pmatrix}, \qquad i = 1, \cdots, p-1.$$

To obtain the expression (3.9) for $y_{k,i}$, we just need to consider the rightmost term in the above equation. Now, using one step of the classical Levinson algorithm to go from T_{k-1} to T_k (cf. § 2), the solution to this system can simply be written as

$$T_k^{-1}\begin{pmatrix} 0 \\ c_i \end{pmatrix} = \frac{c_i}{\gamma^{(k-1)}}\begin{pmatrix} E_{k-1} y_{k-1} \\ 1 \end{pmatrix}.$$

Thus, we have proved (3.9). □

After the p-step, we have two choices for computing the $\gamma^{(k+p)}$ required in the next step. We can use an updating formula similar to (2.8), which becomes $\gamma^{(k+p)} = \gamma^{(k)} - c^T z_p^{(k)}$, or we can compute $\gamma^{(k+p)}$ by the definition (2.4), e.g., $\gamma^{(k+p)} = \rho_0 + r_{k+p}^T y_{k+p}$. We tried both in our numerical experiments, and the latter turned out to be slightly more accurate (although we do not have an explanation for this).

We can determine the computational effort for performing a p-step at the kth stage as follows. Computation of the extraneous $p-1$ vectors $y_{k,i}$ requires $2(p-1)k$ multiplications. Since $\gamma^{(k)}$ is already given by (2.8), setting up $\Gamma_p^{(k)}$ and the right-hand sides in (3.7) requires $(\frac{1}{2}p(p+1) - 1 + 2p)k = (\frac{1}{2}p^2 + (5/2)p - 1)k$ multiplications. The computational effort involved in an LU-factorization of the indefinite matrix $\Gamma_p^{(k)}$ is $O(p^3)$ multiplications. Thus, the effort in preparing the p-step is

$$\text{preparation effort} = (\tfrac{1}{2}p^2 + (9/2)p - 3)k + O(p^3) \text{ multiplications.} \tag{3.14}$$

Solving the two systems in (3.7) using the LU-factorization requires $2p^2$ multiplications, while the updating of x_{k+p} and y_{k+p} in (3.6) requires additional $2pk$ multiplications and the recomputation of $\gamma^{(k+p)}$ (for $p > 1$ only) requires $k + p$ multiplications. Thus, the effort in updating the solution is

$$\text{updating effort} = \begin{cases} 2k & \text{multiplications,} \quad p = 1, \\ (2p+1)k + O(p^2) & \text{multiplications,} \quad p > 1. \end{cases} \tag{3.15}$$

However, note that we save p classical Levinson steps which would have involved approximately $2pk$ multiplications. Hence, the computational *overhead* in the updating phase is only $k + O(p^2)$ multiplications for $p > 1$.

4. A brief error analysis. The purpose of this section is to show that our extended algorithm is numerically stable (in fact, "weakly stable" in the terminology of Bunch [3]). We shall base our discussion on the error analysis by Cybenko [5] and, like Cybenko, restrict our discussion to the Durbin algorithm for the Yule–Walker problem.

Let $\tilde{y}_k$ and $\tilde{y}$ denote the solutions to the Yule–Walker problems computed by the Durbin algorithm. Cybenko's analysis is carried out for positive-definite matrices only. His basic idea is to consider the residual for the kth order system $d_k \equiv T_k(y_k - \tilde{y}_k)$ and in this way derive a bound on the norm of the residual $T_n(y - \tilde{y})$. Most of Cybenko's analysis—including Lemmas 5.1, 5.2, and 5.3 in [5]—are still valid for general, indefinite Toeplitz matrices; only the relation $|\alpha^{(k)}| < 1$ does not hold in general.

Let us consider Cybenko's expression for d_{k+1}, which in our notation becomes

$$d_{k+1} = \begin{pmatrix} d_k + \alpha^{(k)} E_k d_k \\ -\varepsilon_1 \gamma^{(k)} - \varepsilon_2 \alpha^{(k)} \end{pmatrix} + T_{k+1} \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_k \\ 0 \end{pmatrix}. \tag{4.1}$$

Here, ε_1 and ε_2 represent all the accumulated rounding errors involved in computing $\alpha^{(k)}$ and $\gamma^{(k)}$, respectively, while the ξ_i represent the rounding errors produced when updating y_k. Following Cybenko's analysis step by step, it is not difficult to see that his expressions for r_k and s_k [5, p. 317] now (for general T_n) should be multiplied by $|\alpha^{(k)}|$. The upper bound for the 1-norm of d_k thus becomes

$$\|d_{k+1}\|_1 \leq \varepsilon_M \max_{0 \leq j \leq k} |\alpha^{(j)}| (\tfrac{1}{2}k^2 + 11k) \left(\prod_{j=0}^{k} (1 + |\alpha^{(j)}|) - 1 \right) + O(\varepsilon_M^2) \tag{4.2}$$

where ε_M is the machine precision. This result confirms that the key to the failure of the classical Durbin algorithm is the fact that one or several consecutive $|\gamma^{(j)}|$ can become arbitrarily small and hence the product $\prod_{j=0}^{k}(1 + |\alpha^{(j)}|)$ can become arbitrarily large, thus "blowing up" the rounding errors.

Our extended algorithm avoids large $\|d_k\|_1$ by the incorporation of the p-steps, guaranteeing that the norm of

$$\begin{pmatrix} E_k Y_p \\ I_p \end{pmatrix} z_p^{(k)}$$

in (3.6) is not large because p is chosen such that T_{k+p} is well conditioned. Consider again the residual corresponding to step k, immediately *after* taking a p-step and updating the solution y_k. Assuming for simplicity that this is the first p-step, we have

$$d_{k+p} = \begin{pmatrix} d_k + E_k [d_{k,0}, \cdots, d_{k-p,p-1}] z_p^{(k)} \\ -\Gamma_p^{(k)} \varepsilon_1 - E_2 z_p^{(k)} \end{pmatrix} + T_{k+p} \begin{pmatrix} \xi_1 \\ \vdots \\ \xi_k \\ 0 \end{pmatrix}. \tag{4.3}$$

The vector ε_1 represents the accumulated rounding errors involved in computing $z_p^{(k)}$, the matrix E_2 represents all the rounding errors in the elements of the matrix $\Gamma_p^{(k)}$, and the ξ_i represent the rounding errors involved in updating y_k. Of course, all these quantities are still bounded as before; but now we are working with p-vectors and p-by-p matrices. If we use the Frobenius norm of E_2 in our new bounds, we must multiply Cybenko's bounds by p^2 to obtain upper bounds for the perturbations in d_{k+p}. Thus, we have proved the following upper bound corresponding to (4.2).

THEOREM 2. *Immediately after the first p-step, the* 1*-norm of the residual* $d_{k+p} \equiv T_{k+p}(y_{k+p} - \tilde{y}_{k+p})$ *satisfies*

$$\begin{aligned} \|d_{k+p}\|_1 \leq \varepsilon_M \max_{0 \leq j \leq k-1} |\alpha^{(j)}| (\tfrac{1}{2}(k-1)^2 + 11(k-1)) \left(\prod_{j=0}^{k-1} (1 + |\alpha^{(j)}|) - 1 \right) \\ + \varepsilon_M p^2 (3k+5) \|z_p^{(k)}\|_1 \left(\prod_{j=0}^{k-1} (1 + |\alpha^{(j)}|) - 1 \right) + O(\varepsilon_M^2). \end{aligned} \tag{4.4}$$

We see that we can indeed guarantee that $\|d_{k+p}\|_1$ does not become large by controlling the size of $|\alpha^{(j)}|$ and $\|z_p^{(k)}\|_1$.

5. Practical details. In this section we discuss some of the practical details involved in an implementation of our new algorithm. First of all, it is important to decide *when* to perform a p-step and to chose the *correct* value of the block-size p. Our aim is to choose the block size adaptively in order to ensure the best possible accuracy in the computed solution while, at the same time, choosing $p > 1$ only when necessary, because such block steps are more expensive (as measured in multiplications) than p usual Levinson steps. Our approach is to let the user choose a small number $p_{\max}$ and then, in each step k, choose the optimal block-size p within the look-ahead range $p = 1, \cdots, p_{\max}$. To do this, let $\sigma_{\min}(T_{p+k})$ denote the smallest singular value of T_{k+p}, and let $s_{\min}$ denote the smallest $\sigma_{\min}(T_{k+p})$ accepted so far in all the previous steps. We then choose as block size the smallest p for which

$$\sigma_{\min}(T_{k+p}) \geqq 0.1 s_{\min}. \tag{5.1}$$

The reason for comparing with $s_{\min}$ is that there is no point in trying to use a larger pivot in this step than in any previous step as far as the condition of T_{k+p} is concerned, and the factor 0.1 is included in (5.1) to avoid choosing a too large p if $\sigma_{\min}(T_{k+p})$ is only slightly smaller than $s_{\min}$ such that T_{k+p} is still well conditioned. If (5.1) is not satisfied for any $p \leqq p_{\max}$, then we choose as block size the p for which $\sigma_{\min}(T_{k+p})$ is maximum, and only in this case do we update the $s_{\min}$. In this way, we are guaranteed in each step to choose the optimally conditioned T_{k+p} within the allowed look-ahead range $p_{\max}$, and the strategy is numerically stable as long as T_n does not have more than $p_{\max} - 1$ consecutive ill-conditioned leading submatrices.

An important side effect of using $s_{\min}$ in this fashion to keep track of the smallest accepted $\sigma_{\min}(T_{k+p})$ is that it provides us with an inexpensive estimate of the condition number for the algorithm. Since the accuracy of the computed solution depends on the smallest singular value of any T_{k+p} accepted during the extended Levinson algorithm, we define the *algorithm condition number* as

$$\kappa_{\text{Levinson}} \equiv \|T_n\|_2 / \min\{\text{accepted } \sigma_{\min}(T_{k+p})\} \approx \|T_n\|_2 / s_{\min}. \tag{5.2}$$

Thus, without increasing the complexity of the algorithm, we can return reliable estimates of the algorithm condition number κ_{Levinson}, as well as the usual condition number $\|T_n\|_2 / \sigma_{\min}(T_n)$.

In order to keep the computational overhead as small as possible, it is important to compute an estimate of $\sigma_{\min}(T_{k+p})$ efficiently. It is well known that the smallest singular value of the Schur complement $\Gamma_p^{(k)}$ does not give a reliable estimate of the condition of T_{k+p}, and our numerical experiments confirmed this: in some situations an ill-conditioned T_{k+p} was not reflected by a small $\sigma_{\min}(\Gamma_p^{(k)})$, and a necessary p-step was therefore not performed. However, we want our algorithm to be highly reliable.

As an alternative to using $\sigma_{\min}(\Gamma_p^{(k)})$, we will estimate $\sigma_{\min}(T_{k+p})$ directly through an estimate for $\|T_{k+p}^{-1}\|_2 = [\sigma_{\min}(T_{k+p})]^{-1}$. Consider therefore the following expression for T_{k+p}^{-1}, which is obtained by means of (3.2) and (3.4):

$$T_{k+p}^{-1} = \begin{bmatrix} T_k^{-1} + E_k Y_p (\Gamma_p^{(k)})^{-1} Y_p^T E_k & E_k Y_p (\Gamma_p^{(k)})^{-1} \\ (\Gamma_p^{(k)})^{-1} Y_p^T E_k & (\Gamma_p^{(k)})^{-1} \end{bmatrix}. \tag{5.3}$$

Since we are only concerned with identifying the ill-conditioned submatrices T_{k+p} in order to "leap over" them, the accuracy of the estimate of $\|T_{k+p}^{-1}\|_2 = [\sigma_{\min}(T_{k+p})]^{-1}$ is not so important. What we need is a reliable and efficient means for detecting when $\|T_{k+p}^{-1}\|_2$ becomes large. A lower bound is given by

$$\|T_{k+p}^{-1}\|_2 \geqq \max\{\|T_k^{-1} + E_k Y_p (\Gamma_p^{(k)})^{-1} Y_p^T E_k\|_2, \|E_k Y_p (\Gamma_p^{(k)})^{-1}\|_2, \|(\Gamma_p^{(k)})^{-1}\|_2\}. \tag{5.4}$$

There are no simple lower bounds for the matrix norms in (5.4), so instead we use the upper bounds

$$\|T_k^{-1} + E_k Y_p (\Gamma_p^{(k)})^{-1} Y_p^T E_k\|_2 \leqq \|T_k^{-1}\|_2 + \|Y_p\|_2^2 \|(\Gamma_p^{(k)})^{-1}\|_2,$$

$$\|E_k Y_p (\Gamma_p^{(k)})^{-1}\|_2 \leqq \|Y_p\|_2 \|(\Gamma_p^{(k)})^{-1}\|_2.$$

Next, we note that $\|Y_p\|_2$ always lies between 1 and $\|Y_p\|_2^2$, and we use the lower bound $\max\{|(Y_p)_{ij}|\} \leqq \|Y_p\|_2$. Finally, due to our block algorithm we know that T_k is well conditioned such that $\|T_k^{-1}\|_2$ is not large and therefore cannot contribute to a large $\|T_{k+p}^{-1}\|_2$, and we can therefore readily ignore $\|T_k^{-1}\|_2$. This combination of lower and upper bounds leads to the following heuristic estimate, which works well in practice:

$$\|T_{k+p}^{-1}\|_2 \approx \max\{\max\{|(Y_p)_{ij}|\}^2 / \sigma_{\min}(\Gamma_p^{(k)}), 1/\sigma_{\min}(\Gamma_p^{(k)})\}. \tag{5.5}$$

For $p = 1$, $\sigma_{\min}(\Gamma_p^{(k)})$ is simply $|\gamma^{(k)}|$, and for $p > 1$, $\sigma_{\min}(\Gamma_p^{(k)})$ can be estimated by any efficient condition number estimator in $O(p^2)$ multiplications. Determination of the numerically largest element in Y_p requires kp comparisons at the kth stage, i.e., $O(n^2)$ comparisons for the complete algorithm. We do not think it is possible to get good estimates of the smallest singular values with a lower complexity than this.

A nice consequence of combining this strategy for estimating $\|T_{k+p}^{-1}\|_2$ with the block-size decision, based on (4.1), is that a specific threshold for detection of ill-conditioned submatrices is *not* required. This is important both from a numerical and a practical (i.e., a user's) point of view.

At this point, it is interesting to note that for a *symmetric* Toeplitz matrix and in *exact arithmetic*, the number $2l - 1$ of consecutive *singular* leading submatrices is always an odd number, and the nullities of these submatrices are $1, 2, \cdots, l, \cdots, 2, 1$ [11, Thm. 15.6]. In the presence of rounding errors, we do not expect this to hold in general.

We stress that the extended algorithm will certainly not handle all indefinite Toeplitz matrices efficiently, simply because the necessary p could become so large that the overhead becomes too large. As an example of where our algorithm is not suited, take the matrix (for n even):

$$T_n = \begin{bmatrix} 0 & I_{n/2} \\ I_{n/2} & 0 \end{bmatrix}, \qquad I_{n/2} = \text{identity matrix of order } n/2, \tag{5.6}$$

where all leading submatrices are singular except for T_n. However, our algorithm still has its usefulness whenever the number of consecutive ill-conditioned leading principal submatrices is small. One such application is eigenfilter problems, where eigensolutions of indefinite matrices must be computed, and this usually requires the solution of Yule–Walker systems involving the shifted matrix $T_n - \lambda I_n$. The algorithm by Cybenko and Van Loan [6], as well as the more recent algorithm by Trench [19], precludes any singular or ill-conditioned leading principal submatrices. Our algorithm allows this assumption to be removed. In particular, if the eigenvalues are well separated, then Trench's method [19] requires $p_{\max} = 2$ only.

The second important detail in a practical implementation of our algorithm is the fact that two extended steps may follow immediately after each other. Let p' denote the size of the *previous* p'-step, and let k' denote the corresponding dimension of that leading submatrix $T_{k'}$, such that $k = k' + p'$. In such a situation, neither $\gamma^{(k-1)}$ nor y_{k-1} is available because we skipped the step $k - 1$ and the ill-conditioned matrix T_{k-1} when going from $T_{k'}$ to

$$T_k = \begin{pmatrix} T_{k'} & E_k R_{p'} \\ R_{p'}^T E_k & T_{p'} \end{pmatrix}.$$

Hence, we cannot use (3.9) to compute the vectors $y_{k,i}$ because we cannot produce the vector s_k in (3.10) (which is a function of $\gamma^{(k-1)}$ and y_{k-1}). However, in this special situation there is another way to proceed. The idea is to compute $y_{k,1}$ by updating the vector $y_{k',1}$ and, if $p > 2$, then compute an s_k such that (3.9) still gives the solution $y_{k,i}$.

THEOREM 3. *In case of two consecutive extended Levinson steps with $k = k' + p'$, the vector $y_{k,1}$ is given by*

$$y_{k,1} = \begin{pmatrix} y_{k',1} \\ 0 \end{pmatrix} + \begin{pmatrix} E_{k'} Y_{p'} \\ I_{p'} \end{pmatrix} a \tag{5.7}$$

where the vector $a \in \mathbb{R}^{p'}$ is the solution to the system

$$\Gamma_p^{(k')} a = -\begin{pmatrix} \rho_{k'+2} \\ \vdots \\ \rho_{k+1} \end{pmatrix} - R_{p'}^T E_{k'} y_{k',1}. \tag{5.8}$$

If $p > 2$, then the remaining vectors $y_{k,i}$, $i = 2, \cdots, p-1$ can be computed from (3.9) *with the vectors s_k given by*

$$s_k = c_1^{-1}(y_{k,1} - \Delta_k y_k + (y_k)_1 y_k) \tag{5.9}$$

in which c_1 is the first component of the vector c defined in (3.11).

Proof. We recall that the vector $y_{k,1}$ is the solution to the system $T_k y_{k,1} = -[\rho_2, \cdots, \rho_{k+1}]^T = -[r_{k',1}^T, \rho_{k'+2}, \cdots, \rho_{k+1}]^T$. In the previous p'-step, we already computed $y_{k',1}$ satisfying $T_{k'} y_{k',1} = -r_{k',1}$. Since the first k' components of these two right-hand sides are identical, it is immediately clear that we can use an extended Levinson step as described in § 3, equations (3.5)–(3.7), to obtain (5.7) with a given by (5.8). Equation (5.9) follows immediately from (3.9) with $i = 1$. □

Hence, all we need to do is to save the matrix $Y_{p'}$ and the factorization of $\Gamma_{p'}^{(k')}$ from the previous p'-step. Then the vectors $y_{k,i}$, $i = 1, \cdots, p-1$ can be computed recursively using (3.9) and Theorem 2. The effort in computing $y_{k,1}$ and s_k this way is only about $2(p'+1)k + O((p')^2)$ multiplications.

In the situation mentioned in the beginning of this section, where we have looked $p_{\max}$ steps ahead, but chosen a $p < p_{\max}$, we could in principle update the remaining $y_{k',i}$, $i = p+1, \cdots, p_{\max}$ to obtain some of the new vectors $y_{k,i}$. This is, however, not reliable in general because some of these remaining $y_{k',i}$ might correspond to ill-conditioned leading submatrices occurring after the current T_{k+p}. Thus, we prefer to recompute all the $y_{k,i}$ by means of Theorem 3.

The third detail, which is but a minor one, is that the very first leading submatrices $T_1, T_2, \cdots, T_p$ of T_n may be ill conditioned. If this is the case, we cannot perform a p-step, simply because we have no starting vectors to update. Instead, we must compute the solutions x_p and y_p by a standard linear-equation solver, such as LU-factorization with pivoting, ignoring the Toeplitz structure.

The complete version of the extended Levinson algorithm, including condition numbers estimates, is given in Fig. 1.

We conclude this section by giving upper and lower bounds for the computational effort involved in the new algorithm. In the *worst case* (which is very unlikely to occur), for each k we will look $p_{\max}$ steps ahead and choose $p = 1$. According to (3.14) and

init.: let T_k = best-conditioned T_i, $i = 1 : p_{\max}$
solve $T_k \mathbf{x}_k = \mathbf{b}_k$ and $T_k Y_k = -R_k$, e.g., by LU-factorization
the first column of Y_k is $\mathbf{y}_k$
$\gamma^{(k)} = \rho_o + \mathbf{r}_k^T \mathbf{y}_k$
$s_{\min} = \sigma_{\min}(T_k)$
loop: for $i = k : n-1$
for $p = 1 : \min(p_{\max}, n-k)$
if last step was a block-step
set up Y_p using Theorem 3
else
set up Y_p using Theorem 1
end
set up the systems in (3.7)
estimate $\sigma_{\min}(T_{k+p})$ by (5.5)
if $\sigma_{\min}(T_{k+p}) > 0.1\, s_{\min}$ goto *update*
end
let T_{k+p} = best-conditioned T_{k+i}, $i = 1 : \min(p_{\max}, n-k)$
if $\sigma_{\min}(T_{k+p}) < s_{\min}$ then $s_{\min} = \sigma_{\min}(T_{k+p})$
update: solve (3.7) for updates $\mathbf{w}_p^{(k)}$ and $\mathbf{z}_p^{(k)}$ e.g. by LU-factorization
update $\mathbf{x}_{k+p}$ and $\mathbf{y}_{k+p}$ by (3.6)
update $\gamma^{(k+p)}$
end
$\kappa_{\text{Levinson}} = \|T_n\|_2 / s_{\min}$
$\kappa(T_n) = \|T_n\|_2 / \sigma_{\min}(T_n)$

FIG. 1. *The look-ahead Levinson algorithm.*

(3.15), and taking into account the overhead in estimating $\sigma_{\min}(T_{k+p})$ as well as in Theorem 3, each step thus requires about

$$\sum_{p=1}^{p_{\max}} \left(\frac{1}{2}p^2 + \frac{9}{2}p - 3\right)k + (2p_{\max}+1)k + 2(p_{\max}+1)k$$

$$= \left(\frac{p_{\max}^3}{6} + \frac{5}{2}p_{\max}^2 + \frac{10}{3}p_{\max} + 3\right)k \text{ multiplications.}$$

Summing over all k, we then obtain the following approximate expression for the very pessimistic upper bound

$$\text{maximum effort} = \frac{1}{2}\left(\frac{p_{\max}^3}{6} + \frac{5}{2}p_{\max}^2 + \frac{10}{3}p_{\max} + 3\right)n^2 \text{ multiplications.} \tag{5.10}$$

For $p_{\max}$ equal to 2, 3, and 4, this amounts to $10n^2$, $20n^2$, and $33n^2$ multiplications, respectively. As we shall see in the next section, we never obtain this upper bound in practice. The minimum computational effort corresponds to matrices where all the $\sigma_{\min}(T_{k+1})$ form an increasing sequence, such that only 1-steps are performed, and we only look one step ahead in each stage. For such matrices, Theorem 3 is never used, and (3.14) and (3.15) lead to $4k$ multiplications in each stage, thus giving a total computational effort of

$$\text{minimum effort} = 2n^2 \text{ multiplications.} \tag{5.11}$$

Note that this is identical to the complexity of the classical Levinson algorithm, i.e., for such matrices there is no overhead. Also note that this is *not* the case for well-conditioned symmetric positive-definite Toeplitz matrices T_n. For such matrices, it can be shown that the $\sigma_{\min}(T_{k+1})$ form a slowly but strictly decreasing sequence. Hence, due to our strategy for choosing the block-size p, our algorithm will always take classical 1-steps only. Also, it will usually only look one step ahead until it reaches a stage k where the estimate of $\sigma_{\min}(T_{k+1})$ has dropped so much that (5.1) is not satisfied any more. Then the algorithm will look $p_{\max}$ steps ahead, choose $p = 1$, update $s_{\min}$, and continue the process. Depending on the order n and the condition number of T_n, this may occur more than once; but only very rarely as long as T_n is well conditioned. Thus, the overhead compared to the classical Levinson algorithm is only very small for well-conditioned symmetric positive-definite Toeplitz matrices, and in addition we obtain the condition number estimates.

6. Numerical results. In this section we present some numerical results from applying our extended Levinson algorithm to a series of test problems. Our tests were performed with symmetric test matrices with at least one leading submatrix being ill conditioned. It is difficult to generate large test problems with an arbitrary number of ill-conditioned leading submatrices, so we concentrated our testing effort on test matrices with one ill-conditioned leading submatrix. Such test matrices are easily generated: generate a random symmetric Toeplitz matrix $\bar{T}_n$, compute the largest eigenvalue λ of its order-k leading submatrix $\bar{T}_k$ for some k, and set

$$T_n = \bar{T}_n - (\lambda - \delta) I_n, \tag{6.1}$$

where $\delta > 0$ is a real parameter, and I_n is the identity matrix of order n. Thus, we are guaranteed that the computed T_n has a leading submatrix T_k with $\sigma_{\min}(T_k) \approx \delta$, and in this way we can control the ill-conditioning of T_k by means of δ. Shifted matrices of this form are common in inverse iterations.

We implemented the extended Levinson algorithm in Matlab [16] and restricted the maximal block size to 4. The tests were carried out on a computer with machine precision $\varepsilon_M = 2.22 \cdot 10^{-16}$ and with the following values of δ and $p_{\max}$:

$$\delta = 0, \quad 10^3 \varepsilon_M, \quad 10^6 \varepsilon_M, \quad 10^9 \varepsilon_M, \quad 1, \tag{6.2a}$$

$$p_{\max} = 1 \text{ (classical Levinson)}, \quad 2, \quad 3, \quad 4. \tag{6.2b}$$

For each δ we generated 300 test matrices: 100 of order $n = 16$, 100 of order $n = 32$, and 100 of order $n = 64$. In all cases, we chose the k for which T_k is guaranteed to be ill conditioned to $k = n/2$, and we chose the right-hand side b such that the exact solution is $x = [1, \cdots, 1]^T$.

Figure 2 illustrates the behavior of our algorithm for a typical test example with $n = 64$, $\delta = 10^6 \varepsilon_M$, $p_{\max} = 4$, and $\|T_n\|_2 = 28.4$. The top figure shows how many steps we look ahead in each stage k (typically one step); the middle figure shows the chosen value of the block-size p in each stage k (typically one); and the bottom figure shows the accepted estimates of $\sigma_{\min}(T_{k+p})$ in each stage k. Note that one 2-step was taken in the beginning of the process, when $\sigma_{\min}(T_{k+p})$ was gradually decreasing, and that one 4-step was needed for $k = 29$ in order to skip the severely ill conditioned submatrix T_{32}.

Figure 3, which supplements Fig. 2, shows for the same test matrix a plot of all the true $\sigma_{\min}(T_k)$ (solid line) as well as the estimates computed by (5.5) (circles). This figure illustrates that our rough estimates actually follow the same variation as the $\sigma_{\min}(T_k)$, such that the choice of block-size p can, indeed, be based on these estimates. In Fig. 3 we have also included, by dotted lines, the 2-step and the 4-step mentioned above. Note

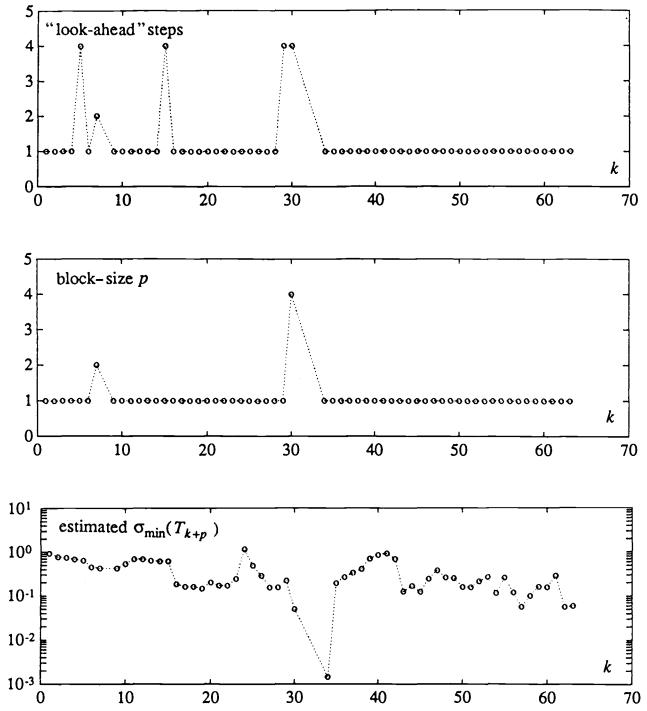


FIG. 2. *A typical example of the behavior of the number of "look-ahead" steps (top), the block-size p (middle), and the accepted estimates of* $\sigma_{\min}(T_k)$ *(bottom).*

that the "canyon" of the $\sigma_{\min}(T_k)$-curve is fairly wide around $k = 31$. This is actually an example of a matrix for which $p_{\max} = 2$ would be bad (the algorithm would jump from $k = 30$ to $k + p = 32$), $p_{\max} = 4$ is better, and $p_{\max} > 4$ would be even better.

To monitor the accuracy of the *computed* solutions $\tilde{x}$ we computed the relative errors

$$\rho = \frac{\|\tilde{x} - x\|_2}{\|x\|_2}. \tag{6.3}$$

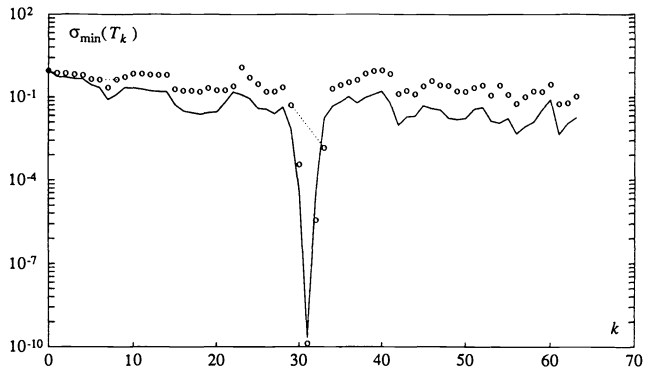


FIG. 3. *A typical example of all the true* $\sigma_{\min}(T_k)$ *(solid line) and the estimated values (circles). The 2-step and the 4-step are shown by dotted lines (same matrix as in Fig. 2).*

Figure 4 shows histograms of ρ for four tests with $n = 64$, $\delta = 10^3\,\varepsilon_M$, and $p_{\max}$ equal to 1, 2, 3, and 4. For $p_{\max} = 1$ (classical Levinson), ρ is never smaller than 10^{-5} due to the ill-conditioned submatrix T_{32}. On the other hand, for $p_{\max} = 4$ the relative error ρ is never larger than 10^{-9}, and typically it is even smaller, about 10^{-12}. The largest relative errors ρ for $p_{\max} = 2$ and 3 correspond to test matrices that actually required a block-size $p > p_{\max}$ in order to "leap over" a sequence of ill-conditioned leading submatrices. We have never encountered a situation where our algorithm missed an ill-conditioned submatrix.

For each combination of δ and $p_{\max}$, we computed the maximum relative error $\rho_{\max} = \max\{\rho\}$ over all 300 test matrices. Figure 5 shows a plot of $\rho_{\max}$ as a function of δ and $p_{\max}$. For $p_{\max} = 1$ (classical Levinson), the maximum relative error $\rho_{\max}$ is approximately proportional to δ^{-1}, as expected, while $\rho_{\max}$ is almost independent of δ for $p_{\max} > 1$. We see that the extended Levinson algorithm is *always* doing better than the classical algorithm, even in the case $\delta = 1$ (where all the leading submatrices are well conditioned). For $\delta < 1$, we are actually doing much better than the classical Levinson algorithm, with a maximum relative error of about 10^{-8} almost independent of δ.

In Fig. 6 we show the average number of p-steps required in our experiments. For $p_{\max}$ equal to 2 or 3, the average number of 2-steps is about 2, and for $p_{\max} = 4$ it is about

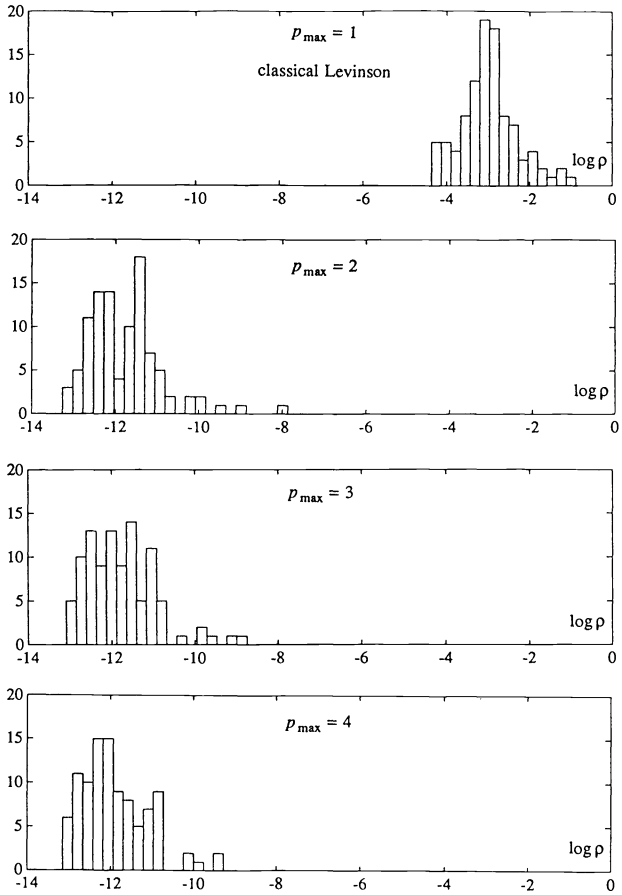


FIG. 4. *Histograms of the relative error* ρ *for* 100 *test matrices with* $n = 64$ *and* $\delta = 10^3\varepsilon_M$, *using* $p_{\max} = 1, 2, 3,$ *and* 4.

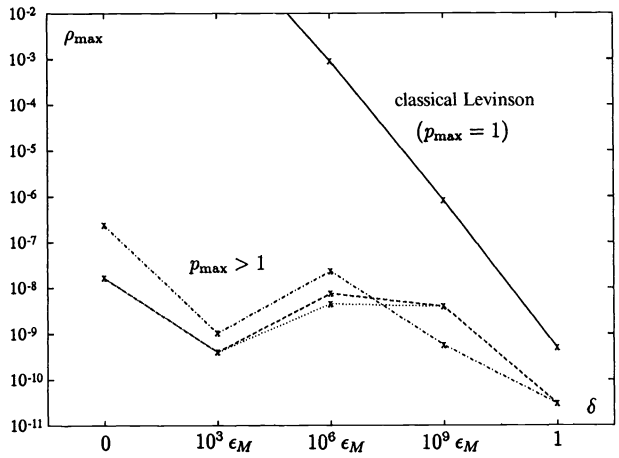


FIG. 5. *The maximum relative error of the computed solution.*

1.5. This is the "price" we have to pay for ensuring that we can handle these indefinite Toeplitz matrices. The average number of 3-steps and 4-steps are much smaller, about 0.5.

Finally, in Table 1 we compare the number of multiplications required in our new algorithm with those of the classical Levinson algorithm, still using the above-mentioned test matrices with one ill-conditioned leading submatrix. Each entry in Table 1 shows, for each combination of n and $p_{\max}$, the average number of multiplications over all 300 test matrices (left) and the maximum number of multiplications (right) over the same 300 test matrices. We do not show the dependency on δ because, due to the uniform way in which we generate the test matrices, the average multiplication count for $p_{\max} > 1$ is almost independent of δ—although somewhat smaller for $\delta = 1$ than for $\delta < 1$—and completely independent of δ for $p_{\max} = 1$ (classical Levinson). We note that the multiplication count for our new method with $p_{\max} \leqq 4$ is never more than three times the multiplication count for the classical Levinson algorithm, the average ratio, in fact,

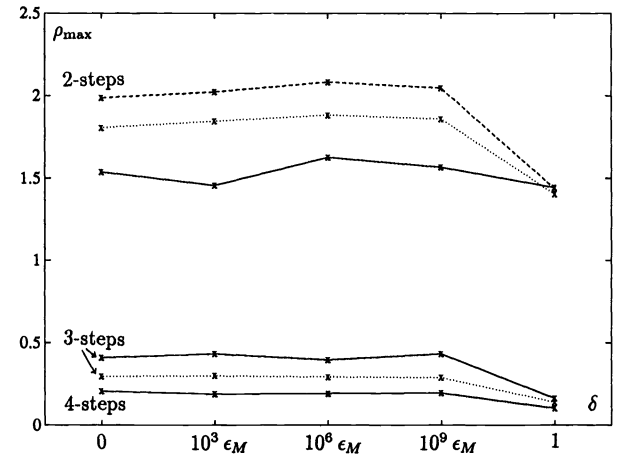


FIG. 6. *The average number of p-steps for* $p_{\max} = 2$ (*dashed line*), $p_{\max} = 3$ (*dotted line*), *and* $p_{\max} = 4$ (*solid line*).

TABLE 1
The average (left) and maximum (right) number of multiplications.

n	$p_{\max}$ = 1		2		3		4	
16	510	510	634	774	689	876	744	1057
32	2046	2046	2322	2573	2451	2912	2577	3268
64	8190	8190	8795	9502	9061	10012	9369	10710

being only 1.25. Also, note that the average ratio decreases as n increases, from 1.4 (for $n = 16$) to 1.1 (for $n = 64$). Hence, for this particular class of matrices the complexity of our new algorithm, including condition estimation, is about 25 percent larger than that of the classical Levinson algorithm.

Following Trench [19], we also tried our algorithm on the KMS matrices given by

$$\rho_0 = 0, \qquad \rho_i = (\tfrac{1}{2})^{i-1}, \qquad i = 1, \cdots, n-1, \tag{6.4}$$

for which all the leading submatrices T_k, $k = 1, 4, 7, \cdots$ are exactly singular. In order to compare our algorithm with the classical Levinson algorithm, which breaks down whenever $\rho_0 = 0$, we modified the matrix slightly and put $\rho_0 = 10^{-14}$. The results, given in Table 2, show that our algorithm computes an accurate solution with an overhead of about 50 percent, including condition estimation.

As a final example, we applied the new algorithm to a highly ill conditioned test matrix with elements given by $\rho_i = 2^i$, $i = 0, \cdots, n-1$, and $n = 64$ (Matlab produced $\|T_n\|_2 = 1.23 \cdot 10^{19}$ and $\sigma_{\min}(T_n) = 0$). For this matrix, the $\gamma^{(k)}$ form an increasing sequence. Only classical 1-steps were performed, and the algorithm never looked more than 1 step ahead, so this is an example of a matrix for which the overhead is minimum. The ill-conditioning was detected by our algorithm, which returned a condition number estimate equal to $1.84 \cdot 10^{19}$. In comparison, the classical Levinson algorithm applied to this matrix never breaks down and simply produces an extremely bad solution without any warning to the user.

7. Conclusion. We have presented an extension of the classical Levinson algorithm based on the ability to take block steps instead of classical steps of size one. Combining this feature with an inexpensive and reliable estimate of the condition of all the leading submatrices encountered during the process, the algorithm in each step chooses the optimal block size and thus is able to skip consecutive ill-conditioned submatrices. This condition estimate also yields reliable estimates of both the matrix and the algorithm condition numbers.

TABLE 2
Numerical results for the KMS *test-matrices given by* (6.4).

n	$p_{\max} = 1$		$p_{\max} > 1$	
	ρ	Multiplications	ρ	Multiplications
15	$1.0 \cdot 10^{-3}$	448	$9.3 \cdot 10^{-13}$	698
30	$7.5 \cdot 10^{-4}$	1798	$1.3 \cdot 10^{-12}$	2678
60	$7.2 \cdot 10^{-4}$	7198	$1.3 \cdot 10^{-12}$	10463
120	$7.0 \cdot 10^{-4}$	28798	$1.3 \cdot 10^{-12}$	41333

The only limitation of the algorithm is the maximum block-size $p_{\max}$, which determines the maximum number of consecutive ill-conditioned submatrices the algorithm is able to skip over—at the cost of a computational overhead which increases slightly with $p_{\max}$, compared to the classical Levinson algorithm. As a good choice of this maximum block size, we suggest $p_{\max} = 2$ for a fast extended algorithm which is numerically stable for a much larger class of indefinite Toeplitz matrices than the classical Levinson algorithm. In the rare cases where a larger $p_{\max}$ is needed, a comparison of the estimated algorithm and matrix condition numbers will reveal this.

Our algorithm is particularly suited for problems which do not have many consecutive ill-conditioned leading submatrices. One important application of our algorithm is in fast algorithms for computing eigenvalues of Toeplitz matrices.

REFERENCES

[1] O. B. ARUSHANIAN, M. K. SAMARIN, V. V. VOEVODIN, E. E. TYRTYSHNIKOV, B. S. GARBOW, J. M. BOYLE, W. R. COWELL, AND K. W. DRITZ, *The Toeplitz package users' guide*, Report ANL-83-16, Argonne National Laboratory, Argonne, IL, 1983.

[2] J. R. BUNCH, *Stability of methods for solving Toeplitz systems of equations*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 349–364.

[3] ———, *The weak and strong stability of algorithms in numerical linear algebra*, Linear Algebra Appl., 88/89 (1987), pp. 49–66.

[4] T. F. CHAN AND P. C. HANSEN, *A look-ahead Levinson algorithm for general Toeplitz systems*, CAM Report 90-11, Department of Mathematics, University of California, Los Angeles, CA, May 1990; IEEE Trans. Acoust. Speech Signal Process., to appear.

[5] G. CYBENKO, *The numerical stability of the Levinson–Durbin algorithm for Toeplitz systems of equations*, SIAM J. Sci. Statist. Comput., 1 (1980), pp. 303–319.

[6] G. CYBENKO AND C. F. VAN LOAN, *Computing the minimum eigenvalue of a symmetric positive definite Toeplitz matrix*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 123–131.

[7] P. DELSARTE, Y. V. GENIN, AND Y. G. KAMP, *A generalization of the Levinson algorithm for Hermitian Toeplitz matrices with any rank profile*, IEEE Trans. Acoust. Speech Signal Process., 33 (1985), pp. 964–971.

[8] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[9] M. J. C. GOVER AND S. BARNETT, *Inversion of Toeplitz matrices which are not strongly non-singular*, IMA J. Numer. Anal., 5 (1985), pp. 101–110.

[10] P. C. HANSEN AND T. F. CHAN, *Fortran subroutines for general Toeplitz systems*, CAM Report 90-21, Department of Mathematics, University of California, Los Angeles, CA, 1990; ACM Trans. Math. Software, submitted.

[11] I. S. IOHVIDOV, *Hankel and Toeplitz Matrices and Forms*, Birkhäuser, Boston, 1982.

[12] T. KAILATH, *A view of three decades of linear filtering theory*, IEEE Trans. Inform. Theory, 20 (1974), pp. 145–181.

[13] T. KAILATH, A. VIEIRA, AND M. MORF, *Inverses of Toeplitz operators, innovations, and orthogonal polynomials*, SIAM Rev., 20 (1978), pp. 106–119.

[14] N. LEVINSON, *The Wiener rms (root mean square) error criterion in filter design and prediction*, J. Math. Phys., 25 (1946), pp. 261–278.

[15] F. T. LUK AND S. QIAO, *A fast but unstable orthogonal triangularization technique for Toeplitz matrices*, Linear Algebra Appl., 88/89 (1987), pp. 495–506.

[16] C. B. MOLER, J. N. LITTLE, AND S. BANGERT, *PC-Matlab User's Guide*, The MathWorks, Sherborn, MA, 1987.

[17] D. R. SWEET, *Fast Toeplitz orthogonalization*, Numer. Math., 43 (1984), pp. 1–21.

[18] ———, *The use of pivoting to improve the numerical performance of Toeplitz solvers*, in Advanced Algorithms and Architectures for Signal Processing, J. M. Speiser, ed., in Proc. Society of Photo-Optical Instrumentation Engineers 696, 1986, pp. 8–18.

[19] W. F. TRENCH, *Numerical solution of the eigenvalue problem for Hermitian Toeplitz matrices*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 135–146.

[20] E. E. TYRTYSHNIKOV, *New cost-efficient and fast algorithms for special classes of Toeplitz systems*, Soviet J. Numer. Anal. Math. Modelling, 3 (1988), pp. 63–76.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 507–529, April 1992

006

AN EIGENVALUE REGION FOR LESLIE MATRICES*

STEVE KIRKLAND†

Abstract. Leslie matrices arise in a mathematical model for population growth and the eigenvalues of a Leslie matrix are important in describing the limiting behaviour of the corresponding population model. While it is well known that such matrices have exactly one positive eigenvalue, which dominates the others in modulus, little has been said about the nonpositive eigenvalues of a Leslie matrix, which are also of interest. In this paper, a brief description of the Leslie model is given, and a sharp containment region for the nonpositive spectrum of a Leslie matrix is constructed, characterizing the region in terms of some simple equations.

Key words. Leslie model, nonpositive eigenvalue, row-stochastic Leslie matrix

AMS(MOS) subject classifications. 15A18, 15A48, 92A15

1. Introduction. There are many models which attempt to describe the growth of populations; one of the best known is the Leslie model, which first appeared in the early 1940s [6], [5]. In this model we take a population that is closed to migration and consider only one sex—usually females. Each female may live up to a maximum of k time units, and the interval $[0, k)$ is partitioned into $\{[i-1, i)\}i = 1, \cdots, k$; we say that a female is in the ith age group if her exact age falls in $[i-1, i)$. Letting p_i, $i = 1, \cdots, k-1$ be the (positive) probability that a female in the ith age group survives one time unit, and letting m_i, $i = 1, \cdots, k$ be the (nonnegative) expected number of surviving daughters that a female in the ith age group contributes in one time unit, we arrive at the following equation, where $f_{i,t}$, $i = 1, \cdots$, $t = 1, 2, \cdots$ is the expected number of females in the ith age group after t time units have passed:

$$\begin{bmatrix} f_{1,t+1} \\ f_{2,t+1} \\ \vdots \\ f_{k,t+1} \end{bmatrix} = \begin{bmatrix} m_1 & m_2 & \cdots & m_{k-1} & m_k \\ p_1 & & & & \\ & p_2 & & & \\ & & \ddots & & \\ & & & p_{k-1} & 0 \end{bmatrix} \begin{bmatrix} f_{1,t} \\ f_{2,t} \\ \vdots \\ f_{k-1,t} \\ f_{k,t} \end{bmatrix}.$$

If the age-dependent mortality and fertility has remained constant since time 0 we see that $\mathbf{f}_t = \mathbf{M}^t\mathbf{f}_0$, where $\mathbf{f}_s$, $s = 0, 1, \cdots$ is the k-vector whose entries are the $f_{i,s}$, $i = 1, \cdots, k$, and $\mathbf{M}$ is the matrix above.

This last equation indicates that if we are interested in the long-term effects of this age-dependent schedule of mortality and fertility, we need to examine the eigenvalues of $\mathbf{M}$, which is commonly called a Leslie matrix. If $m_k > 0$ then the matrix is nonsingular, but if there is some n such that $m_n > 0$ and $m_{n+1} = m_{n+2} = \cdots = m_k = 0$, then $\mathbf{M}$ may be partitioned as

$$\left[\begin{array}{c|c} \mathbf{L} & \mathbf{0} \\ \hline \mathbf{P} & \mathbf{N} \end{array}\right],$$

where

$$\mathbf{L} = \begin{bmatrix} m_1 & \cdots & m_{n-1} & m_n \\ p_1 & & & \\ & \ddots & & \\ & & p_{n-1} & 0 \end{bmatrix},$$

* Received by the editors December 11, 1989; accepted for publication (in revised form) August 16, 1990.
† Department of Mathematics and Statistics, Queen's University, Kingston, Ontario, Canada (mathematics.department@queensu.ca).

$\mathbf{0}$ is an $n \times k - n$ matrix of zeros, $\mathbf{P}$ is $k - n \times n$ with its only nonzero entry being p_n in the upper right corner, and $\mathbf{N}$ is $k - n \times k - n$ with all entries zero except for the subdiagonal, which consists of $p_{n+1}, \cdots, p_{k-1}$. This partition reveals that the eigenvalue 0 has multiplicity $k - n$, and that the only nonzero eigenvalues are those of the matrix $\mathbf{L}$. In either case, the problem is reduced to studying the spectrum of $\mathbf{L}$, where we take $n = k$ if $m_k > 0$.

It is well known to mathematical demographers (amongst others) that $\mathbf{L}$ has exactly one strictly positive eigenvalue, say ρ, which has algebraic multiplicity one, and that $\rho \geqq \lambda$ for all $\lambda \in \sigma(\mathbf{L})$. Further, if $\gcd\{i \mid m_i > 0\} = 1$, then $\rho > |\lambda|$ for all $\lambda \in \sigma(\mathbf{L}) - \{\rho\}$, while if $\gcd\{i \mid m_i > 0\} = d \geqq 2$, then $\mathbf{L}$ has exactly d eigenvalues of modulus ρ, namely $\rho e^{2\pi i a/d}$, $a = 0, \cdots, d - 1$, and all other eigenvalues of $\mathbf{L}$ have modulus strictly less than ρ. The reader seeking some discussion of these results is referred to [4], or to the exposition of Pollard [7], which uses the characteristic equation of $\mathbf{L}$ to derive them, or to a general reference on nonnegative matrix theory (Seneta [9], for example).

If ρ strictly dominates the other eigenvalues of $\mathbf{L}$ in modulus (which is often the case in demographic applications), we see from the Jordan form for $\mathbf{L}$ that

$$\mathbf{L}^t \sim \frac{\rho^t \mathbf{w}\mathbf{v}'}{\mathbf{v}'\mathbf{w}} + O(t^{n-1}|\lambda_1^t|) \quad \text{as } t \to \infty,$$

where $\mathbf{w}$ and $\mathbf{v}'$ are the right and left eigenvectors corresponding to ρ, respectively, and λ_1 is an eigenvalue of $\mathbf{L}$ with next largest modulus after ρ. Elementary calculations show that $\mathbf{w}$ may be written as

$$\begin{bmatrix} 1 \\ \left(\frac{p_1}{\rho}\right) \\ \left(\frac{p_1 p_2}{\rho^2}\right) \\ \vdots \\ \left(\frac{p_1 \cdots p_{n-1}}{\rho^{n-1}}\right) \end{bmatrix},$$

while $\mathbf{v}'$ may be written as a vector with strictly positive entries. Since $\mathbf{f}_0$ has nonnegative entries (and is assumed to have at least one positive entry), we see that $\mathbf{v}'\mathbf{f}_0 > 0$. In particular, we have

$$\mathbf{f}_t \sim \frac{\rho^t \mathbf{w}\mathbf{v}'}{\mathbf{v}'\mathbf{w}}\mathbf{f}_0 + O(t^{n-1}|\lambda_1^t|) = (\text{constant})\rho^t \mathbf{w} + O(t^{n-1}|\lambda_1^t|) \quad \text{as } t \to \infty,$$

where the constant is strictly positive. As time goes on, our population forgets its initial distribution and heads towards a stable age distribution that is proportional to $\mathbf{w}$. Note that the kind of convergence of our population to the stable age distribution depends on λ/ρ, where $\lambda \in \sigma(\mathbf{L}) - \{\rho\}$.

On the other hand, if $\mathbf{L}$ has d eigenvalues of modulus ρ for some $d \geqq 2$, then the behaviour of the age distributions is somewhat different. If $d = n$, the age distributions are cyclic with period n in general, while if $d < n$, the age distributions are asymptotically cyclic with period d in general. This cycling arises from the fact that the collection of λ/ρ, $\lambda \in \sigma(\mathbf{L})$ consists of the dth roots of unity and, if $d < n$, $n - d$ complex numbers whose moduli are strictly less than 1. As in the noncyclic case, the behaviour of the age distributions in the Leslie model depends on these λ/ρ.

In order to study these λ/ρ, consider the similarity transformation $\mathbf{S}^{-1}\mathbf{L}\mathbf{S}$, where $\mathbf{S} = \mathbf{diag}\{\mathbf{w}\}$. It is not difficult to see that

$$\mathbf{S}^{-1}\mathbf{L}\mathbf{S} = \rho \begin{bmatrix} \frac{m_1}{\rho} & \frac{p_1 m_2}{\rho^2} & \cdots & \frac{p_1 \cdots p_{n-2} m_{n-1}}{\rho^{n-1}} & \frac{p_1 \cdots p_{n-1} m_n}{\rho^n} \\ 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & 0 \end{bmatrix} = \rho \bar{\mathbf{L}},$$

so that the eigenvalues of $\bar{\mathbf{L}}$ are just λ/ρ, $\lambda \in \sigma(\mathbf{L})$, precisely the quantities in which we are interested. Letting $\mathbf{1}$ be the column vector of size n with each entry equal to unity, we see that

$$\bar{\mathbf{L}}\mathbf{1} = \frac{1}{\rho}\mathbf{S}^{-1}\mathbf{L}\mathbf{S}\mathbf{1} = \frac{1}{\rho}\mathbf{S}^{-1}\mathbf{L}\mathbf{w} = \mathbf{S}^{-1}\mathbf{w} = \mathbf{1},$$

and hence that $\bar{\mathbf{L}}$ is a row-stochastic matrix. For obvious reasons, we will refer to $\bar{\mathbf{L}}$ as a row-stochastic Leslie matrix; this is not to be confused with the term "stochastic Leslie matrix," which denotes a Leslie matrix whose entries are random. In attempting to say something about the nonpositive eigenvalues of $\bar{\mathbf{L}}$, we may look to $M_n = \{\lambda \in \mathbb{C} \mid \lambda \in \sigma(\mathbf{A}), \mathbf{A}\ n \times n \text{ and row-stochastic}\}$, for certainly any eigenvalue of $\bar{\mathbf{L}}$ will lie in M_n. The problem of characterizing M_n was partially solved by Dmitriev and Dynkin [1], who found M_n for $n = 2, \cdots, 5$ and was fully solved by Karpelevich [3], who characterized M_n for all $n \geqq 2$. However, $\bar{\mathbf{L}}$ has a very special structure which could perhaps be used to obtain a sharper containment region for the nonpositive eigenvalues of $\bar{\mathbf{L}}$. Further, by investigating the region analogous to M_n for row-stochastic Leslie matrices, we will get information about which values of λ/ρ are admitted by Leslie matrices, and hence about the kinds of convergence to the stable age distribution which are admitted by the Leslie model. What follows is the construction and characterization of just such a region.

2. Basic properties of the region. In light of our interest in the nonpositive eigenvalues of an $n \times n$ row-stochastic Leslie matrix, we make the following definition.

DEFINITION 1. For each $n \geqq 2$, let $L_n = \{\lambda \in \mathbb{C} - \{1\} \mid \lambda \in \sigma(\mathbf{A})$ for some $n \times n$ row-stochastic Leslie matrix $\mathbf{A}\}$.

Noting that an $n \times n$ row-stochastic Leslie matrix is determined by its top row, which is nonnegative and has unit sum, we make another definition.

DEFINITION 2. Let

$$K_n = \left\{(a_1, a_2, \cdots, a_n) \in \mathbb{R}^n \mid a_i \geqq 0, i = 1, \cdots, n, \sum_{i=1}^{n} a_i = 1\right\}.$$

It is useful to know which entries of a vector in K_n are positive and which are zero; this suggests the next definition.

DEFINITION 3. Given $a = (a_1, \cdots, a_n) \in K_n$, let $\text{supp}(a) = \{i \mid a_i > 0\}$.

So given a row-stochastic Leslie matrix $\mathbf{A}$, we have $\mathbf{A}_{1,i} = a_i$, $i = 1, \cdots, n$ for some $(a_1, \cdots, a_n) \in K_n$, and it is easily seen that the characteristic equation for the eigenvalues of $\mathbf{A}$ is $\lambda^n - a_1\lambda^{n-1} - a_2\lambda^{n-2} - \cdots - a_n = 0$. Dividing out a factor of $\lambda - 1$ from the equation, we see that $\lambda \in L_n$ if and only if there exists $(a_1, \cdots, a_n) \in K_n$ such that $\lambda^{n-1} + (1 - a_1)\lambda^{n-2} + (1 - a_1 - a_2)\lambda^{n-3} + \cdots + (1 - a_1 - a_2 - \cdots - a_{n-1}) = 0$. From this formulation, we find that $L_2 = [-1, 0]$ and that $L_n \subseteq L_{n+1}$ for all $n \geqq 2$. Note

that $L_n \cap \mathbb{R} = L_2$, because L_n is contained in the unit disc and has no positive elements. Since L_2 is known, we shall take $n \geqq 3$ in what follows, unless otherwise indicated.

It is evident that L_n is both closed and symmetric with respect to the real axis; the latter observation prompts our next definition.

DEFINITION 4. Let $L_n^+ = \{\lambda \in L_n | \operatorname{Im}(\lambda) \geqq 0\}$.

For the sake of brevity, we will often only deal with L_n^+ since results for the lower half of L_n will follow immediately. We are now ready for our first result about the nature of L_n.

THEOREM 1. *L_n is star-shaped with respect to the origin for all $n \geqq 3$.*

Proof. $0 \in L_2 \subseteq L_n$, so we need only show that if $\lambda \in L_n$, then $\rho\lambda \in L_n$ for all $\rho \in (0, 1)$. Fix such a ρ. Since $\lambda \in L_n$ there exists $(a_1, \cdots, a_n) \in K_n$ such that $\lambda^{n-1} + (1 - a_1)\lambda^{n-2} + \cdots + (1 - a_1 - \cdots - a_{n-1}) = 0$. Multiplication of this equation by ρ^{n-1} yields $(\rho\lambda)^{n-1} + \rho(1 - a_1)(\rho\lambda)^{n-2} + \rho^2(1 - a_1 - a_2)(\rho\lambda)^{n-3} + \cdots + \rho^{n-1}(1 - a_1 - \cdots - a_{n-1}) = 0$. We now define $b_1 = 1 - \rho(1 - a_1)$, $b_{i+1} = \rho^i(1 - a_1 - \cdots - a_i) - \rho^{i+1}(1 - a_1 - \cdots - a_{i+1})$, $i = 1, \cdots, n-2$, and $b_n = \rho^{n-1}(1 - a_1 - \cdots - a_{n-1})$. Then $(b_1, \cdots, b_n) \in K_n$ and $(\rho\lambda)^{n-1} + (1 - b_1)(\rho\lambda)^{n-2} + \cdots + (1 - b_1 - \cdots - b_{n-1}) = 0$, and hence $\rho\lambda \in L_n$. □

Thus L_n can be viewed as a collection of ray segments emanating from the origin, and we may well ask: For which $\theta_0 \in (0, 2\pi)$ can the ray described in polar coordinates by $\theta = \theta_0$ intersect L_n nontrivially? We answer this question in the following theorem.

THEOREM 2. *If $\lambda \in L_n^+ - \{0\}$, then* $\arg\lambda \in (\pi/(n-1), \pi]$. *Conversely, for all $\theta \in (\pi/(n-1), \pi]$ there exists $\lambda \in L_n^+ - \{0\}$ such that* $\arg\lambda = \theta$.

Proof. To prove the first statement, we note that if $\lambda = re^{i\theta} \in L_n^+ - \{0\}$, then $\lambda^{n-1} + (1 - a_1)\lambda^{n-2} + \cdots + (1 - a_1 - \cdots - a_{n-1}) = 0$ for some $(a_1, \cdots, a_n) \in K_n$ with $a_1 \neq 1$. Taking imaginary parts of the polynomial equation gives

$$r^{n-1}\sin((n-1)\theta) + (1-a_1)r^{n-2}\sin((n-2)\theta) + \cdots + (1-a_1-\cdots-a_{n-2})r\sin(\theta) = 0;$$

since the left-hand side is strictly positive for all $\theta \in (0, \pi/(n-1)]$, we must have $\arg(\lambda) = \theta \in (\pi/(n-1), \pi]$.

To prove the second statement, we will use induction on n. If $n = 3$ we find that as α runs from 0 to 1, the argument of the solution to $\lambda^2 + (1 - \alpha)\lambda + 1 - \alpha = 0$ which falls in L_3^+ covers the interval $(\pi/2, 2\pi/3]$, while as β runs from 0 to 1, the argument of the solution to $\lambda^2 + \lambda + 1 - \beta = 0$ which falls in L_3^+ covers the interval $[2\pi/3, \pi]$. Thus the result holds when $n = 3$.

Now suppose that the result holds for some $n \geqq 3$. If $\theta \in (\pi/(n-1), \pi]$ there exists $\lambda \in L_n^+ - \{0\}$ with $\arg(\lambda) = \theta$, and hence such a λ also lies in $L_{n+1}^+ - \{0\}$. To complete the induction step, we need only show that if

$$\theta \in \left(\frac{\pi}{n}, \frac{\pi}{n-1}\right] \quad \exists \lambda \in L_{n+1}^+ - \{0\}$$

with $\arg(\lambda) = \theta$. Fix such a θ;

$$\pi > \frac{2\pi}{n+1} \geqq \frac{\pi}{n-1},$$

so $\sin((n+1)\theta) \leqq 0$, $\sin(\theta) > 0$, and $\sin(n\theta) < 0$. Note that there exists $r \in (0, 1]$ such that $r\sin((n+1)\theta) - r^{n+1}\sin(\theta) = \sin(n\theta)$ (since the left side is 0 when $r = 0$ and is $\sin((n+1)\theta) - \sin(\theta) \leqq \sin(n\theta)$ when $r = 1$). Letting

$$\alpha = \frac{r\sin((n+1)\theta)}{\sin(n\theta)}$$

(so that $\alpha > 0$), we see from the equation for r that

$$\alpha = 1 + \frac{r^{n+1} \sin(\theta)}{\sin(n\theta)}$$

so that $\alpha < 1$. We have $r^{n+1} \sin((n+1)\theta) = \alpha r^n \sin(n\theta)$, and by expanding $\sin(\theta)$ as $\sin((n+1)\theta - n\theta)$, we find that

$$\alpha - 1 = \frac{r^{n+1} \sin(\theta)}{\sin(n\theta)} = \alpha r^n \cos(n\theta) - r^{n+1} \cos((n+1)\theta),$$

so that both the real and imaginary parts of $(re^{i\theta})^{n+1} = \alpha(re^{i\theta})^n + 1 - \alpha$ are satisfied. Thus $re^{i\theta} \in L_{n+1}^+ - \{0\}$, and the induction is finished. □

We now have a picture of L_n^+ as a star-shaped set which has a nontrivial intersection with the ray $\theta = \theta_0$ for all $\theta_0 \in (\pi/(n-1), \pi]$. It is natural to wonder: when $\theta \in (\pi/(n-1), \pi]$, at what point does the ray at angle θ exit from L_n^+? If $\theta = \pi$, then the answer is -1, but what about the other values of θ? This question prompts the following definition.

DEFINITION 5. Fix $n \geqq 3$ and $\theta \in (\pi/(n-1), \pi)$. Let $e_{n,\theta}$ be the complex number $re^{i\theta}$ such that $re^{i\theta} \in L_n^+$ but $\gamma re^{i\theta} \notin L_n^+$ for all $\gamma > 1$.

Evidently $e_{n,\theta} \in \partial L_n$ for all $\theta \in (\pi/(n-1), \pi)$ and the question arises as to whether these are the only points in the open upper half plane on the boundary of L_n. The following definition and lemma (in the spirit of [2]) will help us to answer this question by telling us something about int L_n.

DEFINITION 6. Given $z \in \mathbb{C} - \mathbb{R}$ and $a = (a_1, \cdots, a_n) \in K_n$, let P_z^a be the convex hull of $\{z^{n-i} \mid i \in \operatorname{supp}(a)\}$.

We note that P_z^a is either a single point, a line segment, or a polygon with nonvoid interior in $\mathbb{C}$, and that if $\lambda \notin \mathbb{R}$, $\lambda \in L_n$ if and only if $\lambda^n \in P_\lambda^a$ for some $a \in K_n$.

LEMMA 1. *Suppose that* $\lambda \notin \mathbb{R}$ *and* $\lambda^n - \sum_{i=1}^n a_i \lambda^{n-i} = 0$ *for some* $a = (a_1, \cdots, a_n) \in K_n$. *If* P_λ^a *has nonvoid interior in* $\mathbb{C}$, *then* $\lambda \in \operatorname{int} L_n$.

Proof. $\lambda^n \in P_\lambda^a$ and since P_λ^a has nonvoid interior and λ^n is a strictly positive convex combination of all of the points in $\{\lambda^{n-i} \mid i \in \operatorname{supp}(a)\}$, we see that $\lambda^n \in \operatorname{int} P_\lambda^a$. The vertices of P_λ^a are continuous in λ, so there is a neighbourhood of λ such that for each λ_0 in the neighbourhood, $\lambda_0 \neq 1$ and $\lambda_0^n \in P_{\lambda_0}^a$, so that $\lambda_0 \in L_n$. Thus $\lambda \in \operatorname{int} L_n$. □

Lemma 1 will help us to establish the following result.

THEOREM 3. *If* $\lambda \in L_n^+ - \mathbb{R}$ *and* $\arg(\lambda) = \theta$, *then* $\lambda \in \partial L_n$ *if and only if* $\lambda = e_{n,\theta}$.

Proof. Certainly, if $\lambda = e_{n,\theta}$, then $\lambda \in \partial L_n$; we need only show that if $\lambda \in L_n^+ - \mathbb{R}$ and $\lambda \neq e_{n,\theta}$, then $\lambda \in \operatorname{int} L_n$.

If $\lambda \in L_n^+ - \mathbb{R}$ and $\lambda \neq e_{n,\theta}$, then there exists $\gamma > 1$ such that $\gamma\lambda \in L_n^+ - \mathbb{R}$. Hence there exist $(a_1, \cdots, a_n) \in K_n$ such that $(\gamma\lambda)^{n-1} + (1 - a_1)(\gamma\lambda)^{n-2} + \cdots + (1 - a_1 - a_2 - \cdots - a_{n-1}) = 0$. Let $m = \min\{j \mid \sum_{i=1}^j a_i = 1\}$; evidently $n \geqq m \geqq 3$ and $1 - a_1 - \cdots - a_{m-1} > 0$. Dividing the polynomial equation by $\gamma^{n-1}\lambda^{n-m}$ yields

$$\lambda^{m-1} + \frac{(1-a_1)}{\gamma}\lambda^{m-2} + \frac{(1-a_1-a_2)}{\gamma^2}\lambda^{m-3} + \cdots + \frac{(1-a_1-a_2-\cdots-a_{m-1})}{\gamma^{m-1}} = 0.$$

Setting

$$b_1 = 1 - \frac{(1-a_1)}{\gamma},$$

$$b_{i+1} = \frac{(1-a_1-\cdots-a_i)}{\gamma^i} - \frac{(1-a_1-\cdots-a_{i+1})}{\gamma^{i+1}}, \qquad i = 1, \cdots, m-2,$$

$$b_m = \frac{(1-a_1-\cdots-a_{m-1})}{\gamma^{m-1}} \quad \text{and} \quad b = (b_1, \cdots, b_m),$$

gives $\lambda^{m-1} + (1 - b_1)\lambda^{m-2} + \cdots + (1 - b_1 - \cdots - b_{m-1}) = 0$ and hence $\lambda^m = \sum_{i=1}^{m} b_i\lambda^{m-i}$. Note that $b \in K_m$, that $b_i > 0$, $i = 1, \cdots, m$, and that $\lambda^m \in P_\lambda^b$. Since $m \geq 3$, and $\{m, m-1, m-2\} \in \text{supp}(b)$, it follows that 1, λ, and λ^2 are all in P_λ^b. But $\lambda \notin \mathbb{R}$, so 1, λ, and λ^2 are not collinear and hence int P_λ^b is not void, so by Lemma 1, $\lambda \in \text{int } L_m \subseteq \text{int } L_n$. □

COROLLARY 3.1. *If $\lambda \notin \mathbb{R}$ and λ is an eigenvalue of an $n \times n$ substochastic Leslie matrix, at least one of whose row sums is strictly less than unity, then $\lambda \in \text{int } L_n$.*

Proof. Let $\arg(\lambda) = \theta_0 \in (\pi/(n-1), \pi)$ without loss of generality. If λ is an eigenvalue of a strictly substochastic Leslie matrix, then there exists $a_i \geq 0$, $i = 1, \cdots, n$ with $\sum_{i=1}^{n} a_i \in (0, 1)$ such that $\lambda^n - \sum_{i=1}^{n} a_i\lambda^{n-i} = 0$. Thus $(\gamma\lambda)^n - \sum_{i=1}^{n} a_i\gamma^i(\gamma\lambda)^{n-i} = 0$ for all $\gamma > 1$. Since $\sum_{i=1}^{n} a_i\bar{\gamma}^i = 1$ for some $\bar{\gamma} > 1$, we see that $\bar{\gamma}\lambda \in L_n$ and hence $\lambda \in L_n^+ - \mathbb{R}$, but $\lambda \neq e_{n,\theta_0}$ so that $\lambda \in \text{int } L_n$ by Theorem 3. □

COROLLARY 3.2. *$e_{n,\theta}$ is continuous (in θ) for $\theta \in (\pi/(n-1), \pi)$.*

Proof. Suppose that $e_{n,\theta}$ is not continuous at $\bar{\theta}$ for some $\bar{\theta} \in (\pi/(n-1), \pi)$. Then there exists θ_k, $k = 1, 2, \cdots$ and there exists $\varepsilon > 0$ such that $\theta_k \to \bar{\theta}$ but $|e_{n,\theta_k} - e_{n,\bar{\theta}}| > \varepsilon$ for all k. ∂L_n is compact, so a subsequence of e_{n,θ_k} converges, say, to $\bar{\lambda} \in \partial L_n$. Writing $e_{n,\theta_k} = |e_{n,\theta_k}|e^{i\theta_k}$, we see that $\arg(\bar{\lambda}) = \bar{\theta}$. But $|\bar{\lambda} - e_{n,\bar{\theta}}| \geq \varepsilon$, so $\lambda \neq e_{n,\bar{\theta}}$ and hence $\bar{\lambda} \in \text{int } L_n$ by Theorem 3—a contradiction. □

Recalling that $L_n^+ \cap \mathbb{R} = L_2 = [-1, 0]$, our primary interest is in describing the nonreal elements of L_n^+. Since L_n^+ is star-shaped with respect to the origin, a characterization of $e_{n,\theta}$ for $\theta \in (\pi/(n-1), \pi)$ will, in turn, give rise to a complete description of $L_n^+ - \mathbb{R}$, and hence of L_n. In the next section, we will find just such a characterization of $e_{n,\theta}$.

3. Characterizing the boundary of the region. It is evident from our discussion in the introduction that if $\theta = 2\pi k/j$ for some $k, j \in \mathbb{N}$ with $n \geq j > 2k \geq 2$, then $e_{n,\theta} = e^{2\pi ik/j}$, and that these are the only points in the open upper half plane where ∂L_n intersects the unit circle. Fortunately, Lemma 1 suggests a way to find the rest of the nonreal portion of ∂L_n.

LEMMA 2. *If $\lambda \in (\partial L_n) - \mathbb{R}$ there exist $\alpha \in [0, 1]$ and $p, q \in \mathbb{N}$ with $n \geq p > q \geq 1$ such that $\lambda^p - \alpha\lambda^q - (1 - \alpha) = 0$.*

Proof. Since $\lambda \in L_n$, there exists $a \in K_n$ such that $\lambda^n \in P_\lambda^a$. $\lambda \in \partial L_n$, so P_λ^a must have void interior by Lemma 1, and hence P_λ^a is either a line segment or a single point. In either case, we may write $\lambda^n = \alpha\lambda^{n-p+q} + (1 - \alpha)\lambda^{n-p}$ for some $\alpha \in [0, 1]$ and some p, q with $n \geq p > q \geq 1$, which yields the result. □

The exponents arising in Lemma 2 prompt the following definition.

DEFINITION 7. Let $E_n = \{(p, q) \in \mathbb{N} \times \mathbb{N} | n \geq p > q \geq 1\}$.

Lemma 2 also suggests that given $\theta \in (\pi/(n-1), \pi)$, we should attempt to characterize $e_{n,\theta}$ as a root of $\lambda^p - \alpha\lambda^q - (1 - \alpha)$ for some p, q, and α. Notice that $e_{n,\theta}$ is a function of θ (when n is fixed) while the roots of $\lambda^p - \alpha\lambda^q - (1 - \alpha)$ are parameterized by α (when p and q are fixed). The following lemma presents some properties of the roots of $\lambda^p - \alpha\lambda^q - (1 - \alpha)$ which will help us to reparameterize these roots in terms of θ.

LEMMA 3. (i) *If $\lambda = re^{i\theta}$ for some $\theta \in (0, \pi)$ and $r > 0$, and λ satisfies* (a) $\lambda^n = \alpha\lambda^{n-p} + 1 - \alpha$ *for some $\alpha \in [0, 1]$ and some $(n, p) \in E_n$, then the pair (r, θ) satisfies* (b) $r^p \sin(n\theta) - r^n \sin(p\theta) = \sin((n-p)\theta)$.

(ii) *If $\theta \in (0, \pi)$ and either $*\{\sin(n\theta) \geq 0, \sin(p\theta) \leq 0, \sin((n-p)\theta) > 0\}$ or $**\{\sin(n\theta) \leq 0, \sin(p\theta) \geq 0, \sin((n-p)\theta) < 0\}$ holds, then there exists $r(\theta)$ with values in $(0, 1]$ which is differentiable in a neighbourhood of θ, such that the pair $(r(\theta), \theta)$ satisfies* (b). *Further, there exists $\alpha(\theta)$ with values in $[0, 1]$ such that*

$(r(\theta)e^{i\theta})^n = \alpha(\theta)(r(\theta)e^{i\theta})^{n-p} + 1 - \alpha(\theta)$, *and if* $\lambda \neq 0$, $\arg(\lambda) = \theta$ *and* λ *solves* (a) *for some* $\alpha \in [0, 1]$, *then* λ *must equal* $r(\theta)e^{i\theta}$.

Proof. (i) If $\alpha = 0$, then $\lambda = e^{2\pi il/n}$ for some $l \in \mathbb{N}$ with $n/2 > l \geqq 1$, and certainly the pair $(1, 2\pi l/n) = (r, \theta)$ satisfies (b). If $\alpha > 0$ in (a), substitution of $\lambda = re^{i\theta}$ into (a) gives $r^n \cos(n\theta) - 1 = \alpha(r^{n-p}\cos((n-p)\theta) - 1)$ and $r^n \sin(n\theta) = \alpha r^{n-p}\sin((n-p)\theta)$. Crossmultiplying these two equations and cancelling the common nonzero factor of αr^{n-p} from each side yields $r^n \cos(n\theta)\sin((n-p)\theta) - \sin((n-p)\theta) = r^n \cos((n-p)\theta)\sin(n\theta) - r^p \sin(n\theta)$, which reduces to (b).

(ii) Suppose that $*$ holds for some $\theta \in (0, \pi)$ and consider the function $g(r) = r^p \sin(n\theta) - r^n \sin(p\theta) - \sin((n-p)\theta)$ for $r \in [0, 1]$. $g(0) = -\sin((n-p)\theta) < 0$, $g(1) = \sin(n\theta) - \sin(p\theta) - \sin((n-p)\theta) \geqq 0$ and g is strictly increasing in r, so there exists $r(\theta)$ with $1 \geqq r(\theta) > 0$ such that $g(r(\theta)) = 0$. Since $\partial g(r)/\partial r|_{r(\theta)} > 0$ the Implicit Function Theorem applies and so $r(\theta)$ is differentiable in a neighbourhood of θ.

Now let

$$\alpha(\theta) = \frac{r^p(\theta)\sin(n\theta)}{\sin((n-p)\theta)};$$

$\alpha(\theta) \geqq 0$ follows from $*$, while (b) tells us that

$$\alpha(\theta) = 1 + \frac{r^n(\theta)\sin(p\theta)}{\sin((n-p)\theta)} \leqq 1,$$

the inequality following from $*$. From the definition of $\alpha(\theta)$, we have that (1) $r^n(\theta)\sin(n\theta) = \alpha(\theta)r^{n-p}(\theta)\sin((n-p)\theta)$. Further,

$$\alpha(\theta) - 1 = \frac{r^n(\theta)\sin(p\theta)}{\sin((n-p)\theta)} = \alpha(\theta)r^{n-p}(\theta)\cos((n-p)\theta) - r^n(\theta)\cos(n\theta),$$

and hence (2) $r^n(\theta)\cos(n\theta) = \alpha(\theta)r^{n-p}(\theta)\cos((n-p)\theta) + 1 - \alpha(\theta)$. (1) and (2) are the imaginary and real parts, respectively, of $(r(\theta)r^{i\theta})^n = \alpha(\theta)(r(\theta)e^{i\theta})^{n-p} + 1 - \alpha(\theta)$.

Finally, if $\lambda \neq 0$, $\arg(\lambda) = \theta$ and λ solves (a) for some $\alpha \in [0, 1]$, then writing $\lambda = \rho e^{i\theta}$, we have (from (i)) $g(\rho) = 0$ and hence $\rho = r(\theta)$. This proves (ii) when $*$ holds; the proof of (ii) when $**$ holds is analogous, and is omitted. □

Recall that Lemma 2 tells us that every point on $(\partial L_n) - \mathbb{R}$ satisfies at least one equation of the form $\lambda^p - \alpha\lambda^q - (1 - \alpha) = 0$, $\alpha \in [0, 1]$, $(p, q) \in E_n$; it is clear that some points on $(\partial L_n) - \mathbb{R}$, for example $e^{2\pi ik/j}$ when $(j, k) \in E_n$, may satisfy more than one such equation. The lemma which follows establishes which elements of $L_n - \mathbb{R}$ can satisfy two (or more) such equations.

LEMMA 4. *Fix* $n \geqq 3$ *and suppose that* $\lambda \in \mathbb{C} - \mathbb{R}$ *and* $\lambda \neq e^{2\pi ik/j}$ *for all* $(j, k) \in E_n$. *Further suppose that* λ *solves both* (1) $\lambda^n = \alpha\lambda^p + 1 - \alpha$ *for some* $\alpha \in [0, 1]$ *and some* $p \in \mathbb{N}$ *with* $n > p \geqq 1$, *and* (2) $\lambda^m = \beta\lambda^q + 1 - \beta$ *for some* $\beta \in [0, 1]$ *and some* $(m, q) \in E_n - (n, p)$. *Then* $\lambda \in \operatorname{int} L_n$.

Proof. First we note that $\alpha, \beta \in (0, 1)$ since $\lambda \neq e^{2\pi ik/j}$ for all $(j, k) \in E_n$ and $\lambda \neq 0$. If $\lambda^n \in \mathbb{R}$, then λ^p and λ^{n-p} are also real, and one of λ^n, λ^p, and λ^{n-p} must be positive. But $\lambda \notin \mathbb{R}$ and $\lambda \neq e^{2\pi ik/j}$ for all $(j, k) \in E_n$, so $\lambda \neq e_{n,\arg(\lambda)}$ and hence $\lambda \in \operatorname{int} L_n$ by Theorem 3. To complete the proof, we will take $\arg(\lambda^n) \in (0, \pi)$ without loss of generality, and proceed by extended induction on n.

For $n = 3$, we note that the solutions of $\lambda^2 = \alpha\lambda + 1 - \alpha$, $\alpha \in [0, 1]$ are real, and when $\alpha, \beta \in [0, 1]$, the only nonreal solutions common to both $\lambda^3 = \alpha\lambda^2 + 1 - \alpha$ and $\lambda^3 = \beta\lambda + 1 - \beta$ are $e^{\pm 2\pi i/3}$, when $\alpha = \beta = 0$. Thus the result holds trivially for $n = 3$.

Next we suppose that the result holds for all $l \in \mathbb{N}$ with $n-1 \geqq l \geqq 3$; we must show that it holds for n. Either $n < m$ or $n = m$, and we suppose the latter for the moment. When $n = m$ then $p \neq q$ and so we will take $p > q$. This gives rise to the picture shown in Fig. 1. From Fig. 1 we have (3) $\lambda^p = \gamma\lambda^q + 1 - \gamma$ for some $\gamma \in (0, 1)$, and recalling that $1 \leqq q < p < n$, we see that the case $n = m$ can be reduced to the case $n > m$ by considering (1) and (3) instead of (1) and (2).

We now take $n > m$. (1) tells us that λ^n is on the segment joining λ^p to 1, while multiplying (2) by λ^{n-m} tells us that λ^n is also on the segment joining λ^{n-m+q} to λ^{n-m}. Thus λ^n is at the intersection of two line segments, and if they are not collinear, then the convex hull of $\{1, \lambda^p, \lambda^{n-m}, \lambda^{n-m+q}\}$ has nonvoid interior. Defining $a = (a_1, \cdots, a_n)$ by setting supp $(a) = \{n-p, m-q, n, m\}$ and letting $a_{n-p} = \alpha/2 > 0$, $a_{m-q} = \beta/2 > 0$, $a_m = (1-\beta)/2 > 0$, and $a_n = (1-\alpha)/2 > 0$, we have $a \in K_n$ and (considering $\frac{1}{2}(1) + \frac{1}{2}\lambda^{n-m}(2))\lambda^n \in P_\lambda^q$. But as noted above, P_λ^q has nonvoid interior, so $\lambda \in \text{int } L_n$ by Lemma 1.

If the segment joining λ^p to 1 is collinear with that joining λ^{n-m+q} to λ^{n-m}, we recall that λ^n sites on the segment joining 1 to λ^p, and we note that λ^{n-m+q} and λ^{n-m} must "bracket" λ^n on the line; in particular, one of λ^{n-m+q} and λ^{n-m}, say λ^b, falls strictly between 1 and λ^n on the line. The configuration is shown in Fig. 2. From Fig. 2, we see that (4) $\lambda^n = \gamma\lambda^p + (1-\gamma)\lambda^b$ for some $\gamma \in (0, 1)$. If $p > b$ and $(n-b, p-b) \neq (m, q)$, then (5) $\lambda^{n-b} = \gamma\lambda^{p-b} + 1 - \gamma$ and (2) $\lambda^m = \beta\lambda^q + 1 - \beta$, so by the induction step, $\lambda \in \text{int } L_{\max(n-b,m)} \subseteq \text{int } L_n$. If $p < b$ and $(n-p, b-p) \neq (m, q)$, then (6) $\lambda^{n-p} = \gamma + (1-\gamma)\lambda^{b-p}$, which, along with (2), yields $\lambda \in \text{int } L_{\max(n-p,m)} \subseteq \text{int } L_n$ by the induction step. The only remaining cases are (A) $(n-b, p-b) = (m, q)$ and (B) $(n-p, b-p) = (m, q)$. We shall deal with (A) first.

If (A) holds, then $b = n - m = p - q$ and λ^p, λ^n, λ^{p-q}, and 1 are all collinear in that order (see Fig. 2), and hence λ^q, λ^{n+q-p}, 1 and λ^{q-p} are all collinear in that order, and λ^{q-p} is in the open lower half plane. The configurations are shown in Figs. 3 and 4. Evidently $\phi_1, \phi_2 \in (0, \pi/2)$, as indicated in Figs. 3 and 4. Letting $\lambda^{p-q} = \rho e^{i\theta_0}$, we see that since $|\lambda^{p-q} - 1| > 0$, then $1 - 2\rho\cos(\theta_0) + \rho^2 > 0$, which gives

$$1 > \frac{\rho\cos(\theta_0) - \rho^2}{1 - \rho\cos(\theta_0)} = \frac{\tan(\phi_1)}{\tan(\phi_2)}.$$

Thus $\phi_1 < \phi_2$ and we can combine Figs. 3 and 4 into the picture shown in Fig. 5. From Fig. 5, we see that λ^n is in the interior of the convex hull of $\{1, \lambda^q, \lambda^p, 0\}$ and

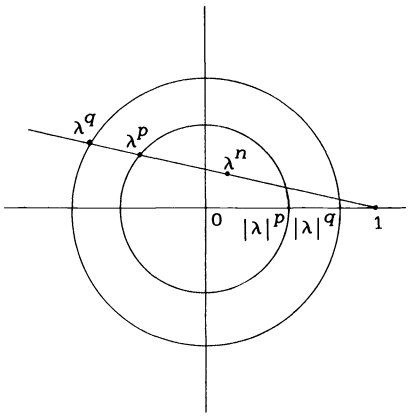


FIG. 1

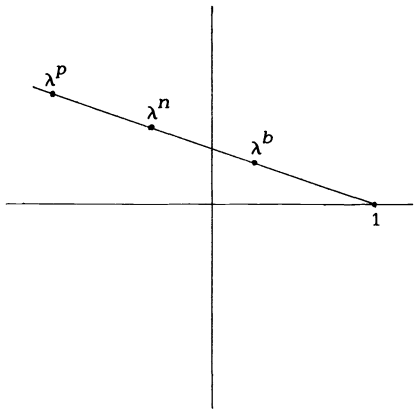


FIG. 2

hence $\lambda^n = c_1 + c_2\lambda^q + c_3\lambda^p + (1 - c_1 - c_2 - c_3)(0)$ for some $c_i > 0$, $i = 1, 2, 3$ with $\sum_{i=1}^{3} c_i < 1$. So λ is a nonreal eigenvalue of a strictly substochastic Leslie matrix, and hence $\lambda \in \text{int } L_n$ by Corollary 3.1.

If (B) holds, then $p = n - m$, $b = p + q = n - m + q$, and λ^p, λ^n, λ^{p+q}, and 1 are collinear in that order (see Fig. 2) and hence 1, $\lambda^{n-p} = \lambda^m$, λ^q, and λ^{-p} are collinear in that order with λ^{-p} in the open lower half plane. Note that the map $z \mapsto 1/z$ preserves the order of points encountered on a straight line, and takes the line through 1, λ^m, λ^q, and λ^{-p} to the circle through 0, 1, λ^{-m}, λ^{-q}, and λ^p and takes the lower half plane to the upper half plane. Noticing that (2) yields $\lambda^{m-q} = \beta + (1 - \beta)\lambda^{-q}$, we arrive at the picture shown in Fig. 6. From Fig. 6, λ^n is in the interior of the convex hull of $\{1, \lambda^{m-q}, \lambda^p, 0\}$ and so as in the preceding paragraph, λ is a nonreal eigenvalue of a strictly substochastic Leslie matrix. Thus $\lambda \in \text{int } L_n$ by Corollary 3.1, which completes the induction step and the proof. □

We know that for each $\theta \in (\pi/(n - 1), \pi)$, $e_{n,\theta}$ will be a root of $\lambda^m - \alpha\lambda^l - (1 - \alpha)$ for some m, l, and α. Lemma 4 suggests that as θ runs between two neighbouring angles of the form $2\pi k/j$, $(j, k) \in E_n$, the m and l will be fixed, and only α will change with θ. The angles $2\pi k/j$, $(j, k) \in E_n$ give rise to a partition of

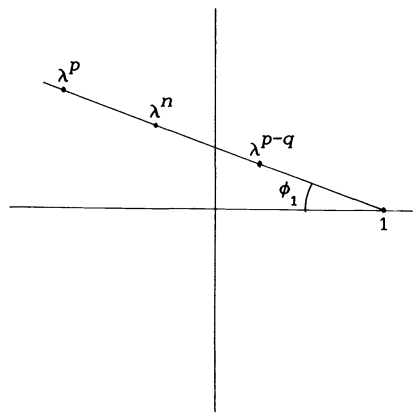


FIG. 3

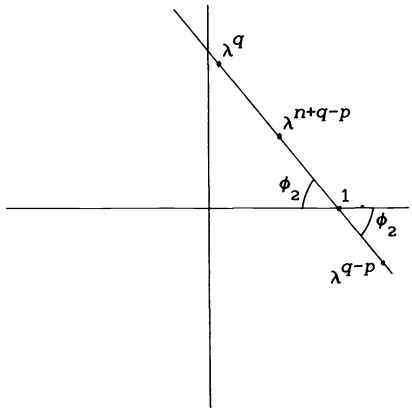


FIG. 4

$(\pi/(n-1), \pi)$ and a brief discussion of their properties will prove useful in our characterization of ∂L_n.

DEFINITION 8. For each $n \geqq 3$, let

$$A_n = \left\{ \frac{2\pi k}{j} \middle| (j,k) \in E_n, \frac{k}{j} < \frac{1}{2} \text{ and } \frac{k}{j} \text{ is in lowest terms} \right\}.$$

We order A_n increasingly as

$$\frac{2\pi}{n} = \frac{2\pi k_1}{j_1} < \frac{2\pi k_2}{j_2} < \cdots < \frac{2\pi k_u}{j_u}.$$

The fractions k_s/j_s, $s = 1, \cdots, u$ are actually a subset of the nth Farey Series,

$$F_n = \left\{ \frac{k}{j} \middle| k,j \in \mathbb{Z}, 0 \leqq k \leqq j \leqq n, j \neq 0 \text{ and } \frac{k}{j} \text{ is in lowest terms} \right\}.$$

We shall require the following facts about consecutive terms in a Farey Series, which can be found in Roberts [8], or can be easily demonstrated by the reader.

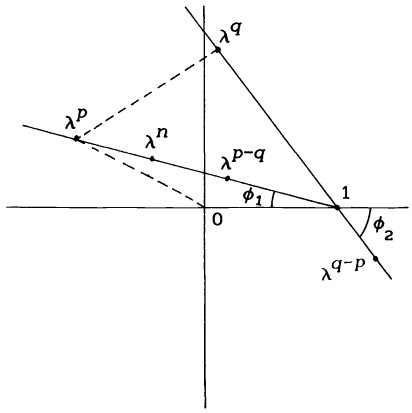


FIG. 5

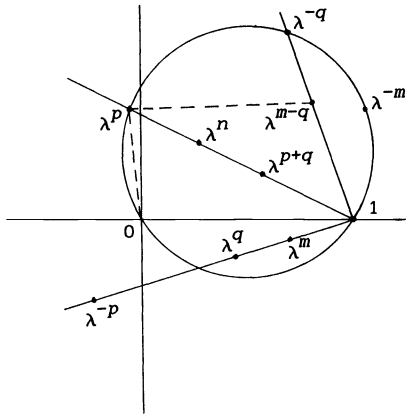


FIG. 6

FACTS. (i) $j_s + j_{s+1} > n$ when $1 \leqq s \leqq u - 1$.

(ii) If $1 \leqq s \leqq u - 1$, then

$$j_{s+1} = m + i_0 j_s \quad \text{and} \quad k_{s+1} = \frac{k_s(m + i_0 j_s) + 1}{j_s},$$

where m is defined by $0 < m < j_s$ and $k_s m = -1 \bmod (j_s)$ and

$$i_0 = \left[\frac{n-m}{j_s}\right].$$

(Here the [ ] denotes the greatest integer function.)

(iii) If $1 \leqq s \leqq u - 1$ then j_s and j_{s+1} are mutually prime.

As mentioned above, the set A_n gives rise to a partition of $(\pi/(n-1), \pi)$, namely into the subintervals

$$\left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right], \quad \left[\frac{2\pi k_s}{j_s}, \frac{2\pi k_{s+1}}{j_{s+1}}\right], \qquad s = 1, \cdots, u-1$$

and $[2\pi k_u/j_u, \pi)$. The lemma which follows will confirm our suspicions about the implications of Lemma 4.

LEMMA 5. (i) *Suppose that* $n \geqq 3$ *and that* $2\pi k_s/j_s \in A_n$, $s = 1, \cdots, u$. *Let* I *be one of the intervals*

$$\left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right], \quad \left[\frac{2\pi k_s}{j_s}, \frac{2\pi k_{s+1}}{j_{s+1}}\right], \qquad s = 1, \cdots, u-1$$

or $[2\pi k_u/j_u, \pi)$ *and let* $\theta_0 \in \operatorname{int} I$. *If there exists* $\alpha \in [0, 1]$ *such that* e_{n,θ_0} *solves* $\lambda^p = \alpha\lambda^q + 1 - \alpha$ *for some* $(p, q) \in E_n$, *then for all* $\theta \in I$ *there exists* $\alpha \in [0, 1]$ *such that* $e_{n,\theta}$ *solves* $\lambda^p = \alpha\lambda^q + 1 - \alpha$.

(ii) *If there exist* $\alpha_0, \alpha_1 \in [0, 1]$ *such that* $e^{2\pi i k_s/j_s}$ *solves* $\lambda^p = \alpha_0\lambda^q + 1 - \alpha_0$ *and* $e^{(2\pi i k_{s+1})/(j_{s+1})}$ *solves* $\lambda^p = \alpha_1\lambda^q + 1 - \alpha_1$ *for some* $s = 1, \cdots, u - 1$ *and some* $(p, q) \in E_n$, *then* (α_0, α_1) *is either* $(0, 1)$ *or* $(1, 0)$.

Proof. If there exists $\bar{\theta} \in I$ such that $e_{n,\bar{\theta}}$ does not solve $\lambda^p = \alpha\lambda^q + 1 - \alpha$ for any $\alpha \in [0, 1]$, then without loss of generality we may take $\bar{\theta} > \theta_0$, and let θ_1 be the first such element of I which is larger than θ_0. By Lemma 2 e_{n,θ_1} solves $\lambda^{p_1} = \beta\lambda^{q_1} + 1 - \beta$ for some $\beta \in [0, 1]$, $(p_1, q_1) \in E_n - \{(p, q)\}$. Thus for all $\theta \in [\theta_0, \theta_1)$ there exists $\alpha_\theta \in [0, 1]$ such

that $(e_{n,\theta})^p = \alpha_\theta(e_{n,\theta})^q + 1 - \alpha_\theta$; letting $\{\tau_i\}$ be a sequence in $[\theta_0, \theta_1)$ such that $\tau_i \to \theta_1$ and α_{τ_i} converges, say to $\alpha_{\theta_1} \in [0, 1]$, we see that since $e_{n,\theta}$ is continuous in θ, e_{n,θ_1} solves $\lambda^p = \alpha_{\theta_1}\lambda^q + 1 - \alpha_{\theta_1}$. By Lemma 4, θ_1 must be one of the closed end points of I (and hence $\bar{\theta} = \theta_1$) and for all $\theta \in [\theta_0, \theta_1]$ there exists $\alpha \in [0, 1]$ such that $e_{n,\theta}$ solves $\lambda^p = \alpha\lambda^q + 1 - \alpha$, which proves the result.

(ii) If $\alpha_0 \in (0, 1)$, then the $p \times p$ row-stochastic Leslie matrix with an α_0 in the $(p-q)$th spot of the top row and a $1 - \alpha_0$ in the pth spot of the top row has $e^{2\pi i k_s/j_s}$ as an eigenvalue, so $j_s | p$ and $j_s | (p-q)$. Since $e^{(2\pi i k_{s+1})/(j_{s+1})}$ solves $\lambda^p = \alpha_1\lambda^q + 1 - \alpha_1$, we see that j_{s+1} divides one of p and $(p-q)$. Fact (iii) tells us that j_s and j_{s+1} are mutually prime, so $p \geqq j_s j_{s+1}$. But $j_s j_{s+1} \geqq j_s + j_{s+1} > n$ (the last by Fact (i))—a contradiction. A similar argument applies if $\alpha_1 \in (0, 1)$, so $\alpha_0, \alpha_1 \in \{0, 1\}$.

If $\alpha_0 = \alpha_1 = 0$, then $p \geqq j_s j_{s+1}$, while if $\alpha_0 = \alpha_1 = 1$, then $(p-q) \geqq j_s j_{s+1}$ and each of these leads to the contradiction above. Thus (α_0, α_1) is either $(0, 1)$ or $(1, 0)$. □

The subdivision of $(\pi/(n-1), \pi)$ by A_n into subintervals gives rise to the $u - 1$ (interior) intervals

$$\left[\frac{2\pi k_s}{j_s}, \frac{2\pi k_{s+1}}{j_{s+1}}\right], \qquad s = 1, \cdots, u-1,$$

and two end ones,

$$\left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right] \quad \text{and} \quad \left[\frac{2\pi k_u}{j_u}, \pi\right);$$

(in light of Lemma 5(i)), it is natural to attempt to characterize $e_{n,\theta}$ (as roots of some $\lambda^p - \alpha\lambda^q - (1-\alpha)$) as θ ranges over each subinterval. Typically, a subinterval may have several such polynomials as candidates, and the problem is to determine which of the candidates has $e_{n,\theta}$ as a root. Recalling that we also want to express the roots of such polynomials in terms of θ (instead of α), we should look for a condition like $*$ or $**$ in Lemma 3 to hold when θ is in one of these subintervals. We address this matter now.

LEMMA 6. *Fix $n \geqq 3$; suppose that*

$$\frac{2\pi k_s}{j_s}, \frac{2\pi k_{s+1}}{j_{s+1}} \in A_n$$

for some $s = 1, \cdots, u-1$ and that

$$\frac{a}{b} = \frac{k_s}{j_s}, \qquad \frac{c}{d} = \frac{k_{s+1}}{j_{s+1}}$$

for some $a, b, c, d \in \mathbb{N}$ with $1 \leqq b, d \leqq n$ (note that neither a/b nor c/d need be in lowest terms). Further suppose that if $d > b$ then $(c, d) \neq (2, 6)$. Then

$$\forall \theta \in \left[\frac{2\pi a}{b}, \frac{2\pi c}{d}\right]$$

the following inequalities hold:

(i) $(2a+1)\pi > b\theta \geqq 2a\pi$,

(ii) $2\pi c \geqq d\theta > (2c-1)\pi$,

(iii) $(2a - 2c + 1)\pi > (b-d)\theta > (2a-2c)\pi$.

Proof. First we note that $b \neq d$, for if $b = d$, then both j_s and j_{s+1} divide b, which yields $b \geqq j_s j_{s+1} \geqq j_s + j_{s+1} > n$ (by Facts (i) and (iii))—a contradiction.

(i) $2a/b < 1$, so $2a \leqq b - 1$. We claim that

$$\frac{2a+1}{b} > \frac{2c}{d},$$

which is obvious if $2a = b - 1$. If $2a < b - 1$, then

$$\frac{2a+1}{b} > \frac{2a}{b-1}.$$

But by the ordering of A_n,

$$\frac{2a}{b-1} \geqq \frac{2k_{s+1}}{j_{s+1}} = \frac{2c}{d}, \quad \text{so } \frac{2a+1}{b} > \frac{2c}{d},$$

as claimed. Thus if

$$\theta \in \left[\frac{2\pi a}{b}, \frac{2\pi c}{d}\right],$$

then

$$(2a+1)\pi > \frac{2\pi cb}{d} \geqq b\theta \geqq 2\pi a.$$

(ii) We claim that

$$\frac{2a}{b} > \frac{2c-1}{d}.$$

If $b > d$, then from the claim in (i), we have

$$\frac{2a+1}{b} > \frac{2c}{d},$$

which yields

$$\frac{2a}{b} > \frac{2c}{d} - \frac{1}{b} > \frac{2c-1}{d},$$

as desired. Now we take $b < d$, and note that if this is the case, we must have $c \geqq 2$ and $d \geqq 5$. We shall first suppose that $2c/d \neq 2/3$. Consider the collection of intervals $(4/3r, 2/r)$, $r = 2, \cdots, [d/3] + 1$. Since $r \geqq 2$ we have

$$\frac{2}{r+1} \geqq \frac{4}{3r}$$

(with equality if and only if $r = 2$) and so the union of the intervals covers the set $[4/d, 1) - \{2/3\}$ because the intervals overlap for $r \geqq 3$. Thus

$$\frac{2c}{d} \in \left(\frac{4}{3r}, \frac{2}{r}\right)$$

for some $r \in \mathbb{N}$ with $2 \leqq r \leqq [d/3] + 1$. So for some such r, $2d > 2cr$ and hence

$$\frac{2c}{d} > \frac{2c-2}{d-r}.$$

Since

$$\frac{4}{3r} \geqq \frac{2c}{(2c-1)r},$$

we have

$$\frac{2c}{d} > \frac{4}{3r} \geqq \frac{2c}{(2c-1)r},$$

so that $(2c-1)r > d$ and hence

$$\frac{2(c-1)}{d-r} > \frac{2c-1}{d}.$$

Thus for some suitable r,

$$\frac{2c}{d} > \frac{2(c-1)}{d-r} > \frac{2c-1}{d}$$

and so by the ordering of A_n, we must have

$$\frac{2c}{d} > \frac{2a}{b} \geqq \frac{2(c-1)}{d-r} > \frac{2c-1}{d},$$

as claimed.

If $d > b$ and $(2c/d) = (2/3)$ but $(c, d) \neq (2, 6)$, then $d > 6$, $c > 2$, and $d = 3c$. This yields

$$\frac{2c}{d} > \frac{2(c-1)}{d-2},$$

as well as

$$\frac{2(c-1)}{d-2} > \frac{2c-1}{d},$$

so that by the ordering of A_n,

$$\frac{2c}{d} > \frac{2a}{b} \geqq \frac{2(c-1)}{d-2} \geqq \frac{2c-1}{d}$$

and the claim holds. Thus if $\theta \in [2\pi a/b, 2\pi c/d]$, then

$$2\pi c \geqq d\theta \geqq \frac{2\pi ad}{b} > (2c-1)\pi.$$

(iii) If $\theta \in [2\pi a/b, 2\pi c/d]$, then

$$2\pi a - \frac{2\pi ad}{b} \leqq (b-d)\theta \leqq \frac{2\pi cb}{d} - 2\pi c.$$

We showed in (i) that $2a + 1 > (2cb/d)$ and obviously $(2ad/b) < 2c$, so that $(2a - 2c)\pi < (b - d)\theta < (2a + 1 - 2c)\pi$. □

COROLLARY 6.1. *Under the hypotheses of Lemma* 6, *if* $b > d (b < d)$, *then for all* $\theta \in [2\pi a/b, 2\pi c/d]$ *there exists* $r(\theta)$ *with values in* $(0, 1]$ *and there exists* $\alpha(\theta)$ *with values in* $[0, 1]$, *with* $r(\theta)$ *differentiable in* θ *such that* $r(\theta)e^{i\theta}$ *solves* $\lambda^b = \alpha(\theta)\lambda^{b-d} + 1 - \alpha(\theta)$ $(\lambda^d = \alpha(\theta)\lambda^{d-b} + 1 - \alpha(\theta))$. *Further,* $r(2\pi a/b) = r(2\pi c/d) = 1$ *and* $r(\theta)e^{i\theta}$ *is the*

unique nonzero solution of $\lambda^b = \alpha\lambda^{b-d} + 1 - \alpha$ ($\lambda^d = \alpha\lambda^{d-b} + 1 - \alpha$), $\alpha \in [0, 1]$ *with argument* θ.

Proof. If $b > d$, then by Lemma 6, if $\theta \in [2\pi a/b, 2\pi c/d]$, condition $*$ in Lemma 3(ii) holds for $\lambda^b = \alpha\lambda^{b-d} + 1 - \alpha$ so that Lemma 3(ii) provides the desired $r(\theta)$ and $\alpha(\theta)$ and it is easily checked (using (b) in Lemma 3(ii)) that $r(2\pi a/b) = r(2\pi c/d) = 1$. If $b < d$, then by Lemma 6, if $\theta \in [2\pi a/b, 2\pi c/d]$, condition $**$ in Lemma 3(ii) holds for $\lambda^d = \alpha\lambda^{d-b} + 1 - \alpha$ and again the result follows. □

We pause to make a brief remark about the case $d > b$, $(c, d) = (2, 6)$, which was omitted from Lemma 6 and its corollary. In conjunction with the other hypotheses of the lemma, it is easy to see that this condition requires that $n = 6$, $a = 1$, and $b = 4$. Observe that if this is the case, the strict lower bound in (ii) and the strict upper bound in (iii) both fail when $\theta = \pi/2$. The corollary also fails when $\theta = \pi/2$: although the polynomial $\lambda^6 - \alpha\lambda^2 - (1 - \alpha)$ has, for each $\alpha \in [0, \frac{3}{4})$, one root $\lambda(\alpha)$ such that $\arg(\lambda(\alpha)) \in (\pi/2, 2\pi/3]$, it has at $\alpha = \frac{3}{4}$ the double root $i/\sqrt{2}$ and for $\alpha \in (\frac{3}{4}, 1]$ it has two pure imaginary roots in the upper half plane, one of which goes to 0 while the other goes to i as $\alpha \to 1^-$. In particular, neither $r(\pi/2)$ nor $\alpha(\pi/2)$ are well defined in this case.

Having established technical conditions for the interior subintervals, we now turn our attention to some properties of the subintervals at either end: $(\pi/(n-1), 2\pi/n]$ and $[2\pi k_u/j_u, \pi)$; these properties will help us to decide which polynomials can have $e_{n,\theta}$ as a root when θ is in one of these two end intervals.

LEMMA 7. (i) *Fix* $q \in \mathbb{N}$ *such that* $1 \leqq q \leqq n - 1$ *and suppose that*

$$\forall\theta \in \left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right] \quad \exists\alpha \in [0, 1]$$

such that $\lambda^n - \alpha\lambda^q - (1 - \alpha)$ *has a nonzero root with argument* θ. *Then* $q = n - 1$.

(ii) *Fix* $l \in \mathbb{N}$ *such that* $1 \leqq l \leqq j_u - 1$ *and suppose that*

$$\forall\theta \in \left[\frac{2\pi k_u}{j_u}, \pi\right) \quad \exists\alpha \in [0, 1]$$

such that $\lambda^{j_u} - \alpha\lambda^{j_u - l} - (1 - \alpha)$ *has a nonzero root with argument* θ. *Then* l *is even. Further, if* $1 \leqq q \leqq j_u$, *there exists* $\theta \in [2\pi k_u/j_u, \pi)$ *such that no polynomial of the form* $\lambda^{j_u+1} - \alpha\lambda^q - (1 - \alpha)$, $\alpha \in [0, 1]$, *can have a nonzero root with argument* θ.

Proof. (i) Writing $\lambda = re^{i\theta}$ and taking the imaginary parts of $\lambda^n = \alpha\lambda^q + 1 - \alpha$, we see from the hypothesis that

$$\forall\theta \in \left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right) \quad \exists r, \alpha \in [0, 1]$$

such that $r^n \sin(n\theta) = \alpha r^q \sin(q\theta)$. Since $\lambda \neq 0$ then $r > 0$ and since $\sin(n\theta) < 0$, we must have $\alpha > 0$. But if $1 \leqq q \leqq n - 2$, there exists

$$\bar{\theta} \in \left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right)$$

such that $\sin(q\bar{\theta}) > 0$, which is impossible. Thus q must be $n - 1$.

(ii) First we note that j_u is n if n is odd while j_u is $n - 1$ if n is even, and that $2k_u = j_u - 1$. To prove the first statement in part (ii), suppose that the hypothesis holds for some odd l. Writing $\lambda = re^{i\theta}$ and applying Lemma 3(i), we see that

$$\forall\theta \in \left[\frac{2\pi k_u}{j_u}, \pi\right) \quad \exists r \in (0, 1]$$

such that $r^l \sin(j_u\theta) - r^{j_u}\sin(l\theta) = \sin((j_u - l)\theta)$. The right-hand side is not zero for such θ, so the equation is equivalent to

$$\frac{r^l \sin(j_u\theta)}{\sin((j_u-l)\theta)} - \frac{r^{j_u}\sin(l\theta)}{\sin((j_u-l)\theta)} = 1.$$

As $\theta \to \pi^-$,

$$\frac{\sin(j_u\theta)}{\sin((j_u-l)\theta)} \to \frac{-j_u}{j_u-l} \quad \text{and} \quad \frac{\sin(l\theta)}{\sin((j_u-l)\theta)} \to \frac{-l}{j_u-l},$$

so that when θ is sufficiently close to π, we see that

$$\frac{r^l \sin(j_u\theta)}{\sin((j_u-l)\theta)} - \frac{r^{j_u}\sin(l\theta)}{\sin((j_u-l)\theta)} \leqq 0 \quad \forall r \in [0,1],$$

and hence for such θ, there is no $\alpha \in [0, 1]$ such that $\lambda^{j_u} - \alpha\lambda^{j_u-1} - (1-\alpha)$ has a root with argument θ. Thus l is even, as desired.

To prove the second statement, first suppose that q is odd and that $\alpha \in [0, 1]$. Writing $\lambda = re^{i\theta}$ and taking imaginary parts of $\lambda^{j_u+1} = \alpha\lambda^q + 1 - \alpha$, we see that any nonzero root with argument θ must satisfy $r^{j_u+1}\sin((j_u+1)\theta) = \alpha r^q \sin(q\theta)$. When $\theta \in [2\pi k_u/j_u, \pi)$ and is sufficiently close to π, the left side is negative while the right side is positive, so that $\lambda^{j_u+1} - \alpha\lambda^q - (1-\alpha)$ cannot have a nonzero root with argument θ. If q is even and $\alpha \in [0, 1]$, writing $j_u + 1 = 2p$, $q = 2m$, and $\gamma = \lambda^2$, we see that $\lambda^{j_u+1} = \alpha\lambda^q + 1 - \alpha$ yields $\gamma^p = \alpha\gamma^m + 1 - \alpha$. If $\arg(\lambda) = \pi - \varepsilon$ for $\varepsilon \in (0, \pi/2(p-1))$, then

$$\gamma \in L_p \quad \text{and} \quad \arg(\gamma) = 2\pi - 2\epsilon \in \left(2\pi - \frac{\pi}{p-1}, 2\pi\right),$$

contradicting Theorem 2. Thus $\lambda^{j_u+1} - \alpha\lambda^q - (1-\alpha)$ cannot have a nonzero root with argument $\pi - \varepsilon$, and the result is proved. □

Upon making one final definition, we will at last be ready to characterize $e_{n,\theta}$ for $\theta \in (\pi/(n-1), \pi)$.

DEFINITION 9. Fix $n \geqq 3$, suppose that

$$\frac{2\pi k_s}{j_s}, \frac{2\pi k_{s+1}}{j_{s+1}} \in A_n, \qquad s = 1, \cdots, u-1,$$

and recall that $j_s \neq j_{s+1}$. Let $g_s = \max(j_s, j_{s+1})$ and $l_s = \min(j_s, j_{s+1})$. We also let

$$d_s = \begin{cases} k_s & \text{if } g_s = j_s, \\ k_{s+1} & \text{if } g_s = j_{s+1}, \end{cases}$$

and

$$m_s = \begin{cases} k_s & \text{if } l_s = j_s, \\ k_{s+1} & \text{if } l_s = j_{s+1}. \end{cases}$$

THEOREM 4. *Fix $n \geqq 3$ and consider the intervals*

$$\left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right], \quad \left(\frac{2\pi k_s}{j_s}, \frac{2\pi k_{s+1}}{j_{s+1}}\right], \qquad s = 1, \cdots, u-1$$

and $[2\pi k_u/j_u, \pi)$, *where*

$$\frac{2\pi k_s}{j_s} \in A_n, \qquad s = 1, \cdots, u.$$

The following assertions hold.

(i) *If* $\theta \in (\pi/(n-1), 2\pi/n]$, *then* $e_{n,\theta}$ *is a root of* $\lambda^n - \alpha\lambda^{n-1} - (1-\alpha)$ *for some* $\alpha \in [0, 1]$. *Further,* $e_{n,\theta} = re^{i\theta}$ *where* r *is the positive solution to* $r\sin(n\theta) - r^n \sin(\theta) = \sin((n-1)\theta)$, *and*

$$\alpha = \frac{r\sin(n\theta)}{\sin((n-1)\theta)}.$$

(ii) *If* $\theta \in [2\pi k_u/j_u, \pi)$, *then* $e_{n,\theta}$ *is a root of* $\lambda^{j_u} - \alpha\lambda^{j_u-2} - (1-\alpha)$ *for some* $\alpha \in [0, 1]$. *Further,* $e_{n,\theta} = re^{i\theta}$ *where* r *is the positive solution to* $r^2 \sin(j_u\theta) - r^{j_u}\sin(2\theta) = \sin((j_u - 2)\theta)$ *and*

$$\alpha = \frac{r^2\sin(j_u\theta)}{\sin((j_u-2)\theta)}.$$

(iii) *If* $\theta \in [2\pi k_s/j_s, 2\pi k_{s+1}/j_{s+1}]$ *for some* $s = 1, \cdots, u-1$, $e_{n,\theta}$ *is a root of* $\lambda^{g_s} - \alpha\lambda^{g_s - l_s} - (1-\alpha)$ *for some* $\alpha \in [0, 1]$. *Further,* $e_{n,\theta} = re^{i\theta}$ *where* r *is the positive solution to* $r^{l_s}\sin(g_s\theta) - r^{g_s}\sin(l_s\theta) = \sin((g_s - l_s)\theta)$ *and*

$$\alpha = \frac{r^{l_s}\sin(g_s\theta)}{\sin((g_s - l_s)\theta)}.$$

Proof. (i) Fix $\theta_0 \in (\pi/(n-1), 2\pi/n)$. By Lemma 2, there exists $\alpha_0 \in [0, 1]$ and $(p, q) \in E_n$ such that e_{n,θ_0} is a root of $\lambda^p - \alpha_0\lambda^q - (1-\alpha_0)$, and so by Lemma 5(i),

$$\forall\theta \in \left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right] \quad \exists\alpha \in [0, 1]$$

such that $e_{n,\theta}$ is a root of $\lambda^p - \alpha\lambda^q - (1-\alpha)$, and in particular this holds for $\theta = 2\pi/n$. Since $e_{n,2\pi/n} = e^{2\pi i/n} \notin L_{n-1}$, we must have $p = n$. Thus there exists $q \in \mathbb{N}$ with $1 \leqq q \leqq n-1$ such that

$$\forall\theta \in \left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right] \quad \exists\alpha \in [0, 1]$$

such that $e_{n,\theta}$ is a root of $\lambda^n - \alpha\lambda^q - (1-\alpha)$; by Lemma 7(i), $q = n-1$. So

$$\forall\theta \in \left(\frac{\pi}{n-1}, \frac{2\pi}{n}\right] \quad \exists\alpha \in [0, 1]$$

such that $e_{n,\theta}$ is a root of $\lambda^n - \alpha\lambda^{n-1} - (1-\alpha)$. Writing $e_{n,\theta} = re^{i\theta}$ and applying Lemma 3(i) gives the equation for r, while taking imaginary parts of $(re^{i\theta})^n = \alpha(re^{i\theta})^{n-1} + 1 - \alpha$ yields the equation for α.

(ii) For notational simplicity, we shall suppress the subscript u when referring to k_u and j_u in what follows. Fix $\theta_0 \in [2\pi k/j, \pi)$; by Lemma 2, there exist $\alpha_0 \in [0, 1]$ and $(p, q) \in E_n$ such that e_{n,θ_0} is a root of $\lambda^p - \alpha_0\lambda^q - (1-\alpha_0)$ and so by Lemma 5(i) $e_{n,2\pi k/j} = e^{2\pi ik/j}$ is a root of $\lambda^p - \alpha\lambda^q - (1-\alpha)$ for some $\alpha \in [0, 1]$. If n is odd,

then $j = n$ and we have $p = j$, since $e^{2\pi k/j} \notin L_{n-1}$. If n is even, then $j = n - 1$ and p is either j or $j + 1$ since $e^{2\pi k/j} \notin L_{n-2}$. If $p = j + 1$, then again by Lemma 5(i),

$$\forall \theta \in \left[\frac{2\pi k}{j}, \pi\right) \quad \exists \alpha \in [0, 1]$$

such that $e_{n,\theta}$ is a root of $\lambda^{j+1} - \alpha\lambda^q - (1 - \alpha)$—a contradiction to Lemma 7(ii). So in either case, we have $p = j$ and hence, applying Lemma 5(i) and Lemma 7(ii), $q = j - 2a$ for some $a \in \mathbb{N}$ such that $j > 2a \geqq 2$.

Thus there exists $a \in \mathbb{N}$ with $j > 2a \geqq 2$ such that

$$\forall \theta \in \left[\frac{2\pi k}{j}, \pi\right) \quad \exists \alpha \in [0, 1]$$

such that $e_{n,\theta}$ is a root of $\lambda^j - \alpha\lambda^{j-2a} - (1 - \alpha)$. For each such a and θ we have $\sin(j\theta) \geqq 0$, $\sin(2a\theta) < 0$, and $\sin((j - 2a)\theta) > 0$. So for each such a and θ, $*$ holds in Lemma 3(ii) and there exists $r_a(\theta)$, a differentiable function on $[2\pi k/j, \pi)$ with values in $(0, 1]$ such that $\otimes$ $r_a^{2a}(\theta)\sin(j\theta) - r_a^j(\theta)\sin(2a\theta) = \sin((j - 2a)\theta)$. Further, if $\arg(\lambda) = \theta \in [2\pi k/j, \pi)$ and $\lambda^j = \alpha\lambda^{j-2a} + 1 - \alpha$ for some $\alpha \in [0, 1]$, then $\lambda = r_a(\theta)e^{i\theta}$. Applying the Implicit Function Theorem to $\otimes$ and using the fact that $r_a(2\pi k/j) = 1$ (which is easily checked by using $\otimes$), we have

$$r_a'\left(\frac{2\pi k}{j}\right) = \frac{1 - \cos\left(\frac{4\pi ka}{j}\right)}{\sin\left(\frac{4\pi ka}{j}\right)},$$

so that when $\varepsilon > 0$ is sufficiently small,

$$r_a\left(\frac{2\pi k}{j} + \varepsilon\right) = 1 + \varepsilon\left(\frac{1 - \cos\left(\frac{4\pi ka}{j}\right)}{\sin\left(\frac{4\pi ka}{j}\right)}\right) + o(\varepsilon)$$

for any such a.

Writing $2\pi k/j = \pi - (\pi/j)$, we have $4\pi ka/j = 2\pi a - (2\pi a/j)$ so that $(4\pi ka/j) \bmod (2\pi) \in (\pi, 2\pi)$. Since $(1 - \cos(x))/\sin(x)$ is negative and strictly increasing for $x \in (\pi, 2\pi)$, we see that

$$\frac{1 - \cos\left(\frac{4\pi k}{j}\right)}{\sin\left(\frac{4\pi k}{j}\right)} > \frac{1 - \cos\left(\frac{4\pi ka}{j}\right)}{\sin\left(\frac{4\pi ka}{j}\right)}$$

for any $a \geqq 2$. Thus when $\varepsilon > 0$ is sufficiently small,

$$r_1\left(\frac{2\pi k}{j} + \varepsilon\right) > r_a\left(\frac{2\pi k}{j} + \varepsilon\right) \quad \forall a \in \mathbb{N}$$

with $j > 2a > 2$. Consequently, $e_{n,(2\pi k/j)+\varepsilon}$ must be a root of $\lambda^j - \alpha\lambda^{j-2} - (1 - \alpha)$ for some $\alpha \in [0, 1]$ and so by Lemma 5(i),

$$\forall \theta \in [2\pi k/j, \pi) \quad \exists \alpha \in [0, 1]$$

such that $e_{n,\theta}$ is a root of $\lambda^j - \alpha\lambda^{j-2} - (1-\alpha)$, and $e_{n,\theta} = r_1(\theta)e^{i\theta}$. We have already seen that r_1 satisfies the desired equation, and taking imaginary parts of $(r_1 e^{i\theta})^j = \alpha(r_1 e^{i\theta})^{j-2} + 1 - \alpha$ yields the equation for α.

(iii) For notational simplicity, we shall suppress the subscript s when referring to g_s, l_s, d_s, and m_s in what follows. Fix θ_0 strictly between $2\pi d/g$ and $2\pi m/l$. By Lemma 2, there exists $\beta \in [0, 1]$ and $(p, q) \in E_n$ such that e_{n,θ_0} is a root of $\lambda^p - \beta\lambda^{p-q} - (1-\beta)$ and so by Lemma 5(i), there exists $\alpha_0, \alpha_1 \in [0, 1]$ such that $e^{2\pi id/g}$ is a root of $\lambda^p - \alpha_0\lambda^{p-q} - (1-\alpha_0)$ and $e^{2\pi m/l}$ is a root of $\lambda^p - \alpha_1\lambda^{p-q} - (1-\alpha_1)$; by Lemma 5(ii), (α_0, α_1) is either $(0, 1)$ or $(1, 0)$. If $(\alpha_0, \alpha_1) = (0, 1)$, then $g \mid p$ and $l \mid q$. Since

$$2 > 1 + \frac{l}{g} > \frac{n}{g} \geqq \frac{p}{g}$$

(recall that $g + l > n$ by Fact (i)), we must have $g = p$ so that $(p, q) = (g, al)$ for some $a \in \mathbb{N}$ such that $1 \leqq al < g$. If $(\alpha_0, \alpha_1) = (1, 0)$ then $g \mid q$ and $l \mid p$. As above,

$$2 > 1 + \frac{l}{g} > \frac{q}{g},$$

so we must have $g = q$ and hence $(p, q) = (al, g)$ for some $a \in \mathbb{N}$ such that $n \geqq al > g$.

Thus the polynomials which are candidates to have e_{n,θ_0} as a root are (1) $\lambda^g - \alpha\lambda^{g-al} - (1-\alpha)$, $\alpha \in [0, 1]$ for $a \in \mathbb{N}$ with $1 \leqq al > g$, and (2) $\lambda^{al} - \alpha\lambda^{al-g} - (1-\alpha)$, $\alpha \in [0, 1]$ for $a \in \mathbb{N}$ with $g < al \leqq n$. Note that if $al > g$ and $(am, al) = (2, 6)$ (recall that this was the exception in Corollary 6.1), then $g = 4$, $d = 1$, and $n = 6$; the root, $\lambda(\alpha)$, of $\lambda^6 - \alpha\lambda^2 - (1-\alpha)$, which has argument in $(\pi/2, 2\pi/3]$ cannot be on ∂L_n since, as remarked after Corollary 6.1,

$$\lambda(\alpha) \to \frac{i}{\sqrt{2}} \neq e_{n,\pi/2} \quad \text{as } \alpha \to \frac{3^-}{4}.$$

Having disposed of this special case, we will not consider it further.

Applying Corollary 6.1 to the remaining candidates of the form (1) and (2), we see that for each $a \in \mathbb{N}$ with $n \geqq al \geqq 1$ and each θ between $2\pi d/g$ and $2\pi m/l$, there exists $r_a(\theta)$, a differentiable function taking values in $(0, 1]$ such that $\oplus\ r_a^{al}(\theta)\sin(g\theta) - r_a^g(\theta)\sin(al\theta) = \sin((g-al)\theta)$. Further, if $\arg(\lambda) = \theta$ for such a θ and λ is a root of a polynomial of type (1) or (2) for some $\alpha \in [0, 1]$, then $\lambda = r_a(\theta)e^{i\theta}$. We recall that $r_a(2\pi d/g) = 1$ (by Corollary 6.1) and using this fact in our application of the Implicit Function Theorem to $\oplus$ we have

$$r_a'\left(\frac{2\pi d}{g}\right) = \frac{1 - \cos\left(\frac{2\pi d}{g} al\right)}{\sin\left(\frac{2\pi d}{g} al\right)}.$$

If $2\pi d/g < 2\pi am/al$, then using Lemma 6(ii) and writing $2\pi d/g = 2\pi am/al - \phi$ gives

$$2\pi am > \frac{2\pi d}{g} al = 2\pi am - al\phi > 2\pi am - \pi.$$

Since $(1 - \cos(x))/\sin(x)$ is negative and increasing for $x \in (\pi, 2\pi)$, we see that

$$r_1'\left(\frac{2\pi d}{g}\right) > r_a'\left(\frac{2\pi d}{g}\right)$$

for any $a > 1$ with $n \geqq al > 1$. Thus if $\varepsilon > 0$ is sufficiently small,

$$r_1\left(\frac{2\pi d}{g}+\varepsilon\right) > r_a\left(\frac{2\pi d}{g}+\varepsilon\right)$$

for any $a > 1$ with $n \geqq al > 1$. Similarly, if

$$\frac{2\pi d}{g} > \frac{2\pi am}{al},$$

writing

$$\frac{2\pi d}{g} = \frac{2\pi am}{al} + \phi$$

and using Lemma 6(i) yields

$$2\pi am < 2\pi am + al\phi = \frac{2\pi d}{g} al < 2\pi am + \pi.$$

Since $(1 - \cos(x))/\sin(x)$ is positive and increasing for $x \in (0, \pi)$, we have

$$r_1'\left(\frac{2\pi d}{g}\right) < r_a'\left(\frac{2\pi d}{g}\right)$$

for any $a > 1$ with $n \geqq al > 1$, and hence when $\varepsilon > 0$ is sufficiently small,

$$r_1\left(\frac{2\pi d}{g}-\varepsilon\right) > r_a\left(\frac{2\pi d}{g}-\varepsilon\right)$$

for any such a.

Thus when θ is strictly between $2\pi d/g$ and $2\pi m/l$ and sufficiently close to $2\pi d/g$, $e_{n,\theta}$ must be a root of $\lambda^g - \alpha\lambda^{g-l} - (1 - \alpha)$ for some $\alpha \in [0, 1]$; hence by Lemma 5(i) for any θ between $2\pi d/g$ and $2\pi m/l$, there exists $\alpha \in [0, 1]$ such that $e_{n,\theta}$ is a root of $\lambda^g - \alpha\lambda^{g-l} - (1 - \alpha)$ and $e_{n,\theta} = r_1(\theta)e^{i\theta}$. We have already seen that r_1 satisfies the desired equation, and taking imaginary parts of $(r_1(\theta)e^{i\theta})^g = \alpha(r_1(\theta)e^{i\theta})^{g-l} + 1 - \alpha$ gives the equation for α. □

COROLLARY 4.1. *Fix $n \geqq 3$ and let*

$$\frac{2\pi k_s}{j_s} \in A_n, \quad s = 1, \cdots, u.$$

Let $R_0 = \{\rho e^{i\theta} | \theta \in (\pi/(n-1), 2\pi/n], \rho \in [0, r_0(\theta)]$, where $r_0 > 0$ solves $r\sin(n\theta) - r^n \sin(\theta) = \sin((n-1)\theta)\}$; let $R_s = \{\rho e^{i\theta} | \theta \in [2\pi k_s/j_s, 2\pi k_{s+1}/j_{s+1}], \rho \in [0, r_s(\theta)]$, where $r_s > 0$ solves $r^{l_s}\sin(g_s\theta) - r^{g_s}\sin(l_s\theta) = \sin((g_s - l_s)\theta)\}$, $s = 1, \cdots, u-1$; let $R_u = \{\rho e^{i\theta} | \theta \in [2\pi k_u/j_u, \pi), \rho \in [0, r_u(\theta)]$, where $r_u > 0$ solves $r^2\sin(j_u\theta) - r^{j_u}\sin(2\theta) = \sin((j_u - 2)\theta)\}$; and let $\overline{R_s} = \{\bar{z} | z \in R_s\}$, $s = 0, \cdots, u$. Then

$$L_n = \left(\bigcup_{s=0}^{u} R_s\right) \cup \left(\bigcup_{s=0}^{u} \overline{R_s}\right) \cup [-1, 0].$$

Proof. The result follows from Theorem 1, Theorem 4, and the symmetry of L_n with respect to the real axis. □

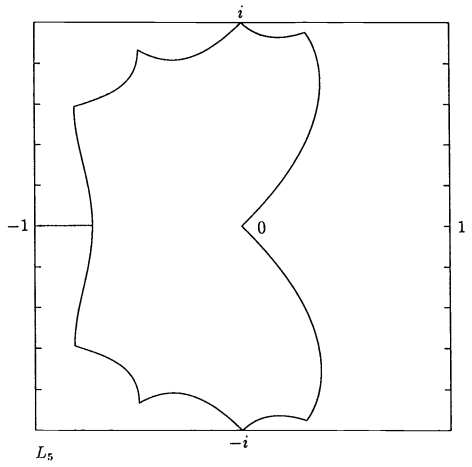


FIG. 7

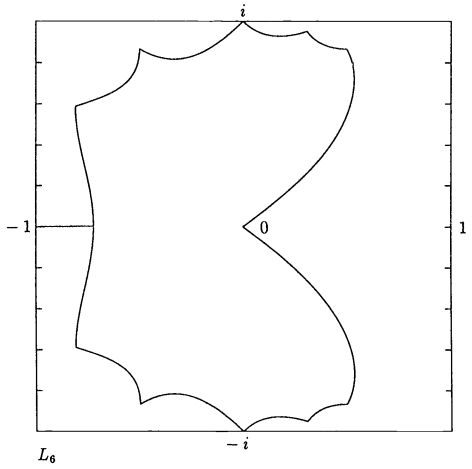


FIG. 8

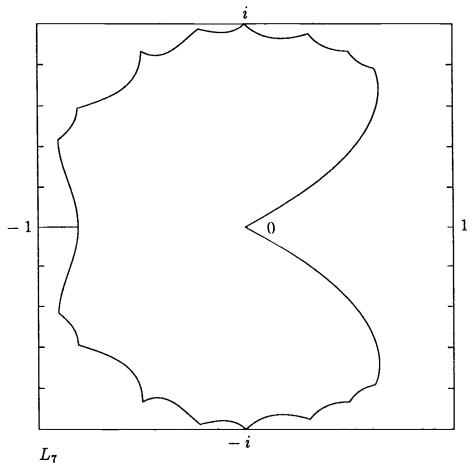


FIG. 9

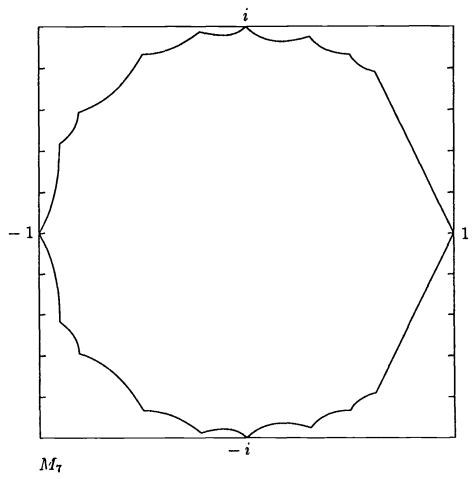


FIG. 10

So we really have accomplished the task of describing L_n, for it is now possible to pick an arbitrary n and construct L_n as follows. Knowing n, we can generate the successive elements of A_n by starting with $2\pi k_1/j_1 = 2\pi/n$ and using Fact (ii) concerning Farey Series (which is stated just after Definition 8) repeatedly. This gives us $2\pi k_s/j_s$, $s = 1, \cdots, u$ and hence the regions R_s and $\overline{R_s}$, $s = 0, \cdots, u$, which, along with $[-1, 0]$, form the set L_n. In Figs. 7–10, we present L_5, L_6, L_7, and M_7 for the reader's consideration.

Recall that the motivation for determining L_n came from looking at eigenvalues of Leslie matrices. Specifically, given a Leslie matrix $\mathbf{L}$ that has ρ as its positive eigenvalue, say, we are interested in λ/ρ, $\lambda \in \sigma(\mathbf{L}) - \{\rho\}$, since these quantities determine the convergence (if any) of the successive age distributions. Plotting these λ/ρ in the plane gives us an idea of how these complex numbers sit inside the unit disc, but our knowledge of L_n also enables us to see how these quantities compare to those of other Leslie matrices; we can see not only how quickly our age distributions are converging (by looking at $|\lambda/\rho|$), but also how this convergence compares to the range of possibilities available to Leslie matrices of the same order. Thus L_n provides us with a sharp frame of reference in which to view the λ/ρ by giving precise information about which values of λ/ρ are admitted by Leslie matrices of order n.

Acknowledgments. I would like to thank Professor Chandler Davis of the University of Toronto and Professor Beresford Parlett of the University of California, Berkeley, both of whom made a number of helpful comments concerning this material when it was presented in my Ph.D. thesis. I would also like to thank the referees for suggesting improvements to this manuscript.

REFERENCES

[1] N. A. DMITRIEV AND E. DYNKIN, *On the characteristic roots of stochastic matrices*, Izv. Akad. Nauk SSSR Ser. Mat., 10 (1946), pp. 167–184.

[2] C. R. JOHNSON, R. B. KELLOG, AND A. B. STEPHENS, *Complex eigenvalues of a nonnegative matrix with a specified graph* II, Linear and Multilinear Algebra, 7 (1979), pp. 129–143.

[3] F. I. KARPELEVICH, *On the characteristic roots of matrices with nonnegative elements*, Izv. Akad. Nauk SSSR Ser. Mat., 15 (1951), pp. 361–383.
[4] N. KEYFITZ, *Introduction to the Mathematics of Population with Revisions*, Addison–Wesley, Reading, MA, 1977.
[5] P. H. LESLIE, *On the use of matrices in certain population mathematics*, Biometrika, 33 (1945), pp. 183–212.
[6] E. G. LEWIS, *On the generation and growth of a population*, Sankhya, 6 (1942), pp. 93–96.
[7] J. H. POLLARD, *Mathematical Models for the Growth of Human Populations*, Cambridge University Press, Cambridge, U.K., 1973.
[8] J. ROBERTS, *Elementary Number Theory: A Problem Oriented Approach*, MIT Press, Massachusetts Institute of Technology, Cambridge, MA, 1977.
[9] E. SENETA, *Non-Negative Matrices and Markov Chains*, Springer-Verlag, New York, 1981.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 530–549, April 1992

007

ON INVERSION OF SYMMETRIC TOEPLITZ MATRICES*

LEIBA RODMAN† AND TAMIR SHALOM‡

Abstract. It is shown that the inverse of a symmetric Toeplitz matrix is determined by at most two of its columns, when properly chosen. A formula for the inverse matrix is given in terms of these columns, generalizing the version of the Gohberg–Semencul formula for the symmetric case. Similar results for the Hermitian case are also given.

Key words. symmetric Toeplitz matrices, Hermitian Toeplitz matrices, inversion

AMS(MOS) subject classifications. 15A09, 15A57

1. Introduction.

Let A be a symmetric Toeplitz matrix:

$$A=\begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{k-1} \\ a_1 & a_0 & a_1 & \cdots & a_{k-2} \\ a_2 & a_1 & a_0 & \cdots & a_{k-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{k-1} & a_{k-2} & a_{k-3} & \cdots & a_0 \end{pmatrix},$$

where $a_0, a_1, \cdots, a_{k-1}$ are complex numbers. General, not necessarily symmetric, Toeplitz matrices play an important role in many applications. Therefore, algorithms (starting with [L]) and formulae (starting with [GS]) for inversion of such matrices have been extensively developed. A partial list of further developments of related algorithms includes [T], [BGY], and some of the generalizations of [GS] appear in [GK], [HR], [BS], and [KC].

In this paper we study the invertibility of A and produce formulae for A^{-1} in terms of the solutions of linear equations of the form

$$AX=E,$$

where E is a column vector which belongs to the standard basis in $\mathbb{C}^n$. Namely, we obtain formulae for A^{-1} in terms of a minimal number of its columns. Problems of this type (for general nonsymmetric Toeplitz matrices) were studied first in [GS] and later in [GK] and [BS]. In particular, the celebrated Gohberg–Semencul formula is in this category. The symmetric version of the theorem of Gohberg and Semencul [GS] states, in particular, that the inverse of a symmetric Toeplitz matrix can sometimes be represented via its first column. Hereafter, we use $\delta_{i,j}$ to denote the Kronecker index, namely,

$$\delta_{i,j}=\begin{cases} 1 & \text{if } i=j, \\ 0 & \text{if } i\neq j. \end{cases}$$

THEOREM 1.1 (Gohberg and Semencul). *Let $A=(a_{i-j})_{i,j=0}^{k-1}$ be a symmetric Toeplitz matrix, i.e., $a_{-p}=a_p$ for $p=0, 1, \cdots, k-1$. If the following system of equations*

$$\sum_{j=0}^{k-1} a_{i-j}x_j=\delta_{0,i}, \qquad (i=0,1,\cdots,k-1),$$

* Received by the editors January 17, 1990; accepted for publication (in revised form) November 16, 1990.

† Department of Mathematics, The College of William and Mary, Williamsburg, Virginia 23187-8795 (lxrodm@wmmvs.bitnet). This author's research was partially supported by National Science Foundation grant DMS-8802836 and United States–Israel Binational Science Foundation grant 88-00304/1.

‡ Department of Computer Science, Columbia University, New York, New York 10027 (shalom@cs.columbia.edu). This research was done while visiting the Department of Mathematics, The College of William and Mary, Williamsburg, Virginia 23187.

is solvable and $x_0 \neq 0$, *then* A *is invertible and*

$$A^{-1} = \frac{1}{x_0}\left[\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} x_0 & x_1 & \cdots & x_{k-1} \\ 0 & x_0 & \cdots & x_{k-2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_0 \end{pmatrix}\right. \tag{1.1}$$
$$\left. - \begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_{k-1} & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_1 & \cdots & x_{k-1} & 0 \end{pmatrix}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\right].$$

However, this is not always the case. It might happen that an invertible symmetric Toeplitz matrix is not determined by a single column of its inverse. Indeed, the matrix

$$A = \begin{pmatrix} 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

is an invertible symmetric Toeplitz matrix with

$$A^{-1} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & 1 \\ 1 & -1 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}.$$

Note that

$$\begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix},$$

which has the same first and last columns as A^{-1} while

$$\begin{pmatrix} -1 & -1 & 0 & 0 \\ -1 & -1 & -1 & 0 \\ 0 & -1 & -1 & -1 \\ 0 & 0 & -1 & -1 \end{pmatrix}^{-1} = \begin{pmatrix} -1 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \\ 1 & -1 & 0 & 0 \\ -1 & 1 & 0 & -1 \end{pmatrix},$$

which has the same second and third columns as A^{-1}.

Let us remark that the same matrix A was given in [GK] as an example of a Toeplitz matrix whose inverse cannot be determined (in the class of inverses of general, not necessarily symmetric, Toeplitz matrices) by any pair of its columns.

Ben-Artzi and Shalom have shown in [BS] that three columns of the inverse of a general (not necessarily symmetric) Toeplitz matrix are always enough to reconstruct it.

THEOREM 1.2 (Ben-Artzi and Shalom). *Let* $A = (a_{i-j})_{i,j=0}^{k-1}$ *be a Toeplitz matrix. If each of the following system of equations*

$$\sum_{j=0}^{k-1} a_{i-j}x_j = \delta_{0,i}, \qquad (i = 0, 1, \cdots, k-1),$$

$$\sum_{j=0}^{k-1} a_{i-j}y_j = \delta_{k-1-p,i}, \qquad (i = 0, 1, \cdots, k-1),$$

$$\sum_{j=0}^{k-1} a_{i-j}z_j = \delta_{k-p,i}, \qquad (i = 0, 1, \cdots, k-1),$$

is solvable and $x_p \neq 0$, *then* A *is invertible and*

$$A^{-1}=\frac{1}{x_p}\left[\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} y_{k-1} & y_{k-2}-z_{k-1} & \cdots & y_0-z_1 \\ 0 & y_{k-1} & \cdots & y_1-z_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & y_{k-1} \end{pmatrix}\right. \tag{1.2}$$
$$\left.+\begin{pmatrix} z_0 & \cdots & 0 & 0 \\ z_1-y_0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ z_{k-1}-y_{k-2} & \cdots & z_1-y_0 & z_0 \end{pmatrix}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\right].$$

The main result of this paper shows that at most two columns of the inverse of a symmetric Toeplitz matrix, when properly chosen, are always enough to represent the inverse matrix, hence, to determine the matrix itself.

THEOREM 1.3. *Let* $A=(a_{i-j})_{i,j=0}^{k-1}$ *be a symmetric Toeplitz matrix, i.e.,* $a_{-p}=a_p$ *for* $p=0, 1, \cdots, k-1$. *The matrix is invertible if and only if the following conditions hold.*

(a) *There exists a solution for the following system of equations*:

$$\sum_{j=0}^{k-1} a_{i-j}x_j=\delta_{0,i}, \qquad (i=0,1,\cdots,k-1).$$

(b) *Let* p *be such that* $x_p \neq 0$ *and* $x_q=0$ *for all* $q<p$; *then* $p \leqq k/2$ *and* $x_q=0$ *for all* $q>k-p$.

(c) *There exists a solution for the following system of equations*

$$\sum_{j=0}^{k-1} a_{i-j}y_{k-1-j}=\delta_{p,i}, \qquad (i=0,1,\cdots,k-1).$$

In case $p=0$, *then*

$$A^{-1}=\frac{1}{x_0}\left[\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} x_0 & x_1 & \cdots & x_{k-1} \\ 0 & x_0 & \cdots & x_{k-2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_0 \end{pmatrix}\right. \tag{1.3}$$
$$\left.-\begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_{k-1} & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_1 & \cdots & x_{k-1} & 0 \end{pmatrix}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\right].$$

In case $p>0$, *then* $x_{k-q}=\alpha x_q$ *for* $q=1, 2, \cdots, k-1$ *with* $\alpha=1$ *or* $\alpha=-1$ *and*

$$A^{-1}=\frac{1}{x_p}\left[\begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k-1} & \cdots & x_1 & 0 \end{pmatrix}\begin{pmatrix} y_{k-1} & y_{k-2}-z_{k-1} & \cdots & y_0-z_1 \\ 0 & y_{k-1} & \cdots & y_1-z_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & y_{k-1} \end{pmatrix}\right.$$
$$+\begin{pmatrix} y_{k-1} & 0 & \cdots & 0 \\ y_{k-2}-z_{k-1} & y_{k-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ y_0-z_1 & y_1-z_2 & \cdots & y_{k-1} \end{pmatrix}\begin{pmatrix} 0 & x_1 & \cdots & x_{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_1 \\ 0 & 0 & \cdots & 0 \end{pmatrix} \tag{1.4}$$
$$\left.-\frac{y_{k-p-1}}{x_p}\begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k-1} & \cdots & x_1 & 0 \end{pmatrix}\begin{pmatrix} 0 & x_1 & \cdots & x_{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_1 \\ 0 & 0 & \cdots & 0 \end{pmatrix}\right]$$

for

$$\begin{pmatrix} z_{k-1} \\ z_{k-2} \\ \vdots \\ z_0 \end{pmatrix} = \frac{1}{x_p} \begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k-1} & \cdots & x_1 & 0 \end{pmatrix} \begin{pmatrix} y_{k-p} \\ \vdots \\ y_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{1.5}$$

The result of Theorem 1.3, together with its proof, holds verbatim for symmetric Toeplitz matrices over any (commutative) field.

Now let us consider a Hermitian Toeplitz matrix that is a matrix of the form

$$A = \begin{pmatrix} a_0 & \bar{a}_1 & \bar{a}_2 & \cdots & \bar{a}_{k-1} \\ a_1 & a_0 & \bar{a}_1 & \cdots & \bar{a}_{k-2} \\ a_2 & a_1 & a_0 & \cdots & \bar{a}_{k-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{k-1} & a_{k-2} & a_{k-3} & \cdots & a_0 \end{pmatrix},$$

where a_0 is a real number and $a_1, a_2, \cdots, a_{k-1}$ are complex numbers. Hereafter, $\bar{a}$ denotes the complex conjugate of the complex number a and $|a|$ denotes its absolute value. The following result is the analogue of Theorem 1.3 for the Hermitian case.

THEOREM 1.4. *Let $A = (a_{i-j})_{i,j=0}^{k-1}$ be a Hermitian Toeplitz matrix, i.e., $a_{-p} = \bar{a}_p$ for $p = 0, 1, \cdots, k-1$. The matrix is invertible if and only if the following conditions hold.*

(a) *There exists a solution for the following system of equations*:

$$\sum_{j=0}^{k-1} a_{i-j} x_j = \delta_{0,i}, \qquad (i = 0, 1, \cdots, k-1).$$

(b) *Let p be such that $x_p \neq 0$ and $x_q = 0$ for all $q < p$; then $p \leqq k/2$ and $x_q = 0$ for all $q > k - p$.*

(c) *There exists a solution for the following system of equations*:

$$\sum_{j=0}^{k-1} a_{i-j} y_{k-1-j} = \delta_{p,i}, \qquad (i = 0, 1, \cdots, k-1).$$

(*Note that $y_{k-p-1} = (A^{-1})_{p,p}$ and hence it is a real number.*)
In case $p = 0$, then

$$A^{-1} = \frac{1}{x_0} \left[\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix} \begin{pmatrix} x_0 & \bar{x}_1 & \cdots & \bar{x}_{k-1} \\ 0 & x_0 & \cdots & \bar{x}_{k-2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_0 \end{pmatrix} - \begin{pmatrix} 0 & \cdots & 0 & 0 \\ \bar{x}_{k-1} & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \bar{x}_1 & \cdots & \bar{x}_{k-1} & 0 \end{pmatrix} \begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix} \right]. \tag{1.6}$$

In case $p > 0$, then $x_{k-q} = \alpha \bar{x}_q$ for $q = 1, 2, \cdots, k-1$ with a constant complex number α such that $|\alpha| = 1$ and

$$A^{-1} = \frac{1}{x_p}\begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k-1} & \cdots & x_1 & 0 \end{pmatrix}\begin{pmatrix} \bar{y}_{k-1} & \bar{y}_{k-2} - \bar{z}_{k-1} & \cdots & \bar{y}_0 - \bar{z}_1 \\ 0 & \bar{y}_{k-1} & \cdots & \bar{y}_1 - \bar{z}_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{y}_{k-1} \end{pmatrix}$$

$$+\frac{1}{\bar{x}_p}\begin{pmatrix} y_{k-1} & 0 & \cdots & 0 \\ y_{k-2} - z_{k-1} & y_{k-1} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ y_0 - z_1 & y_1 - z_2 & \cdots & y_{k-1} \end{pmatrix}\begin{pmatrix} 0 & \bar{x}_1 & \cdots & \bar{x}_{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{x}_1 \\ 0 & 0 & \cdots & 0 \end{pmatrix} \tag{1.7}$$

$$-\frac{y_{k-p-1}}{|x_p|^2}\begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k-1} & \cdots & x_1 & 0 \end{pmatrix}\begin{pmatrix} 0 & \bar{x}_1 & \cdots & \bar{x}_{k-1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{x}_1 \\ 0 & 0 & \cdots & 0 \end{pmatrix}$$

for

$$\begin{pmatrix} z_{k-1} \\ z_{k-2} \\ \vdots \\ z_0 \end{pmatrix} = \frac{1}{x_p}\begin{pmatrix} 0 & \cdots & 0 & 0 \\ x_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ x_{k-1} & \cdots & x_1 & 0 \end{pmatrix}\begin{pmatrix} \bar{y}_{k-p} \\ \vdots \\ \bar{y}_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{1.8}$$

Note that in case $p = 0$, Theorems 1.3 and 1.4 state the result of Gohberg and Semencul for the symmetric and the Hermitian cases, respectively.

The paper consists of four additional sections. In § 2 we give some notation and preliminaries, in § 3 we prove Theorem 1.3, and in § 4 we prove Theorem 1.4. In § 5 we present some examples.

2. Notation and preliminaries. We denote the row with entries $b_1, b_2, \cdots, b_s$ either by $row(b_1, b_2, \cdots, b_s)$ or by $row(b_j)_{j=1}^{s}$. The column with entries $b_1, b_2, \cdots, b_s$ is denoted either by $col(b_1, b_2, \cdots, b_s)$ or by $col(b_i)_{i=1}^{s}$. Let $E^{(0)}, E^{(1)}, \cdots, E^{(k-1)}$ and $F^{(0)}, F^{(1)}, \cdots, F^{(k-1)}$ be the k unit columns and unit rows, i.e.,

$$E^{(j)} = col(\delta_{i,j})_{i=0}^{k-1}, \qquad (j = 0, 1, \cdots, k-1),$$

and

$$F^{(i)} = row(\delta_{i,j})_{j=0}^{k-1}, \qquad (i = 0, 1, \cdots, k-1).$$

For any matrix A we denote its transposed matrix by A^T. We denote by S and by J the following $k \times k$ matrices:

$$S = \begin{pmatrix} 0 & 0 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & 0 \\ 0 & 0 & \cdots & 1 & 0 \end{pmatrix}, \qquad J = \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 \\ 0 & 0 & \cdots & 1 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 1 & \cdots & 0 & 0 \\ 1 & 0 & \cdots & 0 & 0 \end{pmatrix}.$$

LEMMA 2.1. *If A is a symmetric Toeplitz matrix and*

$$A\, col(b_i)_{i=0}^{k-1} = col(c_i)_{i=0}^{k-1},$$

then

$$A\ col(b_{k-1-i})_{i=0}^{k-1} = col(c_{k-1-i})_{i=0}^{k-1},$$

$$row(b_j)_{j=0}^{k-1} A = row(c_j)_{j=0}^{k-1},$$

and

$$row(b_{k-1-j})_{j=0}^{k-1} A = row(c_{k-1-j})_{j=0}^{k-1}.$$

This well-known lemma follows from the fact that $JAJ = A$ for any symmetric Toeplitz matrix A.

For any $k \times k$ matrix $D = (d_{i,j})_{i,j=0}^{k-1}$ we denote by ΔD the $(k+1) \times (k+1)$ matrix defined by

$$\Delta D = (d_{i,j} - d_{i-1,j-1})_{i,j=0}^{k},$$

with $d_{i,j} = 0$ if $i \notin \{0, 1, \cdots, k-1\}$ or $j \notin \{0, 1, \cdots, k-1\}$. We shall use the following result, which appears in [HR].

LEMMA 2.2 (Heinig and Rost). *Let A be a nonsingular Toeplitz matrix; then*

$$\operatorname{rank}\Delta(A^{-1}) = 2.$$

The proof follows essentially from

$$SA - AS = SAE^{(k-1)}F^{(k-1)} - E^{(0)}F^{(0)}AS,$$

which holds for any Toeplitz matrix A, and from

$$\Delta(A^{-1}) = \begin{pmatrix} A^{-1}E^{(0)} & A^{-1}S - SA^{-1} \\ 0 & F^{(k-1)}A^{-1} \end{pmatrix}.$$

It is proved in [HR] that the converse of Lemma 2.2 holds as well, however, the converse statement is not used in this paper.

3. Proof of Theorem 1.3. We divide the proof into two parts. In the necessity part, we show that if A is an invertible symmetric matrix, then the conditions hold and A^{-1} admits the stated formula.

Proof of necessity. We assume that A is an invertible symmetric Toeplitz matrix. It is clear that if we define $col(x_i)_{i=0}^{k-1}$ and $col(y_i)_{i=0}^{k-1}$ by

$$col(x_i)_{i=0}^{k-1} = A^{-1}E^{(0)}, \qquad col(y_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p)},$$

where p is such that $x_p \neq 0$ and $x_q = 0$ for all $q < p$, then these are the solutions for the given systems of equations. Let us consider the cases of $p = 0$ and of $p > 0$ separately.

It is clear that if $p = 0$, then $y_{k-1-i} = x_i$ for $i = 0, 1, \cdots, k-1$. In this case, condition b holds trivially and the formula for A^{-1} follows from Theorem 1.2, noting that $z_i = 0$ in this case.

From now on we assume that $p > 0$. It follows from Lemma 2.1 that A^{-1} is of the form

$$A^{-1} = \begin{pmatrix} x_0 & x_1 & \cdots & x_{k-2} & x_{k-1} \\ x_1 & & & & x_{k-2} \\ \vdots & & * & & \vdots \\ x_{k-2} & & & & x_1 \\ x_{k-1} & x_{k-2} & \cdots & x_1 & x_0 \end{pmatrix}, \tag{3.1}$$

and hence,

$$\Delta(A^{-1}) = \begin{pmatrix} x_0 & x_1 & \cdots & x_{k-1} & 0 \\ x_1 & & & & -x_{k-1} \\ \vdots & & * & & \vdots \\ x_{k-1} & & & & -x_1 \\ 0 & -x_{k-1} & \cdots & -x_1 & -x_0 \end{pmatrix}.$$

It follows from Lemma 2.2 that $\operatorname{rank}\Delta(A^{-1}) = 2$ and hence the determinant of any 3×3 submatrix of $\Delta(A^{-1})$ must equal zero. Let q be any integer such that $k - p < q < k$, then

$$0 = \det \begin{pmatrix} x_0 & x_{k-q} & x_p \\ x_p & * & * \\ 0 & -x_q & -x_{k-p} \end{pmatrix} = \det \begin{pmatrix} 0 & 0 & x_p \\ x_p & * & * \\ 0 & -x_q & -x_{k-p} \end{pmatrix},$$

as $k - q < p$. Now, $x_p \neq 0$, therefore, $x_q = 0$. We have just proved that $x_q = 0$ for all $q > k - p$, but $x_p \neq 0$, hence, $p \leqq k - p$ or $p \leqq k/2$.

Define $col(z_i)_{i=0}^{k-1} = A^{-1}E^{(k-p)}$. Then by Lemma 2.1, $col(z_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p-1)}$. Now we apply (1.2) and obtain

$$col(z_{k-1-i})_{i=0}^{k-1} = \frac{1}{x_p}\left[\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} y_{k-p} - z_{k-p+1} \\ \vdots \\ y_{k-2} - z_{k-1} \\ y_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}\right.$$
$$\left. + \begin{pmatrix} z_0 & \cdots & 0 & 0 \\ z_1 - y_0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ z_{k-1} - y_{k-2} & \cdots & z_1 - y_0 & z_0 \end{pmatrix}\begin{pmatrix} x_{k-p+1} \\ \vdots \\ x_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}\right].$$

Note that the second summand is zero as $x_q = 0$ for $q > k - p$. Moreover, $z_q = 0$ for $q \geqq k - p$ as $x_0 = x_1 = \cdots = x_{p-1} = 0$. This proves (1.5).

Now, it is clear that

$$\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 & x_0 & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots & x_1 & x_0 & \cdots & 0 \\ 0 & 0 & \cdots & x_{k-1} & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}$$
$$\times \begin{pmatrix} -z_0 & \cdots & 0 & 0 \\ y_0 - z_1 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ y_{k-2} - z_{k-1} & \cdots & y_0 - z_1 & -z_0 \\ y_{k-1} & \cdots & y_1 - z_2 & y_0 - z_1 \\ 0 & \cdots & y_2 - z_3 & y_1 - z_2 \\ \vdots & \ddots & \vdots & \vdots \\ 0 & \cdots & 0 & y_{k-1} \end{pmatrix}$$

$$= \begin{pmatrix} y_{k-1} & y_{k-2}-z_{k-1} & \cdots & y_0-z_1 & -z_0 & \cdots & 0 & 0 \\ 0 & y_{k-1} & \cdots & y_1-z_2 & y_0-z_1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & y_{k-1} & y_{k-2}-z_{k-1} & \cdots & y_0-z_1 & -z_0 \end{pmatrix}$$

$$\times \begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \\ 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}.$$

Therefore, it follows from (1.2) that A^{-1} admits the following representation as well.

$$A^{-1} = \frac{1}{x_p}\left[\begin{pmatrix} y_{k-1} & y_{k-2}-z_{k-1} & \cdots & y_0-z_1 \\ 0 & y_{k-1} & \cdots & y_1-z_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & y_{k-1} \end{pmatrix}\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\right.$$

(3.2)

$$\left. + \begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} z_0 & \cdots & 0 & 0 \\ z_1-y_0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ z_{k-1}-y_{k-2} & \cdots & z_1-y_0 & z_0 \end{pmatrix}\right].$$

Now, according to (3.1),

$$A^{-1}E^{(k-1)} = col(x_{k-1-i})_{i=0}^{k-1},$$

and therefore, it follows from (3.2) that

$$col(x_{k-1-i})_{i=0}^{k-1} = \frac{x_0}{x_p} col(y_i - z_{i+1})_{i=0}^{k-1} + \frac{z_0}{x_p} col(x_{i+1})_{i=0}^{k-1},$$

setting $x_k = z_k = 0$. But, $z_0 = F^{(k-1)}A^{-1}E^{(p-1)} = x_{k-p}$ and $x_0 = 0$, since $p > 0$. Consequently,

$$\begin{pmatrix} x_{k-1} \\ \vdots \\ x_1 \\ 0 \end{pmatrix} = \frac{x_{k-p}}{x_p}\begin{pmatrix} x_1 \\ \vdots \\ x_{k-1} \\ 0 \end{pmatrix}. \tag{3.3}$$

In particular $x_p = (x_{k-p}/x_p)x_{k-p}$ and hence $x_{k-p}/x_p = 1$ or $x_{k-p}/x_p = -1$.

It remains to prove (1.4). Note that $col(y_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p)}$ and therefore, it follows from (1.2) that

$$col(y_{k-1-i})_{i=0}^{k-1} = \frac{x_{k-p}}{x_p}\begin{pmatrix} z_0 \\ z_1-y_0 \\ \vdots \\ z_{k-1}-y_{k-2} \end{pmatrix}$$

$$+ \frac{1}{x_p}\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} y_{k-p-1}-z_{k-p} \\ \vdots \\ y_{k-2}-z_{k-1} \\ y_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

or,

$$col(y_i)_{i=0}^{k-1} = \frac{x_{k-p}}{x_p}\begin{pmatrix} z_{k-1}-y_{k-2} \\ \vdots \\ z_1 - y_0 \\ z_0 \end{pmatrix} + \frac{1}{x_p}\begin{pmatrix} x_0 & x_1 & \cdots & x_{k-1} \\ 0 & x_0 & \cdots & x_{k-2} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ y_{k-1} \\ y_{k-2}-z_{k-1} \\ \vdots \\ y_{k-p-1}-z_{k-p} \end{pmatrix}.$$

It follows from (3.3) that

$$col(y_i)_{i=0}^{k-1} = \frac{x_{k-p}}{x_p}\begin{pmatrix} z_{k-1}-y_{k-2} \\ \vdots \\ z_1 - y_0 \\ z_0 \end{pmatrix}$$

(3.4)

$$+\frac{1}{x_{k-p}}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ y_{k-1} \\ y_{k-2}-z_{k-1} \\ \vdots \\ y_{k-p-1}-z_{k-p} \end{pmatrix}.$$

But according to Lemma 2.1, $col(y_i)_{i=0}^{k-1} = A^{-1}E^{(k-p-1)}$, and hence, it follows from (3.2) that

$$col(y_i)_{i=0}^{k-1} = \begin{pmatrix} y_0 - z_1 \\ y_1 - z_2 \\ \vdots \\ y_{k-1} \end{pmatrix} + \frac{1}{x_p}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_0 \\ z_1 - y_0 \\ \vdots \\ z_p - y_{p-1} \end{pmatrix},$$

which, combined with (3.4) gives

$$\frac{x_{k-p}}{x_p}\begin{pmatrix} z_{k-1}-y_{k-2} \\ \vdots \\ z_1 - y_0 \\ z_0 \end{pmatrix} - \frac{1}{x_p}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_0 \\ z_1 - y_0 \\ \vdots \\ z_p - y_{p-1} \end{pmatrix}$$

$$= \begin{pmatrix} y_0 - z_1 \\ y_1 - z_2 \\ \vdots \\ y_{k-1} \end{pmatrix} - \frac{1}{x_{k-p}}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ y_{k-1} \\ y_{k-2}-z_{k-1} \\ \vdots \\ y_{k-p-1}-z_{k-p} \end{pmatrix}.$$

Note that $z_p = F^{(p)}A^{-1}E^{(k-p)} = y_{p-1}$ and $z_{k-p} = 0$, so

$$\frac{x_{k-p}}{x_p}\begin{pmatrix} z_{k-1}-y_{k-2} \\ \vdots \\ z_1-y_0 \\ z_0 \end{pmatrix} - \frac{1}{x_p}\begin{pmatrix} 0 & 0 & x_{k-1} & \cdots & x_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_0 \\ z_1-y_0 \\ \vdots \\ z_{p-1}-y_{p-2} \end{pmatrix}$$

$$= \begin{pmatrix} y_0-z_1 \\ \vdots \\ y_{k-2}-z_{k-1} \\ y_{k-1} \end{pmatrix} - \frac{1}{x_{k-p}}\begin{pmatrix} 0 & 0 & x_{k-1} & \cdots & x_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ y_{k-1} \\ y_{k-2}-z_{k-1} \\ \vdots \\ y_{k-p}-z_{k-p+1} \end{pmatrix}$$

$$- \frac{y_{k-p-1}}{x_{k-p}}\begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{k-1} \\ 0 \end{pmatrix}.$$

Define

$$w_i = y_i - z_{i+1} - \frac{y_{k-p-1}}{x_{k-p}}x_{i+1}$$

for $i = 0, 1, \cdots, k-1$ (where, as before, $x_k = z_k = 0$). Then

$$\frac{x_{k-p}}{x_p}\begin{pmatrix} z_{k-1}-y_{k-2} \\ \vdots \\ z_1-y_0 \\ z_0 \end{pmatrix} - \frac{1}{x_p}\begin{pmatrix} 0 & 0 & x_{k-1} & \cdots & x_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_0 \\ z_1-y_0 \\ \vdots \\ z_{p-1}-y_{p-2} \end{pmatrix}$$

$$= \begin{pmatrix} w_0 \\ \vdots \\ w_{k-1} \end{pmatrix} - \frac{1}{x_{k-p}}\begin{pmatrix} 0 & 0 & x_{k-1} & \cdots & x_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ w_{k-1} \\ w_{k-2} \\ \vdots \\ w_{k-p} \end{pmatrix},$$

since $x_{k-p+1} = \cdots = x_{k-2} = x_{k-1} = 0$. Equivalently,

$$\frac{x_{k-p}}{x_p}\begin{pmatrix} z_{k-1}-y_{k-2} \\ \vdots \\ z_1-y_0 \\ z_0 \end{pmatrix} - \frac{1}{x_p}\begin{pmatrix} x_{p+1} & \cdots & x_2 & & \\ \vdots & \ddots & \vdots & & \\ x_{k-1} & \cdots & x_{k-p} & & \\ 0 & \cdots & x_{k-p+1} & 0_{k\times(k-p)} & \\ \vdots & \ddots & \vdots & & \\ 0 & \cdots & x_{k-1} & & \\ 0 & \cdots & 0 & & \\ 0 & \cdots & 0 & & \end{pmatrix}\begin{pmatrix} z_0 \\ z_1-y_0 \\ \vdots \\ z_{k-1}-y_{k-2} \end{pmatrix}$$

(3.5)

$$= \begin{pmatrix} w_0 \\ \vdots \\ w_{k-1} \end{pmatrix} - \frac{1}{x_{k-p}}\begin{pmatrix} x_{p+1} & \cdots & x_2 & & \\ \vdots & \ddots & \vdots & & \\ x_{k-1} & \cdots & x_{k-p} & & \\ 0 & \cdots & x_{k-p+1} & 0_{k\times(k-p)} & \\ \vdots & \ddots & \vdots & & \\ 0 & \cdots & x_{k-1} & & \\ 0 & \cdots & 0 & & \\ 0 & \cdots & 0 & & \end{pmatrix}\begin{pmatrix} w_{k-1} \\ \vdots \\ w_0 \end{pmatrix}.$$

Note that the $k \times k$ matrix B defined by

$$B = \frac{1}{x_{k-p}}\begin{pmatrix} & x_2 & \cdots & x_{p+1} \\ & \vdots & \ddots & \vdots \\ & x_{k-p} & \cdots & x_{k-1} \\ 0_{k\times(k-p)} & x_{k-p+1} & \cdots & 0 \\ & \vdots & \ddots & \vdots \\ & x_{k-1} & \cdots & 0 \\ & 0 & \cdots & 0 \\ & 0 & \cdots & 0 \end{pmatrix}$$

is upper triangular with zero diagonal as $x_q = 0$ for $q > k - p$. But (3.5) implies

$$[I_k - B]\left[\frac{x_{k-p}}{x_p} col(z_{k-1-i}-y_{k-2-i})_{i=0}^{k-1} - col(w_i)_{i=0}^{k-1}\right] = 0,$$

setting $y_{-1} = 0$. Now $[I_k - B]$ is nonsingular, so

$$z_{k-1-i}-y_{k-2-i} = \frac{x_p}{x_{k-p}} w_i$$

for $i = 0, 1, \cdots, k-1$. So, it follows from the definition of w_i and from (3.3) that

$$z_{k-1-i}-y_{k-2-i} = \frac{x_p}{x_{k-p}}\left(y_i - z_{i+1} - \frac{y_{k-p-1}}{x_p} x_{k-1-i}\right), \qquad (i = 0, 1, \cdots, k-1). \tag{3.6}$$

Plugging (3.3) and (3.6) into (1.2) proves (1.4).

In order to complete the proof of the theorem it is enough to show that if conditions (a)–(c) hold, then the matrix is invertible. Here we check separately the cases of $p = 0$ and $p \neq 0$.

Proof of sufficiency. In case $p = 0$, the assertion follows from Theorem 1.1 of Gohberg and Semencul.

Now let us consider the case of $p \neq 0$. We assume that conditions (a)–(c) hold. Let us show that if $A\ col(b_i)_{i=0}^{k-1} = 0$, then $col(b_i)_{i=0}^{k-1}$ must equal zero. It follows from Lemma 2.1 that $A\ col(b_{k-1-i})_{i=0}^{k-1} = 0$ as well. Therefore, we may assume that either $b_i = b_{k-1-i}$

for $i = 0, 1, \cdots, k-1$ or $b_i = -b_{k-1-i}$ for $i = 0, 1, \cdots, k-1$ as we may define $col(c_i)_{i=0}^{k-1} = col(b_i)_{i=0}^{k-1} + col(b_{k-1-i})_{i=0}^{k-1}$ and $col(d_i)_{i=0}^{k-1} = col(b_i)_{i=0}^{k-1} - col \times (b_{k-1-i})_{i=0}^{k-1}$, and it is clear that $A\ col(c_i)_{i=0}^{k-1} = A\ col(d_i)_{i=0}^{k-1} = 0$ and that $col(b_i)_{i=0}^{k-1}$ is a linear combination of $col(c_i)_{i=0}^{k-1}$ and of $col(d_i)_{i=0}^{k-1}$.

We need only a weaker assumption that $b_r \neq 0$ if and only if $b_{k-1-r} \neq 0$ for $r = 0, 1, \cdots, k-1$. Now it follows from Lemma 2.1 that $row(b_j)_{j=0}^{k-1} A = 0$ and hence

$$b_0 = row(b_j)_{j=0}^{k-1} A\ col(x_i)_{i=0}^{k-1} = 0.$$

Similarly, we can replace $col(x_i)_{i=0}^{k-1}$ by $col(y_i)_{i=0}^{k-1}$, $col(x_{k-1-i})_{i=0}^{k-1}$, or $col \times (y_{k-1-i})_{i=0}^{k-1}$, to show that

$$b_p = b_{k-1} = b_{k-1-p} = 0.$$

Let q be such that $b_q \neq 0$ but $b_i = 0$ for all $i < q$. Then $b_{k-1-q} \neq 0$, but $b_i = 0$ for all $i > k-1-q$. In particular, $q \leqq (k-1)/2$. Our aim is to show that the existence of such a q implies contradiction. It follows from $row(b)_{j=0}^{k-1} A = 0$ that

$$\sum_{r=q}^{k-1-q} b_r row(a_{r-j})_{j=0}^{k-1} = 0. \tag{3.7}$$

First let us assume that $0 < q < p$. In this case, (3.7) implies

$$\sum_{r=q}^{k-1-q} b_r row(q_{r-j})_{j=p+q}^{k-p+q} = 0,$$

or, equivalently,

$$\sum_{r=0}^{k-1-2q} b_{r+q} row(a_{r-j})_{j=p}^{k-p} = 0.$$

Thus

$$\sum_{r=0}^{k-1-2q} b_{r+q} row(a_{r-j})_{j=p}^{k-p} col(x_i)_{i=p}^{k-p} = 0.$$

Recall that $x_i = 0$ for $i < p$ or $i > k-p$, so we conclude that

$$\sum_{r=0}^{k-1-2q} b_{r+q} row(a_{r-j})_{j=0}^{k-1} col(x_i)_{i=0}^{k-1} = 0,$$

which is impossible, as $(a_{r-j})_{j=0}^{k-1} col(x_i)_{i=0}^{k-1} = \delta_{0,r}$ and hence $b_q = 0$.

Assume now that $q > p$. It follows from $A\ col(y_{k-1-i})_{i=0}^{k-1} = E^{(p)}$ that

$$row(a_{r-j})_{j=0}^{k-1} col(y_{k-1-i})_{i=0}^{k-1} = 0 \quad \text{for } r \geqq q, \tag{3.8}$$

since $p < q$. In particular,

$$\sum_{r=q}^{k-1-q} b_r row(a_{r+1-j})_{j=0}^{k-1} col(y_{k-1-i})_{i=0}^{k-1} = 0.$$

Considering (3.7), we conclude that

$$\sum_{r=q}^{k-1-q} b_r a_{r+1} y_{k-1} = 0.$$

But $y_{k-1} = row(x_i)_{i=0}^{k-1} A\, col(y_{k-1-i})_{i=0}^{k-1} = x_p$, and therefore, $y_{k-1} \neq 0$. Consequently,

$$\sum_{r=q}^{k-1-q} b_r a_{r+1} = 0,$$

which, together with (3.7), gives

$$\sum_{r=q+1}^{k-q} b_{r-1} row(a_{r-j})_{j=0}^{k-1} = 0.$$

We can similarly use $k - 2q$ consecutive r values in (3.8) and conclude that

$$\sum_{r=q+i}^{k-1-q+i} b_{r-i} row(a_{r-j})_{j=0}^{k-1} = 0,$$

for $i = 0, 1, \cdots, q - p$. In the case of $i = q - p$, we obtain

$$\sum_{r=2q-p}^{k-1-p} b_{r-q+p} row(a_{r-j})_{j=0}^{k-1} = 0,$$

and therefore,

$$\sum_{r=2q-p}^{k-1-p} b_{r-q+p} row(a_{r-j})_{j=0}^{k-1} col(y_i)_{i=0}^{k-1} = 0,$$

which is impossible as $(a_{r-j})_{j=0}^{k-1} col(y_i)_{i=0}^{k-1} = \delta_{k-1-p,r}$, and hence $b_{k-1-q} = 0$. □

4. Proof of Theorem 1.4. Essentially, this is the same as the proof of Theorem 1.3, so we will indicate only the differences. Here we use the fact that if A is a Hermitian Toeplitz matrix and

$$A\, col(b_i)_{i=0}^{k-1} = col(c_i)_{i=0}^{k-1},$$

then

$$A\, col(\bar{b}_{k-1-i})_{i=0}^{k-1} = col(\bar{c}_{k-1-i})_{i=0}^{k-1},$$

$$row(\bar{b}_j)_{j=0}^{k-1} A = row(\bar{c}_j)_{j=0}^{k-1},$$

and

$$row(b_{k-1-j})_{j=0}^{k-1} A = row(c_{k-1-j})_{j=0}^{k-1},$$

as for a Toeplitz matrix A we have $JAJ = A^T$ and A^T is Hermitian as well.

Proof of necessity. We assume that A is an invertible Hermitian Toeplitz matrix. Again, we define $col(x_i)_{i=0}^{k-1}$ and $col(y_i)_{i=0}^{k-1}$ by

$$col(x_i)_{i=0}^{k-1} = A^{-1}E^{(0)}, \qquad col(y_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p)},$$

where p is such that $x_p \neq 0$ and $x_q = 0$ for all $q < p$. So, these are the solutions for the given systems of equations. Note that if $p = 0$, then $y_{k-1-i} = \bar{x}_i$ for $i = 0, 1, \cdots, k - 1$, and (1.6) follows from Theorem 1.2 with $z_i = 0$.

From now on, we assume that $p > 0$. It is clear that since A is an invertible Hermitian Toeplitz matrix, then its inverse is of the form

$$A^{-1} = \begin{pmatrix} x_0 & \bar{x}_1 & \cdots & \bar{x}_{k-2} & \bar{x}_{k-1} \\ x_1 & & & & \bar{x}_{k-2} \\ \vdots & & * & & \vdots \\ x_{k-2} & & & & \bar{x}_1 \\ x_{k-1} & x_{k-2} & \cdots & x_1 & x_0 \end{pmatrix},$$

and hence,

$$\Delta(A^{-1}) = \begin{pmatrix} x_0 & \bar{x}_1 & \cdots & \bar{x}_{k-1} & 0 \\ x_1 & & & & -\bar{x}_{k-1} \\ \vdots & & * & & \vdots \\ x_{k-1} & & & & -\bar{x}_1 \\ 0 & -x_{k-1} & \cdots & -x_1 & -x_0 \end{pmatrix}.$$

Again, as $\operatorname{rank}\Delta(A^{-1}) = 2$, the determinant of any 3×3 submatrix of $\Delta(A^{-1})$ must equal zero, and for any integer q such that $k - p < q < k$, then

$$0 = \det \begin{pmatrix} x_0 & \bar{x}_{k-q} & \bar{x}_p \\ x_p & * & * \\ 0 & -x_q & -x_{k-p} \end{pmatrix} = \det \begin{pmatrix} 0 & 0 & \bar{x}_p \\ x_p & * & * \\ 0 & -x_q & -x_{k-p} \end{pmatrix},$$

so $x_q = 0$ for all $q > k - p$ and $p \leqq k/2$ as well.

Define $col(z_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p-1)}$. It follows from our definitions that $A\ col(\bar{y}_i)_{i=0}^{k-1} = E^{(k-p-1)}$ and that $A\ col(\bar{z}_i)_{i=0}^{k-1} = E^{(k-p)}$. Thus, according to Theorem 1.2,

$$A^{-1} = \frac{1}{x_p}\left[\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} \bar{y}_{k-1} & \bar{y}_{k-2} - \bar{z}_{k-1} & \cdots & \bar{y}_0 - \bar{z}_1 \\ 0 & \bar{y}_{k-1} & \cdots & \bar{y}_1 - \bar{z}_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{y}_{k-1} \end{pmatrix}\right.$$

$$\left. + \begin{pmatrix} \bar{z}_0 & \cdots & 0 & 0 \\ \bar{z}_1 - \bar{y}_0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \bar{z}_{k-1} - \bar{y}_{k-2} & \cdots & \bar{z}_1 - \bar{y}_0 & \bar{z}_0 \end{pmatrix}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\right], \tag{4.1}$$

as well as

$$A^{-1} = \frac{1}{x_p}\left[\begin{pmatrix} \bar{y}_{k-1} & \bar{y}_{k-2} - \bar{z}_{k-1} & \cdots & \bar{y}_0 - \bar{z}_1 \\ 0 & \bar{y}_{k-1} & \cdots & \bar{y}_1 - \bar{z}_2 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{y}_{k-1} \end{pmatrix}\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\right.$$

$$\left. + \begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} \bar{z}_0 & \cdots & 0 & 0 \\ \bar{z}_1 - \bar{y}_0 & \cdots & 0 & 0 \\ \vdots & \ddots & \vdots & \vdots \\ \bar{z}_{k-1} - \bar{y}_{k-2} & \cdots & \bar{z}_1 - \bar{y}_0 & \bar{z}_0 \end{pmatrix}\right]. \tag{4.2}$$

It follows from (4.1) that

$$col(z_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p-1)} = \frac{1}{x_p}\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} \bar{y}_{k-p} - \bar{z}_{k-p+1} \\ \vdots \\ \bar{y}_{k-2} - \bar{z}_{k-1} \\ \bar{y}_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

as $x_q = 0$ for $q > k - p$. Moreover, $z_q = 0$ for $q \geqq k - p$ as $x_0 = x_1 = \cdots = x_{p-1} = 0$, so (1.8) is proved.

Now, according to (4.2),

$$\begin{pmatrix} \bar{x}_{k-1} \\ \vdots \\ \bar{x}_1 \\ \bar{x}_0 \end{pmatrix} = A^{-1}E^{(k-1)} = \frac{\bar{z}_0}{x_p}\begin{pmatrix} x_1 \\ \vdots \\ x_{k-1} \\ 0 \end{pmatrix},$$

as $x_0 = 0$. But, $\bar{z}_0 = F^{(0)}A^{-1}E^{(k-p)} = \bar{x}_{k-p}$ and therefore,

$$\begin{pmatrix} \bar{x}_{k-1} \\ \vdots \\ \bar{x}_1 \\ 0 \end{pmatrix} = \frac{\bar{x}_{k-p}}{x_p}\begin{pmatrix} x_1 \\ \vdots \\ x_{k-1} \\ 0 \end{pmatrix}. \tag{4.3}$$

In particular $\bar{x}_p = (\bar{x}_{k-p}/x_p)x_{k-p}$ and hence $\alpha \triangleq \bar{x}_{k-p}/x_p$ satisfies $|\alpha| = 1$.

It remains to prove (1.7). It follows from (4.1) that as $col(y_{k-1-i})_{i=0}^{k-1} = A^{-1}E^{(p)}$, then

$$col(y_{k-1-i})_{i=0}^{k-1} = \frac{x_{k-p}}{x_p}\begin{pmatrix} \bar{z}_0 \\ \bar{z}_1 - \bar{y}_0 \\ \vdots \\ \bar{z}_{k-1} - \bar{y}_{k-2} \end{pmatrix} + \frac{1}{x_p}\begin{pmatrix} x_0 & 0 & \cdots & 0 \\ x_1 & x_0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ x_{k-1} & x_{k-2} & \cdots & x_0 \end{pmatrix}\begin{pmatrix} \bar{y}_{k-p-1} - \bar{z}_{k-p} \\ \vdots \\ \bar{y}_{k-2} - \bar{z}_{k-1} \\ \bar{y}_{k-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

or, according to (4.3),

$$col(y_i)_{i=0}^{k-1} = \frac{x_{k-p}}{x_p}\begin{pmatrix} \bar{z}_{k-1} - \bar{y}_{k-2} \\ \vdots \\ \bar{z}_1 - \bar{y}_0 \\ \bar{z}_0 \end{pmatrix} + \frac{1}{\bar{x}_{k-p}}\begin{pmatrix} 0 & \bar{x}_{k-1} & \cdots & \bar{x}_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & \bar{x}_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \bar{y}_{k-1} \\ \bar{y}_{k-2} - \bar{z}_{k-1} \\ \vdots \\ \bar{y}_{k-p-1} - \bar{z}_{k-p} \end{pmatrix}. \tag{4.4}$$

Now, $col(\bar{y}_i)_{i=0}^{k-1} = A^{-1}E^{(k-p-1)}$. Thus, according to (4.2),

$$col(\bar{y}_i)_{i=0}^{k-1} = \begin{pmatrix} \bar{y}_0 - \bar{z}_1 \\ \bar{y}_1 - \bar{z}_2 \\ \vdots \\ \bar{y}_{k-1} \end{pmatrix} + \frac{1}{x_p}\begin{pmatrix} 0 & x_{k-1} & \cdots & x_1 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & x_{k-1} \\ 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \bar{z}_0 \\ \bar{z}_1 - \bar{y}_0 \\ \vdots \\ \bar{z}_p - \bar{y}_{p-1} \end{pmatrix},$$

which, combined with (4.4) gives

$$\frac{x_{k-p}}{x_p}\begin{pmatrix}\bar{z}_{k-1}-\bar{y}_{k-2}\\ \vdots\\ \bar{z}_1-\bar{y}_0\\ \bar{z}_0\end{pmatrix}-\frac{1}{\bar{x}_p}\begin{pmatrix}0 & \bar{x}_{k-1} & \cdots & \bar{x}_1\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \bar{x}_{k-1}\\ 0 & 0 & \cdots & 0\end{pmatrix}\begin{pmatrix}0\\ \vdots\\ 0\\ z_0\\ z_1-y_0\\ \vdots\\ z_p-y_{p-1}\end{pmatrix}$$

$$=\begin{pmatrix}y_0-z_1\\ y_1-z_2\\ \vdots\\ y_{k-1}\end{pmatrix}-\frac{1}{\bar{x}_{k-p}}\begin{pmatrix}0 & \bar{x}_{k-1} & \cdots & \bar{x}_1\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \bar{x}_{k-1}\\ 0 & 0 & \cdots & 0\end{pmatrix}\begin{pmatrix}0\\ \vdots\\ 0\\ \bar{y}_{k-1}\\ \bar{y}_{k-2}-\bar{z}_{k-1}\\ \vdots\\ \bar{y}_{k-p-1}-\bar{z}_{k-p}\end{pmatrix}.$$

Note that $z_p = F^{(k-p-1)}A^{-1}E^{(p-1)} = y_{p-1}$ and $\bar{z}_{k-p} = 0$, so

$$\frac{x_{k-p}}{x_p}\begin{pmatrix}\bar{z}_{k-1}-\bar{y}_{k-2}\\ \vdots\\ \bar{z}_1-\bar{y}_0\\ \bar{z}_0\end{pmatrix}-\frac{1}{\bar{x}_p}\begin{pmatrix}0 & 0 & \bar{x}_{k-1} & \cdots & \bar{x}_2\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & \bar{x}_{k-1}\\ 0 & 0 & 0 & \cdots & 0\\ 0 & 0 & 0 & \cdots & 0\end{pmatrix}\begin{pmatrix}0\\ \vdots\\ 0\\ z_0\\ z_1-y_0\\ \vdots\\ z_{p-1}-y_{p-2}\end{pmatrix}$$

$$=\begin{pmatrix}y_0-z_1\\ \vdots\\ y_{k-2}-z_{k-1}\\ y_{k-1}\end{pmatrix}-\frac{1}{\bar{x}_{k-p}}\begin{pmatrix}0 & 0 & \bar{x}_{k-1} & \cdots & \bar{x}_2\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \cdots & \bar{x}_{k-1}\\ 0 & 0 & 0 & \cdots & 0\\ 0 & 0 & 0 & \cdots & 0\end{pmatrix}\begin{pmatrix}0\\ \vdots\\ 0\\ \bar{y}_{k-1}\\ \bar{y}_{k-2}-\bar{z}_{k-1}\\ \vdots\\ \bar{y}_{k-p}-\bar{z}_{k-p+1}\end{pmatrix}$$

$$-\frac{\bar{y}_{k-p-1}}{\bar{x}_{k-p}}\begin{pmatrix}\bar{x}_1\\ \bar{x}_2\\ \vdots\\ \bar{x}_{k-1}\\ 0\end{pmatrix}.$$

Note that $y_{k-p-1} = (A^{-1})_{p,p}$, and hence is a real number. Define

$$w_i = y_i - z_{i+1} - \frac{y_{k-p-1}}{\bar{x}_{k-p}}\bar{x}_{i+1}$$

for $i = 0, 1, \cdots, k-1$, setting $\bar{x}_k = z_k = 0$. Then

$$\frac{x_{k-p}}{x_p}\begin{pmatrix} \bar{z}_{k-1}-\bar{y}_{k-2} \\ \vdots \\ \bar{z}_1-\bar{y}_0 \\ \bar{z}_0 \end{pmatrix} - \frac{1}{\bar{x}_p}\begin{pmatrix} 0 & 0 & \bar{x}_{k-1} & \cdots & \bar{x}_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \bar{x}_{k-1} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ z_0 \\ z_1-y_0 \\ \vdots \\ z_{p-1}-y_{p-2} \end{pmatrix}$$

$$= \begin{pmatrix} w_0 \\ \vdots \\ w_{k-1} \end{pmatrix} - \frac{1}{\bar{x}_{k-p}}\begin{pmatrix} 0 & 0 & \bar{x}_{k-1} & \cdots & \bar{x}_2 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & \bar{x}_{k-1} \\ 0 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 0 & \cdots & 0 \end{pmatrix}\begin{pmatrix} 0 \\ \vdots \\ 0 \\ \bar{w}_{k-1} \\ \bar{w}_{k-2} \\ \vdots \\ \bar{w}_{k-p} \end{pmatrix},$$

since $x_{k-p-1} = \cdots = x_{k-2} = x_{k-1} = 0$. Equivalently,

$$\frac{x_{k-p}}{x_p}\begin{pmatrix} \bar{z}_{k-1}-\bar{y}_{k-2} \\ \vdots \\ \bar{z}_1-\bar{y}_0 \\ \bar{z}_0 \end{pmatrix} - \frac{1}{\bar{x}_p}\begin{pmatrix} \bar{x}_{p+1} & \cdots & \bar{x}_2 & \\ \vdots & \ddots & \vdots & \\ \bar{x}_{k-1} & \cdots & \bar{x}_{k-p} & \\ 0 & \cdots & \bar{x}_{k-p+1} & 0_{k\times(k-p)} \\ \vdots & \ddots & \vdots & \\ 0 & \cdots & \bar{x}_{k-1} & \\ 0 & \cdots & 0 & \\ 0 & \cdots & 0 & \end{pmatrix}\begin{pmatrix} z_0 \\ z_1-y_0 \\ \vdots \\ z_{k-1}-y_{k-2} \end{pmatrix}$$

$$= \begin{pmatrix} w_0 \\ \vdots \\ w_{k-1} \end{pmatrix} - \frac{1}{\bar{x}_{k-p}}\begin{pmatrix} \bar{x}_{p+1} & \cdots & \bar{x}_2 & \\ \vdots & \ddots & \vdots & \\ \bar{x}_{k-1} & \cdots & \bar{x}_{k-p} & \\ 0 & \cdots & \bar{x}_{k-p+1} & 0_{k\times(k-p)} \\ \vdots & \ddots & \vdots & \\ 0 & \cdots & \bar{x}_{k-1} & \\ 0 & \cdots & 0 & \\ 0 & \cdots & 0 & \end{pmatrix}\begin{pmatrix} \bar{w}_{k-1} \\ \vdots \\ \bar{w}_0 \end{pmatrix}.$$

Consequently,

$$\frac{x_{k-p}}{x_p} col(\bar{z}_{k-1-i}-\bar{y}_{k-2-i})_{i=0}^{k-1} - col(w_i)_{i=0}^{k-1} = B\left[\frac{\bar{x}_{k-p}}{\bar{x}_p} col(z_{k-1-i}-y_{k-2-i})_{i=0}^{k-1} - col(\bar{w}_i)_{i=0}^{k-1}\right], \tag{4.5}$$

(with $y_{-1} = 0$) for the $k \times k$ matrix B defined by

$$B = \frac{1}{\bar{x}_{k-p}}\begin{pmatrix} & \bar{x}_2 & \cdots & \bar{x}_{p+1} \\ & \vdots & \ddots & \vdots \\ & \bar{x}_{k-p} & \cdots & \bar{x}_{k-1} \\ 0_{k\times(k-p)} & \bar{x}_{k-p+1} & \cdots & 0 \\ & \vdots & \ddots & \vdots \\ & \bar{x}_{k-1} & \cdots & 0 \\ & 0 & \cdots & 0 \\ & 0 & \cdots & 0 \end{pmatrix}.$$

Denote by $\bar{B}$ the matrix with entries being the complex conjugates of the entries of B; then (4.5) implies

$$col(c_i)_{i=0}^{k-1} = B\ col(\bar{c}_i)_{i=0}^{k-1} = B\bar{B}col(c_i)_{i=0}^{k-1} = \cdots = B\bar{B}B\cdots\bar{B}col(c_i)_{i=0}^{k-1},$$

for

$$c_i \triangleq \frac{x_{k-p}}{x_p}(\bar{z}_{k-1-i} - \bar{y}_{k-2-i}) - w_i, \qquad (i = 0, 1, \cdots, k-1).$$

Note that B, and hence $\bar{B}$, are both upper triangular matrices with zero diagonal as $x_q = 0$ for $q > k - p$. Therefore, k products of B and $\bar{B}$ equal zero and consequently, $col(c_i)_{i=0}^{k-1} = 0$, or equivalently,

$$\frac{1}{x_p}(\bar{z}_i - \bar{y}_{i-1}) = \frac{1}{x_{k-p}}\left(y_{k-1-i} - z_{k-i} - \frac{y_{k-p-1}}{\bar{x}_{k-p}}\bar{x}_{k-i}\right), \qquad (i = 0, 1, \cdots, k-1),$$

which, together with (4.1) and (4.3), prove (1.7).

The proof of the sufficiency is almost identical to the one in Theorem 1.3.

Proof of sufficiency. We consider the case of $p \neq 0$, assuming that conditions (a)–(c) hold. Let us show that if $A\ col(b_i)_{i=0}^{k-1} = 0$, then $col(b_i)_{i=0}^{k-1}$ must equal zero. In the Hermitian case, it follows that $A\ col(\bar{b}_{k-1-i})_{i=0}^{k-1} = 0$ as well. Therefore, we may assume that either $b_i = \bar{b}_{k-1-i}$ for $i = 0, 1, \cdots, k-1$, or $b_i = \bar{b}_{k-1-i}$ for $i = 0, 1, \cdots, k-1$, as we may define $col(c_i)_{i=0}^{k-1} = col(b_i)_{i=0}^{k-1} + col(\bar{b}_{k-1-i})_{i=0}^{k-1}$ and $col(d_i)_{i=0}^{k-1} = col(b_i)_{i=0}^{k-1} - col(\bar{b}_{k-1-i})_{i=0}^{k-1}$, and it is clear that $A\ col(c_i)_{i=0}^{k-1} = A\ col(d_i)_{i=0}^{k-1} = 0$ and that $col(b_i)_{i=0}^{k-1}$ is a linear combination of $col(c_i)_{i=0}^{k-1}$ and of $col(d_i)_{i=0}^{k-1}$. As in the proof of sufficiency of Theorem 1.3, we need only the weaker assumption that $b_r \neq 0$ if and only if $b_{k-1-r} \neq 0$ for $r = 0, 1, \cdots, k-1$, and the proof continues similarly. □

5. Examples. In this section we give several examples that illustrate the necessity of the conditions in Theorem 1.3.

Example 5.1. Consider the 9×9 symmetric Toeplitz matrix

$$A = \begin{pmatrix} 2 & 0 & 2 & -4 & 2 & -4 & 2 & -8 & 18 \\ 0 & 2 & 0 & 2 & -4 & 2 & -4 & 2 & -8 \\ 2 & 0 & 2 & 0 & 2 & -4 & 2 & -4 & 2 \\ -4 & 2 & 0 & 2 & 0 & 2 & -4 & 2 & -4 \\ 2 & -4 & 2 & 0 & 2 & 0 & 2 & -4 & 2 \\ -4 & 2 & -4 & 2 & 0 & 2 & 0 & 2 & -4 \\ 2 & -4 & 2 & -4 & 2 & 0 & 2 & 0 & 2 \\ -8 & 2 & -4 & 2 & -4 & 2 & 0 & 2 & 0 \\ 18 & -8 & 2 & -4 & 2 & -4 & 2 & 0 & 2 \end{pmatrix}.$$

The inverse matrix A^{-1} is

$$A^{-1} = \frac{-1}{8}\begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 1 & 2 & 1 \\ 1 & 1 & 1 & 1 & 0 & 3 & 1 & 1 & 1 \\ 1 & 2 & 1 & 0 & -1 & 2 & 3 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 3 & 2 & -1 & 0 & 1 & 2 & 1 \\ 1 & 1 & 1 & 3 & 0 & 1 & 1 & 1 & 1 \\ 1 & 2 & 1 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix}.$$

In this case $k = 9$, $p = 2$, and according to Theorem 1.3, the matrix A is determined by the two columns of A^{-1}, which are $A^{-1}E^{(0)}$ and $A^{-1}E^{(2)}$. It is clear that $A^{-1}E^{(8)}$ can replace $A^{-1}E^{(0)}$ and that $A^{-1}E^{(6)}$ can replace $A^{-1}E^{(2)}$. However, the columns $A^{-1}E^{(0)}$, $A^{-1}E^{(1)}, A^{-1}E^{(3)}, A^{-1}E^{(4)}, A^{-1}E^{(5)}, A^{-1}E^{(7)}$, and $A^{-1}E^{(8)}$ do not determine A. Indeed, the symmetric Toeplitz matrix

$$B = \begin{pmatrix} 1 & 1 & 1 & -3 & 1 & -3 & 1 & -7 & 17 \\ 1 & 1 & 1 & 1 & -3 & 1 & -3 & 1 & -7 \\ 1 & 1 & 1 & 1 & 1 & -3 & 1 & -3 & 1 \\ -3 & 1 & 1 & 1 & 1 & 1 & -3 & 1 & -3 \\ 1 & -3 & 1 & 1 & 1 & 1 & 1 & -3 & 1 \\ -3 & 1 & -3 & 1 & 1 & 1 & 1 & 1 & -3 \\ 1 & -3 & 1 & -3 & 1 & 1 & 1 & 1 & 1 \\ -7 & 1 & -3 & 1 & -3 & 1 & 1 & 1 & 1 \\ 17 & -7 & 1 & -3 & 1 & -3 & 1 & 1 & 1 \end{pmatrix}$$

has an inverse of the form

$$B^{-1} = \frac{-1}{8}\begin{pmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 2 & 1 & 0 & 1 & 2 & 1 \\ 1 & 1 & 0 & 1 & 0 & 3 & 0 & 1 & 1 \\ 1 & 2 & 1 & 0 & -1 & 2 & 3 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & -1 & 0 & 1 & 0 \\ 0 & 0 & 3 & 2 & -1 & 0 & 1 & 2 & 1 \\ 1 & 1 & 0 & 3 & 0 & 1 & 0 & 1 & 1 \\ 1 & 2 & 1 & 0 & 1 & 2 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{pmatrix},$$

and $B^{-1}E^{(j)} = A^{-1}E^{(j)}$ for $j = 0, 1, 3, 4, 5, 7, 8$. Furthermore, consider the symmetric Toeplitz matrix

$$C = 2B - A = \begin{pmatrix} 0 & 2 & 0 & -2 & 0 & -2 & 0 & -6 & 16 \\ 2 & 0 & 2 & 0 & -2 & 0 & -2 & 0 & -6 \\ 0 & 2 & 0 & 2 & 0 & -2 & 0 & -2 & 0 \\ -2 & 0 & 2 & 0 & 2 & 0 & -2 & 0 & -2 \\ 0 & -2 & 0 & 2 & 0 & 2 & 0 & -2 & 0 \\ -2 & 0 & -2 & 0 & 2 & 0 & 2 & 0 & -2 \\ 0 & -2 & 0 & -2 & 0 & 2 & 0 & 2 & 0 \\ -6 & 0 & -2 & 0 & -2 & 0 & 2 & 0 & 2 \\ 16 & -6 & 0 & -2 & 0 & -2 & 0 & 2 & 0 \end{pmatrix}.$$

Then, there exist column vectors $X^{(j)}$, which satisfy $CX^{(j)} = E^{(j)}$ for $j = 0, 1, 3, 4, 5, 7, 8$, and these are the same vectors that satisfy both $AX^{(j)} = E^{(j)}$ and $BX^{(j)} = E^{(j)}$. However, there is no solution for $CX^{(2)} = E^{(2)}$ and condition (c) does not hold. The matrix C is indeed singular as $F^{(2)}C = -F^{(6)}C$.

Example 5.2. Look at the 3×3 matrix

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

Note that $A\,col(0, 0, 1) = col(1, 0, 0)$, so $p = 2$ and $A\,col(1, 0, 0) = col(0, 0, 1) = E^{(p)}$. But $p > k/2$ and the matrix is not invertible.

Example 5.3. Now let us consider a case in which $p \leqq k/2$. Look at the 5×5 matrix

$$A = \begin{pmatrix} -1 & 0 & 1 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 \\ 1 & 0 & -1 & 0 & 1 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & 0 & -1 \end{pmatrix}.$$

Here, $k = 5$ and $p = 2$ as $A\, col(0, 0, 1, 0, 1) = col(1, 0, 0, 0, 0)$. Moreover, $A\, col(1, 0, 1, 0, 1) = col(0, 0, 1, 0, 0) = E^{(p)}$, yet the matrix is not invertible, as $x_4 = 1$ but $4 > k - p$.

Example 5.4. Let us show that the case of $k = 5$ and $p = 2$ is possible. The matrix

$$A_t = \begin{pmatrix} -1 & 1 & -1 & 2 & t \\ 1 & -1 & 1 & -1 & 2 \\ -1 & 1 & -1 & 1 & -1 \\ 2 & -1 & 1 & -1 & 1 \\ t & 2 & -1 & 1 & -1 \end{pmatrix}$$

is invertible for any value of t. Indeed, $p = 2$ as $A_t\, col(0, 0, 1, 1, 0) = col(1, 0, 0, 0, 0)$. Regarding $E^{(p)}$, note that $A_t\, col(1, -t - 2, -2t - 7, -t - 2, 1) = col(0, 0, 1, 0, 0)$.

Example 5.5. Now we show that both $x_p = x_{k-p}$ and $x_p = -x_{k-p}$ might occur. Indeed,

$$\begin{pmatrix} -1 & 1 & 0 \\ 1 & -1 & 1 \\ 0 & 1 & -1 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix},$$

while

$$\begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 1 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 1 & -1 \\ 1 & -1 & 1 \\ -1 & 1 & 0 \end{pmatrix}.$$

Acknowledgment. It is our pleasure to thank I. Gohberg for bringing the problems solved in this paper to our attention.

REFERENCES

[BGY] R. P. BRENT, F. G. GUSTAVSON, AND D. Y. Y. YUN, *Fast solution of Toeplitz systems of equations and computation of Pade approximants*, J. Algorithms, 1 (1980), pp. 259–295.

[BS] A. BEN-ARTZI AND T. SHALOM, *On inversion of Toeplitz and close to Toeplitz matrices*, Linear Algebra Appl., 75 (1986), pp. 173–192.

[GK] I. C. GOHBERG AND N. Y. KRUPNIK, *A formula for the inversion of finite Toeplitz matrices*, Mat. Issled., 7 (1972), pp. 272–283. (In Russian.)

[GS] I. C. GOHBERG AND A. A. SEMENCUL, *On the inversion of finite Toeplitz matrices and their continual analogs*, Mat. Issled., 7 (1972), pp. 201–223. (In Russian.)

[HR] G. HEINIG AND K. ROST, *Algebraic methods for Toeplitz-like matrices and operators*, in Operator Theory: Advances and Applications, Vol. 13, Birkhäuser, Boston, 1984.

[KC] T. KAILATH AND J. CHUN, *Generalized Gohberg–Semencul formulas for matrix inversion*, in Operator Theory: Advances and Applications, Vol. 40, Birkhäuser, Boston, 1989, pp. 231–246.

[L] N. LEVINSON, *The Wiener* RMS *(root-mean-square) error criterion in filter design and prediction*, J. Math. Phys., 25 (1947), pp. 261–278.

[T] W. F. TRENCH, *An algorithm for the inversion of finite Toeplitz matrices*, SIAM J. Appl. Math., 12 (1964), pp. 515–522.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 550–566, April 1992

008

CYCLIC STRONG ERGODICITY IN NONHOMOGENEOUS MARKOV SYSTEMS*

IOANNIS I. GERONTIDIS†

Abstract. This paper investigates the ergodic behaviour of the vector of means and the covariance matrix of the grade sizes for a nonhomogeneous Markov system that undergoes a cyclic behaviour, both in discrete and continuous time. It is shown that the first and second central moments converge to a cyclic family of multinomial type with the same period, independently of the initial distribution. The regions of cyclically ergodic distributions are determined as convex hulls of certain points as the recruitment distribution varies. The rate of convergence to the cyclic distribution is also examined, together with some transient aspects of the system concerning stability and quasi stationarity. Two numerical examples from the literature on manpower planning illustrate the theory.

Key words. limiting distribution, manpower planning, nonstationary Markov replacement process, quasi stationarity, rate of convergence, stability

AMS(MOS) subject classifications. primary 60J20; secondary 92A15

1. Introduction. In recent years there has been a rapidly growing literature on the asymptotic behaviour of nonhomogeneous Markov systems (NHMS) (see, for example, Vassiliou (1981), (1982), (1984), (1986); Vassiliou and Gerontidis (1985); Tsaklidis and Vassiliou (1988); Vassiliou and Tsaklidis (1989); and Gerontidis (1990a), (1991)). These studies extend earlier results in Leslie (1945); Young and Almond (1961); Pollard (1966), (1973); Feichtinger (1976); Bartholomew (1982); Bartholomew and Forbes (1979); and Gerontidis (1990b). NHMS were first introduced by Vassiliou (1982) in order to provide a general framework for a number of Markov chain models in manpower systems.

In an earlier investigation (Gerontidis (1991)), the concept of periodic strong ergodicity in NHMS, i.e., when the sequence of transition matrices of the associated replacement chain converges to an irreducible stochastic matrix with $d \geqq 1$ eigenvalues of modulus 1, was examined in detail. In the present paper, which is a sequel to the above study, we exhibit a unified treatment of the ergodic behaviour of the vector of means and the covariance matrix of the grade sizes for an NHMS that undergoes a cyclic behaviour, both in discrete and continuous time. Related work appears in Vassiliou (1986), but we intend to present here an alternative point of view. Our main objective is to exploit the properties of the associated nonhomogeneous Markov replacement chain, an approach which provides insight not only into the direction into which the structure is changing, but also into the probabilistic nature of the convergence (by characterizing cyclic strong ergodicity) in NHMS. In § 2 the discrete time case is considered, where it is proved that for a system that undergoes a cyclic behaviour of period $d > 1$, where d is the period of the cycle, the vector of means and the covariance matrix converge to a cyclic sequence of multinomial type with the same period independently of the initial distribution. This fact reveals a kind of "cyclic strong ergodicity." The reasoning is based on the development of the asymptotic theory for products of stochastic matrices in a cyclic environment. By representing a cyclically ergodic distribution as convex combination of certain points in $\mathbb{R}^k$, where k is the number of the states of the system, we

* Received by the editors November 27, 1989; accepted for publication (in revised form) November 29, 1990.

† Mathematics Department, University of Thessaloniki, 54006 Thessaloniki, Greece.

determine the d sets of limiting distributions as convex hulls of these points for varying recruitment distribution. In § 3 some transient aspects of the system during the period of a cycle with respect to stability and quasi stationarity are investigated and those quasi-stationary structures that are stable are identified. In § 4 the continuous time analogue of the discrete case is developed, where it is shown that for an NHMS undergoing a cyclic behaviour of period $\tau > 0$, the first and second central moments of the grade sizes converge to a periodic family of multinomial type. Section 5 examines the rate of convergence problem to the cyclically ergodic distribution and conditions are provided for the convergence to be exponentially fast. Finally, in § 6, two examples from the literature on manpower planning illustrate the theoretical results.

2. Cyclic strong ergodicity in discrete time. For an NHMS in discrete time with k states (grades), let $\bar{n}_i(t)$, $i = 1, \cdots, k$, be the expected number of members in the ith grade of the system at time $t = 0, 1, \cdots$ and in vector notation, $\bar{n}(t) = [\bar{n}_1(t), \cdots, \bar{n}_k(t)]$. Consider also the $k \times k$ matrix $V(t) = [v_{ij}(t)]$ with entries being the variances and covariances of the grade sizes. Define by $T(t)$ the total membership in the system, which is assumed to be a given function of time, and denote by $M(t) = T(t+1) - T(t)$ the change of the total size in the time interval $[t, t+1)$. Also let $p_0(t) = [p_{01}(t), \cdots, p_{0k}(t)]$ be the recruitment distribution at time t and denote by $V\{p_0(t)\}$ the covariance matrix attributable to $p_0(t)$. For a vector $a = (a_i)$, denote by Diag (α) the matrix with elements $[\delta_{ij} a_i]$, where δ_{ij} is the Kronecker delta. It is known (see, for example, Gerontidis (1991)) that $\bar{n}(t)$ and $V(t)$ satisfy the difference equations

$$\bar{n}(t+1) = \bar{n}(t)Q(t) + M(t)p_0(t) \tag{2.1}$$

and

$$\begin{aligned} V(t+1) - \text{Diag}\{\bar{n}(t+1)\} &= Q(t)'[V(t) - \text{Diag}\{\bar{n}(t)\}]Q(t) \\ &\quad + M(t)[V\{p_0(t)\} - \text{Diag}\{p_0(t)\}], \end{aligned} \tag{2.2}$$

where

$$Q(t) = P(t) + p_{k+1}(t)'p_0(t) \tag{2.3}$$

(prime denotes transposition) is the $k \times k$ stochastic matrix of the associated nonhomogeneous Markov replacement chain (Keilson (1979, p. 41)). $P(t) = [p_{ij}(t)]$ is the $k \times k$ matrix of transition probabilities between the states, i.e., $p_{ij}(t)$ is the probability that someone who is in grade i in the beginning of the time interval $[t, t+1)$ will be in grade j during that interval. Also, $p_{k+1}(t) = [p_{1,k+1}(t), \cdots, p_{k,k+1}(t)]$ is the vector of wastage probabilities from the various states (state $k+1$ representing the world outside the system). Finally, for two integers s, $m (s < m)$ define

$$Q(s,m) = Q(s)Q(s+1)\cdots Q(m)$$

and $Q(s+1, s) = I$ for any s. Iterating (2.1) and (2.2) and assuming that $n(0)$ is given (nonrandom) so that $V(0) = 0$, the $k \times k$ zero matrix, we get

$$\bar{n}(t+1) = n(0)Q(0,t) + \sum_{\tau=0}^{t} M(\tau)p_0(\tau)Q(\tau+1,t) \tag{2.4}$$

and

$$\begin{aligned} V(t+1) &= \text{Diag}\{\bar{n}(t+1)\} - Q(0,t)'\,\text{Diag}\{n(0)\}Q(0,t) \\ &\quad - \sum_{\tau=0}^{t} M(\tau)Q(\tau+1,t)'p_0(\tau)'p_0(\tau)Q(\tau+1,t). \end{aligned} \tag{2.5}$$

The importance of the cyclic behaviour was first pointed out by Bartholomew (1982, p. 71), who was concerned with Gani's (1963) study of student enrollment at the Michigan State University. There the academic year was divided into three quarters and transitions took place at the end of each quarter. As Bartholomew (1982) argued, it would be reasonable to assume that for the case study under consideration, the transition and recruitment probabilities are periodic functions of time, i.e., that they are the same in the first quarter of one year as they had been in the first quarter of the previous year. There are also several other references supporting the assumption of cyclically varying parameters; see, for example, Young (1971); Conlisk (1976); Grinold and Marshall (1977, p. 62); Vassiliou (1984), (1986); Davies (1985); and Gerontidis (1990a). From Vassiliou (1986), we borrow the following definition.

DEFINITION 2.1. An NHMS in discrete time undergoes a cyclic behaviour with period d if and only if $P(md+r) = P(r)$, $p_{k+1}(md+r) = p_{k+1}(r)$, and $p_0(md+r) = p_0(r)$ for any $m = 1, 2, \cdots$ and $r = 0, 1, \cdots, d-1$ for some integer $d > 1$, called the period of the cycle.

In the rest of this section we first develop the asymptotic theory for products of stochastic matrices in a cyclic environment and later we characterize cyclic strong ergodicity in NHMS. For an NHMS under cyclic behaviour from (2.3) we get

$$Q(md+r) = P(md+r) + p_{k+1}(md+r)' p_0(md+r) = Q(r). \tag{2.6}$$

Define $Q(0, d-1) = Q(0)Q(1)\cdots Q(d-1)$ to be the "cycle matrix." The following lemmas show that much of the structure (i.e., the eigenvalues and eigenvectors) of the sequence $Q(s, t)$ for $s = 0, 1, \cdots, t = 0, 1, \cdots,$ (and later to be shown of the system) is contained in the structure of the cycle matrix $Q(0, d-1)$.

LEMMA 2.1. *If the sequence of replacement matrices $Q(t)$, $t = 0, 1, \cdots$, undergoes a cyclic behaviour with period $d > 1$, then for τ, $\pi = 1, 2, \cdots$, $\tau < \pi$, and $m, r = 0, 1, \cdots, d-1$, it holds that*

$$Q(\tau d+m, \pi d+r) = Q(m, d-1)Q(0, d-1)^{\pi-\tau-1}Q(0, r). \tag{2.7}$$

Proof. We have

$$\begin{aligned}
&Q(\tau d+m, \pi d+r) \\
&\quad = Q(\tau d+m)Q(\tau d+m+1)\cdots Q(\tau d+d-1) \\
&\qquad \left.\begin{aligned} &\times Q\{(\tau+1)d\}Q\{(\tau+1)d+1\}\cdots Q\{(\tau+1)d+d-1\} \\ &\quad \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots \\ &\times Q\{(\pi-1)d\}Q\{(\pi-1)d+1\}\cdots Q\{(\pi-1)d+d-1\} \end{aligned}\right\} (\pi-\tau-1) \text{ products} \\
&\qquad \times Q(\pi d)Q(\pi d+1)\cdots Q(\pi d+r).
\end{aligned}$$

The cyclic behaviour of the sequence $Q(t)$, $t = 0, 1, \cdots$, i.e., (2.6), implies (2.7). □

For a real matrix $A = [a_{ij}]$ and a real vector $u = (u_j)$ we define their norms as $\|A\| = \max_i \sum_{j=1}^{k} |a_{ij}|$ and $\|u\| = \max_j |u_j|$, respectively. In what follows, convergence of matrices and vectors is assumed with respect to these norms. A stochastic matrix Q is called strongly ergodic (Isaacson and Luecke (1978)) if it has no characteristic roots ($\neq 1$) of modulus 1 and if 1 is a simple root of the characteristic equation of Q.

LEMMA 2.2. *If $Q(0, d-1)$ is strongly ergodic with left normalized eigenvector u_{d-1} corresponding to the eigenvalue 1, then*

$$u_r = u_{d-1}Q(0, r) \tag{2.8}$$

is the left normalized eigenvector of $Q(\tau d + r + 1, \pi d + r)$ *and*

$$u_m Q(\tau d + m + 1, \pi d + r) = u_r,$$

for $m, r = 0, 1, \cdots, d - 1$, $m < r$, *and* $\pi, \tau = 1, 2, \cdots, \tau < \pi$.

Proof. From (2.7) and (2.8) we have

$$u_r Q(\tau d + r + 1, \pi d + r) = u_{d-1} Q(0, r) Q(r + 1, d - 1) Q(0, d - 1)^{\pi - \tau - 1} Q(0, r)$$

$$= u_{d-1} Q(0, d - 1)^{\pi - \tau} Q(0, r) = u_r$$

and

$$u_m Q(\tau d + m + 1, \pi d + r) = u_{d-1} Q(0, m) Q(m + 1, d - 1) Q(0, d - 1)^{\pi - \tau - 1} Q(0, r)$$

$$= u_r. \qquad \square$$

If $Q(0, d - 1)$ is strongly ergodic with left normalized eigenvector u_{d-1}, from (2.7) and (2.8) we get that uniformly in m,

$$\lim_{\pi \to \infty} Q(\tau d + m, \pi d + r) = Q(m, d - 1) 1' u_{d-1} Q(0, r) = 1' u_r, \tag{2.9}$$

where $1 = (1, \cdots, 1)$. We also provide the following lemma (Isaacson and Madsen (1976)).

LEMMA 2.3. *Let* $\{a_s\}$ *be a sequence of nonnegative numbers such that* $\{a_s / \sum_{q=0}^{s} a_q\} \to 0$ *as* $s \to \infty$. *Also let* $\{B_s\}$ *be a sequence of finite matrices converging to* B; *then*

$$\lim_{s \to \infty} (B_0 a_s + B_1 a_{s-1} + \cdots + B_s a_0) \Big/ \left(\sum_{q=0}^{s} a_q \right) = B.$$

Define $n_r = (n_{r,i})$ and $V_r = (v_{r,ij})$, for $i, j = 1, \cdots, k$, $r = 0, 1, \cdots, d - 1$, to be a sequence of vectors and matrices with double index notation. The following theorem is the main result of this section, which characterizes ergodic cyclicity in NHMS.

THEOREM 2.1. *Consider an* NHMS *in discrete time such that the vector of means and the covariance matrix of the grade sizes satisfy the difference equations* (2.1) *and* (2.2), *respectively. Assume that the system undergoes a cyclic behaviour with period* $d > 1$, *i.e.,* $P(md + r) = P(r)$, $p_{k+1}(md + r) = p_{k+1}(r)$, *and* $p_0(md + r) = p_0(r)$ *for any* $m = 1, 2, \cdots$ *and* $r = 0, 1, \cdots, d - 1$. *Suppose further that* $\lim_{t \to \infty} T(t) = T < \infty$ *and* $M(t) \geqq 0$, *for* $t = 0, 1, 2, \cdots$, *i.e., the system is always expanding. If* $Q(0, d - 1)$ *is strongly ergodic with left normalized eigenvector* u_{d-1} *corresponding to the eigenvalue* 1, *then for any initial distribution,*

$$\lim_{s \to \infty} \bar{n}(sd + r + 1) = \bar{n}_{r+1}(\infty) = T u_r \tag{2.10}$$

and

$$\lim_{s \to \infty} V(sd + r + 1) = V_{r+1}(\infty) = T[\mathrm{Diag}\,(u_r) - u_r' u_r], \tag{2.11}$$

where

$$u_r = u_{d-1} Q(0, r), \qquad r = 0, 1, \cdots, d - 1.$$

Proof. Without loss of generality, let $t = sd + r$. Since $Q(0, d - 1)$ is strongly ergodic with left normalized eigenvector u_{d-1}, from (2.9) we get

$$\lim_{t \to \infty} n(0) Q(0, t) = \lim_{s \to \infty} n(0) Q(0, sd + r) = n(0) 1' u_r = T(0) u_r. \tag{2.12}$$

Denote by $U(t)$ the sum on the right side of (2.4). The cyclic behaviour of the system together with (2.7) implies

$$
\begin{aligned}
U(t) = U(sd+r) &= \sum_{\tau=0}^{sd+r} M(\tau)p_0(\tau)Q(\tau+1, sd+r) \\
&= \sum_{m=0}^{d-1}\sum_{h=0}^{s-1} M(hd+m)p_0(hd+m)Q(hd+m+1, sd+r) \\
&\quad + \sum_{m=0}^{r} M(sd+m)p_0(sd+m)Q(sd+m+1, sd+r) \qquad (2.13) \\
&= \sum_{m=0}^{d-1}\sum_{h=0}^{s-1} M(hd+m)p_0(m)Q(m+1, d-1)Q(0, d-1)^{s-h-1}Q(0, r) \\
&\quad + \sum_{m=0}^{r} M(sd+m)p_0(m)Q(m+1, r).
\end{aligned}
$$

Since $\lim_{t\to\infty} T(t) = T < \infty$ and $M(t)$ is nonnegative, we have that

$$\lim_{t\to\infty} M(t) = \lim_{t\to\infty} [T(t+1) - T(t)] = 0.$$

This ensures that the second term on the right side of (2.13) goes to zero as $s \to \infty$. It is sufficient to look at

$$
\begin{aligned}
\lim_{s\to\infty} U(sd+r) &= \lim_{s\to\infty} \sum_{m=0}^{d-1}\sum_{h=0}^{s-1} M(hd+m)p_0(m) \\
&\quad \times Q(m+1, d-1)Q(0, d-1)^{s-h-1}Q(0, r) \qquad (2.14) \\
&= \lim_{s\to\infty} \sum_{m=0}^{d-1}\sum_{q=0}^{s-1} M(qd+m)p_0(m)Q(m+1, d-1) \\
&\quad \times \sum_{h=0}^{s-1} \frac{M(hd+m)Q(0, d-1)^{s-h-1}}{\sum_{q=0}^{s-1} M(qd+m)} Q(0, r).
\end{aligned}
$$

The assumption $\lim_{t\to\infty} T(t) = T(0) + \sum_{h=0}^{\infty} M(h) < \infty$ implies that

$$\sum_{m=0}^{s} M(md+r) < \sum_{h=0}^{sd+r} M(h) < T - T(0) < \infty.$$

Thus the bounded nonnegative series

$$\sum_{m=0}^{\infty} M(md+r), \qquad r = 0, 1, \cdots, d-1,$$

are convergent. From Lemma 2.3 we get

$$\lim_{s\to\infty} \sum_{h=0}^{s} \frac{M(hd+m)Q(0, d-1)^{s-h}}{\sum_{q=0}^{s} M(qd+m)} = 1' u_{d-1}. \qquad (2.15)$$

From (2.14) and (2.15),

$$\lim_{s\to\infty} U(sd+r) = \sum_{m=0}^{d-1}\sum_{q=0}^{\infty} M(qd+m)p_0(m)1' u_{d-1}Q(0, r). \qquad (2.16)$$

Since $p_0(m)1' = 1$, for $m = 1, 2, \cdots$, we have

$$p_0(m)1'u_{d-1}Q(0,r) = u_r, \qquad r = 0, 1, \cdots, d-1. \tag{2.17}$$

Then from (2.16) and (2.17), we get

$$\lim_{s\to\infty} U(sd+r) = \sum_{q=0}^{\infty} M(q)u_r = \{T - T(0)\}u_r, \tag{2.18}$$

and from (2.4), (2.12), and (2.18),

$$\lim_{s\to\infty} \bar{n}(sd+r+1) = \bar{n}_{r+1}(\infty) = T(0)u_r + \{T - T(0)\}u_r = Tu_r. \tag{2.19}$$

We can similarly show that

$$\begin{aligned}\lim_{s\to\infty} \sum_{\tau=0}^{sd+r} & M(\tau)Q(\tau+1, sd+r)'p_0(\tau)'p_0(\tau)Q(\tau+1, sd+r) \\ &= \{T - T(0)\}u_r'u_r\end{aligned} \tag{2.20}$$

and

$$\lim_{s\to\infty} Q(0, sd+r)' \operatorname{Diag}\{n(0)\}Q(0, sd+r) = T(0)u_r'u_r. \tag{2.21}$$

Finally, from (2.5), (2.19), (2.20), and (2.21), we arrive at (2.11). □

Remark 2.1. Relations (2.10) and (2.11) show that the system is cyclically strongly ergodic and the limiting first and second central moments of the grade sizes are of multinomial type with size T and probability vectors u_r, for $r = 0, 1, \cdots, d-1$.

A matrix $A = [a_{ij}]$ is said to be nonnegatively invertible if A^{-1} exists and has nonnegative elements. We also need the following lemma (Seneta (1981, p. 30)).

LEMMA 2.4. *Let A be a nonnegative square matrix and let λ be the largest eigenvalue of A. The matrix $(\alpha I - A)$ is nonnegatively invertible if and only if $\alpha > \lambda$.*

Consider the stochastic k-simplex

$$\mathscr{D}^k \equiv \{q \mid q \geqq 0, q1' = 1\}$$

(where $0 = (0, \cdots, 0)$) of the k-dimensional Euclidean space $\mathbb{R}^k$. Geometrically, $\mathscr{D}^k$ has dimension $k-1$, being a subset in the positive orthant of the $(k-1)$-dimensional hyperplane

$$\mathscr{H}^{k-1} = \{q \mid q1' = 1\}$$

contained in $\mathbb{R}^k$. For a set $\mathbf{A}$ denote by $\mathscr{C}\{\mathbf{A}\}$ its convex hull. Let $q(t) = \bar{n}(t)/T(t)$ be the relative structure of the system at time t and $q_r = \bar{n}_{r+1}(\infty)/T$, $r = 0, 1, \cdots, d-1$. Denote by $\mathscr{L}_r(d)$ the set of limiting distributions q_r, obtained as the vectors of the sequence $p_0(i)$, $i = 0, 1, \cdots, d-1$, are varying over $\mathscr{D}^k$, for $r = 0, 1, \cdots, d-1$. From (2.10) and Lemma 2.2,

$$\mathscr{L}_r(d) \equiv \{q_r \mid q_r = q_r Q(r+1, d+r), q_r \in \mathscr{D}^k\}, \qquad r = 0, 1, \cdots, d-1.$$

The following result determines these sets.

THEOREM 2.2. *As the sequence of vectors, $p_0(i)$, $i = 0, 1, \cdots, d-1$, varies over $\mathscr{D}^k$, the set $\mathscr{L}_r(d)$ of limiting distributions q_r is that subset of $\mathscr{D}^k$ characterized by the property*

$$\mathscr{L}_r(d) \equiv \{q_r \mid q_r \geqq q_r P(r+1, d+r), q_r \in \mathscr{D}^k\}, \tag{2.22}$$

whereas the boundary of $\mathscr{L}_r(d)$ is the convex hull of points with vertices of the normalized rows of

$$[I-P(r+1,d+r)]^{-1}, \qquad r=0,1,\cdots,d-1.$$

Proof. Since q_r is the left normalized eigenvector of $Q(r+1, d+r)$ corresponding to the eigenvalue 1, from (2.3) we get

$$q_r = q_r \prod_{i=r+1}^{d+r} [P(i)+p_{k+1}(i)'p_0(i)] \tag{2.23}$$

$$= q_r P(r+1,d+r)+x^r(d),$$

where

$$x^r(d)=q_r \sum_{i=1}^{2^d-1} \prod_{j=0}^{d-1} S_{i_j}(j) \geqq 0, \qquad r=0,1,\cdots,d-1,$$

with $S_0(j) = P(j)$, $S_1(j) = p_{k+1}(j)'p_0(j)$, $j = 0, 1, \cdots, d-1$; and the sequence of indices $i_0, i_1, \cdots, i_{d-1}$ is evaluated from the binary representation of the index i. Thus (2.23) implies (2.22). We next determine the boundary of (2.22). The matrix $P(r+1, d+r)$ is substochastic, so that $[I - P(r+1, d+r)]$ is nonnegatively invertible. From (2.23) we get

$$q_r = x^r(d)[I-P(r+1,d+r)]^{-1}, \qquad r=0,1,\cdots,d-1. \tag{2.24}$$

By setting $\mu_i^r(d) = e_i[I - P(r+1, d+r)]^{-1}1'$, where e_i is the row vector with 1 in the ith position and 0 elsewhere, (2.24) becomes

$$q_r = \sum_{i=1}^{k} x_i^r(d)\mu_i^r(d) \frac{e_i[I-P(r+1,d+r)]^{-1}}{\mu_i^r(d)},$$

where $x_i^r(d)\mu_i^r(d) \geqq 0$, $i = 1, \cdots, k$, and from (2.24), $\sum_{i=1}^{k} x_i^r(d)\mu_i^r(d) = 1$, $r = 0, 1, \cdots, d-1$. Thus for a given sequence $p_0(j), j = 0, 1, \cdots, d-1$, the vector q_r has been represented as a convex combination of $\{\mu_i^r(d)^{-1}e_i[I - P(r+1, d+r)]^{-1}, i = 1, \cdots, k\}$, $r = 0, 1, \cdots, d-1$. As the vectors $p_0(j), j = 0, 1, \cdots, d-1$ range over $\mathscr{D}^k$, q_r ranges over all points in the convex hull with vertices of the normalized rows of $[I - P(r+1, d+r)]^{-1}$, $r = 0, 1, \cdots, d-1$. Suppose conversely that $\bar{q}_r$ is an element of $\mathscr{C}\{\mu_i^r(d)^{-1}e_i[I - P(r+1, d+r)]^{-1}, i = 1, \cdots, k\}$. Then there exists a set of (nonnegative) weights $f^r(d) = [f_1^r(d), \cdots, f_k^r(d)], f^r(d)1' = 1$, such that

$$\bar{q}_r = \sum_{i=1}^{k} f_i^r(d) \frac{e_i[I-P(r+1,d+r)]^{-1}}{\mu_i^r(d)}, \qquad r=0,1,\cdots,d-1.$$

By setting $y_i^r(d) = f_i^r(d)/\mu_i^r(d) > 0$, $i = 1, \cdots, k$, and $y^r(d) = [y_1^r(d), \cdots, y_k^r(d)]$, we get

$$\bar{q}_r[I-P(r+1,d+r)] = y^r(d) \geqq 0$$

and thus (2.22) is valid, i.e., $\bar{q}_r \in \mathscr{L}_r(d)$ for $r = 0, 1, \cdots, d-1$. □

3. Transient behaviour within the cycle. In this section we investigate the transient behaviour of the system during the period of a cycle with respect to stability and quasi stationarity. These are two concepts of central importance in the study of populations (see, for example, Feichtinger (1976), Vajda (1978), Bartholomew (1984), Bartholomew and Forbes (1979), and Vassiliou (1981), (1984)). In what follows, for an NHMS in

discrete time that undergoes a cyclic behaviour, we are going to identify those quasi-stationary structures that are stable. This type of characterization was first addressed in Gerontidis (1990a) for the system with given input, with time-dependent growth factor under the continuous time formulation.

DEFINITION 3.1. An NHMS is α-stable during the sth cycle for $s = 0, 1, \cdots$, if

$$\bar{n}(sd+r+1)=\alpha\bar{n}(sd+r), \qquad r=0,1,\cdots,d-1, \tag{3.1}$$

where α is the stable growth factor per unit time.

DEFINITION 3.2. An NHMS under cyclic behaviour of period d has a quasi-stationary (or maintainable (Forbes (1971))) relative structure q during the sth cycle for $s = 0, 1, \cdots$ if

$$q(sd+r)=q(sd), \qquad r=0,1,\cdots,d-1. \tag{3.2}$$

If a structure $\bar{n}(sd + r)$ is stable at the sth cycle, it is also quasi-stationary since

$$q(sd+r+1)=\frac{\bar{n}(sd+r+1)}{\bar{n}(sd+r+1)1'}=\frac{\alpha\bar{n}(sd+r)}{\alpha\bar{n}(sd+r)1'}=q(sd+r), \tag{3.3}$$

for $r = 0, 1, \cdots, d - 1$, $s = 0, 1, \cdots$. Conversely, the quasi-stationarity property does not necessarily imply the stability of $\bar{n}(sd + r)$ in the sense of (3.1) (Feichtinger (1976)). It is thus of interest to characterize stability in quasi-stationary NHMS, which undergoes a cyclic behaviour. The following theorem is the main result of this section.

THEOREM 3.1. *An* NHMS *that undergoes a cyclic behaviour of period d has an α-stable structure during the sth cycle, $s = 0, 1, \cdots$:*

(i) *For $\alpha > 1$, if and only if it is quasi-stationary with initial structure given by*

$$n(0)=M(0)p_0(0)[\alpha I-Q(0)]^{-1} \tag{3.4}$$

and expansion sequence of the form

$$M(sd+r)=M(r)\alpha^{sd}, \tag{3.5}$$

where

$$M(r)=M(0)\frac{w(0)}{w(r)}\alpha^r \tag{3.6}$$

with $w(r) = p_0(r)[\alpha I - Q(r)]^{-1}1'$, $r = 0, 1, \cdots, d - 1$, and $M(0)$ a given constant.

(ii) *For $\alpha = 1$, if and only if $M(r) = 0$ and $\bar{n}(r)$ is the left eigenvector of $Q(r)$ corresponding to the eigenvalue 1, for $r = 0, 1, \cdots, d - 1$.*

In proving Theorem 3.1 we need the following lemma.

LEMMA 3.1. *For a quasi-stationary* NHMS *undergoing a cyclic behaviour of period d, relations* (3.4) *and* (3.5) *are equivalent to*

$$\bar{n}(sd+r)=M(r)\alpha^{sd}p_0(r)[\alpha I-Q(r)]^{-1}, \tag{3.7}$$

where $M(r)$ is as in (3.6) *with $\alpha > 1$, for $r = 0, 1, \cdots, d - 1$, $s = 0, 1, \cdots$.*

Proof of Lemma 3.1. Let (3.7) hold; then by setting $s = 0$ and $r = 0$, we get (3.4). Since the system is quasi-stationary from (3.7), we have

$$\frac{p_0(r+1)[\alpha I-Q(r+1)]^{-1}}{p_0(r+1)[\alpha I-Q(r+1)]^{-1}1'}=\frac{p_0(r)[\alpha I-Q(r)]^{-1}}{p_0(r)[\alpha I-Q(r)]^{-1}1'} \tag{3.8}$$

for $r = 0, 1, \cdots, d-1$, and from (2.1) and (3.5)–(3.8),

$$\begin{aligned} M(sd+r)p_0(sd+r) &= \bar{n}(sd+r+1) - \bar{n}(sd+r)Q(sd+r) \\ &= M(r+1)\alpha^{sd}p_0(r+1)[\alpha I - Q(r+1)]^{-1} \\ &\quad - M(r)\alpha^{sd}p_0(r)[\alpha I - Q(r)]^{-1}Q(r) \\ &= M(0)w(0)\alpha^{sd+r}\left\{\alpha\frac{p_0(r+1)[\alpha I - Q(r+1)]^{-1}}{w(r+1)} - \frac{p_0(r)[\alpha I - Q(r)]^{-1}}{w(r)}Q(r)\right\} \\ &= M(0)\frac{w(0)}{w(r)}\alpha^{sd+r}p_0(r). \end{aligned}$$

Postmultiplying both sides by $1'$ together with (3.6), we get (3.5). The converse, i.e., that (3.4) and (3.5) imply (3.7), can be proved by induction. □

Proof of Theorem 3.1. (i) Assuming stability from (2.1) and (3.1),

$$\bar{n}(sd+r) = M(sd+r)p_0(r)[\alpha I - Q(r)]^{-1}. \tag{3.9}$$

By setting $r = 0$ and $s = 0$ in (3.9), we get (3.4). Combining (3.1) and (3.9) we have

$$M(sd+r+1)p_0(r+1)[\alpha I - Q(r+1)]^{-1} = \alpha M(sd+r)p_0(r)[\alpha I - Q(r)]^{-1}.$$

Postmultiplying both sides by $1'$ gives

$$M(sd+r+1) = \alpha M(sd+r)\frac{w(r)}{w(r+1)}, \quad r = 0, 1, \cdots, d-1, \quad s = 0, 1, \cdots,$$

from which (3.5) follows, by means of (3.6).

Assume now that the system is quasi-stationary and that (3.4) and (3.5) both hold. Then by Lemma 3.1, relation (3.7) is also valid, giving

$$\begin{aligned} \bar{n}(sd+r)[\alpha I - Q(r)] &= M(r)\alpha^{sd}p_0(r) \\ &= M(sd+r)p_0(r), \end{aligned}$$

from which we get stability.

(ii) The proof in this case is similar to that in Feichtinger (1976) and is omitted here. □

4. Ergodic cyclicity in continuous time. In this section we are going to develop the continuous time analogue of the discrete time result derived in § 2. For an NHMS in continuous time (Gerontidis (1990a)), define $r_{ij}(t)$ to be the intensity of transition from grade i to j, for $i, j = 1, \cdots, k$ and in matrix notation $R(t) = [r_{ij}(t)]$. Also let $r_{k+1}(t) = [r_{1,k+1}(t), \cdots, r_{k,k+1}(t)]$ be the row vector of loss intensities from the various states. Then $z_{ij}(t) = r_{ij}(t) + r_{i,k+1}(t)p_{0j}(t)$ is the total force of an i to j transition (of any kind) and

$$Z(t) = R(t) + r_{k+1}(t)'p_0(t) \tag{4.1}$$

is the intensity matrix of the associated nonhomogeneous Markov replacement process with all row sums equal to zero. We assume that for any $t \geqq 0$, $\int_0^t -z_{ii}(u)\,du < \infty$, $i = 1, \cdots, k$, (Iosifescu (1980)) and $\sup\{\|Z(t)\|, 0 \leqq t < \infty\} < \infty$. Denote by

$Q(s, t)$ the stochastic probability matrix associated with $Z(t)$, having the product integral representation

$$Q(s,t)=\prod_{x=s}^{t}[I+Z(x)dx]. \tag{4.2}$$

$T(t)$ is now a given continuous function of time; define $M(t) = dT(t)/dt$ to be the rate of increase (expansion) of the system. For an expanding system following the conditional argument described in Gerontidis (1990b) for the homogeneous case, we can show that $\bar{n}(t)$ and $V(t)$ satisfy the integral equations

$$\bar{n}(t)=n(0)Q(0,t)+\int_0^t M(s)p_0(s)Q(s,t)\,ds \tag{4.3}$$

and

$$\begin{aligned} V(t)=&\operatorname{Diag}\{\bar{n}(t)\}-Q(0,t)'\operatorname{Diag}\{n(0)\}Q(0,t)\\ &-\int_0^t M(s)Q(s,t)'p_0(s)'p_0(s)Q(s,t)\,ds. \end{aligned} \tag{4.4}$$

DEFINITION 4.1. An NHMS in continuous time undergoes a cyclic behaviour with period $\tau > 0$, if and only if $R(t + \tau) = R(t)$, $r_{k+1}(t + \tau) = r_{k+1}(t)$, and $p_0(t + \tau) = p_0(t)$ for any $t \geqq 0$ and some real $\tau > 0$, called the period of the cycle.

For a system in continuous time undergoing a cyclic behaviour, from (4.1) we get

$$Z(t+\tau)=R(t+\tau)+r_{k+1}(t+\tau)'p_0(t+\tau)=Z(t)$$

and

$$\frac{\partial Q(0,t+\tau)}{\partial t}=Q(0,t+\tau)Z(t). \tag{4.5}$$

Since $Q(0, t)$ is also a solution of (4.5) the theory of differential equations with periodic coefficients (see also Gerontidis (1990a)) implies

$$Q(0,t+\tau)=Q(0,\tau)Q(0,t) \tag{4.6}$$

and

$$Q(0,n\tau)=Q(0,\tau)^n. \tag{4.7}$$

Defining

$$B=-\frac{1}{\tau}\sum_{n=1}^{\infty}\frac{[I-Q(0,\tau)]^n}{n}=\frac{1}{\tau}\ln Q(0,\tau),$$

where ln $(\cdot)$ is the logarithm matrix function (Cuthbert (1972)), we get that

$$C(t)=\exp(-tB)Q(0,t) \tag{4.8}$$

is nonsingular, stochastic, and periodic of period τ, with the property $C(0) = C(\tau) = I$. From the Chapman–Kolmogorov equation, we have

$$Q(0,h+t')=Q(0,h)Q(h,h+t')$$

and for $t = h + t'$, this is

$$Q(h,t)=Q(0,h)^{-1}Q(0,t). \tag{4.9}$$

Theorem 1.4 in Dollard and Friedman (1984, p. 9) states that the matrix $Q(0, h)$, $0 \leqq h \leqq t$, is nonsingular; thus $Q^{-1}(0, h)$ always exists. From (4.8) we get that

$$Q(0,t) = \exp(tB)C(t). \tag{4.10}$$

Substitution of (4.10) into (4.9) gives

$$Q(h,t) = C(h)^{-1} \exp\{(t-h)B\}C(t). \tag{4.11}$$

Define $Q(0, \tau)$ to be the "cycle matrix." The following lemmas are the continuous time analogues of Lemmas 2.1 and 2.2.

LEMMA 4.1. *For any* $0 \leqq h < t < \infty$ *it holds that*

$$Q(h,t) = Q(x,\tau)Q(0,\tau)^{[(t-h)/\tau]-1}Q(0,r), \tag{4.12}$$

where $0 \leqq x, r < \tau$, *and* $[\cdot]$ *is the lowest integer function.*

Proof. Without loss of generality, let $h = m\tau + x$ and $t = s\tau + r$, for $0 \leqq x, r < \tau$, $m, s = 0, 1, \cdots, m < s$. Using (4.7) and (4.11) we get

$$\begin{aligned} Q(m\tau+x, s\tau+r) &= Q\{m\tau+x,(m+1)\tau\}Q\{(m+1)\tau, s\tau\}Q(s\tau, s\tau+r) \\ &= C(x)^{-1}\exp\{(\tau-x)B\}\exp\{(s-m-1)\tau B\}\exp(rB)C(r) \\ &= Q(x,\tau)Q(0,\tau)^{s-m-1}Q(0,r). \end{aligned}$$

□

LEMMA 4.2. *If* $Q(0, \tau)$ *is strongly ergodic with left normalized eigenvector* $u(0)$ *corresponding to the eigenvalue* 1, *then*

$$u(r) = u(0)Q(0,r)$$

is the left normalized eigenvector of $Q(m\tau + r, s\tau + r)$ *and*

$$\lim_{s\to\infty} Q(m\tau+x, s\tau+r) = Q(x,\tau)1'u(0)Q(0,r) = 1'u(r) \tag{4.13}$$

uniformly in x, $0 \leqq x < r < \tau$, $m, s = 0, 1, \cdots, m < s$.

The following theorem characterizes cyclic strong ergodicity in NHMS in continuous time.

THEOREM 4.1. *Consider an* NHMS *in continuous time such that the vector of means and the covariance matrix satisfy the integral equations* (4.3) *and* (4.4), *respectively. Let us also assume that the system undergoes a cyclic behaviour with period* $\tau > 0$ *and that* $\lim_{t\to\infty} T(t) = T < \infty$ *and* $M(t)$ *is nonnegative and monotone nonincreasing. If* $Q(0, \tau)$ *is strongly ergodic with left normalized eigenvector* $u(0)$ *corresponding to the eigenvalue* 1, *then whatever the initial distribution,*

$$\lim_{s\to\infty} \bar{n}(s\tau+r) = \bar{n}_r(\infty) = Tu(r)$$

and

$$\lim_{s\to\infty} V(s\tau+r) = V_r(\infty) = T[\operatorname{Diag}\{u(r)\} - u(r)'u(r)],$$

where

$$u(r) = u(0)Q(0,r), \qquad 0 \leqq r < \tau.$$

Proof. We provide an indication of the proof. Let $t = s\tau + r$ and denote by $W(t)$ the integral of (4.3). The cyclic behaviour of the system together with (4.13) implies

$$\begin{aligned}
&\lim_{t\to\infty} W(t)\\
&= \lim_{s\to\infty} W(s\tau+r) = \lim_{s\to\infty}\int_0^{s\tau+r} M(x)p_0(x)Q(x,s\tau+r)\,dx\\
&= \lim_{s\to\infty}\int_0^{\tau}\sum_{m=0}^{s-1} M(m\tau+x)p_0(x)Q(x,\tau)Q\{0,(s-m-1)\tau\}Q(0,r)\,dx\\
&\qquad + \lim_{s\to\infty}\int_0^{r} M(s\tau+x)p_0(x)Q(x,r)\,dx \qquad (4.14)\\
&= \lim_{s\to\infty}\int_0^{\tau}\sum_{q=0}^{s-1} M(q\tau+x)p_0(x)Q(x,\tau)\sum_{m=0}^{s-1}\frac{M(m\tau+x)Q(0,\tau)^{s-m-1}}{\sum_{q=0}^{s-1}M(q\tau+x)}Q(0,r)\,dx\\
&= \int_0^{\tau}\sum_{m=0}^{\infty} M(m\tau+x)p_0(x)Q(x,\tau)1'u(0)Q(0,r)\,dx = \int_0^{\infty} M(x)\,dxu(r)\\
&= \{T-T(0)\}u(r).
\end{aligned}$$

The rest of the proof follows from that of Theorem 2.1. □

Remark 4.1. Relation (4.14) is a multidimensional generalization of the key renewal theorem result in a cyclic environment.

Let $\mathscr{L}_\tau(r)$ be the family of limiting distributions $q(r) = \bar{n}_r(\infty)/T$, $0 \leqq r < \tau$.

THEOREM 4.2. *$\mathscr{L}_\tau(r)$ is the set characterized by*

$$\mathscr{L}_\tau(r) \equiv \left\{ q(r) \mid q(r) \geqq q(r)\prod_{t=r}^{\tau+r}[I+R(t)dt], q(r)\in\mathscr{D}^k \right\},$$

moreover, $\mathscr{L}_\tau(r)$ is the convex hull of points with vertices of the normalized rows of

$$\prod_{t=r}^{\tau+r}\{I+R(t)dt\}^{-1}, \qquad 0\leqq r<\tau.$$

Proof. From (4.1) and (4.2) we get

$$\begin{aligned}
q(r) &= q(r)\prod_{t=r}^{\tau+r}[I+\{R(t)+r_{k+1}(t)'p_0(t)\}dt]\\
&= q(r)\prod_{t=r}^{\tau+r}[I+R(t)dt]+y_\tau(r),
\end{aligned}$$

where

$$\prod_{t=r}^{\tau+r}[I+R(t)dt]$$

is substochastic and $y_\tau(r) \geqq 0$, $0 \leqq r < \tau$. □

5. Rate of convergence to the cyclic distribution. In this section we will investigate the important question of the rate at which the system converges to the cyclic distribution.

In fact, we intend to provide conditions under which $\bar{n}(t)$ and $V(t)$ converge to their limits exponentially fast. Rates of convergence for the first and second central moments of the grade sizes for the discrete time case have been studied in Vassiliou and Tsaklidis (1989), whereas in continuous time, the strongly ergodic case is treated in Gerontidis and Vassiliou (1990).

DEFINITION 5.1. The matrix $A(t)$ converges to the limit matrix A exponentially fast, if there exist constants $c > 0$ and $\beta > 0$ such that

$$\|A(t)-A\| < c \cdot \exp(-\beta t), \qquad t \geqq 0.$$

The following lemma provides conditions on the transition matrix of a cyclic continuous time Markov process to converge to its limit exponentially fast.

LEMMA 5.1. *Consider a cyclic nonhomogeneous Markov process with intensity matrix function $Z(t)$ such that $Z(t+\tau) = Z(t)$ for some $\tau > 0$ and any $t \geqq 0$. Assume further that $Q(0, \tau)$ is strongly ergodic with limit matrix Q^*. Then there exist constants $c > 0$ and $\lambda > 0$ such that for any $0 \leqq h < t < \infty$:*

(i) *If $h = [h/\tau]\tau$, then*

$$\|Q(h,t)-Q^*Q(0,t-[t/\tau]\tau)\| < c \cdot \exp\{-\lambda[(t-h)/\tau]\tau\}.$$

(ii) *If $h > [h/\tau]\tau$, then*

$$\|Q(h,t)-Q^*Q(0,t-[t/\tau]\tau)\| < c \cdot \exp\{-\lambda([(t-h)/\tau]-1)\tau\},$$

where $[\cdot]$ is the lowest integer function.

Proof. (i) Suppose that $h = m\tau$ and $t = s\tau + r$, $0 \leqq r < \tau$, $m, s = 0, 1, \cdots, s \geqq m$. Since $Q(0, \tau)$ is strongly ergodic, there exist constants $c > 0$, $\lambda > 0$ such that

$$\begin{aligned} \|Q(m\tau, s\tau+r)-Q^*Q(0,r)\| &= \|Q(m\tau, s\tau)-Q^*\|\|Q(0,r)\| \\ &< c \cdot \exp\{-\lambda(s-m)\tau\}. \end{aligned} \tag{5.1}$$

(ii) Without loss of generality, let $h = m\tau + x$ and $t = s\tau + r$, for $0 \leqq x, r < \tau$, $m, s = 0, 1, \cdots, s > m$. From (5.1), since Q^* is a row constant matrix, we get

$$\begin{aligned} \|Q(m\tau+x, s\tau+r)-Q^*Q(0,r)\| &\leqq \|Q(x,\tau)\|\|Q\{(m+1)\tau, s\tau\}-Q^*\|\|Q(0,r)\| \\ &< c \cdot \exp\{-\lambda(s-m-1)\tau\}. \end{aligned}$$

□

Rewrite (4.4) by using double subscripts and setting the elements of $V(t)$ in vector form as follows

$$\begin{aligned} v(t) &= \bar{n}(t)J - n(0)J\{Q(0,t)\times Q(0,t)\} \\ &\quad - \int_0^t M(s)\{p_0(s)\times p_0(s)\}\{Q(s,t)\times Q(s,t)\}\, ds, \end{aligned} \tag{5.2}$$

where $v(t)$ is a k^2-vector corresponding to $V(t)$, $p_0(s) \times p_0(s)$ is the k^2-direct product vector and $Q(s, t) \times Q(s, t)$ is the $k^2 \times k^2$-direct product matrix as defined in Peace (1965, p. 322). J is a $k \times k^2$-matrix, the ith row of which has 1 in the $\{(i-1)k + i\}$th column and zeros elsewhere. Postmultiplying a k-vector by J has the effect of expanding it to a k^2-vector by inserting zeros in positions labelled (i, i') for which $i \neq i'$. Note that $\|J\| = 1$. We shall also make use of the following norm properties.

(i) $\|p_0\| = \|p_0 \times p_0\| \leqq 1$, p_0 stochastic vector.

(ii) $\|Q \times Q - Q^* \times Q^*\| \leqq 2\|Q - Q^*\|$, Q, Q^* stochastic matrices.

We now provide the main result of this section, which characterizes exponential convergence of $\bar{n}(s\tau + r)$ and $v(s\tau + r)$, $0 \leqq r < \tau$.

THEOREM 5.1. *Assume that for an* NHMS *in continuous time that undergoes a cyclic behaviour with period* $\tau > 0$, $\bar{n}(t)$ *and* $v(t)$ *satisfy the integral equations* (4.3) *and* (5.2), *respectively. Assume further that*

(i) $Q(0, \tau)$ *is strongly ergodic with limit matrix* Q^*.

(ii) $M(t)$ *is nonnegative and* $\lim_{t\to\infty} M(t) = 0$ *at an exponential rate.*

Then $\bar{n}(s\tau + r)$ *and* $v(s\tau + r)$, $0 \leqq r < \tau$, *converge to their limits exponentially fast.*

Proof. There exist constants $c_1 > 0$, $\nu > 0$ such that for any $t \geqq 0$,

$$|M(t)| < c_1 \exp(-\nu t). \tag{5.3}$$

Let $t = s\tau + r$. Since $Q(0, \tau)$ is strongly ergodic with limit Q^*, by Lemma 5.1 we can choose constants $c_0 > 0$ and $0 < \lambda < \nu$ such that

$$\|Q(m\tau + x, s\tau + r) - Q^*Q(0, r)\| < c_0 \exp\{-\lambda(s - m - 1)\tau\}. \tag{5.4}$$

The cyclic behaviour of the system together with (5.3), (5.4) implies

(5.5)

$$\begin{aligned}
&\Big\| \int_0^\tau \sum_{m=0}^{s-1} M(m\tau + x) p_0(x) Q(m\tau + x, s\tau + r)\, dx + \int_0^r M(s\tau + r) p_0(x) Q(0, r)\, dx \\
&\qquad - \int_0^\tau \sum_{m=0}^{\infty} M(m\tau + x) p_0(x) Q(x, \tau) Q^*Q(0, r)\, dx \Big\| \\
&\leqq \int_0^\tau \Big\{ \sum_{m=0}^{s-1} |M(m\tau + x)| \, \|Q(m\tau + x, s\tau + r) - Q^*Q(0, r)\| \\
&\qquad + \sum_{m=s}^{\infty} |M(m\tau + x)| \, \|Q^*Q(0, r)\| \Big\} dx \\
&\qquad + \int_0^r |M(s\tau + x)|\, dx < c^*(r) \exp(-\lambda s\tau),
\end{aligned}$$

where

$$c^*(r) = \frac{c_1}{\nu}\left[c_0 \frac{\{1 - \exp(-\nu\tau)\}}{[1 - \exp\{(\lambda - \nu)\tau\}]} \exp(\lambda\tau) + 2 - \exp(-\nu r) \right] > 0, \qquad 0 \leqq r < \tau.$$

From (5.3) we get $\int_0^\infty M(u)\, du < c_1/\nu$ so that there exists $\lim_{t\to\infty} T(t) = T$, say, and from Theorem 4.1, $\lim_{s\to\infty} \bar{n}(s\tau + r) = \bar{n}_r(\infty)$, $0 \leqq r < \tau$. Finally, from (5.3)–(5.5),

$$\|\bar{n}(s\tau + r) - \bar{n}_r(\infty)\| < c(r) \exp(-\lambda s\tau), \tag{5.6}$$

where $c(r) = \{c_0 \|n(0)\| + c^*(r)\}$, for $0 \leqq r < \tau$. From (5.2), (5.4), (5.5), and (5.6) we get

$$\|v(s\tau + r) - v_r(\infty)\| < 3c(r) \exp(-\lambda s\tau), \qquad 0 \leqq r < \tau. \qquad \square$$

Remark 5.1. The assumption $\lambda < \nu$ has for $\bar{n}(s\tau + r)$ and $v(s\tau + r)$, $0 \leqq r < \tau$, the effect of following the rate by which $Q(0, \tau)$ converges to Q^*. Accordingly, the best possible rate of convergence of the strongly ergodic matrix $Q(0, \tau)$ to its limit Q^* is given by the second largest eigenvalue ρ of $Q(0, \tau)$ (Isaacson and Luecke (1978)).

6. Two illustrative examples. In this section we illustrate the previously derived theoretical results with two examples from the literature on manpower planning. Consider a hierarchical manpower system with five grades, grade 5 being the top and grade 1 the

bottom. Assume further that the system undergoes a cyclic behaviour with period $d = 3$ and basic data

$$P(3t) = \begin{bmatrix} 0.4 & 0.3 & 0 & 0 & 0 \\ 0 & 0.5 & 0.2 & 0 & 0 \\ 0 & 0 & 0.6 & 0.2 & 0 \\ 0 & 0 & 0 & 0.7 & 0.1 \\ 0 & 0 & 0 & 0 & 0.9 \end{bmatrix},$$

$$P(3t+1) = \begin{bmatrix} 0.5 & 0.2 & 0 & 0 & 0 \\ 0 & 0.6 & 0.2 & 0 & 0 \\ 0 & 0 & 0.7 & 0.1 & 0 \\ 0 & 0 & 0 & 0.7 & 0.2 \\ 0 & 0 & 0 & 0 & 0.9 \end{bmatrix},$$

$$P(3t+2) = \begin{bmatrix} 0.5 & 0.3 & 0 & 0 & 0 \\ 0 & 0.5 & 0.3 & 0 & 0 \\ 0 & 0 & 0.6 & 0.2 & 0 \\ 0 & 0 & 0 & 0.7 & 0.2 \\ 0 & 0 & 0 & 0 & 0.9 \end{bmatrix},$$

for $t = 0, 1, \cdots$. Then the three sets of limiting distributions $\mathscr{L}_r(3)$, $r = 0, 1$, are the convex hulls of points with vertices w_r^m, $m, r = 0, 1, 3$, i.e.,

$$\mathscr{L}_0(3) \equiv \mathscr{C} \begin{bmatrix} w_0^1(3) = (0.582 & 0.137 & 0.111 & 0.068 & 0.102) \\ w_0^2(3) = (0 & 0.517 & 0.166 & 0.127 & 0.190) \\ w_0^3(3) = (0 & 0 & 0.517 & 0.182 & 0.301) \\ w_0^4(3) = (0 & 0 & 0 & 0.446 & 0.554) \\ w_0^5(3) = (0 & 0 & 0 & 0 & 1.000) \end{bmatrix},$$

$$\mathscr{L}_1(3) \equiv \mathscr{C} \begin{bmatrix} w_1^1(3) = (0.557 & 0.145 & 0.124 & 0.062 & 0.112) \\ w_1^2(3) = (0 & 0.518 & 0.170 & 0.112 & 0.200) \\ w_1^3(3) = (0 & 0 & 0.504 & 0.167 & 0.329) \\ w_1^4(3) = (0 & 0 & 0 & 0.456 & 0.544) \\ w_1^5(3) = (0 & 0 & 0 & 0 & 1.000) \end{bmatrix},$$

$$\mathscr{L}_2(3) \equiv \mathscr{C} \begin{bmatrix} w_2^1(3) = (0.569 & 0.127 & 0.127 & 0.066 & 0.111) \\ w_2^2(3) = (0 & 0.543 & 0.170 & 0.109 & 0.331) \\ w_2^3(3) = (0 & 0 & 0.499 & 0.170 & 0.331) \\ w_2^4(3) = (0 & 0 & 0 & 0.470 & 0.530) \\ w_2^5(3) = (0 & 0 & 0 & 0 & 1.000) \end{bmatrix}.$$

In continuous time, the internal transitions of a hierarchical manpower system with four grades may be modeled by the intensity matrix

$$R(t) = \begin{bmatrix} -\left\{3 + \sin^2\left(\frac{2\pi}{3}t\right)\right\} & 4\sin^2\left(\frac{2\pi}{3}t\right) & 0 & 0 \\ 0 & -\left\{2 + \sin^2\left(\frac{2\pi}{3}t\right)\right\} & 3\sin^2\left(\frac{2\pi}{3}t\right) & 0 \\ 0 & 0 & -\left\{1 + \sin^2\left(\frac{2\pi}{3}t\right)\right\} & 2\sin^2\left(\frac{2\pi}{3}t\right) \\ 0 & 0 & 0 & -\cos^2\left(\frac{2\pi}{3}t\right) \end{bmatrix},$$

whereas losses can occur according to the wastage vector

$$r_5(t) = \left[3\cos^2\left(\frac{2\pi}{3}t\right), 2\cos^2\left(\frac{2\pi}{3}t\right), \cos^2\left(\frac{2\pi}{3}t\right), \cos^2\left(\frac{2\pi}{3}t\right)\right],$$

which are periodic with period $\tau = 1.5$. For a recruitment policy determined by $p_0(t) = p_0 = (1, 0, 0, 0)$, $Z(t)$ is also periodic with the same period and $Q(0, 1.5)$ is strongly ergodic with left normalized eigenvector $u(0) = (0.404, 0.054, 0.176, 0.366)$. For an initial structure $n(0) = (950, 650, 300, 100)$ and expansion given by $M(t) = 100\exp(-2.5t)$, we have $\lim_{t\to\infty} T(t) = 2040$. Applying Theorem 4.1 and simulating the system, we found that whatever the initial distribution, $n(t)$ and $v(t)$ converge to periodic limits with period $\tau = 1.5$ and some limiting values:

$$\bar{n}_{0.0}(\infty) = (824.9, 110.2, 358.4, 746.5),$$

$$\bar{n}_{0.3}(\infty) = (877.7, 277.8, 261.8, 622.7),$$

$$\bar{n}_{0.6}(\infty) = (306.1, 618.6, 407.2, 708.1),$$

$$\bar{n}_{0.9}(\infty) = (90.6, 362.1, 615.8, 971.5),$$

$$\bar{n}_{1.2}(\infty) = (345.8, 200.5, 494.1, 999.6),$$

and

$$v_{0.0}(\infty) = (491.4, -44.5, -144.9, -301.9, -44.5, 104.3, -19.3, -40.3, -144.5, -19.3, 295.4, -131.1, -301.9, -40.3, -131.1, 473.4),$$

$$v_{0.3}(\infty) = (500.1, -199.5, -112.6, -267.9, -119.5, 240.1, -35.6, -84.8, -112.6, -35.6, 288.3, -79.9, -267.9, -84.8, -79.9, 432.6),$$

$$v_{0.6}(\infty) = (260.2, -92.8, -61.1, -106.2, -92.8, 431.0, -123.5, -214.7, -61.1, -123.5, 326.0, -141.4, -106.2, -214.7, -141.4, 462.3),$$

$$v_{0.9}(\infty) = (86.6, -16.1, -27.3, -43.2, -16.1, 297.8, -109.3, -172.4, -27.3, -109.3, 429.9, -293.3, -43.2, -172.4, -293.3, 508.9),$$

$$v_{1.2}(\infty) = (287.2, -34.0, -83.7, -169.4, -34.0, 180.8, -48.6, -98.2, -83.7, -48.6, 374.5, -242.2, -169.4, -98.2, -242.1, 509.8),$$

with $\bar{n}_{1.5}(\infty) = \bar{n}_{0.0}(\infty)$ and $v_{1.5}(\infty) = v_{0.0}(\infty)$. Thus the limiting first and second central moments of the grade sizes are of multinomial type with size 2040 and probability vectors $u(r)$, $0 \leqq r < 1.5$.

Acknowledgment. The author would like to thank the referee for his constructive comments, which improved the presentation on an earlier version of this paper.

REFERENCES

[1] D. J. BARTHOLOMEW, *Stochastic Models for Social Processes*, Third Edition, John Wiley, New York, 1982.

[2] ———, *Recent developments in nonlinear stochastic modelling of social processes*, Canad. J. Statist., 12(1984), pp. 39–52.

[3] D. J. BARTHOLOMEW AND A. F. FORBES, *Statistical Techniques for Manpower Planning*, John Wiley, New York, 1979.

[4] J. CONLISK, *Interactive Markov chains*, J. Math. Sociol., 4(1976), pp. 157–185.

[5] J. R. CUTHBERT, *On the uniqueness of the logarithm for Markov semigroups*, J. London Math. Soc., 4(1972), pp. 623–630.
[6] G. S. DAVIES, *A note on a continuous-time Markov manpower model*, J. Appl. Probab., 22(1985), pp. 932–938.
[7] J. D. DOLLARD AND C. N. FRIEDMAN, *Product Integration*, Cambridge University Press, Cambridge, U.K., 1984.
[8] G. FEICHTINGER, *On the generalization of stable age distributions to Gani-type person flow models*, Adv. in Appl. Probab., 8(1976), pp. 433–445.
[9] A. F. FORBES, *Promotion and recruitment policies for control of quasi-stationary hierarchical systems*, in Models for Manpower Systems, A. R. Smith, ed., English University Press, London, 1971, pp. 401–414.
[10] J. GANI, *Formulae for projecting enrollments and degrees awarded in universities*, J. Roy. Statist. Soc. Ser. A, 126(1963), pp. 400–409.
[11] I. I. GERONTIDIS, *On certain aspects of nonhomogeneous Markov systems in continuous time*. J. Appl. Probab., 27(1990a), pp. 530–544.
[12] ———, *On the variance-covariance matrix in Markovian manpower systems in continuous time*, Austral. J. Statist. 32(1990b), pp. 271–280.
[13] ———, *Periodic strong ergodicity in nonhomogeneous Markov systems*, J. Appl. Probab., 28(1991), pp. 58–73.
[14] I. I. GERONTIDIS AND P.–C. G. VASSILIOU, *Exponential ergodicity in nonhomogeneous Markov systems in continuous time*, manuscript, Mathematics Department, University of Thessaloniki, Thessaloniki, Greece, 1990.
[15] R. C. GRINOLD AND K. T. MARSHALL, *Manpower Planning Models*, North-Holland, New York, 1977.
[16] M. IOSIFESCU, *Finite Markov Processes and their Applications*, John Wiley, New York, 1980.
[17] D. L. ISAACSON AND W. R. MADSEN, *Markov Chains*, John Wiley, New York, 1976.
[18] D. ISAACSON AND G. R. LUECKE, *Strongly ergodic Markov chains and rates of convergence using spectral conditions*, Stochastic Process. Appl., 7(1978), pp. 113–121.
[19] J. KEILSON, *Markov Chain Models—Rarity and Exponentiality*, Springer-Verlag, Berlin, 1979.
[20] H. P. LESLIE, *On the use of matrices in certain population mathematics*, Biometrika, 33(1945), pp. 182–212.
[21] M. C. PEACE, III, *Matrix Methods of Algebra*, Academic Press, New York, 1965.
[22] J. H. POLLARD, *On the use of direct product matrix in analysing certain stochastic population models*, Biometrika, 53(1966), pp. 397–415.
[23] ———, *Mathematical Models for the Growth of Human Populations*, Cambridge University Press, Cambridge, U.K., 1973.
[24] E. SENETA, *Nonnegative Matrices and Markov Chains*, Second Edition, Springer-Verlag, Berlin, 1981.
[25] G. TSAKLIDIS AND P.–C. G. VASSILIOU, *Asymptotic periodicity of the vector of variances and covariances in nonhomogeneous Markov systems*, J. Appl. Probab., 25(1988), pp. 21–33.
[26] S. VAJDA, *Mathematics of Manpower Planning*, John Wiley, New York, 1978.
[27] P.–C. G. VASSILIOU, *Stability in a nonhomogeneous Markov chain model in manpower systems*, J. Appl. Probab., 18(1981), pp. 924–930.
[28] ———, *Asymptotic behaviour of Markov systems*, J. Appl. Probab., 19(1982), pp. 851–857.
[29] ———, *Cyclic behaviour and asymptotic stability of nonhomogeneous Markov systems*, J. Appl. Probab., 21(1984), pp. 315–325.
[30] ———, *Asymptotic variability of nonhomogeneous Markov systems under cyclic behaviour*, European J. Oper. Res., 27(1986), pp. 215–228.
[31] P.–C. G. VASSILIOU AND I. GERONTIDIS, *Variances and covariances of the grade sizes in manpower systems*, J. Appl. Probab., 22(1985), pp. 583–597.
[32] P.–C. G. VASSILIOU AND G. TSAKLIDIS, *The rate of convergence of the vector of variances and covariances in nonhomogeneous Markov systems*, J. Appl. Probab., 27(1989), pp. 776–783.
[33] A. YOUNG, *Demographic and ecological models for manpower planning*, in Aspects of Manpower Planning, D. J. Bartholomew and B. R. Morris, eds., English University Press, London, 1971, pp. 75–97.
[34] A. YOUNG AND G. ALMOND, *Predicting distributions of staff*, Comput. J., 3(1961), pp. 246–250.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 567–593, April 1992

009

REDUCTION TO TRIDIAGONAL FORM AND MINIMAL REALIZATIONS*

BERESFORD N. PARLETT†

Abstract. This paper presents the theoretical background relevant to any method for producing a tridiagonal matrix similar to an arbitrary square matrix. Gragg's work on factoring Hankel matrices and the Kalman–Gilbert structure theorem from systems theory both find a place in the development.

Tridiagonalization is equivalent to the application of the generalized Gram–Schmidt process to a pair of Krylov sequences. In Euclidean space proper normalization allows one to monitor a tight lower bound on the condition number of the transformation. The various possibilities for breakdown find a natural classification by the ranks of certain matrices.

The theory is illustrated by some small examples and some suggestions for restarting are evaluated.

Key words. Lanczos algorithm, linear systems theory, tridiagonal form, minimal realizations

AMS(MOS) subject classifications. 65F15, 93B10, 93C75

1. Introduction and summary.

No one has presented a finite algorithm that is guaranteed to compute a tridiagonal matrix similar to an arbitrary given square complex matrix while avoiding huge intermediate quantities. Section 2 presents a brief sketch of the history of these attempts, along with a parameterization of the possible tridiagonals.

This paper describes theoretical results that are relevant to any method for producing a tridiagonal representation. In particular, the exceptional parameter values, for which the reduction fails, fall into two classes that we call curable and incurable. Cures come with acceptance of a block tridiagonal form. One impetus for this study was the desire to explain the intriguing observation of Taylor, in his dissertation [26], that incurable breakdown is a blessing in disguise because every eigenvalue of the tridiagonal matrix at breakdown is an eigenvalue of the original matrix despite the failure to find any invariant subspace. A satisfactory explanation comes from the canonical structure theorem of linear systems theory and we thank J.W. Demmel for pointing out that we had actually rediscovered that theorem. However the main goal of this essay is to provide the "right" setting for discussing any attempt to produce a stable algorithm for a tridiagonal representation.

Tridiagonal matrices are associated with three-term recurrence relations and with systems of orthogonal polynomials. The literature on these classical topics is vast. Analysts have studied the moment problem, control theorists have studied the sequence of impulse responses for time-invariant linear dynamical systems, and approximation theorists have studied continued fractions. The focus of all these studies is quite different from our goal of reducing a matrix to tridiagonal form but, as indicated above, some of their results help us to answer our questions. To keep this essay to a reasonable length, we have refrained from pointing out connections to the moment problem and to orthogonal polynomials. However, the recent interest in polynomials

* Received by the editors January 29, 1990; accepted for publication (in revised form) October 29, 1990. This paper was completed while the author visited the Numerical Analysis Group of the Oxford University Computing Laboratory, Oxford, U.K. This work was partially supported by Office of Naval Research contract N00014-90-J-1372.

† Department of Mathematics and the Computer Science Division of the Electrical Engineering and Computer Science Department, University of California, Berkeley, California 94720 (parlett@math.berkeley.edu).

orthogonal with respect to an indefinite inner product and to the associated modified moment problem can be thought of as an ascent of the same mountain range that we approach, but by a different route. Matrix factorizations arrived late on the mathematical scene but surely deserve a place alongside the traditional problems mentioned above. A few recent references are [14], [3], [11], and [18]. We do point out some connections with [14] in the final section.

Much of the theory is pure linear algebra and is independent of norms and angles. For this reason, at the risk of seeming pedantic, we make a distinction between $\mathbb{C}^n$ (complex column vectors) and its dual space $\mathbb{C}_*^n$ of linear functionals on $\mathbb{C}^n$ (row vectors). In this setting, the Gram–Schmidt (GS) process is seen as a method for producing a basis in $\mathbb{C}^n$ and a dual basis in $\mathbb{C}_*^n$. This approach is standard in control theory and although numerical analysts seem to favor the term bi-orthogonal over dual, the notion of angle (and hence right angle) is not needed for theoretical purposes. Nevertheless, a plain vector space is an inadequate setting for numerical analysis. A norm is needed to distinguish bad bases from good ones. In §10 we show how to monitor the condition number of the similarity transformation when Euclidean space is an appropriate setting.

The perspective we have reached, after several revisions, is indicated in the following synopsis of the rest of the essay. After a little history, §2 shows the representation of the class of similar tridiagonals by vector pairs and also urges the use of a pair $(\hat{T}, \Omega)$, with $\hat{T}$ symmetric tridiagonal and Ω diagonal, rather than a single matrix $\Omega^{-1}\hat{T}$. Section 3 presents the GS process, GS factorization, and a new extended GS algorithm to overcome breakdown. Section 4 presents the basic ideas of systems theory and the associated Hankel matrices $H^{(1)}$ and $H^{(0)}$. The rank of $H^{(0)}$ and those of related Krylov matrices give a nice characterization of the exceptional vector pairs. That is the end of the preparation. Section 5 discusses triangular factorization of Hankel matrices and the fundamental result for successful reduction: the three pencils $(H_n^{(1)}, H_n^{(0)})$, (B, I_n), and $(\hat{T}_n, \Omega_n)$ are equivalent. Here B is the given $n \times n$ matrix. In different words this result says that tridiagonal reduction is equivalent to the GS process applied to two Krylov sequences. The next two sections concern failure. Section 6 presents Gragg's result on block triangular factorization and mentions the interesting result of Kailath and his coworkers on Schur complements in Hankel matrices. Section 7 explains Taylor's observation and shows that incurable breakdown at step j occurs only when $(\hat{T}_j, \Omega_j)$ is a so-called minimal realization of the transfer function associated with the "initial" vectors. Section 8 is a table that summarizes the theory and §9 presents some small examples. Section 10 shows that, with proper normalization, the matrix Ω reveals the condition number of the transformations. It follows that, in all cases, there is a straightforward algorithm that can force stability and produce a block tridiagonal $\Omega^{-1}\hat{T}$, each of whose eigenvalues is an eigenvalue of B. This stable reduction is mentioned in §11, along with comments of a practical nature that are inspired by the preceding theory. Section 12 puts the results in perspective, makes connections with other approaches, and points to work left undone.

Finally, we point out here that a straightforward attempt to explain breakdown using the Jordan form of B becomes heavily burdened with irrelevant complications. The geometric approach of the Kalman–Gilbert structure theorem provides just the right level of abstraction; the controllable, observable subspace is easy to define and all we need is its dimension. However, a direct attack on the breakdown problem involves finding a basis for this subspace, explicitly or implicitly, and that can be

very complicated.

1.1. Notation. With few exceptions we follow Householder notational conventions: i, j, k, l, m, n for indices; lower case greek for scalars; lower case roman for column vectors; upper case roman for matrices; script upper case for vector spaces.

In addition,

- $\mathbb{C}$ denotes the complex numbers, $\mathbb{C}^{m\times n}$ the space of $m \times n$ complex matrices, $\mathbb{C}^n = \mathbb{C}^{n\times 1}$, $\mathbb{C}^n_* = \mathbb{C}^{1\times n}$.
- If $x \in \mathbb{C}^n$ then x^t denotes its transpose and x^* denotes $(\bar{x})^t$, the conjugate transpose of x.
- The identity matrix $I_n = [e_1, e_2, \cdots, e_n]$, $\tilde{I}_n = [e_n, e_{n-1}, \cdots, e_1]$.
- $\mathbf{0}$ denotes the zero vector in $\mathbb{C}^n$.
- $\mathbb{N}$ denotes the set of natural numbers.
- range (B) denotes the span of B's columns.

2. Tridiagonal form: History and basics.

DEFINITION. A matrix is *tridiagonal* if the (i, j) entry vanishes whenever $|i - j| > 1$.

A consequence of Galois' theory [1] is that there is no finite algebraic procedure for computing the eigenvalues of an $n\times n$ matrix, real or complex, in the general case when $n \geq 5$. Since diagonal and bidiagonal matrices reveal their eigenvalues immediately, the tridiagonal form is the most compact representation that can be expected from a finite process invoking the four basic arithmetic operators and the extraction of roots. It may turn out that this form is too sparse to be achieved in a stable way for the more difficult cases. Nevertheless, there are infinitely many tridiagonal matrices in the similarity class of a given matrix and a parametric representation of them is given later in this section.

Here is a brief history of the search for satisfactory algorithms to reduce a matrix by similarity transformations $B \rightarrow SBS^{-1}$ to tridiagonal form. In [21] Lanczos presented his method of "minimized iterations" that applied to real symmetric matrices and self-adjoint linear operators. However, he also showed the natural generalization to arbitrary square matrices and noted that this general process can break down. Even the symmetric version was sensitive to the effects of roundoff error and the nonsymmetric version received little serious implementation until the 1980s. In 1954, Givens presented a method for reducing a full $n \times n$ real symmetric (or complex Hermitian) matrix to tridiagonal form using plane rotations and fewer than n^3 scalar multiplications. Attempts to generalize this method appeared in [4], [25], and [21]. Demonstrations and explanations of their instabilities appeared in [28] and [23]. However the search lost its sense of importance in the early 1960s with the general acceptance of the Householder/QR method [5], [6] as a fast, stable solution to the nonsymmetric eigenvalue problem for dense matrices. The Householder reduction to upper Hessenberg form (the (i, j) entry vanishes whenever $i - j > 1$) requires only $(5/3)n^3$ multiplications, and the QR phase that diminishes the subdiagonal entries in positions $(i + 1, i)$ requires about $8n^3$ multiplications in practice, though infinitely many in exact arithmetic.

Now consider the fact that the fastest-known way to reduce a full matrix to tridiagonal form (the Lanczos algorithm) requires little more than $2n^3$ multiplications, if it does not break down. So the potential reduction in arithmetic cost from use of tridiagonal rather than Hessenberg form is about 80 percent ($10n^3$ to $2n^3$); in the 1960s a factor of 5 was significant. However, no stable version, one that is guaranteed to avoid huge intermediate quantities, has been found.

Circumstances have changed since the 1960s. Matrices are often larger and sparser. New computer hardware has reduced the dominance of arithmetic operations in assessing speed and cost. Moreover, there are applications for the tridiagonal form that are independent of the eigenvalue problem (see [27]).

Interest in finding better algorithms has revived in the 1980s and is nicely reviewed in [7]. Attention has been concentrated on how to recover from breakdown and this focus is of practical importance. Though breakdown (the vanishing of a denominator) is rare, near breakdowns are not and they provoke instability. Remedies for breakdown lead to remedies for instability.

Next we turn to the degrees of freedom present in the reduction to tridiagonal form.

2.1. Diagonal scaling. If $Q^{-1}BQ = T$, a tridiagonal matrix, and if D is diagonal and invertible, then $(QD)^{-1}B(QD) = D^{-1}TD$ is another tridiagonal matrix similar to B. T and $D^{-1}TD$ are equivalent for theoretical purposes. Note that $T(i,i)$ and $T(i+1,i)T(i,i+1)$, $\quad i = 1, \cdots, n$ are invariant under diagonal scaling.

2.2. Reduced matrices. A tridiagonal matrix is *reduced* if one (or more) of its next-to-diagonal entries vanishes. Observe that an unreduced $n \times n$ tridiagonal T has the property that rank $[T - \xi I]$ can never drop below $n - 1$. It follows that the eigenspace of each eigenvalue is one-dimensional. Such matrices are sometimes called nonderogatory and this property is invariant under similarity transformations. Consequently, a derogatory matrix (such as the identity I) can never be transformed to unreduced tridiagonal form.

Derogatory matrices are not "difficult" in any practical sense but the parametric representation of the tridiagonal forms does break down; a uniqueness property is lost. The implication of this is that a large class of methods for reducing an $n \times n$ matrix are liable to terminate early, having found an invariant subspace, range (Q_1), where

$$BQ_1 = Q_1T_1, \quad Q_1 \in \mathbb{C}^{n\times m}, \quad m < n.$$

There are infinitely many ways of adding more columns to obtain an invertible $Q = (Q_1, Q_2)$ and

$$Q^{-1}BQ = \begin{pmatrix} T_1 & * \\ 0 & T_2 \end{pmatrix},$$

where the $*$ may or may not vanish.

For the eigenvalue problem the occurrence of a split in T is an advantage. For a theoretical discussion it is not unreasonable to stop with $BQ_1 = Q_1T_1$ and regard this as benign early termination. The possible continuation of the reduction is just a new problem on a smaller space. This point of view will be taken in what follows.

Note that with a nonderogatory B the tridiagonal representation *may* split, but for a derogatory matrix it *must* split.

2.3. Parametric representation. The following useful result is well known and is not usually attributed to any one person. It is closely related to the implicit Q theorem in [10].

THEOREM 2.1. *If $B \in \mathbb{C}^{n\times n}$ is similar to an unreduced tridiagonal $T \in \mathbb{C}^{n\times n}$, i.e., if*

$$Q^{-1}BQ = T,$$

then Q and T are determined, to within diagonal scaling, by the first (or last) column of Q and the first (or last) row of Q^{-1}.

An equivalent form of this result is less well known despite the fact that it is more elegant theoretically and offers practical advantages as well; see §10.

THEOREM 2.2. *If $B \in \mathbb{C}^{n\times n}$ admits a representation of the form*

$$P^*IQ = \Omega = \operatorname{diag}(\omega_1, \cdots, \omega_n), \quad \omega_i \neq 0, \quad i = 1, \cdots, n,$$

$$P^*BQ = \hat{T} = \text{ unreduced tridiagonal,}$$

with invertible matrices P and Q, then P, Q, Ω, and $\hat{T}$ are determined, to within column scaling, by the first (or last) columns of P and Q.

Remark. By exercising the freedom to scale columns of P and Q, we can arrange that

$$\hat{T}(i, i+1) = \hat{T}(i+1, i) = \omega_{i+1}, \qquad i = 1, \cdots, n-1.$$

Thus $\hat{T}$ is symmetric but not necessarily Hermitian.

Remark. The connection between Theorems 2.1 and 2.2 is that

$$T = \Omega^{-1}\hat{T}.$$

Theorem 2.1 shows that there is a mapping from the projective space, called $\mathbb{CP}^{n-1} \otimes \mathbb{CP}^{n-1}$, of pairs of lines in $\mathbb{C}^n$ into the unreduced tridiagonals[1] in the similarity class of B. However, this mapping is partial; not all pairs yield an image. The emphasis in §§6–9 is on the characterization of the exceptional pairs.

- If B is nonderogatory then the mapping is densely defined. Later sections show that certain determinants depending on the pair must vanish if the pair is exceptional.
- If the pair (q, p) maps to T with $q_1 = q$, $p_1 = p$ then the same pair would map to $\tilde{I}T\tilde{I}$ with $q_n = q, p_n = p$, where $\tilde{I}$ is the reversing matrix, $\tilde{I} = (e_n, e_{n-1}, \cdots, e_2, e_1)$. The use of last columns instead of first columns leads to a map that is isomorphic to the first and will not be mentioned again.
- In the light of this parameterization we may distinguish a special class of methods. The *fixed start* methods never change q_1 and p_1. This class is to be distinguished from those methods that attempt to generalize the Givens method or to reduce bandwidth and so change column 1 of the current Q and P at later steps. Unfortunately, known generalizations of Givens reduction and of bandwidth reduction are also plagued by breakdown and instability. There is more than one member of the fixed start class because there are various possibilities for the way in which the unique Q and P are built up.
- A proof of Theorem 2.2 is given because it is constructive and introduces the Lanczos algorithm [21], the earliest of the *fixed start* methods.

Proof of Theorem 2.2. Let

$$\hat{T} = \operatorname{tridiag}\begin{pmatrix} & \omega_2 & & \omega_3 & & * & & * & & \omega_n & \\ \alpha_1 & & \alpha_2 & & \alpha_3 & & * & & * & & \alpha_n \\ & \omega_2 & & \omega_3 & & * & & * & & \omega_n & \end{pmatrix} = \hat{T}^t.$$

[1] These tridiagonals are normalized by $T(i+1, i) = 1$.

Since Ω is invertible, by hypothesis, the governing equations may be rewritten

$$BQ = Q\Omega^{-1}\hat{T}, \tag{2.1}$$

$$P^*B = \hat{T}\Omega^{-1}P^*. \tag{2.2}$$

Knowledge of p_1 and q_1 yields directly

$$\omega_1 = p_1^* q_1 \neq 0, \qquad \alpha_1 = p_1^* B q_1.$$

To start the construction of Q and P equate column 1 on each side of (2.1) and row 1 on each side of (2.2):

$$\begin{aligned} q_2 &= q_2\left(\frac{\omega_2}{\omega_2}\right) = Bq_1 - q_1\left(\frac{\alpha_1}{\omega_1}\right), \\ p_2^* &= \left(\frac{\omega_2}{\omega_2}\right)p_2^* = p_1^*B - \left(\frac{\alpha_1}{\omega_1}\right)p_1^*, \\ \omega_2 &= p_2^* q_2, \\ \alpha_2 &= p_2^* B q_2, \end{aligned}$$

are each determined by q_1 and p_1.

Next assume that $(q_1, \cdots, q_{j-1})$, $(p_1, \cdots, p_{j-1})$, α_{j-1}, ω_{j-1}, ω_{j-2} are all known. Equate columns $j-1$ on each side of (2.1), rows $j-1$ on each side of (2.2), and rearrange terms to find that

$$\begin{aligned} q_j &= q_j\left(\frac{\omega_j}{\omega_j}\right) = Bq_{j-1} - q_{j-1}\left(\frac{\alpha_{j-1}}{\omega_{j-1}}\right) - q_{j-2}\left(\frac{\omega_{j-1}}{\omega_{j-2}}\right) =: r_j, \\ p_j^* &= \left(\frac{\omega_j}{\omega_j}\right)p_j^* = p_{j-1}^*B - \left(\frac{\alpha_{j-1}}{\omega_{j-1}}\right)p_{j-1}^* - \left(\frac{\omega_{j-1}}{\omega_{j-2}}\right)p_{j-2}^* =: s_j^*, \\ \omega_j &= p_j^* q_j \ \ (\neq 0), \\ \alpha_j &= p_j^* B q_j, \end{aligned}$$

are each determined by previous quantities. Hence, by induction on j, all columns of Q and P are determined by q_1 and p_1.

A similar argument shows that q_n and p_n determine Q, P, Ω and $\hat{T}$. □

The proof shows that (q_1, p_1^*) is an exceptional pair if and only if, for some $l < n$,

$$\omega_{l+1} = p_{l+1}^* q_{l+1} = 0.$$

If $q_{l+1} = 0$ and $p_{l+1}^* \neq 0$ then the algorithm may be continued by redefining q_{l+1} to be *any* vector annihilated by $p_1^*, \cdots, p_l^*$, but not by p_{l+1}^*, and setting $\hat{T}(l+1, l) = 0$. Note that $\hat{T}$ will no longer be symmetric and the dependence of q_i, p_i^*, $i > l$ on q_1 and p_1 has been lost. However, the column space of $[q_1, \cdots, q_l]$ is B-invariant.

If $q_{l+1} \neq 0$, $p_{l+1}^* \neq 0$ but $\omega_{l+1} = p_{l+1}^* q_{l+1} = 0$ then the breakdown is called serious; see [29]. Clearly, there is no pair Q, P with the given q_1 and p_1 that satisfies both equations in Theorem 2.2. Local changes to p_{l+1}^* and q_{l+1} will not suffice.

The important result is that the top unreduced submatrix of every T similar to B is completely determined, to within diagonal scaling, by the pair of directions (q_1, p_1).

3. The two-sided GS algorithm. This section presents a familiar process in an unfamiliar format in order to emphasize the fact that the GS process does not require an inner product. The extended GS procedure (introduced below), though natural, appears to be new.

Given a sequence $\langle u_1, u_2, u_3, \cdots \rangle$ in $\mathbb{C}^n$ and a sequence $\langle v_1^*, v_2^*, v_3^*, \cdots \rangle$ in the dual space $\mathbb{C}_*^n$ of linear functionals on $\mathbb{C}^n$, the GS procedure produces a new pair of sequences that form *dual bases* in the associated subspaces. However, in contrast to the familiar form of the algorithm (when $u_i = v_i$), two-sided GS can break down.

Two-sided GS Algorithm: (GS)
Input: $u_i \in \mathbb{C}^n, u_i \neq \mathbf{0}, i = 1, \cdots, m; v_i^* \in \mathbb{C}_*^n, v_i^* \neq \mathbf{0}^*, i = 1, \cdots, m$
Output:
- $l \in \mathbb{N}$ and $q_i \in \mathbb{C}^n, p_i^* \in \mathbb{C}_*^n, i = 1, \cdots, l, (l \leq m)$ satisfying $p_i^* q_k = 0, i \neq k$, and $p_i^* q_i = \omega_i \neq 0$, $i = 1, \cdots, l$.
- (if $l < m$) p_{l+1}^*, q_{l+1} with $p_{l+1}^* q_{l+1} = 0$. Both, one, or neither of p_{l+1}^* and q_{l+1} may vanish.

Initialization: $l := -1, invar = false;$
repeat
$l := l + 1;$
$q_{l+1} := u_{l+1} - \sum_{i=1}^{l} q_i (p_i^* u_{l+1})/\omega_i,$
$p_{l+1}^* := v_{l+1}^* - \sum_{i=1}^{l} (v_{l+1}^* q_i) p_i^*/\omega_i,$
if $q_{l+1} = \mathbf{0}$ **or** $p_{l+1}^* = \mathbf{0}^*$
then $invar := true; \omega_{l+1} = 0;$
else $\omega_{l+1} := p_{l+1}^* q_{l+1}$
until $l + 1 = m$ **or** $\omega_{l+1} = 0$

Remark 3.1. In practice, when norms are defined on $\mathbb{C}^n$ and $\mathbb{C}_*^n$, it is advisable to normalize the output vectors but this feature is not theoretically necessary.

Remark 3.2. If either q_{l+1} or p_{l+1}^* vanishes (or both), we say that the breakdown is benign, not serious. GS may be continued by choosing as q_{l+1} any vector $\in \operatorname{span}\{u_i\}_{i=1}^m$ annihilated by $p_1^*, \cdots, p_l^*$ but not p_{l+1}^*, but when neither vanishes we have serious breakdown at the end of step l.

The results of the GS will be expressed compactly in terms of the following $n \times l$ matrices:

$$U_l := [u_1, \cdots, u_l], \qquad V_l := [v_1, \cdots, v_l],$$

$$Q_l := [q_1, \cdots, q_l], \qquad P_l := [p_1, \cdots, p_l],$$

together with unit triangular matrices

$$R_l = [r_{ij}] \in \mathbb{C}^{l \times l}, \qquad r_{ij} = \begin{cases} 0, & i > j, \\ 1, & i = j, \\ p_i^* u_j/\omega_j, & i < j, \end{cases}$$

$$L_l = [l_{ij}] \in \mathbb{C}^{l \times l}, \qquad l_{ij} = \begin{cases} v_i^* q_j/\omega_j, & i > j, \\ 1, & i = j, \\ 0, & i < j. \end{cases}$$

These definitions give the GS factorizations

$$U_l = Q_l R_l, \qquad V_l^* = L_l P_l^* \tag{3.1}$$

subject to

$$P_l^* Q_l = \operatorname{diag}(\omega_1, \cdots, \omega_l) = \Omega_l.$$

In practice the modified GS process (MGS) is preferred to GS but MGS is not appropriate in our application in §5.

It is useful to represent the output quantities Q_l, P_l^* in a way that is independent of the algorithm. There is a unique (oblique) projection Π_k onto $\operatorname{span}(u_1, u_2, \cdots, u_k)$ "along" $\operatorname{span}(v_1^*, v_2^*, \cdots, v_k^*)$, for each $k \leq l$. It is readily verified that a convenient matrix representation is

$$\Pi_k = Q_k (P_k^* Q_k)^{-1} P_k^*.$$

Moreover,

$$q_{k+1} = (I - \Pi_k) u_{k+1}, \qquad p_{k+1}^* = v_{k+1}^* (I - \Pi_k).$$

3.1. Extended GS. It is possible to continue GS despite some serious breakdowns by working with several vectors at each step. This has useful applications. The array dim will hold the dimensions of the diagonal blocks of Ω while b counts the blocks and l counts the columns in P and Q.

Initialization: $b := 0$; $l := 0$; $\dim(0) := 1$; $invariant := false$;
repeat
  $l := l + \dim(b)$; $b := b + 1$;
  $q_b := u_l - \sum_{\nu=1}^{b-1} Q_\nu \Omega_\nu^{-1} P_\nu^* u_l$;
  $p_b^* := v_l^* - \sum_{\nu=1}^{b-1} v_l^* Q_\nu \Omega_\nu^{-1} P_\nu^*$;
  if $q_b = \mathbf{0}$ **or** $p_b^* = \mathbf{0}^*$ **then**
    $invariant := true$; $\dim(b) := 0$;
  else
    $\delta := 1$; $Q_b := (q_b)$; $P_b := (p_b)$;
    while $\Omega_b := P_b^* Q_b$ is singular **and** $l + \delta \leq m$ **do**
      $\hat{u} := u_{l+\delta} - \sum_{\nu=1}^{b-1} Q_\nu \Omega_\nu^{-1} P_\nu^* u_{l+\delta}$;
      $\hat{v}^* := v_{l+\delta}^* - \sum_{\nu=1}^{b-1} v_{l+\delta}^* Q_\nu \Omega_\nu^{-1} P_\nu^*$;
      $Q_b := [Q_b, \hat{u}]$;
      $P_b := [P_b, \hat{v}]$;
      $\delta := \delta + 1$;
    $\dim(b) := \delta$;
until $l + \dim(b) > m$ **or** $invariant$

This algorithm yields decompositions like the one in (3.1) with R_ℓ and L_ℓ triangular, but now the scalar ω_i is replaced by a special Hankel matrix Ω_i whenever there is a breakdown. There is more on this in §6.

4. Linear systems and their Hankel forms. Here we describe those parts of systems theory that are germane to the reduction to tridiagonal form. Associated with any triple (B, q, p^*) with $B \in \mathbb{C}^{n \times n}, q \in \mathbb{C}^n, p^* \in \mathbb{C}_*^n$ is a "dynamical system" that evolves according to the linear laws

$$\dot{x}(\tau) = Bx(\tau) + qv(\tau), \tag{4.1}$$

$$\eta(\tau) = p^* x(\tau), \tag{4.2}$$

where $x(\tau) \in \mathbb{C}^n$ represents the state of the system at time τ, $\dot{x}$ is its time derivative, and $v(\tau)$ (read as *upsilon*, the greek u) represents a scalar control (or input) variable. Without loss of generality, it is assumed that $x(0) = \mathbf{0}$. Normally systems theory is set in $\mathbb{R}^n$, not $\mathbb{C}^n$, but the extra generality comes without penalty. Our system is called a single input, single output, time-invariant system. An excellent introduction to the subject is [18].

It is assumed that the state x is only knowable indirectly through the output function η and the input function v. The connection between the v and the η is most simply expressed via the Laplace transform

$$\tilde{f}(\sigma) := \int_0^\infty e^{-\sigma\tau} f(\tau) d\tau, \tag{4.3}$$

as

$$\tilde{\eta}(\sigma) = \Gamma(\sigma)\tilde{v}(\sigma) \tag{4.4}$$

where Γ is the *transfer function*

$$\Gamma(\sigma) := p^*(\sigma I - B)^{-1} q \tag{4.5}$$

$$= \sum_{k=0}^{\infty} (p^* B^k q)/\sigma^{k+1}. \tag{4.6}$$

The series representation is convergent for $|\sigma|$ large enough and the coefficients $\{p^* B^k q\}_{k=0}^{\infty}$ are called the *Markov parameters* or *impulse responses* of the system. The triple (B, q, p^*) is called a *realization* of Γ. In principle, Γ can be recovered from complete knowledge of just the input and the output functions via (4.4).

Systems theory examines various questions concerning v, x, and η but the one that is most relevant for tridiagonalization is the determination of all the realizations (B, q, p^*) that yield a given rational function Γ or, equivalently, its Markov parameters. This realization problem cuts across our reduction problem (where B is given, not Γ) but the connection is nevertheless illuminating.

The Cauchy–Binet theorem applied to $\sigma I - B$ gives

$$(\sigma I - B) \cdot \operatorname{adj}[\sigma I - B] = \det(\sigma I - B) \cdot I$$

where $\operatorname{adj}[M]$ stands for the classical adjugate matrix made up of $(n-1) \times (n-1)$ cofactors of M. Thus for $\sigma \notin B$'s spectrum,

$$\Gamma(\sigma) = p^* \operatorname{adj}[\sigma I - B] q / \chi_B(\sigma)$$

where $\chi_B(\sigma)$ is the characteristic polynomial of B. Since both numerator and denominator are polynomials in σ, cancellation of common factors is possible and, in

that event, not every eigenvalue of B will be a pole of Γ. Thus it is possible to have a triple (B', q', p'^*), with $B' \in \mathbb{C}^{m\times m}$ and $m < n$, giving rise to the same transfer function Γ as does (B, q, p^*).

DEFINITION. A *minimal realization* (B, q, p^*) of Γ is a realization in which B has minimal order (or dimension).

This minimal order is called the *McMillan degree* of Γ and of the Markov parameter sequence p^*B^iq. Clearly, it is of interest to determine the minimal realizations and one way is to associate with a sequence of Markov parameters the Hankel matrices (and their quadratic forms):

$$\mathbf{H}_l^{(k)} := [p^*B^{k+i+j-2}q] \qquad (i,j = 1,\cdots,l) \tag{4.7}$$

$$= \begin{bmatrix} \mu_k & \mu_{k+1} & \mu_{k+2} & \cdot \\ \mu_{k+1} & \mu_{k+2} & \cdot & \cdot \\ \mu_{k+2} & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix} \qquad (i,j = 1,\cdots,l), \tag{4.8}$$

$$\mathbf{H}^{(k)} = \mathbf{H}_\infty^{(k)}. \tag{4.9}$$

The Hankel property is that the (i,j) entry depends only on $i+j$. It turns out that the rank of $\mathbf{H}^{(0)}$ is the McMillan degree of Γ. For tridiagonalization, an unfortunate choice of $q_1 = q$ and $p_1 = p$ may yield rank $\mathbf{H}^{(0)} < n$ but that is not the sole cause of breakdown in the tridiagonalization process.

The controllable subspace and the observable subspace of the system and the canonical structure theorem will be introduced in §7. We mention in passing that systems theory also considers the *partial* realization problem that arises when only the first m Markov parameters are known. See [13] for an exhaustive treatment of this topic.

5. Hankel factorization; equivalent pencils. The two Hankel matrices $H_n^{(0)}$ and $H_n^{(1)}$, defined in the previous section, play a leading role in the analysis of an attempted reduction of B to tridiagonal form. That is the message of the equivalence theorems in this and later sections. To discuss the factorization of $H_n^{(0)}$ and $H_n^{(1)}$, the following four Krylov matrices are needed:

- $K_m(q,B) := [q, Bq, \cdots, B^{m-1}q] \in \mathbb{C}^{n\times m}$.
- $K(q,B) := K_\infty(q,B)$, the controllability matrix.
- $K_m(p^*,B) := K_m(p,B^*)^*$.
- $K(p^*,B) := K_\infty(p^*,B)$, the observability matrix.

Next we list some elementary but fundamental facts.

LEMMA 5.1. *For any* $j \in \mathbb{N}$,

$$H_m^{(j)} = K_m(p^*,B)B^jK_m(q,B),$$
$$H^{(j)} = K(p^*,B)B^jK(q,B).$$

Proof. The (i,k) entry on each side is $(p^*B^{i-1})B^j(B^{k-1}q)$. □

Note that $H^{(j)}$ is a submatrix of $H^{(0)}$.

COROLLARY 5.2.

$$\text{rank}\,[H^{(j)}] \le \text{rank}\,[H^{(0)}] \le \min\{\text{rank}\,[K(p^*,B)], \text{rank}\,[K(q,B)]\} \le n.$$

LEMMA 5.3. range$[K(q,B)]$ *is the smallest* B*-invariant subspace of* $\mathbb{C}^n$ *that contains* q. range$[K(p^*,B)]$ *is the smallest* B*-invariant subspace of* $\mathbb{C}_*^n$ *that contains* p^*.

Proof. If $\mathcal{S}$ is B-invariant and contains q, then it must contain Bq, $B(Bq)$, etc. Hence, range $[K(q,B)] \subset \mathcal{S}$. Moreover,

$$B \text{ range } [K(q,B)] = \text{range } [Bq, B^2q, \cdots] \subset \text{range } [K(q,B)]$$

and so range $[K(q,B)]$ is invariant. Similarly range $[K(p^*,B)]$ is invariant too. □

Together with n, the following three numbers furnish a complete classification of the various cases in the mapping $(q,p^*) \longrightarrow (\hat{T},\Omega)$.

DEFINITION.

$$\begin{aligned} l &:= \min\{j : H_{j+1}^{(0)} \text{is singular}\}, \\ r &:= \text{rank } [H^{(0)}], \\ m &:= \min\{\text{rank}[K(p^*,B)], \text{rank}[K(q,B)]\}. \end{aligned}$$

The corollary given above and these definitions yield

$$l \le r \le m \le n.$$

It is strict inequalities that lead to early termination in tridiagonalization. Anticipating later sections, we can summarize the situation.

- $l = m < n$ yields benign early termination with an invariant subspace. Tridiagonalization may be continued in infinitely many ways.
- $l < r = m$ yields serious breakdown that can be cured by permitting block tridiagonalization.
- $r < m$ yields incurable breakdown but a minimal realization of the transfer function (see §7).

THEOREM 5.4 (Hankel factorization). *Let* $H^{(0)} = H^{(0)}(B,q,p^*)$. *The two-sided* GS *process applied to the columns of* $K(q,B)$ *and the rows of* $K(p^*,B)$ *yields a whole number* l *and rank* l *matrices* Q_l, P_l^*, *and a unit lower triangular* $l \times l$ *matrix* L_l *such that*

$$K_l(q,B) = Q_l L_l^t, \quad K_l(p^*,B) = L_l P_l^*,$$

and

$$\begin{aligned} H_l^{(0)} &= L_l \Omega_l L_l^t, \\ \Omega_l &:= P_l^* Q_l = \text{diag}(\omega_1, \cdots, \omega_l), \qquad \omega_i \neq 0, i = 1, \cdots, l. \end{aligned}$$

In addition, GS *produces* q_{l+1}, p_{l+1} *satisfying* $\omega_{l+1} = p_{l+1}^* q_{l+1} = 0$.

Proof. Algorithm GS (see §3) applied to $K(q,B)$ and $K(p^*,B)$ yields Q_l, P_l so that

$$K_l(q,B) = Q_l R_l, \quad K_l(p^*,B) = L_l P_l^*, \quad P_l^* Q_l = \Omega_l,$$

where R_l is unit upper triangular. By Lemma 5.1

$$\begin{aligned} H_l^{(0)} &= K_l(p^*,B) K_l(q,B) \\ &= L_l P_l^* Q_l R_l \\ &= L_l \Omega_l R_l. \end{aligned}$$

By the LDU theorem L_l, Ω_l, R_l are unique. By symmetry of $H_l^{(0)}$, $R_l = L_l^t$. By definition of l in GS, $\omega_{l+1} = 0, \omega_i \neq 0, i \leq l$. □

LEMMA 5.5 (tridiagonal form). *With the notation of the Hankel factorization theorem, $\hat{T}_l := P_l^* B Q_l$ is symmetric, tridiagonal, and unreduced.*

Proof. The characteristic property of Krylov matrices and the output of the GS process is that, for $j < l$,

$$q_j \in \text{range}\,[K_j(q, B)], \qquad q_j \notin \text{range}\,[K_{j-1}(q, B)].$$

Hence

$$Bq_j \in B \cdot \text{range}\,[K_j(q, B)] \subset \text{range}\,[K_{j+1}(q, B)],$$
$$Bq_j \notin B \cdot \text{range}[K_{j-1}(q, B)] \oplus \text{span}(q) = \text{range}\,[K_j(q, B)].$$

Furthermore, p_i^* annihilates $K_{j+1}(q, B)$ for all $i > j + 1$ and so p_i^* annihilates column j of BQ_l for all $i > j + 1$, and so the (i, j) entry of $P_l^* BQ_l$ vanishes for $i - j > 1$. Similarly, for $i < l$, $p_i^* B \in \text{range}\,[K_{i+1}(p^*, B)]$ whose null space contains q_j for all $j > i + 1$. Hence the (i, j) entry of $P_l^* BQ_l$ vanishes for $j - i > 1$.

It remains to show that $\hat{T}_l$ is unreduced. By the minimality property of l the GS algorithm does not break down at step j for $j < l$. Thus Bq_j has a nonzero component γ in q_{j+1} when expanded in terms of $(q_1, q_2, \cdots, q_{j+1})$. Since p_{j+1}^* annihilates $q_1, q_2, \cdots, q_j$ but not q_{j+1}, it follows that

$$p_{j+1}^* Bq_j = p_{j+1}^* q_{j+1} \gamma = \omega_{j+1} \gamma \neq 0, \qquad j < l.$$

By similar arguments,

$$p_j^* Bq_{j+1} = \delta p_{j+1}^* q_{j+1} = \delta \omega_{j+1} \neq 0, \qquad j < l. \qquad \square$$

First we consider the generic case when $K(q, B)$ and $K(p^*, B)$ have full rank n and the associated system (B, q, p^*) is said to be controllable and observable.

THEOREM 5.6 (equivalence theorem, version 1). *If invertible $H_n^{(0)}(B, q, p^*)$ permits triangular factorization*

$$H_n^{(0)} = L_n \Omega_n L_n^t,$$

then the following pencils are equivalent:

$$(H_n^{(1)}, H_n^{(0)}), \quad (B, I), \quad (\hat{T}_n, \Omega_n).$$

Here $\hat{T}_n$ is the symmetric, unreduced, tridiagonal matrix

$$\hat{T}_n = L_n^{-1} H_n^{(1)} L_n^{-t} = P_n^* B Q_n.$$

The first and third pencils are symmetric but not necessarily Hermitian.

In systems theory these generic $H_n^{(0)}$ are called strongly regular.

Proof. By Lemma 5.1, $H_n^{(0)} = K_n(p^*, B) K_n(q, B)$ and if either Krylov matrix were rank deficient, then so would be their product. Since $H_n^{(0)}$ permits triangular factorization,

$$l = r = m = n.$$

Thus all the transformation matrices that appear below are invertible:

$$\begin{aligned} H_n^{(1)} - \lambda H_n^{(0)} &= K_n(p^*, B)(B - \lambda I)K_n(q, B) \quad \text{(Lemma 5.1)} \\ &= L_n P_n^*(B - \lambda I)Q_n L_n^t \quad \text{(GS factorization)} \\ &= L_n(\hat{T}_n - \lambda \Omega_n)L_n^t. \end{aligned}$$

Theorem 5.4 yields $P_n^* Q_n = \Omega_n$ =diagonal, Lemma 5.5 yields $P_n^* B Q_n = \hat{T}_n$ = tridiagonal, unreduced. Thus $H_n^{(1)} = L_n \hat{T}_n L_n^t$ and $\hat{T}$ must be symmetric. □

Since the pencils (B, I) and $(\hat{T}_n, \Omega_n)$ are equivalent, $\Omega_n^{-1}\hat{T}_n$ has the same spectrum as B.

To complete the circle of ideas it is necessary to identify the matrix Q_n coming from the GS factorization $K_n(q, B) = Q_n L_n^t$ with the matrix Q generated in the proof of Theorem 2.2 solely from the property of producing a tridiagonal form $BQ = Q(\Omega^{-1}\hat{T})$. This will show that the result in Theorem 5.6 is categorical; that is, Hankel factorization, explicit or implicit, is the only mechanism for producing the desired $(\hat{T}, \Omega)$ representation. In other words,

$$\text{tridiagonal reduction} \equiv \text{Krylov matrices} + \text{GS}.$$

The identification is an immediate consequence of the fact that $Qe_{j+1} = \phi_j(B)q_1$ where ϕ_j is a monic polynomial of degree j. To see this, note that $BQ = Q(\Omega^{-1}\hat{T})$ implies that

$$Bq_j = q_{j+1} + q_j(\hat{T}(j,j)/\omega_j) + q_{j-1}(\omega_j/\omega_{j-1}), \qquad j = 2, \cdots, n-1.$$

Thus, if $q_j = \phi_{j-1}(B)q_1$, $q_{j-1} = \phi_{j-2}(B)q_1$, then $q_{j+1} = \phi_j(B)q_1$, for $j = 2, \cdots, n-1$. But $q_1 = \phi_0(B)q_1$, and $q_2 = Bq_1 - q_1(\hat{T}(1,1)/\omega_1) = \phi_1(B)q_1$, and the principle of induction establishes the polynomial representation. In matrix terms $Q = K_n(q_1, B)R^{-1}$ for some unit upper triangular R. Similarly, $P^* := \Omega Q^{-1}$ satisfies $P^* = L^{-1}K_n(p_1^*, B)$ for some unit lower triangular L. Finally

$$\Omega = P^*Q = L^{-1}K_n(p_1^*, B)K_n(q_1, B)R^{-1} = L^{-1}H_n^{(0)}R^{-1}$$

and the uniqueness of triangular factorization shows that R must be the matrix L_n^t from Theorem 5.6.

Before proceeding to cases of failure to produce a tridiagonal form $(\hat{T}_n, \Omega_n)$, we repeat that the case

$$l = r = m < n$$

is essentially like the one above. Either q_{l+1} or p_{l+1}^* generated by GS vanishes and it is only necessary to replace the zero vectors by suitably chosen nonzero vectors to ensure continuation of GS until step n. However, $\hat{T}_n$ will be reduced. It may be preferable to stop with $(\hat{T}_l, \Omega_l)$. Although this pencil is not equivalent to (B, I), nevertheless, when $q_{l+1} = 0$, it follows that

$$\text{range } K_l(q, B) = \text{range } K(q, B),$$

an invariant subspace. It follows that $(\hat{T}_l, \Omega_l)$ is equivalent to $(\hat{B}, I_l)$ where $\hat{B}$ is the restriction of B to the invariant subspace. Not only is every eigenvalue λ of $\Omega_l^{-1}\hat{T}_l$ an eigenvalue of B, but right eigenvectors are explicitly given by $Q_l v$ where $\hat{T}_l v = \Omega_l v\lambda$.

We will not consider this case in any more detail.

6. Block tridiagonal form. This section examines those exceptional cases when $H_n^{(0)}$ has full rank n but does not permit triangular factorization. In the terminology of the previous section: $l < r = m = n$. This is serious breakdown and there is no tridiagonal form $(\hat{T}, \Omega)$ with the given parameter values q and p^*. In the language of system theory, $H_n^{(0)}$ is said to be regular but not strongly regular.

A natural way to persist with the starting vectors and obtain a sparse representation of B is to accept a block tridiagonal representation. The smaller the blocks, the better, and it is of interest to describe the most refined representation that is possible. This was done in [12] and the result may be found more easily in the definitive paper [13], where the authors give ample references to related earlier work. Here is our description of Gragg's result.

DEFINITION. Let $\nu(1), \nu(2), \cdots, \nu(k)$ be the sequence of index values j such that $H_j^{(0)}$ is invertible. These are the *degree indices* of $H^{(0)}$. Let $\nu(0) = 0$.

Under our assumption on $H^{(0)}$, $\nu(k) = n$ for some $k \leq n$.

THEOREM 6.1 (block triangular factorization). *The most refined block triangular factorization of* $H^{(0)}$ *is*

$$H^{(0)} = L\Omega L^t$$

where $\Omega = \Omega_1 \oplus \Omega_2 \oplus \cdots \oplus \Omega_k \oplus O_\infty$, *and* Ω_j *is a nonsingular right lower triangular Hankel matrix of order* $\nu(j) - \nu(j-1)$. *Here* $\nu(k) = \operatorname{rank} H^{(0)} = n$. *Moreover, columns* $\nu(j)+1$ *to* $\nu(j+1)$ *of* L^{-1} *have unit lower triangular Toeplitz structure, for* $j = 0, \cdots, k-1$. *However, the entries of* Ω_i *below the secondary diagonal are not uniquely determined by* $H^{(0)}$.

An Ω_j of order 3 has the form

$$\begin{bmatrix} 0 & 0 & \pi_3 \\ 0 & \pi_3 & \pi_4 \\ \pi_3 & \pi_4 & \pi_5 \end{bmatrix}, \quad \pi_3 \neq 0, \quad (\pi_4, \pi_5 \text{ not unique}).$$

In [14] the irregular orthogonal polynomials are chosen so that each Ω_i is a multiple of $\tilde{I}$, the reversal matrix, i.e., $\pi_4 = \pi_5 = 0$ in the matrix above. In general, the Schur complement of $H_j^{(0)}$ in $H^{(0)}$ is not a Hankel matrix so the only surprising feature of Theorem 6.1 is the Hankel structure of the Ω_j and the structure of L^{-1}. The extended GS algorithm of §3 is intimately connected with the factorization.

We will not present a proof of Theorem 6.1. Instead, we offer some further discussion. The reduced matrix that remains to be processed after j steps of triangular factorization is called the *Schur complement* of the leading principal $j \times j$ submatrix. Surprisingly, neither [13] nor [16] discuss Schur complements in Hankel matrices. One reason may be that the Schur complement of $H_j^{(0)}$ in $H^{(0)}$ is not a Hankel matrix although it does possess an interesting structure. Kailath and his coworkers have studied these Schur complements in the course of their work on *displacement rank*.

THEOREM 6.2 (Schur complements). *If* $H_j^{(0)}$ *is invertible then its Schur complement* $H_{\langle j \rangle}^{(0)}$ *in* $H^{(0)}$ *exists and is triangularly congruent to a Hankel matrix*

$$H_{\langle j \rangle}^{(0)} = LHL^t$$

where L *is unit lower triangular and Toeplitz.*

The reader is referred to [22] for proofs that use the bilinear forms associated with matrices. Matrices with the structure shown in Theorem 6.2 are called quasi-Hankel by Kailath.

Applying these results to Gragg's result (Theorem 6.1) shows that a leading principal submatrix of these Schur complements is actually of Hankel form. In fact, we can give a realization of this Hankel part in terms of the so-called Lanczos vectors, which are the columns of Q and P in the GS factorizations

$$K(q,B) = QL^t, \qquad K(p^*,B) = LP^*.$$

Referring to the diagonal blocks Ω_i in Theorem 6.1, we have the following useful result:

- Let $d_{j+1} = \nu(j+1) - \nu(j)$. The leading principal submatrix of order d_{j+1} of $H^{(0)}_{\langle\nu(j)\rangle}$ is

$$\Omega_{j+1} := H^{(0)}_{d_{j+1}}(q_{j+1}, p^*_{j+1}, B).$$

This is the first invertible submatrix of the Schur complement. Note that $q_{j+1} = (I - \Pi_{\nu(j)})B^{\nu(j)}q_1$. Here Π is the GS projector defined in §3.

The hypotheses of Theorem 5.6 may now be weakened.

THEOREM 6.3 (equivalence theorem, version 2). *Let* $H^{(0)} = H^{(0)}(B,q,p^*)$. *If* rank $[H^{(0)}]$ = *order of* $B = n$, *then the following pencils are equivalent:*

$$(H^{(1)}_n, H^{(0)}_n), \quad (B, I_n), \quad (\hat{T}, \Omega),$$

where Ω *is block diagonal as given above and*

$$\hat{T} = L^{-1}H^{(1)}_n L^{-t} = P^*BQ$$

is symmetric block tridiagonal with structure conformable to Ω. *The block sizes in* Ω *are minimal. The off-diagonal blocks of* $\hat{T}$ *are null except in the lower right entries.*

The proof of this theorem is analogous to the proof of Lemma 5.5 and will be omitted. The only difference is that in extending Lemma 5.5 the phrase "$Bq_{j-1} \in$ range $[K_j]$" must be replaced by "range $(Bq_{j-1}) \subset$ range $[K_j]$" since q_{j-1} is now a matrix. The structure of Ω and $\hat{T}$ is shown at the end of this section.

Although the theorems extend readily to block form, the elementary sequential process of §2 (the Lanczos algorithm) for computing $\hat{T}$ and Ω will not suffice alone when the block sizes vary. For example, let the jth (block) row of $\hat{T}$ be

$$(0 \cdots 0\ B_j\ A_j\ B^t_{j+1}\ 0 \cdots 0), \quad A_j \in \mathbb{C}^{d_j \times d_j}, \quad B_j \in \mathbb{C}^{d_j \times d_{j-1}},$$

where

$$d_j := \nu(j) - \nu(j-1)$$

and let

$$\hat{Q} = (Q_1, Q_2, \cdots, Q_k), \qquad Q_j \in \mathbb{C}^{n \times d_j}.$$

Then equate the jth (block) column on each side of

$$B\hat{Q} = \hat{Q}\Omega^{-1}\hat{T}, \qquad \hat{P}^*B = \hat{T}\Omega^{-1}\hat{P}^*$$

to find

$$Q_{j+1}\Omega_{j+1}^{-1}B_{j+1} = R_{j+1} := BQ_j - Q_j\Omega_j^{-1}A_j - Q_{j-1}\Omega_{j-1}^{-1}B_j^t \in \mathbb{C}^{n\times d_j}$$

and

$$B_{j+1}^t\Omega_{j+1}^{-1}P_{j+1}^* = S_{j+1}^* := P_j^*B - A_j\Omega_j^{-1}P_j^* - B_j\Omega_{j-1}^{-1}P_{j-1}^* \in \mathbb{C}^{d_j\times n}.$$

At the jth step of the process the right sides are known but $Q_{j+1}, \Omega_{j+1}, B_{j+1}$, and P_{j+1}^* are not. Moreover, if $d_{j+1} > d_j$ these right sides are not sufficient by themselves to determine any of the unknowns. This is a manifestation of the lack of uniqueness in Theorem 6.1. More work must be done with the matrix B to continue the computation. Nevertheless, the extended GS process (§3) can determine the blocks $Q_1, Q_2, \cdots$, and $P_1^*, P_2^*, \cdots$ and these, in turn, determine $\hat{T}$ and Ω. At each step the block sizes are minimal subject to the constraint that $P_j^*Q_j$ be invertible.

These ideas lead to the look-ahead Lanczos algorithm that was presented in [25]. The practical defect of this method is that the sizes of the blocks are not known in advance and so dynamic allocation of storage is desirable for the blocks A_j and B_j of $\hat{T}$. However, FORTRAN does not permit such an arrangement. We say more about these ideas in §11.

This section has shown that breakdown corresponding to $l < r = n$ can be cured by accepting block tridiagonal representations.

6.1. The structure of $\hat{T}$ and Ω.

$$\begin{aligned}
&Q := [q_1, Bq_1, B^2q_1, q_2, Bq_2, q_3, Bq_3, B^2q_3, B^3q_3],\\
&P := [p_1, B^*p_1, B^{*2}p_1, p_2, B^*p_2, p_3, B^*p_3, B^{*2}p_3, B^{*3}p_3],\\
&\mu_k^j := p_j^*B^kq_j,\\
&\Omega = \Omega_1 \oplus \Omega_2 \oplus \Omega_3\\
&\quad = \begin{bmatrix} 0 & 0 & \mu_2^1\\ 0 & \mu_2^1 & \mu_3^1\\ \mu_2^1 & \mu_3^1 & \mu_4^1 \end{bmatrix} \oplus \begin{bmatrix} 0 & \mu_1^2\\ \mu_1^2 & \mu_2^2 \end{bmatrix} \oplus \begin{bmatrix} 0 & 0 & 0 & \mu_3^3\\ 0 & 0 & \mu_3^3 & \mu_4^3\\ 0 & \mu_3^3 & \mu_4^3 & \mu_5^3\\ \mu_3^3 & \mu_4^3 & \mu_5^3 & \mu_6^3 \end{bmatrix},
\end{aligned}$$

$$\hat{T} = \begin{bmatrix}
0 & \mu_2^1 & \mu_3^1 & 0 & 0 & 0 & 0 & 0 & 0\\
\mu_2^1 & \mu_3^1 & \mu_4^1 & 0 & 0 & 0 & 0 & 0 & 0\\
\mu_3^1 & \mu_4^1 & \mu_5^1 & 0 & \mu_1^2 & 0 & 0 & 0 & 0\\
0 & 0 & 0 & \mu_1^2 & \mu_2^2 & 0 & 0 & 0 & 0\\
0 & 0 & \mu_1^2 & \mu_2^2 & \mu_3^2 & 0 & 0 & 0 & \mu_3^3\\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mu_3^3 & \mu_4^3\\
0 & 0 & 0 & 0 & 0 & 0 & \mu_3^3 & \mu_4^3 & \mu_5^3\\
0 & 0 & 0 & 0 & 0 & \mu_3^3 & \mu_4^3 & \mu_5^3 & \mu_6^3\\
0 & 0 & 0 & 0 & \mu_3^3 & \mu_4^3 & \mu_5^3 & \mu_6^3 & \mu_7^3
\end{bmatrix}.$$

The diagonal blocks of $\hat{T}$ are Hessenberg Hankel matrices. The off-diagonal blocks each have a single nonzero entry in the lower right position whose value is equal to the other nonzero entries in its antidiagonal. By reversing the order of the rows in each block, we see that for each $\sigma \in \mathbb{C}$, $\hat{T} - \sigma\Omega$ is a Hessenberg matrix. Thus its determinant and the derivatives with respect to σ may be evaluated rapidly and stably by Hyman's recurrence. See [24].

7. Incurable breakdown and minimal realizations. In this section we consider the general case and face up to the situation when $H^{(0)} = H^{(0)}(B, q, p^*)$ has rank $r < n (= \textit{order of } B)$. Recall that r is the McMillan degree of the transfer function $p^*(\sigma I - B)^{-1}q$. A classical result of Kronecker (see [16]) states:

$$\text{if rank } [H^{(0)}] = r \quad \text{ then } H_r^{(0)} \text{ is invertible.}$$

It follows that a (block) triangular factorization of $H_n^{(0)}$ must stop when Ω has r rows. With $r < n$ the output pencil $(\hat{T}, \Omega)$ is not equivalent to (B, I). Moreover, (B, I) need not be equivalent to $(H_n^{(1)}, H_n^{(0)})$; the breakdown is incurable.

It is of interest to express r in terms of geometric quantities that are directly related to B, q, and p^*. This expression is a byproduct of the canonical structure theorem of linear systems theory. See [9] and [19]. The rank of $H^{(0)}(B, q, p^*)$ is the dimension of any controllable, observable subspace of the system (B, q, p^*).

THEOREM 7.1 (minimal realization theorem). *Given $B, q,$ and p^* there is a maximally refined block tridiagonal-diagonal pair $(\hat{T}, \Omega)$, not unique, such that $(\Omega^{-1}\hat{T}, e_1, e_1^*)$ is a minimal realization;*

$$p^*(\sigma I - B)^{-1}q = e_1^*(\sigma I - \Omega^{-1}\hat{T})^{-1}e_1 \quad \textit{for all } \sigma \notin B\textit{'s spectrum.}$$

Every eigenvalue of $\Omega^{-1}\hat{T}$ is an eigenvalue of B.

At first glance it is surprising, and pleasing, that incurable breakdown yields such a rich harvest of eigenvalues despite $(\hat{T}, \Omega)$ not being equivalent to the restriction of (B, I) to any B-invariant subspace of $\mathbb{C}^n$. At second glance it is annoying, but interesting, that from the matrices $Q \in \mathbb{C}^{n\times r}$ and $P \in \mathbb{C}^{n\times r}$ that yield $\Omega = P^*Q$, $\hat{T} = P^*BQ$ there is no direct way of computing eigenvectors of B that belong to eigenvalues of $\Omega^{-1}\hat{T}$. This is because there are no invariant subspaces of B in the range of Q and P. The recent paper [2] shows how to append columns to P and Q to obtain bases for $\mathcal{K}(q, B)$ and $\mathcal{K}(p, B^*)$.

Next, we give a brief summary of the canonical structure theorem. See [18] for a full account.

Recall from §4 that the linear system under consideration is

$$\dot{x} = Bx + qv, \quad x(0) = 0,$$

$$\eta = p^*x, \qquad v(0) \neq 0.$$

There are four special subspaces of $\mathbb{C}^n$ associated with the system.

- $\mathcal{K}(q, B) := \textit{range of } K(q, B) = S_c$,
  the *controllable* subspace. It is the smallest B-invariant subspace containing q.
- $\mathcal{N}(p^*, B) := \textit{null space of } K(p^*, B) = S_{\bar{o}}$,
  the *unobservable* subspace, the largest B-invariant subspace annihilated by p^*.

The analysis to follow holds for any complements in $\mathbb{C}^n$ of these two subspaces. However, there is an obvious choice of complement in each case.

- $\mathcal{N}(q^*, B^*) := \textit{null space of } K(q^*, B^*) = S_{\bar{c}}$,
  the *unobservable* subspace for the dual system (B^*, p, q^*), the largest B^*-invariant subspace annihilated by q^*.
- $\mathcal{K}(p, B^*) := \textit{range of } K(p, B^*) = S_o$,
  the *controllable* subspace for the dual system, the smallest B^*-invariant subspace containing p.

Remark 7.1. The controllable subspace should be called the reachable subspace because it consists of all the states x that can be obtained at a given time by suitable choice of the scalar function $v(t)$. On the other hand, $\mathcal{N}(p^*, B)$ is well named since it consists of the states that yield $\eta = 0$.

Remark 7.2. From the abstract point of view, the introduction of the last two subspaces is the only occasion when we invoke the antilinear mapping $v^* \Leftrightarrow v$ to transform a linear functional in $\mathbb{C}_*^n$ to a vector in $\mathbb{C}^n$. This device puts all the important spaces into $\mathbb{C}^n$.

By taking the intersection of these four subspaces in an appropriate order a canonical block structure for B is revealed. The subscripts $\bar{c}$ and $\bar{o}$ signify uncontrollable and unobservable, respectively.

$$\begin{aligned}
\mathcal{S}_{c\bar{o}} &:= \mathcal{N}(p^*, B) \cap \mathcal{K}(q, B), \\
\mathcal{S}_{co} &:= \mathcal{K}(q, B) \cap \mathcal{K}(p, B^*), \\
\mathcal{S}_{\bar{c}\bar{o}} &:= \mathcal{N}(p^*, B) \cap \mathcal{N}(q^*, B^*), \\
\mathcal{S}_{\bar{c}o} &:= \mathcal{N}(q^*, B^*) \cap \mathcal{K}(p, B^*).
\end{aligned}$$

Clearly,

$$\mathbb{C}^n = \mathcal{S}_{c\bar{o}} \oplus \mathcal{S}_{co} \oplus \mathcal{S}_{\bar{c}\bar{o}} \oplus \mathcal{S}_{\bar{c}o}.$$

Also,

$$\begin{aligned}
&\mathcal{S}_{c\bar{o}} \text{ and } \mathcal{S}_c = \mathcal{S}_{c\bar{o}} \oplus \mathcal{S}_{co} \text{ are } B\text{-invariant}, \\
&\mathcal{S}_{\bar{c}o} \text{ and } \mathcal{S}_o = \mathcal{S}_{co} \oplus \mathcal{S}_{\bar{c}o} \text{ are } B^*\text{-invariant}.
\end{aligned}$$

The controllable, observable subspace $\mathcal{S}_{co}$ is not invariant under B nor under B^*.

By taking, in order, any bases for the four subspaces listed above, a new representation $\tilde{B}$ for B is obtained that is block upper triangular.

$$\tilde{B} = \begin{bmatrix} \tilde{B}_{11} & \tilde{B}_{12} & \tilde{B}_{13} & \tilde{B}_{14} \\ 0 & \tilde{B}_{22} & 0 & \tilde{B}_{24} \\ 0 & 0 & \tilde{B}_{33} & \tilde{B}_{34} \\ 0 & 0 & 0 & \tilde{B}_{44} \end{bmatrix} = F^{-1}BF$$

and the starting vectors have the following representation:

$$\tilde{q} = \begin{pmatrix} \tilde{q}^1 \\ \tilde{q}^2 \\ 0 \\ 0 \end{pmatrix} = F^{-1}q, \qquad \tilde{p} = \begin{pmatrix} 0 \\ \tilde{p}^2 \\ 0 \\ \tilde{p}^4 \end{pmatrix} = F^*p.$$

Remark 7.3. It appears to be traditional in systems theory to invert the order of $\mathcal{S}_{\bar{c}\bar{o}}$ and $\mathcal{S}_{\bar{c}o}$. This has the advantage of putting the bad subspace $\mathcal{S}_{\bar{c}\bar{o}}$ in final position, but the disadvantage of forgoing the block triangular form dear to the hearts of matrix theorists.

The form of $\tilde{q}$ and $\tilde{p}$ reveals the final result

$$\Gamma(\sigma) = p^*(\sigma I - B)^{-1}q = (\tilde{p}^2)^*(\sigma I - \tilde{B}_{22})^{-1}\tilde{q}^2.$$

The system $(\tilde{B}_{22}, \tilde{q}^2, (\tilde{p}^2)^*)$ is a *minimal realization* of the transfer function Γ. Moreover,

$$H^{(0)}(q, p^*, B) = H^{(0)}(\tilde{q}^2, (\tilde{p}^2)^*, \tilde{B}_{22})$$

and

$$\text{rank}[H^{(0)}] = \dim \mathcal{S}_{co}.$$

In systems theory we start from the Markov parameters and seek a minimal realization. To obtain a tridiagonal representation, we start with B and seek q and p^* to ensure that (B, q, p^*) is minimal.

We now give the interpretation of the canonical structure theorem for the reduction of (B, I) to a more condensed form. Consider a method that builds up Q and P one column at a time. Suppose that serious breakdown occurs with

$$\omega_{j+1} := p_{j+1}^* q_{j+1} = 0, \quad q_{j+1} \neq 0, \quad p_{j+1}^* \neq 0^*.$$

Rank $[H^{(0)}]$ is not known but the following dichotomy holds. Either

$$p_{j+1}^* B^\nu q_{j+1} = 0, \qquad \nu = 0, 1, \cdots, n-j-1, \tag{7.1}$$

in which case $p_{j+1}^* B^\nu q_{j+1} = 0$ for all ν and rank $[H^{(0)}] = j$. The current matrix pencil $(\hat{T}_j, \Omega_j)$ is a minimal realization of the transfer function and every eigenvalue of $\Omega_j^{-1}\hat{T}_j$ is an eigenvalue of B, or

$$\delta = \min\{\nu \,:\, p_{j+1}^* B^{\nu-1} q_{j+1} \neq 0\} \leq n - j \tag{7.2}$$

and another step of (block) tridiagonalization may be taken with block size δ. No eigenvalue of $\Omega_j^{-1}\hat{T}_j$ is an eigenvalue of B, although some could be close.

This dichotomy is the content of the mismatch theorem in Taylor's dissertation [26]. Incurable breakdown occurs only when a minimal realization has been found. The word mismatch indicates that the eigenvalues associated with q and the eigenvalues associated with p^* are not the same sets. In other words, $\mathcal{S}_{co}$ is neither $\mathcal{K}(q, B)$ nor $\mathcal{K}(p, B^*)$, but a proper subset.

8. Summary table. Given $B \in \mathbb{C}^{n\times n}$, $q \in \mathbb{C}^n$, $p^* \in \mathbb{C}_*^n$,

$$\begin{aligned} H^{(0)} &:= [p^* B^{i+j-2} q], \qquad i, j = 1, 2, \cdots, \\ l &:= \min\{\nu : H_{\nu+1}^{(0)} \text{is singular}\}, \\ r &:= \text{rank}\,[H^{(0)}], \\ m &:= \min\{\text{rank}[K(q, B)], \text{rank}[K(p^*, B)]\}. \end{aligned}$$

There is a unique pair $(\hat{T}, \Omega)$ with

$$\begin{aligned} K_l(q, B) = Q_l L^t, \qquad & K_l(p^*, B) = L P_l^* \quad \text{(by GS)}. \\ \Omega = \text{diag}\,(\omega_1, \cdots, \omega_l) &= P_l^* Q_l, \\ H_l^{(0)} = L\Omega L^t \quad & \text{(triangular factorization)}, \\ H_l^{(1)} = L\hat{T}L^t, \quad & \hat{T}(i, i+1) = \hat{T}(i+1, i) = \omega_{i+1}, \quad i = 1, \cdots, l-1. \end{aligned}$$

The eigenvalues θ of $(\hat{T}, \Omega)$ will be called Ritz values. In Table 8.1 the phrase "invariant subspace" is an abbreviation for the assertion that either the column space of $K_l(q, B)$ or the row space of $K_l(p^*, B)$ is invariant under B. Recall from §5 that

$$l \leq r \leq m \leq n.$$

TABLE 8.1

Description	Case	Invariant subspace	Ritz values
Benign early termination	$l = r = m < n$	Yes	Each $\theta \in$ spec(B)
Incurable breakdown	$l = r < m$	No	Each $\theta \in$ spec(B)
Curable breakdown	$l < r = m$	No	No $\theta \in$ spec(B)

Generically equality holds throughout and $(\hat{T}, \Omega)$ is equivalent to (B, I). Recall that spec(B) denotes the spectrum of B.

When $l < r$ there is a (block) tridiagonal/diagonal pair $(\hat{T}, \Omega)$ of order r such that each $\theta \in$ spec (B). If and only if $r = m$ then at least one of range $K_r(q, B)$, range $K_r(p^*, B)$ is B-invariant.

9. Examples. To benefit from the examples it is preferable to work out both the computed quantities $(Q, P^*, \cdots)$ and the associated theoretical ones $(H^{(0)}, K, \cdots)$. This has been done in Example 9.1, but only the bare essentials are given in Example 9.2. All the relevant inequalities for l, r, and m are covered.

Example 9.1. Tridiagonalization with initial vectors (q, p^*) breaks down incurably at the end of step 1. Nevertheless, the 1×1 reduced pencil $(5, 1)$ delivers an eigenvalue of B.

$$B = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 0 & 5 & 0 & 6 \\ 0 & 0 & 7 & 8 \\ 0 & 0 & 0 & 9 \end{bmatrix}, \quad q = \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad p = \begin{bmatrix} 0 \\ 1 \\ 0 \\ -1 \end{bmatrix},$$

$$K(q, B) = \begin{bmatrix} 1 & 3 & 13 & \cdot & \cdot \\ 1 & 5 & 25 & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & \cdot \\ 0 & 0 & 0 & \cdot & \cdot \end{bmatrix}, \qquad Q_2 = \begin{bmatrix} 1 & -2 \\ 1 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix},$$

$$K(p^*, B) = \begin{bmatrix} 0 & 1 & 0 & -1 \\ 0 & 5 & 0 & -3 \\ 0 & 25 & 0 & 3 \\ \cdot & \cdot & \cdot & \cdot \end{bmatrix}, \qquad P_2^* = \begin{bmatrix} 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 2 \end{bmatrix},$$

$$H^{(0)} = \begin{bmatrix} 1 & 5 & 25 & \cdot \\ 5 & 25 & \cdot & \cdot \\ 25 & \cdot & \cdot & \\ \cdot & \cdot & & \\ \cdot & & & \end{bmatrix} = \begin{pmatrix} 1 \\ 5 \\ 25 \\ \cdot \\ \cdot \end{pmatrix} (1, 5, 25, \cdot, \cdot),$$

$$\mathcal{S}_{c\bar{o}} = \text{span}(e_1), \qquad \mathcal{S}_{co} = \text{span}(e_2),$$
$$\mathcal{S}_{\bar{c}\bar{o}} = \text{span}(e_3), \qquad \mathcal{S}_{\bar{c}o} = \text{span}(e_4),$$
$$\hat{T} = [5], \qquad \Omega = [1].$$

TABLE 9.1

Case	Starting vectors	Markov parameters	$l\ r\ m$
(i)	$q = e_1, p = e_1$	$1\,0\,0\,0\,0\cdots$	1 1 1
(ii)	$q = e_2, p = e_2$	$1\,0\,0\,0\,0\cdots$	1 1 2
(iii)	$q = e_3, p = e_2$	$0\,1\,0\,0\,0\cdots$	0 2 3
(iv)	$q = e_3, p = e_1$	$0\,0\,1\,0\,0\cdots$	0 3 3
(v)	$q = e_4, p = e_1$	$0\,0\,0\,1\,0\cdots$	0 4 4
(vi)	$q = p = e_1 + e_2 + e_3 + e_4$	$4\,3\,2\,1\,0\cdots$	2 4 4

Example 9.2.

$$B = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

- In Table 9.1, case (i) yields benign early termination since q_1 is an eigenvector.
- Case (ii) yields incurable breakdown after one step with $T = [0]$ and this reveals an eigenvalue but no eigenvector.
- In cases (iii), (iv), and (v) the reduction cannot begin and $H^{(0)}$ does not permit triangular factorization. The extended algorithm breaks down incurably after one block step revealing 2, 3, or 4 eigenvalues, respectively.
- Case (vi) suffers curable breakdown at step 2. The extended algorithm yields

$$\hat{T}_4 = \begin{pmatrix} 3 & -1/4 & 0 & 0 \\ -1/4 & -5/16 & 0 & 1 \\ 0 & 0 & 1 & -2 \\ 0 & 1 & -2 & 1 \end{pmatrix}, \qquad \Omega_4 = \begin{pmatrix} 4 & 0 & 0 & 0 \\ 0 & -1/4 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & -2 \end{pmatrix}.$$

The pair $(\hat{T}_2, \Omega_2)$ gives no useful information about B. The Markov parameters of the system (B, q_3, p_3^*) are $(0, 1, -2, 1, 0, \cdots)$. Although $(\hat{T}_4, \Omega_4)$ is equivalent to (B, I), it is less informative than $(\tilde{I}B, \tilde{I})$ but does have smaller bandwidth! Note that the Ω_4 is not unique. The one shown here is produced by the extended Gram–Schmidt algorithm of §3 but another good choice puts the (4,4) entries of T and Ω to -3 and 0, respectively.

10. Monitoring the condition number. For some applications it is meaningful to enrich $\mathbb{C}^n$ with the Euclidean inner product $\langle v, w\rangle := w^*v$, $w \in \mathbb{C}^n, v \in \mathbb{C}^n$. In this setting it is appropriate to normalize all the auxiliary vectors $\{q_i\}$ and $\{p_i\}$ so that

$$\|q_i\|^2 := \langle q_i, q_i\rangle = \langle p_i, p_i\rangle = \|p_i\|^2 = 1, \quad \text{all } i.$$

Let $\sigma_{\min}(X)$ denote the smallest singular value of X. Let

$$\|X\| = \max_{u \neq 0} \|Xu\|/\|u\|,$$

the spectral norm.

THEOREM 10.1. *If the columns of Q_j and P_j are unit vectors and if $P_j^* Q_j = \hat{\Omega} = \operatorname{diag}(\Omega_1, \Omega_2, \cdots, \Omega_k)$, then*

$$\sigma_{\min}(Q_j) \geq \min_{1 \leq i \leq k} \sigma_{\min}(\Omega_i)/\sqrt{j},$$
$$\sigma_{\min}(P_j) \geq \min_{1 \leq i \leq k} \sigma_{\min}(\Omega_i)/\sqrt{j}.$$

Proof. There exist unit vectors $u \in \mathbb{C}^n$, $v \in \mathbb{C}^j$ such that

$$Q_j v = u \sigma_{\min}(Q_j).$$

Then

$$\begin{aligned} \min_{1 \leq i \leq k} \sigma_{\min}(\Omega_i) &\leq \|\hat{\Omega} v\| \\ &= \|P_j^* Q_j v\| \\ &\leq \|P_j^*\| \|Q_j v\| \\ &= \|P_j^*\| \|u \sigma_{\min}(Q_j)\| \\ &\leq \sqrt{j} \cdot \sigma_{\min}(Q_j). \end{aligned}$$

The first inequality uses

$$\|\hat{\Omega} v\|^2 = \sum_{i=1}^{k} \|\Omega_i v^{(i)}\|^2 \geq \min_{1 \leq i \leq k} \sigma_{\min}^2(\Omega_i) \sum_{i=1}^{k} \|v^{(i)}\|^2$$

and $v^{(i)}$ denotes the ith set of entries in v. The last inequality makes use of the relations

$$\|P_j^*\|^2 = \|P_j^* P_j\| = \lambda_{\max}(P_j^* P_j) \leq \operatorname{trace}(P_j^* P_j) = j. \qquad \square$$

When Ω_i is 1×1, then $\sigma_{\min}(\Omega_i) = |\Omega_i|$. In any case the quantities $\|\Omega_i^{-1}\|^{-1}$ are readily computed during or after reduction. When the matrices Q_j and P_j are built up column by column and normalized, it is necessary to compute the quantities $\|r_i\|, \|s_i^*\|$, $i = 1, \cdots, j$. The vectors r_i and s_i^* are defined in the proof of Theorem 2.2. Despite the extra storage required we advocate keeping $\omega_i, \|r_i\|, \|s_i^*\|$, and $\alpha_i = p_i^* B q_i$. The resulting tridiagonal matrix $\hat{T}$ is then defined by

$$\hat{T}(i,i) = \alpha_i, \quad \hat{T}(i, i-1) = \|r_i\| \omega_i, \quad \hat{T}(i-1, i) = \|s_i^*\| \omega_i,$$
$$\Omega = \operatorname{diag}(\omega_1, \cdots, \omega_l).$$

The point is that all the stored quantities give information directly relevant to the stability of the transformation. It is valuable to know whether a small value $\hat{T}(i, i-1)$ comes from a small ω_i or a small $\|r_i\|$.

10.1. A stable reduction. In the block reduction algorithm that would produce the pair $(\hat{T}, \hat{\Omega})$ of §6 there is no need to insist on the lower triangular Hankel form for the blocks

$$\Omega_i = \begin{bmatrix} 0 & 0 & * \\ 0 & * & * \\ * & * & * \end{bmatrix}.$$

When the setting of Euclidean space is appropriate, we can select a tolerance tol and require that

$$\sigma_{\min}(\Omega_i) \geq \text{tol}, \quad \text{all } i.$$

Under this restriction the algorithm will produce in all cases the most refined pair $(\hat{T}_r, \hat{\Omega}_r)$ that satisfies the following conditions:

1. $(\hat{T}_r, \hat{\Omega}_r)$ is a minimal realization.
2. $\sigma_{\min}(Q_r) \geq \text{tol}/\sqrt{r}, \sigma_{\min}(P_r^*) \geq \text{tol}/\sqrt{r}$.

This is a satisfactory method insofar as it extracts as much useful information as possible from the choice (q, p^*). It is not satisfactory insofar as this most compact structure is not known in advance and examples can be constructed in which $\hat{T}_r$ and $\hat{\Omega}_r$ will be of maximal bandwidth. See case (v) in Table 9.1.

11. Comments on implementation.

11.1. Preconditioning for fixed start methods. We know of no way to choose starting pairs (q, p^*) that are guaranteed not to be exceptional. In some applications, e.g., when a sequence of close matrices must be analyzed, good choices for the pair may be known. In the absence of such additional information our theory suggests that q and p^* should be chosen at random from a uniform distribution.

In practice there is merit in having $\hat{T}$ a graded matrix with large entries near the top of the matrix. When $\hat{T}$ has such a structure it is easier to find the eigenvalues of $(\hat{T}, \Omega)$ in a roughly monotonic order (by decreasing absolute value). These considerations suggest that

$$q = B^\nu(\text{ random}), \quad p^* = (\text{random })^* B^\nu, \quad \nu \geq 1,$$

should be preferable to random vectors. On the other hand, each application of B devoted to a starting vector can be regarded as a waste of a step in tridiagonal reduction. Yet an attractive feature of choosing $\nu \geq 1$ is that it forces $\mathcal{K}(q, B) \subset \mathcal{R}(B)$ and $\mathcal{K}(p, B^*) \subset \mathcal{R}(B^*)$. This is essential when B is not just a matrix but an operator with unwanted infinite eigenvalues, as can occur in generalized eigenvalue problems.

We have used $\nu = 1$ in the symmetric case and advocate the same policy here. It is easy to implement and seems to keep breakdowns at bay but we have no theoretical justification for it.

11.2. Stable reduction to block tridiagonal form. When the Euclidean inner product is appropriate, then the idea (in §10) of controlling the condition number of Q and P may be combined with the block reduction of §6. Thus we suppose that $\|q_i\| = \|p_i^*\| = 1$, all i, and there will be diagonal positive definite scaling matrices D_q and D_p. The new matrices Q_j and P_j will satisfy

$$\text{new } \Omega_j = P_j^* Q_j = D_p \Omega_j D_q$$

where Ω_j is the Hankel matrix discussed in §6.

A suitable lower bound tol on $\min_i \sigma_{\min}(\Omega_i)$ may be selected. If ϵ denotes the roundoff unit, then a value such as $\epsilon^{1/2}$ is a natural choice. Then the Lanczos algorithm may be applied in the normal way. However, if, at the end of step j, the new Lanczos vectors q_{j+1} and p_{j+1} satisfy $\omega_{j+1} := p_{j+1}^* q_{j+1} <$tol the algorithm uses $B^\nu q_{j+1}$ for $q_{j+\nu+1}$ and $p_{j+1}^* B^\nu$ for $p_{j+\nu+1}^*$ until

$$\sigma_{\min}[H_\nu^{(0)}(B, q_{j+1}, p_{j+1}^*)] \geq \text{tol}.$$

The $\hat{T}$ and Ω resulting from this strategy differ from the convenient "symmetric Hessenberg" form by matrices of small norm. This should facilitate the solution of the auxiliary problem $(\hat{T} - \lambda\Omega)u = \mathbf{0}$.

For determining eigenvalues alone it is the ratios $|\omega_{i+1}/\omega_i|^{1/2}$ that matter. While these quantities and the $|\alpha_i/\omega_i|$ remain bounded by a small multiple of $\|B\|$ the reduced pair (T_j, Ω_j) is completely satisfactory.

11.3. Local change to the starting pair. If serious breakdown occurs late in the reduction to tridiagonal form, then a restart with a new (random) starting pair amounts to a complete write-off of the expenses of the first attempt. Geist, Lu, and Wachspress, in [8], have studied a compromise in which the cost of reduction with the new pair is very small compared with the initial reduction. However, the theory developed in this essay shows that there are limitations on this technique. Recall the four indices that characterize the realization: l, r, m, and n.

The limitation is that, although l may be increased (which is good), both r and m either remain the same or decrease.

To justify these comments we describe briefly the way that Geist, Lu, and Wachspress carry out the reduction. They perform a sequence of explicit similarity transformations on B; at step j, row j and column j of the current matrix are put into tridiagonal form. Let T_j be the $j \times j$ tridiagonal obtained at the end of step j and suppose that serious breakdown is detected with $B(j+1, j) = 0$, $B(j, j+1) \neq 0$. Their remedy is to apply an elementary similarity transformation on the first two rows and columns of the current array. The transformation matrix is of the form

$$\left[\begin{pmatrix} 1 & 0 \\ \xi & 1 \end{pmatrix} \oplus I_{n-2}\right] \quad [B] \left[\begin{pmatrix} 1 & 0 \\ -\xi & 1 \end{pmatrix} \oplus I_{n-2}\right]$$

and brings nonzero values into position $(3, 1)$. This bulge in the tridiagonal form is chased down the matrix, from $(3, 1)$ to $(4, 2)$ to $(5, 3)$ $\cdots$, in a way that is familiar to those who have studied the symmetric tridiagonal QL algorithm. The cost of this chasing procedure is a small multiple of j and the result is a new value in position $(j + 1, j)$. If the new value is also tiny, then the whole procedure may be tried again.

We claim:

1. The recovering procedure is equivalent to replacing q_1 by $q_1 - \xi q_2 = \theta q_1 + \eta B q_1$, while leaving p_1 unchanged.
2. Let $\mathcal{K}(v) = \text{span}(v, Bv, B^2v, \cdots)$. Then

$$\begin{aligned} \mathcal{K}((\theta I + \eta B)q_1) &= \{(\theta I + \eta B)\phi(B)q_1 : \phi \text{ ranges over all polynomials}\} \\ &\subset \mathcal{K}(q_1). \end{aligned}$$

   Thus

$$(\mathcal{K}(p_1^*)) \cap \mathcal{K}((\theta I + \eta B)q_1\xi) \subset (\mathcal{K}(p_1^*) \cap \mathcal{K}(q_1)) = \mathcal{S}_{co},$$

   and the new values of m and r (see §8) cannot be increased and may decrease.
3. $H_{l+1}^{(0)}(\theta q + \eta B q, p^*, B) = \theta H_{l+1}^{(0)}(q, p^*, B) + \eta H_{l+1}^{(1)}(q, p^*, B)$.
   Thus the new $H_{l+1}^{(0)}$ need not be singular.

12. Conclusion. The last few sections may have deflected the reader's attention away from the big picture so we take the opportunity to recapitulate the main points. Although the canonical structure theorem supplies the right decomposition for understanding incurable breakdown, yet linear systems theory is not itself relevant. The

matrix B is given and, once the vector q and linear functional p^* are chosen, then the "moment" matrix $H^{(0)}$ is fixed and it determines the numbers l and r while B, q, and p^* determine m. See §5 for definitions.

In the generic case $l = r = m = n$ and tridiagonalization succeeds although it need not be stable. However, §10 shows a natural way to monitor the condition number of the transformation and this greatly improves the situation because some of the algorithms proposed for reduction hide the onset of instability by using drastic but hidden scaling of the columns of P and Q to force ${P_j}^* Q_j = I_j$.

In the nongeneric case, there is always a block triangular factorization of $H^{(0)}$ with minimal sizes for the diagonal blocks. We have, in effect, lifted Gragg's block form back to the GS procedure to produce our extended GS algorithm, which shows how to continue the algorithm by working with several vectors simultaneously.

This is not the only way to overcome a curable breakdown. Gutknecht's approach in [14] may be described briefly in the following way. Recall that before breakdown occurs $K_k(q) = Q_k {L_k}^t$ and, with $L_k^{-1} := (\lambda_{ij})$, for $j < k$,

$$q_{j+1} = K_k {L_k}^{-t} e_{j+1} = \sum_{i=1}^{j+1} (B^{i-1} q_1) \lambda_{j+1,i} = \phi_j(B) q_1$$

where

$$\phi_j(t) = \sum_{i=1}^{j+1} \lambda_{j+1,i}\, t^{i-1}, \qquad \lambda_{j+1,j+1} = 1,$$

is a monic polynomial of degree j and is sometimes called the jth (generalized) Lanczos polynomial. The $\{\phi_j\}_1^k$ is called a sequence of **formal** orthogonal polynomials. The proper L^2 inner product function $\langle f, g \rangle$ is replaced by an appropriate linear functional on the pointwise product $f\,g$. The three-term recurrence connecting standard orthogonal polynomials extends to this more general setting—in the absence of breakdown. Gutknecht shows how to define polynomials ϕ_{k+1}, $\phi_{k+2}, \cdots$ (and hence rows of L^{-1}) after each curable breakdown in such a way as to preserve as much as possible of the standard recurrence relations. As mentioned in §6 the result is equivalent to choosing L, whenever there is freedom, to make the diagonal blocks Ω_i in Theorem 6.1 antidiagonal, i.e., all zeros except along the secondary (NE–SW) diagonal.

By accepting blocks that are larger than minimal size given in §6, it is straightforward to set an upper bound on the condition number of Q and P and then to compute the most refined block tridiagonal form $(\hat{T}, \Omega)$ consistent with this bound. Nevertheless triangular factorization cannot proceed beyond the effective rank of $H^{(0)}$ because the Schur complement of ${H_r}^{(0)}$ vanishes. It is at this stage that the Kalman–Gilbert theorem can be invoked to show that $(\hat{T}_r, \Omega_r)$ is a minimal realization of B, q, p^*. In other words, for some conformable r-vectors $\tilde{q}$ and $\tilde{p}$, and for all σ in the resolvent set,

$$\tilde{p}^* (\sigma\, \Omega_r - \hat{T}_r)^{-1}\, \tilde{q} = p^* (\sigma\, I - B)^{-1} q.$$

It is this transfer function approach that shows immediately that each eigenvalue of $\hat{T}$, Ω is an eigenvalue of B.

For specialists in large eigenvalue problems, it is unnerving to have a solution λ to $Bz = z\lambda$ without knowledge of an appropriate invariant subspace. To algebraists who invoke the characteristic polynomial, this absence of z is a natural state of affairs.

Our approach stops with a minimal realization but there are further questions to be asked. If $r < m$ then how is it possible to append further columns q_{r+1}, $q_{r+2}, \cdots$ or further functionals $p^*{}_{r+1}$, $p^*{}_{r+2}, \cdots$ to obtain bases for the controllable and observable subspaces? This problem is addressed in [2]. Their idea is to modify the Lanczos algorithm a little.

After incurable breakdown at step r the remaining pair q_{r+1}, p^*_{r+1} is used to generate more vectors according to

$$q_{r+i+1} = (I - \Pi_r)Bq_{r+i}, \qquad p^*_{r+i+1} = p^*_{r+i}B(I - \Pi_r),$$

until linear independence is lost. The new q's span $\mathcal{S}_{c\bar{o}}$ and the new p's span $\mathcal{S}_{\bar{c}o}$. The final space $\mathcal{S}_{\bar{c}\bar{o}}$ must be obtained, if needed, by more primitive means. It is the null space of the matrix

$$(p_1, \cdots, p_r, p_{r+1}, \cdots, p_m, q_{r+1}, \cdots, q_l)^*$$

obtained by appending the new p's and q's to the columns of P_r. The extra coefficients generated by these modifications permit the calculation of the canonical form in §7 and with it the remaining eigenvalues and all the eigenvectors.

Although the moment matrix $H^{(0)}$ is an essential theoretical tool it is not available in practice nor indeed are the Krylov matrices $K(q, B)$ and $K(p^*, B)$. From the practical point of view the essential, impressive insight of Cornelius Lanczos was to recognize that Bq_j is preferable to $B^j q_1$, and $p_j^* B$ to $p_1^* B^j$, for the purpose of computing Q, P, $\hat{T}$, and Ω.

Plenty of questions remain unanswered. Is it advisable to put more effort into the choice of q_1 and p_1^* ? Is it better to restart after an early serious breakdown or to continue with a block tridiagonal output? What are good ways to compute, or update, the partial eigensolution of the reduced problem $(\hat{T} - \lambda\Omega)s = 0$? Can the residual error bounds presented in [17] be used for terminating the reduction when only a few eigenpairs of B are wanted? How should we compensate for the effect of roundoff error?

This paper sought to provide a good framework for looking at reduction to tridiagonal form.

REFERENCES

[1] G. Birkhoff and S. Maclane, *A Survey of Modern Algebra*, Chapter XV, Macmillan, New York, 1953.

[2] D. Boley and G. H. Golub, *The nonsymmetric Lanczos algorithm and controllability*, Tech. Report NA-90-06, Computer Science Department, Stanford University, Stanford, CA, May 1990.

[3] A. Draux, *Polynômes Orthogonaux Formels-Applications*, Lecture Notes in Mathematics, Vol. 974, Springer-Verlag, Berlin, Heidelberg, New York, 1983.

[4] V. N. Faddeeva, *Computational Methods of Linear Algebra*, Dover, New York, 1959.

[5] J. G. F. Francis, *The* QR *transformation, Part* I, Comput. J., 4 (1961), pp. 265–271.

[6] ———, *The* QR *transformation, Part* II, Comput. J., 4 (1962), pp. 332–345.

[7] G. A. Geist, *Reduction of a general matrix tridiagonal form*, Tech. Report, ORNL/TM-10991, Oak Ridge National Laboratory, Oak Ridge, TN, March 1989.

[8] G. A. Geist, A. Lu and E. L. Wachspress, *Stabilized Gaussian reduction of an arbitrary matrix to tridiagonal form*, Tech. Report ORNL/TM-11089, Oak Ridge National Laboratory, Oak Ridge, TN, June 1989.

[9] E. Gilbert, *Controllability and observability in multivariable control systems*, SIAM J. Control, 1 (1963), pp. 128–151.

[10] G. H. Golub and C. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[11] G. H. GOLUB AND M. H. GUTKNECHT, *Modified moments for indefinite weight functions*, Interdisciplinary Project for Supercomputing Research Report No. 89-04, ETH-Zentrum, CH-8092 Zurich, 1989.
[12] W. B. GRAGG, *Matrix interpretations and applications of the continued fraction algorithm*, Rocky Mountain J. Math., 4 (1974), pp. 213–225.
[13] W. B. GRAGG AND A. LINDQUIST, *On the partial realization problem*, Linear Algebra Appl. 50 (1985), pp. 277–319.
[14] M. H. GUTKNECHT, *A completed theory of the unsymmetric Lanczos process and related algorithms: Part* I, SIAM J. Matrix Anal. Appl., this issue, pp. 594–639.
[15] A. S. HOUSEHOLDER, *The Theory of Matrices in Numerical Analysis*, Blaisdell, New York, 1964.
[16] I. S. IOHVIDOV, *Hankel and Toeplitz Matrices and Forms*, G.P.A. Thijsse, trans., Birkhäuser-Verlag, Basel, 1982.
[17] W. KAHAN, B. N. PARLETT, AND E. JIANG, *Residual error bounds on approximate eigensystems of nonnormal matrices*, SIAM J. Numer. Anal., 19 (1982), pp. 470–484.
[18] T. KAILATH, *Linear Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980.
[19] R. E. KALMAN, *Mathematical description of linear systems*, SIAM J. Control, 1 (1963), pp. 152–192.
[20] ———, *On partial realizations, transfer functions and canonical forms*, Acta Polytech. Scand. Math. Comput. Sci. Ser., MA31 (1979), pp. 9–32.
[21] C. D. LABUDDE, *The reduction of an arbitrary real sparse matrix to tridiagonal form using similarity transformations*, Math. Comp., 17 (1963), pp. 443–447.
[22] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45, (1950), pp. 255–282 (see pp. 266–270).
[23] H. LEV-ARI AND T. KAILATH, *Triangular factorization of structured Hermitian matrices*, Operator Theory: Adv. Appl., 18 (1986), pp. 301–324.
[24] B. N. PARLETT, *A note on LaBudde's algorithm*, Math. Comp., 19 (1964), pp. 505–506.
[25] B. N. PARLETT, D. R. TAYLOR AND Z.-S. LIU, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp., 44 (1985), pp. 105–124.
[26] C. STRACHEY AND J. G. F. FRANCIS, *The reduction of a matrix to codiagonal form by elimination*, Comput. J., 4 (1961), pp. 168–176.
[27] D. R. TAYLOR, *Analysis of the look ahead Lanczos algorithm*, Ph.D. thesis, Center for Pure and Applied Mathematics, University of California, Berkeley, CA, 1982. Also as Tech. Report CPAM-108, Center for Pure and Applied Mathematics, University of California, Berkeley, CA.
[28] E. L. WACHSPRESS, ADI *solution of Lyapunov equations*, presented at MSI Workshop on Practical Iterative Methods for Large-Scale Computations, Minneapolis, MN, October 1988.
[29] J. H. WILKINSON, *Instability of the elimination method of reducing a matrix to tridiagonal form*, Comput. J., 5 (1962), pp. 61–70.
[30] ———, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 594–639, April 1992

010

A COMPLETED THEORY OF THE UNSYMMETRIC LANCZOS PROCESS AND RELATED ALGORITHMS, PART I*

MARTIN H. GUTKNECHT†

Dedicated to Gene H. Golub on the occasion of his 60th birthday

Abstract. The theory of the "unsymmetric" Lanczos biorthogonalization (BO) algorithm, which has so far been restricted to an essentially generic situation (characterized by the nonsingularity of the leading principal submatrices of the associated moment matrix or by the existence of a full set of regular formal orthogonal polynomials) is extended to the nongeneric case. The "serious" breakdowns due to the occurrence of two orthogonal right and left iteration vectors $\mathbf{x}_n$ and $\mathbf{y}_n$ can be overcome. For an operator of finite rank N the nongeneric BO algorithm, which generalizes the look-ahead Lanczos algorithm of Parlett, Taylor, and Liu [*Math. Comp.*, 44 (1985), pp. 105–124], terminates regularly in at most N steps, except when a very special situation depending on the initial vectors occurs; but even then the algorithm produces in at most N steps a block tridiagonal matrix whose blocks are either small or sparse and whose characteristic polynomial is the minimal polynomial of the restriction of the operator to an invariant subspace.

Formulas are also derived for a nongeneric version of the corresponding linear equation solver BIORES (brief for BIORTHORES or Lanczos/ORTHORES). The whole theory is developed as a consequence of known corresponding results on formal orthogonal polynomials and Padé approximants, for many of which new and simpler derivations are given.

Key words. Lanczos algorithm, biorthogonalization algorithm, BO algorithm, BIORES, biconjugate gradient algorithm, formal orthogonal polynomial, recurrence, Padé approximation, continued fraction, quotient difference algorithm, qd algorithm

AMS(MOS) subject classifications. 65F10, 65F15, 30E05, 41A21

Introduction. Both in the *Lanczos biorthogonalization* (BO) *algorithm* [24] (the application of which to solving linear systems is often called *Lanczos/ORTHORES* and will be named briefly *BIORES* here) and in the closely related *Lanczos biconjugate gradient* (BCG) *method* (*BIOMIN* or *Lanczos/ORTHOMIN*) [25], [8] a "serious breakdown" unrelated to roundoff may occur before an invariant subspace of the matrix (or a solution of the linear system) has been found. The same is also true for *BIODIR* (or *Lanczos/ORTHODIR*). The premature breakdowns in these algorithms are due to the fact that the algorithms are well defined only in the generic case where certain determinants are different from zero. Likewise, the related *qd algorithm* [40] for computing the eigenvalues of a tridiagonal matrix or the poles of a rational function is well defined only in this case, although Rutishauser suggested some rules for filling the gaps in a qd table with zeros and ∞-symbols and Draux [7] has since justified and further specified such rules.

The orthogonality properties of the (finite) sequences of Krylov space vectors generated by the BO and BCG algorithms are paralleled by orthogonality properties of sequences of polynomials. However, if the matrix is not positive definite, the underlying "inner product" may be indefinite; only in the generic case can we assume that none of the constructed polynomials is orthogonal to itself. It is well known that such sequences of *formal orthogonal polynomials* are the denominators of *Padé approximants* lying on a diagonal or a staircase of a Padé table and that these Padé

* Received by the editors December 11, 1989; accepted for publication (in revised form) March 29, 1991.

† Interdisciplinary Project Center for Supercomputing, Eidgenössische Technische Hochschule Zürich, ETH-Zentrum, CH-8092 Zürich, Switzerland (mhg@ips.ethz.ch).

approximants are also the convergents of *continued fractions.* In fact, Rutishauser [40] mentioned these connections already, and he presented his progressive qd algorithm both as a process for generating a sequence of continued fractions belonging to the various descending staircases in a normal Padé table and as a special case of his LR algorithm for generating a sequence of similar matrices, which under certain assumptions converges to a diagonal matrix exhibiting the eigenvalues.

In contrast to the breakdown problems of the algorithms mentioned above, Padé theorists are used to dealing with nonnormal (i.e., nongeneric) Padé tables containing square blocks of size greater than 1. The corresponding sequences of formal orthogonal polynomials have been discussed by Struble [44], Gragg [13], Nuttall and Singh [32], Draux [7], Gragg and Lindquist [14], and Stahl [42], [43]. These sequences have already come up implicitly in the Padé and continued fraction theory, in particular in Magnus's (*diagonal*) *P-fractions* [26], [27], which are the continued fractions associated to the distinct entries on a diagonal of a nonnormal Padé table. The recurrence relations for these formal orthogonal polynomials lead in a straightforward way to the generalized, *nongeneric* BO *algorithm.*

In an analogous way there are the *nongeneric* BIOMIN (BCG) *algorithm* and nongeneric extensions for the many other related algorithms and constructions: to the nongeneric BIOMIN algorithm now corresponds a pair of *block bidiagonal matrices* L *and* R which are the factors of a particular block LU decomposition of the *block tridiagonal matrix* T associated to the nongeneric BO algorithm. L and R contain the recurrence coefficients for a *generalized staircase sequence* in the Padé table, and these coefficients also specify a *staircase P-fraction.* The *nongeneric* (progressive) *qd algorithm* can then be introduced either as a block LR algorithm applied to block tridiagonal matrices of a particular structure or by contractions and expansions of P-fractions.

For the generic case these connections are reviewed in a companion paper [15], which also lists many more references to related work. In contrast, the present paper is the first in a series in which the theory for this whole circle of ideas is extended to the nongeneric case.

In §1 of this first part we compile the relevant material on formal orthogonal polynomials (FOPs) and Padé approximants. In view of later applications to a *backward qd algorithm* the Padé table is defined (as in [47]) in a half-plane instead of a quadrant. The quite detailed presentation of the material in this section is justified by the importance it has for the understanding of the later sections and the subsequent parts. In §2 the (formal) orthogonality properties of the FOPs are used to derive the *recurrence formulas* for these FOPs and formulas for computing the coefficients in these recurrences. The existence of these recurrences was established by Struble [44] with an amazingly brief argument. But they are also a direct consequence of the standard recurrence formula for continued fractions applied to Magnus's P-fractions [26], [27]. Another short derivation was given by Nuttall and Singh [32]. The recurrences, together with the underlying Padé theory results, were also derived in detail in Draux's monograph [7], but his formulation and proof of the recurrence formula alone require 20 pages, plus some 50 pages of preliminary material. In contrast, our proof (of Theorem 2.7), which is similar to the one in Gragg and Lindquist [14], takes only two pages and should be easy to understand. Moreover, we emphasize the constructive aspects, which have not found much attention so far. Actually, our results are also more general since in view of the numerical application to the Lanczos process we allow in the needed "inner" iterations the generation of a polynomial basis

by some other, arbitrary three-term recurrence formula (or by an even more general procedure), while the other authors had used the monomial basis. (For the Lanczos process this means that in case of a "serious" breakdown we proceed for a few steps by, say, Chebyshev iteration instead of Jacobi iteration until the next orthogonalization step can be completed.)

In §3 we present *matrix formulations* of the results of §2. For the case of the monomial basis the two most relevant results have first been alluded to in a few lines by Gragg [13, p. 222]; in a system theory framework they come up in Gragg and Lindquist [14]; one of them also appears in Draux [7].

In §4 we then turn to the application of the results of §2 to the *nongeneric Lanczos* BO *algorithm* and the corresponding linear system solver *BIORES*. The first one is actually an immediate consequence of those results (or the corresponding matrix results) and of the well-known connections that are valid in the generic case. Hence we believe Gragg [51] to have known this so far unpublished algorithm ever since he wrote [13]. Also Taylor [45] and Parlett, Taylor, and Liu [38] introduced, with their look-ahead Lanczos (LAL) algorithm, a mathematically essentially equivalent procedure, although their derivation, analysis, and implementation followed other lines, and most details were restricted to 2×2 block pivots. In particular, one of Taylor's merits is the very surprising "Mismatch Theorem," which for a finite rank operator means that even in the case of an "incurable" breakdown, i.e., when the process does not terminate, it produces in finitely many steps a divisor of the minimal polynomial of the operator.

The application to linear systems is quite straightforward, although it seems to be new. For BIORES we give as in the generic case [15] an unnormalized version, which does not threaten to break down due to the possible singularity of a restriction of the operator to one of the generated Krylov spaces.

It is not yet known whether this nongeneric theory will have much impact on the numerical computation of eigenvalues of nonsymmetric matrices since the occurrence of exact "serious breakdowns" seems to be very rare in practice (Parlett [52]) and the numerical difficulties with the Lanczos process are rather due to other effects [33], [34], [36], [37]. However, the situation is different when these ideas are applied to solving linear systems of equations $\mathbf{Az} = \mathbf{b}$. Then it is not so important that the implicitly constructed sequence of residual polynomial ends with a minimal polynomial of some Krylov subspace. It suffices to find a polynomial whose modulus is sufficiently small on the spectrum of $\mathbf{A}$ (or, rather, on the pseudospectrum of $\mathbf{A}$ [46]), but it is important to generate such a polynomial in a stable way. There is hope that the nongeneric theory will be useful here, because it allows us to replace an unstable situation by a nearby degenerate stable one.

Finally, some remarks about the notation: $\mathbb{C}$, $\mathbb{Z}$, $\mathbb{N}$, $\mathbb{N}^+$ are, respectively, the sets of complex numbers, integers, nonnegative integers, and positive integers. $\mathcal{P}_m$ denotes the set of complex polynomials of degree at most m, $\mathcal{P}$ is the set of formal power series (with complex coefficients), and $\mathcal{L}$ the set of formal Laurent series. The actual degree of a polynomial p is denoted by ∂p. A formal Laurent series f in z satisfies $f(z) = O_+(z^m)$ if it contains only terms $\phi_k z^k$ with $k \geq m$, and likewise $f(z) = O_-(z^m)$ if the series contains only terms $k \leq m$. We write $f(z) \equiv O_+(z^m)$ and $f(z) \equiv O_-(z^m)$, respectively, if additionally $\phi_m \neq 0$. A rational function is said to be of *type* (m, n) if it can be represented as p/q with $p \in \mathcal{P}_m$ and $q \in \mathcal{P}_n$ $(q \neq 0)$. It is of *exact type* (m, n) if m and n are smallest possible. The set of rational functions of type (m, n) is denoted by $\mathcal{R}_{m,n}$. If $r \in \mathcal{R}_{m,n}$ has exact type (μ, ν), the nonnegative

integer $\delta := \min\{m - \mu, n - \nu\}$ is called the *defect* of r in $\mathcal{R}_{m,n}$. If $\delta > 0$, r is called *degenerate*.

We have aimed at using consistent notation over the several parts of this work, and to adhere to at least some of the standard notation in the field. Unfortunately, the standard notation from the several areas that play a role here is incompatible: In orthogonal polynomials, the polynomials of the first kind are usually called p_n, those of the second kind q_n, while in Padé approximation, essentially the same polynomials are often denoted in just the opposite way. Further inconsistencies exist, e.g., with respect to the variable used (z or z^{-1}) and the notation for the recurrence coefficients for these polynomials in orthogonal polynomial theory, Padé approximation, and continued fraction theory. Moreover, even in the closely related areas of the Lanczos algorithm and the conjugate gradient method, the names α_n and β_n are used for two related but fundamentally different pairs of coefficient sequences. Hence, inevitably, not every reader can expect to find his favorite notation.

Note on recent related work.[1] While this paper was being worked out, several other people also turned to the subject of curing serious Lanczos breakdowns. Parlett [35] extended the LAL algorithm, thus essentially obtaining what we call the nongeneric BO algorithm. As in [38], [45] the argumentation is based on the "two-sided block Gram–Schmidt process" and avoids the formal orthogonal polynomials. He also gives an elegant account of the relation to the partial realization problem of system theory and discusses in particular the genericness of termination without breakdown and the case of incurable breakdown. His paper is an excellent complement to ours. The relations of the nongeneric Lanczos process to systems theory are also the subject of a paper by Boley and Golub [5]. Moreover, Boley [3] derived a version of the nongeneric BO algorithm from the block Gram–Schmidt biorthogonalization process and, as in Golub and Gutknecht [10], he discusses the connection to the moment and the modified moment problem. In addition, he explores the application to weighted checksum error correction schemes. (An extended version of his notes is published as [4].) Independently from all these people, David Young's student Joubert, starting from [19], [22], [38], developed in his thesis [21] a version of the nongeneric BO algorithm and combined it with BIODIR to obtain a nongeneric BIODIR algorithm for solving linear systems. He also discussed in detail the breakdown conditions for BIORES, BIOMIN, and BIODIR (a subject also treated in a different way in [15]), and proved that termination without breakdown is a generic situation with respect to complex initial vectors, but not necessarily with respect to real initial vectors. For a summary of his results, see [20].

Although we have been aware of the possibility of deriving the nongeneric Lanczos algorithm directly via the Gram–Schmidt biorthogonalization process, we have chosen to use the orthogonal polynomial approach since we believe that

(i) for dealing with Krylov space vectors, and in particular for discussing the nonsymmetric Lanczos algorithm, the polynomial formulation is the most appropriate, since it allows us to replace one or even two sequences of vectors by one sequence of polynomials containing all the essential information;
(ii) the connections to Padé approximation and the moment problem yield additional insight;
(iii) the polynomial approach is the key to the (bi)conjugate gradient squared methods [15];

[1] Added in revision.

(iv) the Padé and continued fraction connection is also a key to the qd algorithm and to the superfast Hankel solvers [15].

1. Formal orthogonal polynomials and Padé approximants. Given a complex sequence $\{\phi_k\}_{k=0}^{\infty}$, let the linear functional $\Phi_0 : \mathcal{P} \to \mathbb{C}$ be defined by the values it takes on the monomial basis

$$\Phi_0(z^k) := \phi_k \qquad (k \in \mathbb{N}).^2$$

DEFINITION. $P_n \in \mathcal{P}_n$ $(n \geq 1)$ is *a normalized true nth formal orthogonal polynomial (of the first kind)* (FOP1) with respect to Φ_0 if P_n is monic of degree n and

$$\Phi_0(pP_n) = 0 \qquad (\forall p \in \mathcal{P}_{n-1}).$$

P_n is called a *regular* FOP1, if it is uniquely determined; otherwise it is called a *singular* FOP1 [7]. $P_0(z) :\equiv 1$.

Note that it is not required that $\Phi_0(P_n^2) \neq 0$. In general, a true nth FOP1 need not exist for all n. We will later define *deficient* FOP1s for those n where no true FOP1 exists.

In view of various relations we want to explore, we generalize the above situation slightly and consider a whole family of sequences of FOP1s.

After replacing $\{\phi_k\}_{k=0}^{\infty}$ by a doubly infinite series $\{\phi_k\}_{k=-\infty}^{\infty}$, a linear functional $\Phi_l : \mathcal{P} \to \mathbb{C}$ can be defined for each $l \in \mathbb{Z}$ by

$$\Phi_l(z^k) := \phi_{k+l} \qquad (k \in \mathbb{N}). \tag{1.1}$$

An nth true FOP1 with respect to Φ_l is denoted by $P_{l;n}$. By definition, $P_{l;n}$ is monic and satisfies

$$\Phi_l(pP_{l;n}) = 0 \quad (\forall p \in \mathcal{P}_{n-1}). \tag{1.2}$$

Again, we set $P_{l;0}(z) :\equiv 1$. Of course, it is assumed that $\phi_k \neq 0$ for some $k \in \mathbb{Z}$. For simplicity, we deal first with the case $l \geq 0$ only, where we can assume $\phi_k := 0$ if $k < 0$. Consider the formal power series

$$f(z) := \sum_{k=0}^{\infty} \phi_k z^k. \tag{1.3}$$

Since $z^n P_{l;n}(z^{-1})$ is a polynomial, the product $f(z)z^n P_{l;n}(z^{-1})$ is also a formal power series. If

$$P_{l;n}(z) = \sum_{j=0}^{n} \pi_{l;j,n} z^j, \tag{1.4}$$

then

$$f(z)z^n P_{l;n}(z^{-1}) = \sum_{i=0}^{\infty} \left(\sum_{j=0}^{n} \phi_{i+j-n}\, \pi_{l;j,n} \right) z^i. \tag{1.5}$$

[2] Concerning notation, we stick with the traditional one of writing z^k instead of $(.)^k$, although the latter would allow us to distinguish the monomial from its value at z.

Now with (1.4) and in view of the definition (1.1), condition (1.2), written down for $p(z) = z^k$, $k = 0, \cdots, n-1$, yields a homogeneous system of n linear equations in $n+1$ unknowns:

$$\sum_{j=0}^{n} \phi_{i+j-n}\, \pi_{l;j,n} = 0, \qquad i = l+n, \cdots, l+2n-1, \tag{1.6}$$

or

$$\begin{bmatrix} \phi_l & \phi_{l+1} & \cdots & \phi_{l+n-1} & \phi_{l+n} \\ \phi_{l+1} & \phi_{l+2} & \cdots & \phi_{l+n} & \phi_{l+n+1} \\ \vdots & \vdots & \iddots & \vdots & \vdots \\ \phi_{l+n-1} & \phi_{l+n} & \cdots & \phi_{l+2n-2} & \phi_{l+2n-1} \end{bmatrix} \begin{bmatrix} \pi_{l;0,n} \\ \pi_{l;1,n} \\ \vdots \\ \pi_{l;n,n} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}.$$

The following well-known result follows immediately.

THEOREM 1.1. *A* FOP1 $P_{l;n}$ *exists if and only if* (1.6) *has a solution with* $\pi_{l;n,n} = 1$. *It is regular* (*i.e., unique*) *if and only if the* $n \times n$ *moment matrix*

$$\mathbf{M}_{l;n} := \begin{bmatrix} \phi_l & \phi_{l+1} & \cdots & \phi_{l+n-1} \\ \phi_{l+1} & \phi_{l+2} & \cdots & \phi_{l+n} \\ \vdots & \vdots & & \vdots \\ \phi_{l+n-1} & \phi_{l+n} & \cdots & \phi_{l+2n-2} \end{bmatrix} \tag{1.7}$$

is nonsingular.

By (1.6) the coefficient of z^i in (1.5) vanishes for $i = l+n, \cdots, l+2n-1$. Therefore, (1.5) becomes

$$f(z) z^n P_{l;n}(z^{-1}) = p_{l+n-1,n}(z) + O_+(z^{l+2n}), \tag{1.8a}$$

with

$$p_{l+n-1,n}(z) := \sum_{i=0}^{l+n-1} \left(\sum_{j=0}^{n} \phi_{i+j-n}\, \pi_{l;j,n} \right) z^i \tag{1.8b}$$

a polynomial of degree at most $n+l-1$.

In order to further investigate the FOP1 $P_{l;n}$ we will now relate it to a Padé approximant of f.

DEFINITION. Given $f \in \mathcal{P}$, $m \in \mathbb{N}$, and $n \in \mathbb{N}$, an (m,n) *Padé form* of f is any pair $(p,q) \in \mathcal{P}_m \times \mathcal{P}_n$ with $(p,q) \neq (0,0)$ for which

$$f(z)q(z) - p(z) = O_+(z^{m+n+1}). \tag{1.9}$$

The corresponding rational function $r_{m,n} = p/q$ is called (m,n) *Padé approximant.* If r is also a solution of

$$f(z) - r(z) = O_+(z^{m+n+1}), \tag{1.10}$$

we call it a *true* (m,n) *Padé interpolant.* Otherwise, we say that it is a *deficient* Padé approximant.

By definition, $r_{m,n} = p/q$ is determined via the *linearized confluent rational interpolation problem* (1.9). While this problem always has a solution (as we will see

in a moment), the corresponding *nonlinear confluent rational interpolation problem* (1.10) may have no solution $r \in \mathcal{R}_{m,n}$.

By comparing (1.8a) and (1.9) it becomes clear that with

$$q_{l+n-1,n}(z) := z^n P_{l;n}(z^{-1}) \tag{1.11}$$

the pair $(p_{l+n-1,n}\,,\ q_{l+n-1,n}) \in \mathcal{P}_{l+n-1} \times \mathcal{P}_n$ is an $(l+n-1,n)$ Padé form of f. On the other hand, if (p,q) is any $(l+n-1,n)$ Padé form of f, then setting $P_{l;n}(z) := z^n q(z^{-1})$ yields (1.8a), which, as a restriction for $P_{l;n} \in \mathcal{P}_n$ is equivalent to (1.2). *Hence, $P_{l;n}$ so defined is a true nth normalized* FOP1 *with respect to Φ_l if and only if it has exact degree n and is monic.* This is the basic relationship between FOP1s and Padé approximants. In the above derivation, $n > 0$ is assumed. If $n = 0$, the linear system (1.6) is void. But since we have set $P_{l;0} := 1$ and since $q(z) \equiv 1$ for a normalized $(m,0)$ Padé fraction, the basic relation persists.

Given f, m, and n, every (m,n) Padé form can be constructed by solving the linear system (1.6) with $l + n - 1 = m$, i.e., $l := m - n + 1$, and defining, for *any* nontrivial solution $\{\pi_{l;j,n}\}_{j=0}^{n}$ of (1.6),

$$q(z) := \sum_{j=0}^{n} \pi_{l;j,n}\, z^{n-j}\,, \qquad p(z) := \sum_{i=0}^{m} \left(\sum_{j=0}^{n} \phi_{i+j-n}\, \pi_{l;j,n} \right) z^i; \tag{1.12}$$

cf. (1.4), (1.11), and (1.8b). Note that the system (1.6) determines the coefficients of q, while p is then determined by q.

The discussion of the Padé forms and of the related question of existence of FOP1s is often based on a lengthy discussion of the linear system (1.6), its possible rank deficiency, etc. Since it is a Hankel system, proofs can be worked out much easier in term of polynomials and formal power series, however. This approach has been applied in [17] to the more difficult Newton–Padé approximation problem (or, rational interpolation problem), where Hankel matrices are replaced by divided difference matrices.

In our opinion the most important basic result is the following one on the general form of Padé forms [12].

THEOREM 1.2. *Given $f \in \mathcal{P}$, $m \in \mathbb{N}$, and $n \in \mathbb{N}$, the general solution $(p,q) \in \mathcal{P}_m \times \mathcal{P}_n$ of* (1.9) *is*

$$(p,q) = (z^\sigma \hat{p}_{m,n} w,\ z^\sigma \hat{q}_{m,n} w), \tag{1.13}$$

where $\hat{p}_{m,n} \in \mathcal{P}_m$ and $\hat{q}_{m,n} \in \mathcal{P}_n$ are uniquely determined, relatively prime polynomials; $\hat{q}_{m,n}(0) = 1$, $\sigma := \sigma_{m,n}$ is a fixed integer with

$$0 \le \sigma \le \delta := \delta_{m,n} := \min\{m - \partial \hat{p}_{m,n}\,,\ n - \partial \hat{q}_{m,n}\}, \tag{1.14}$$

and $w \in \mathcal{P}_{\delta-\sigma}$ is arbitrary.

Hence, there always exist (m,n) Padé forms (p,q), and they all yield the same rational function (Padé approximant) $r_{m,n} := p/q = \hat{p}_{m,n}/\hat{q}_{m,n}$.

Note that the Padé form itself is never unique. It is unique up to a scalar factor if and only if $\delta = \sigma$.

Several well-known corollaries of this result can be derived easily.

COROLLARY 1.3. *The (m,n) Padé approximant $r_{m,n} = \hat{p}_{m,n}/\hat{q}_{m,n}$ is a true Padé interpolant if and only if $\sigma = 0$ in* (1.13).

COROLLARY 1.4. *The homogeneous linear system* (1.6) *has rank* $n - \delta + \sigma$, *i.e., row rank deficiency* $\delta - \sigma$.

COROLLARY 1.5 (Characterization Theorem). *A rational function* $r = p/q \in \mathcal{R}_{m,n}$ *with defect* $\delta := \min\{m - \partial p,\ n - \partial q\}$ *is the* (m,n) *Padé approximant of* f *if and only if*

$$f(z) - r(z) = O_+(z^{m+n+1-\delta}). \tag{1.15}$$

COROLLARY 1.6 (Block Structure Theorem). (i) *A rational function* $r \in \mathcal{R}_{\mu,\nu}$ $(r \neq 0)$ *of exact type* (μ,ν) *is a Padé approximant of* $f \notin \mathcal{R}_{\mu,\nu}$ *if and only if*

$$f(z) - r(z) \equiv O_+(z^{\mu+\nu+\Delta+1}) \tag{1.16}$$

for some $\Delta \in \mathbb{N}$. *If the latter holds,* r *is the* (m,n) *Padé approximant* $r_{m,n}$ *of* f *for all pairs* (m,n) *in the square*

$$\{(m,n)\ ;\ \mu \leq m \leq \mu + \Delta\ ,\ \nu \leq n \leq \nu + \Delta\}, \tag{1.17}$$

but for no other pair.[3] *Then, in* (1.13) *there holds* $\partial \hat{p}_{m,n} = \mu$, $\partial \hat{q}_{m,n} = \nu$, $\delta = \min\{m - \mu, n - \nu\}$ *and*

$$\sigma = \max\{0,\ m + n - \mu - \nu - \Delta\}, \tag{1.18}$$

so that

$$\delta - \sigma = \min\{m - \mu,\ n - \nu,\ \mu + \Delta - m,\ \nu + \Delta - n\}. \tag{1.19}$$

(ii) *If* f *is itself rational, say, of exact type* (μ,ν), *then* $r_{m,n} = f$ *exactly for all* (m,n) *in the quadrant* $\{(m,n);\ m \geq \mu,\ n \geq \nu\}$. *Moreover, in* (1.13), $\partial \hat{p}_{m,n} = \mu$, $\partial \hat{q}_{m,n} = \nu$, $\delta = \min\{m - \mu, n - \nu\}$, *and* $\sigma = 0$.

(iii) *The function* $r = 0$ *with exact type* $(-\infty, 0)$ *is a Padé approximant of* $f \neq 0$ *if and only if* $f(z) \equiv O_+(z^k)$ *for some* $k > 0$, *and then* $r_{m,n} = 0$ *exactly for all* (m,n) *in the half-strip* $\{(m,n);\ 0 \leq m < k,\ n \in \mathbb{N}\}$. *Then, in* (1.13), $\hat{p}_{m,n} = 0$, $\hat{q}_{m,n} = 1$, $\delta = n$, *and* $\sigma = \max\{0, m + n + 1 - k\}$.

COROLLARY 1.7 (Definition by Optimality). *The* (m,n) *Padé approximant* $r_{m,n}$ *is, among all* $r \in \mathcal{R}_{m,n}$, *the one for which*

$$f(z) - r(z) = O_+(z^k) \tag{1.20}$$

holds for the largest possible $k \in \mathbb{Z}$.

According to the Block Structure Theorem, Corollary 1.6, the *Padé table*, which contains as (m,n) entry the (m,n) Padé approximant of f, is made up of square blocks of distinct rational functions. Following Gragg [12] we let the m-axis point downwards and the n-axis point to the right. Then, for each block, the position (μ,ν) of the upper left corner indicates the exact type of the approximant. On the border of the block $\delta = \sigma$ holds, which means that the Padé form is unique up to a common scalar factor there. On and above the antidiagonal, $\sigma = 0$ holds, so that the function r is a true Padé interpolant there, while below the antidiagonal, r is a deficient Padé approximant. (Some authors say that the Padé approximant *does not exist* there [1].)

[3] Note that Δ is assumed to be maximal by definition of $\equiv O_+$.

There are two exceptions to these finite square blocks: If the zero function is a Padé approximant, then its block is a half-strip. If f is itself rational, there is an "infinite square block," since in the whole quadrant $\{(m,n);\ m \geq \mu,\ n \geq \nu\}$ the Padé approximant is $r = f$.

The extension to the functionals Φ_l with $l < 0$ is easy after adapting the Padé construction to formal Laurent series with finitely many terms with negative index. More generally we can define the (one-sided) Padé approximant of an arbitrary formal Laurent series [47, §2].[4]

DEFINITION. Given $f \in \mathcal{L}$, $m \in \mathbb{Z}$, and $n \in \mathbb{N}$, divide f into two pieces according to

$$f(z) := z^{m-n+1}(f^-(z) + f^+(z)), \tag{1.21a}$$

where

$$z^{m-n+1}f^-(z) := \sum_{k=-\infty}^{m-n} \phi_k z^k, \qquad z^{m-n+1}f^+(z) := \sum_{k=m-n+1}^{\infty} \phi_k z^k, \tag{1.21b}$$

and let $r^+ := p^+/q$ be the $(n-1,n)$ Padé approximant of $f^+ \in \mathcal{P}$. (If $n = 0$, then $r^+ := 0$.) Then define the (m,n) *Padé approximant* of $f \in \mathcal{L}$ by

$$r_{m,n}(z) := z^{m-n+1}(f^-(z) + r^+(z)). \tag{1.22}$$

It can be checked easily that in the case $f \in \mathcal{P}$ this definition is consistent with the previous one. The function $r_{m,n}$ of (1.22) belongs to the set

$$\tilde{\mathcal{R}}_{m,n} := z^{m-n+1}(\mathcal{P}^\perp + \mathcal{R}_{n-1,n}), \tag{1.23}$$

where $\mathcal{P}^\perp := \{f \in \mathcal{L};\ f(z) = O_-(z^{-1})\}$ is the space of "coanalytic" formal power series. The *exact type* (μ,ν) of $r_{m,n}$ is defined by setting ν equal to the exact degree of q (assuming p^+ and q relatively prime) and μ equal to the index of the highest nonvanishing term in the formal Laurent series of $z^{m-n+1}(f^-(z)q(z) + p^+(z))$. The zero function has type $(-\infty, 0)$, but there are other functions with type $(-\infty,\nu)$ for some $\nu > 0$, e.g., $f(z) = (\cdots + z^{-3} + z^{-2} + z^{-1}) + 1/(1-z)$, which can be represented in every space $\tilde{\mathcal{R}}_{m1}$ $(m \in \mathbb{Z})$ [47, Ex. 2.2]. Two functions in two spaces of type (1.23) are considered equal if the formal Laurent series, which are obtained by expanding the terms from $\mathcal{R}_{n-1,n}$ into a power series, are identical.

With these definitions the Characterization Theorem (Corollary 1.5), the Block Structure Theorem (Corollary 1.6), and the Definition by Optimality (Corollary 1.7) remain valid if $\mathcal{R}_{m,n}$ is replaced by $\tilde{\mathcal{R}}_{m,n}$, and $m \in \mathbb{N}$ by $m \in \mathbb{Z}$. In part (i) of the Block Structure Theorem the assumption $r \neq 0$ is replaced by $\mu \neq -\infty$, and the statement there concerning (1.13) requires an adaptation of Theorem 1.2 and of the notion of a Padé form. The latter could indeed still be used to define $r_{m,n}$. Basically, we just have to replace $\mathcal{P}_m$ by $\tilde{\mathcal{P}}_m := \tilde{\mathcal{R}}_{m,0} := z^{m+1}\mathcal{P}^\perp := \{f \in \mathcal{L};\ f(z) = O_-(z^m)\}$.

DEFINITION. Given $f \in \mathcal{L}$, $m \in \mathbb{Z}$ and $n \in \mathbb{N}$, an (m,n) *Padé form* of f is any pair $(p,q) \in \tilde{\mathcal{P}}_m \times \mathcal{P}_n$, $(p,q) \neq (0,0)$, for which

$$f(z)q(z) - p(z) = O_+(z^{m+n+1}). \tag{1.24}$$

[4] The following paragraphs are not so important for the understanding of the Lanczos algorithm. Readers who are just interested in the latter may assume $l = 0$ and $\phi_k = 0$ if $k < 0$, so that $m = n-1$, $f^- = 0$, $f^+ = f$, and $r^+ = r$; using this setting they can then proceed to Theorem 1.8.

Note that any second member q of an $(n-1,n)$ Padé form of f^+ is a second member of an (m,n) Padé form of f, and vice versa. In fact, in view of (1.6), q only depends on $\phi_l, \phi_{l+1}, \cdots, \phi_{l+2n-1}$.

The Padé table now occupies the right half-plane $\{(m,n);\ m \in \mathbb{Z},\ n \in \mathbb{N}\}$, and there are three possible exceptions from the finite square blocks of the modified Corollary 1.6: If $f(z) = O_+(z^k)$ for some $k \in \mathbb{Z}$, then $r_{m,n} = 0$ in the quadrant $\{(m,n);\ m < k,\ n \in \mathbb{N}\}$. If $f \in \tilde{\mathcal{R}}_{m,n}$ for some (m,n) and the exact type of f is (μ,ν) with $\mu \neq -\infty$, then $r_{m,n} = f$ in the quadrant $\{(m,n);\ m \geq \mu,\ n \geq \nu\}$; however, if $\mu = \infty$, then $r_{m,n} = f$ in the half-plane $\{(m,n);\ m \in \mathbb{Z},\ n \geq \nu\}$. Of course, the Block Structure Theorem also requires some modifications concerning these exceptional cases. However, in our application here we could always assume that $\phi_i = 0$ for $i < l$, since the Padé approximant r^+ of f^+ is what really matters. Hence, we essentially have the situation $f \in \mathcal{P}$ discussed before. But for the future discussion of the backward qd algorithm it is necessary to introduce Padé approximants for all $(m,n) \in \mathbb{Z} \times \mathbb{N}$.

Let us now return to the formal orthogonal polynomials defined by (1.1). We now allow that $l < 0$ and, using the coefficients ϕ_k from (1.1), we set in accordance with (1.21)

$$f(z) := \sum_{k=-\infty}^{\infty} \phi_k z^k . \tag{1.25}$$

For fixed $l \in \mathbb{Z}$ an nth FOP1 $P_{l;n}$, which is defined as a monic polynomial of degree n satisfying (1.2), is now linked to the $(l+n-1,n)$ Padé approximant of $f \in \mathcal{L}$ in a way analogous to the special case $f \in \mathcal{P}$, $l > 0$ treated before. The formulas (1.4), (1.6), (1.8a), and (1.11) remain valid, while in (1.5), (1.8b), and (1.12) the lower bound 0 for i has to be replaced by $-\infty$.

Since $P_{l;n}$ $(n \geq 1)$ is supposed to have exact degree n, we need in view of (1.11) an $(n-1,n)$ Padé form (p^+,q) of f^+ with $q(0) \neq 0$ or, equivalently, an $(l+n-1,n)$ Padé form (p,q) of f with $q(0) \neq 0$. Hence, $P_{l;n}$ exists if and only if $\sigma = 0$ in Theorem 1.2 (applied either to f^+ (with $m := n-1$) or to f (with $m := l+n-1$)), and there is a unique monic $P_{l;n}$ if and only if $\sigma = \delta = 0$. Also note that δ equals the amount by which n surpasses the next smaller index for which $P_{l;n}$ is unique, i.e., regular. Summarizing, we get the following theorem.

THEOREM 1.8. *Given $f \in \mathcal{L}$, $l \in \mathbb{Z}$, and $n \in \mathbb{N}^+$, there exists a true nth* FOP1 *$P_{l;n}$ if and only if the index pair $(m,n) := (l+n-1,n)$ lies either in an infinite nonzero block or on or above the antidiagonal of a finite square block or a zero block of the Padé table of f, i.e., if and only if f has a true (m,n) Padé interpolant. $P_{l;n}$ then has the general form*

$$P_{l;n}(z) = \hat{P}_{l;n}(z)W(z) = z^n \hat{q}_{m,n}(z^{-1})w(z^{-1}), \tag{1.26}$$

where $\hat{P}_{l;n}(z) = z^{n-\delta}\hat{q}_{m,n}(z^{-1})$ is monic of degree $n-\delta$, $\hat{q}_{m,n}$ is the normalized denominator of the (m,n) Padé approximant, and $W(z) = z^\delta w(z^{-1})$ is an arbitrary monic polynomial of exact degree δ. Moreover, $\hat{P}_{l;n} = P_{l;n_j}$, where n_j denotes the largest integer less than or equal to n for which a regular FOP1 *for Φ_l exists. (Hence, $\delta = n - n_j$.)*

$P_{l;n}$ is regular (i.e., unique) if and only if the pair (m,n) lies on the upper or the left border of a finite or infinite square block of the Padé table. If $P_{l;n}$ is regular, then $W(z) \equiv 1$ in (1.26).

As antidiagonal of the zero block we understand the line $\{(m,n) \in \mathbb{Z}\times\mathbb{N};\ m+n = k-1\}$ if $f(z) \equiv O_+(z^k)$. If $r_{m,n} = 0$, then $\hat{P}_{l;n}(z) \equiv P_{l;0}(z) \equiv 1$ and $\partial W = \delta = n$ in (1.26). According to this theorem, for fixed l, the true FOP1s $P_{l;n}$, as far as they exist, are essentially the denominators of a diagonal sequence $\{r_{l+n-1,n}\}_{n=0}^{\infty}$ of Padé approximants, and the existence and uniqueness of a $P_{l;n}$ depends only on how this diagonal intersects with the blocks of the table (cf. Fig. 1).

From Theorem 1.8 it also follows that the determinant of the $n\times n$ moment matrix $M_{l;n}$ defined in (1.7) is nonzero if and only if the index pair $(m,n) := (l+n-1,n)$ lies on the first row or the first column of a finite or infinite square block of the Padé table. Hence, the zero pattern of the so-called *c-table*, whose (m,n) entry is this determinant, reflects the block structure of the Padé table [12, Thm. 3.2].

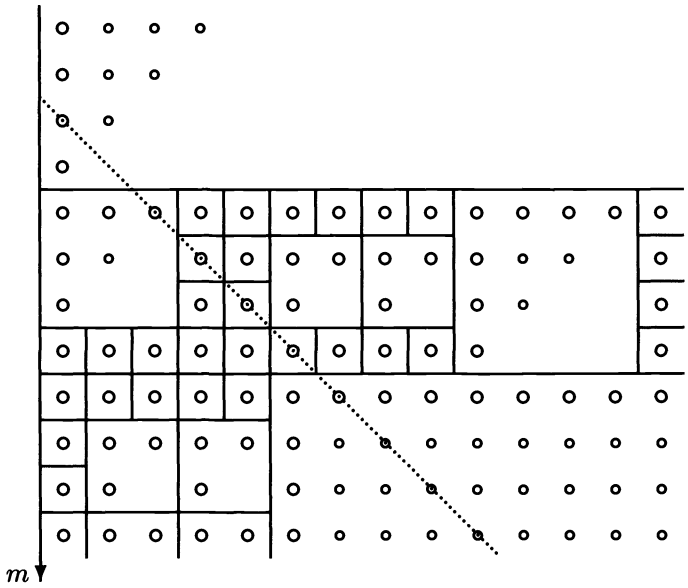


FIG. 1. *The location of regular (large circles) and singular (small circles) true* FOP1*s* $P_{l;n}$ *in the Padé table of* f. *At the top of the table there is a zero block, on the right side at the bottom is an infinite square block. The entries with deficient* FOP1*s are left blank. One diagonal is dotted.*

Theorem 1.8 suggests that we extend the definition of FOP1s in the following way to those values of l and m where no true FOP1 exists.

DEFINITION. If for some n there exists no true FOP1 with respect to Φ_l, and if n_j is the largest integer less than or equal to n for which a regular FOP1 exists, then any polynomial $P_{l;n}(z) := W(z)P_{l;n_j}(z)$, with $W \in \mathcal{P}_{n-n_j}$ monic, is called an *nth normalized deficient* FOP1 with respect to Φ_l.

Draux [7] calls these polynomials *quasi-orthogonal*. Our choice of the attribute deficient is based on their relationship to deficient Padé approximants stated next. These deficient Padé approximants are, on the other hand, special cases of deficient rational interpolants [16]. In view of Theorem 1.2 the following addition to Theorem 1.8 then holds.

THEOREM 1.9. *Given* $f \in \mathcal{L}$, $l \in \mathbb{Z}$, *and* $n \in \mathbb{N}^+$, *a* FOP1 $P_n = P_{l;n}$ *is deficient if and only if the index pair* $(m,n) := (l+n-1,n)$ *lies in the Padé table of* f *below the antidiagonal of a finite square block or a zero block, i.e., if and only if* f *has a deficient* (m,n) *Padé interpolant.* $P_{l;n}$ *then has the general form* (1.26), *where again*

$\hat{P}_{l;n}(z) = P_{l;n_j}(z) = z^{n-\delta}\hat{q}_{m,n}(z^{-1})$ is monic of degree $n-\delta$, $\hat{q}_{m,n}$ is the normalized denominator of the (m,n) Padé approximant, and $W(z) = z^{\delta}w(z^{-1})$ is an arbitrary monic polynomial of exact degree δ.

However, if P_n is deficient, $q(z) := q_{l+n-1,n}(z) := z^n P_n(z^{-1})$ is not the second member of an (m,n) Padé form of f. To obtain such a second member we would have to restrict the degree of W to $\delta - \sigma$ in (1.26), *where*

$$\sigma = n - \left\lfloor \frac{n_{j+1} + n_j - 1}{2} \right\rfloor > 0. \tag{1.27}$$

Here n_j denotes the largest integer less than or equal to n for which a regular FOP1 *exists, n_{j+1} is the smallest integer greater than n for which a regular* FOP1 *exists, and $\lfloor x \rfloor$ denotes the largest integer not greater than x. (Again, $\delta = n - n_j$.)*

For fixed l we now have a complete set of FOP1s, and the following is a useful reformulation of Theorems 1.8 and 1.9.

THEOREM 1.10. *For fixed l, let $0 = n_0 < n_1 < n_2 < \cdots$ be the indices for which a regular (and, hence, unique)* FOP1 *$P_n := P_{l;n}$ exists. (If f^+ is rational, then there are finitely many regular* FOP1*s P_{n_j}, $j = 0, \cdots, J$, and we set $n_{J+1} := \infty$. Otherwise, we set $J := \infty$.) Then, for all $n \in \mathbb{N}$, the* FOP1*s $P_n := P_{l;n}$ have the form*

$$P_n(z) := W_{n-n_j}(z)P_{n_j}(z) \quad \text{if } n_j \le n < n_{j+1} \tag{1.28}$$

with W_{n-n_j} an arbitrary monic polynomial of exact degree $\delta = n - n_j$. Further, if we let

$$h_j := n_{j+1} - n_j\,, \qquad h'_j := \left\lfloor \frac{h_j - 1}{2} \right\rfloor, \tag{1.29}$$

there holds for $n_j \le n < n_{j+1}$, $j \in \mathbb{N}$ (cf. Fig. 2):

(i) *P_n is a true* FOP1 *if and only if $n_j \le n \le n_j + h'_j$;*
(ii) *P_n is a regular* FOP1 *if and only if $n = n_j$;*
(iii) *P_n is a singular* FOP1 *if and only if $n_j < n \le n_j + h'_j$;*
(iv) *P_n is a deficient* FOP1 *if and only if $n_j + h'_j < n < n_{j+1}$.*

Formal orthogonal polynomials of the *second kind* do not play as important a role in numerical analysis as the FOP1s. But they ought to be mentioned for completeness and for symmetry reasons. Moreover, most results for them are obtained nearly for free.[5]

DEFINITION. The *formal orthogonal polynomial of the second kind* (FOP2) $Q_{l;n}$ associated with an nth FOP1 $P_{l;n}$ is

$$Q_{l;n}(\zeta) := \Phi_l\left(\frac{P_{l;n}(\zeta) - P_{l;n}(z)}{\zeta - z}\right), \tag{1.30}$$

where Φ_l, as before, acts on polynomials in z.

LEMMA 1.11. *$Q_{l;n}$ $(n \ge 1)$ is a polynomial of degree at most $n-1$,*

$$Q_{l;n}(\zeta) = \sum_{i=0}^{n-1} \chi_{l;i,n}\zeta^i, \tag{1.31}$$

[5] Those readers who are only interested in the Lanczos process may find it sufficient to know that the FOP2s are related to the numerators of the Padé approximants, and may want to skip to the last paragraph of this section.

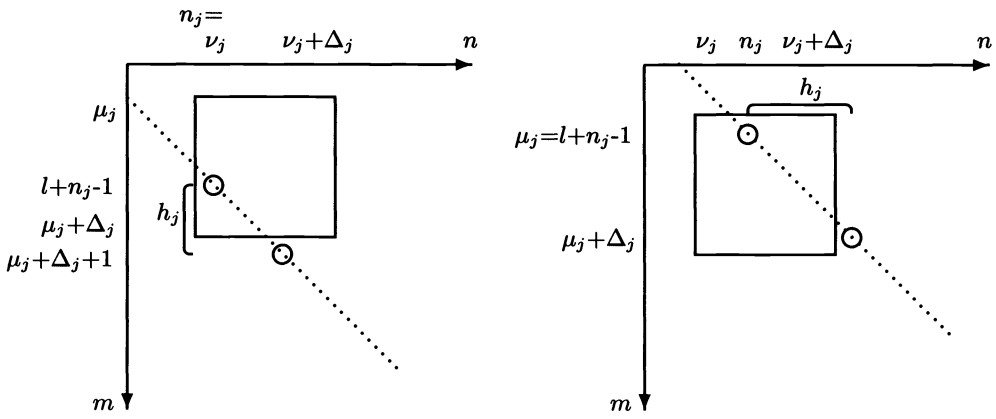


FIG. 2. *The indices of the entries on the lth diagonal of the Padé table, where this diagonal crosses the block containing the* $(l+n_j-1, n_j)$ *Padé approximant, whose denominator is* $z^{n_j}P_{n_j}(1/z)$.

whose coefficients $\chi_{l;i,n}$ *are obtained from those of* $P_{l;n}$ *according to*

$$\chi_{l;i,n} = \sum_{j=i+1}^{n} \phi_{l+j-i-1}\pi_{l;j,n}\,, \qquad i = 0, \cdots, n-1. \tag{1.32}$$

$Q_{l;n}$ *has exact degree* $n-1$ *if and only if* $\phi_l \neq 0$. *Moreover,* $Q_{l;0} = 0$.

Proof. [6, p. 10]. Since $(\zeta^i - z^i)/(\zeta - z) = \zeta^{i-1} + \zeta^{i-2}z + \cdots + \zeta z^{i-2} + z^{i-1}$, we get

$$\begin{aligned}\frac{P_{l;n}(\zeta) - P_{l;n}(z)}{\zeta - z} &= \pi_{l;n,n}(\zeta^{n-1} + \zeta^{n-2}z + \cdots + \zeta z^{n-2} + z^{n-1}) \\ &\quad + \pi_{l;n-1,n}(\zeta^{n-2} + \zeta^{n-3}z + \cdots + \zeta z^{n-3} + z^{n-2}) \\ &\quad + \cdots + \pi_{l;2,n}(\zeta + z) + \pi_{l;1,n}\,.\end{aligned}$$

Applying Φ_l on both sides yields (1.32). □

In analogy to (1.11) let us set

$$p^+_{l+n-1,n}(z) := z^{n-1}Q_{l;n}(z^{-1}) = \sum_{i=0}^{n-1} \chi_{l;i,n} z^{n-1-i}\,. \tag{1.33}$$

If we assume that $P_{l;n}$ is a true FOP1, the comparison with (1.8b) shows that $p^+_{l+n-1,n} = p_{l+n-1,n}$ if $l = 0$ and $\phi_k = 0$ for $k < l$. More generally, $z^l p^+_{l+n-1,n}(z) = p_{l+n-1,n}(z)$ if $\phi_k = 0$ for $k < l$. Setting $m := l + n - 1$ and recalling the definition of $r_{m,n}$ for $f \in \mathcal{L}$ (cf. (1.21a, b) and (1.22)), we notice that $p^+_{m,n}$ coincides with p^+ there. Hence, $p^+_{m,n}/q_{m,n}$ is a representation of r^+, namely the one normalized by $q_{m,n}(0) = 1$. Consequently, in view of (1.13), $(p^+_{m,n}, q_{m,n})$ is a Padé form of r^+. (Note that $\sigma = 0$ since $P_{l;n}$ is a true FOP1.) Hence, we get the following theorem.

THEOREM 1.12. *Let* $P_{l;n}$ *be any normalized true* FOP1, *and let* $Q_{l;n}$ *be the associated* FOP2. *Then, with* $m := l+n-1$, *the pair* $(p^+_{m,n}, q_{m,n})$ *defined by* (1.11) *and* (1.33) *is an* $(n-1, n)$ *Padé form of* f^+ *with the property that* $q_{m,n}(0) = 1$. *Conversely, any such normalized Padé form of* f^+ *yields an nth* FOP1 *and an associated* FOP2.

Moreover, any such Padé form yields the (m,n) *Padé approximant of* f *in the form* (1.22) *with*

$$r^+(z) = \frac{p^+_{m,n}(z)}{q_{m,n}(z)} = \frac{z^{-1}Q_{l;n}(z^{-1})}{P_{l;n}(z^{-1})}. \tag{1.34}$$

In view of Theorems 1.2 and 1.8 we can further conclude that $Q_{l;n} = \hat{Q}_{l;n}W$ is, in the sense of definition (1.30), associated with $P_{l;n} = \hat{P}_{l;n}W$ if the latter is a true FOP1. This follows here in an indirect way. However, in §2 we will give a direct proof based on the definition (1.30) and the orthogonality property of the FOP1s alone, and we will see that this relation extends to deficient FOP1s.

COROLLARY 1.13. *The* FOP2 $Q_{l;n}$ *associated with the* FOP1 $P_{l;n}$ *of* (1.28) *is*

$$Q_{l;n}(z) = \hat{Q}_{l;n}(z)W(z), \tag{1.35}$$

where $\hat{Q}_{l;n}(z) := Q_{l,n_j}$ *is associated with* $\hat{P}_{l;n} = P_{l;n_j}$.

It then follows in turn that Theorem 1.12 is also valid for deficient FOP1s if we replace $(p^+_{m,n}, q_{m,n})$ by $(z^\sigma p^+_{m,n}, z^\sigma q_{m,n})$ in the statement of the theorem.

A natural question is whether the FOP2s are also in a certain sense orthogonal, i.e., whether there holds a relation analogous to (1.2) for $Q_{l;n}$. The answer, which is given in Theorem 2.3 below, is easy to find once the following lemma is established.

LEMMA 1.14. *Assume* $f \in \mathcal{L}$ *has only at most finitely many terms with negative index, i.e.,* $f(z) = z^k\tilde{f}(z)$ *holds for some* $k \in \mathbb{Z}$, *where* $\tilde{f} \in \mathcal{P}$, $\tilde{f}(0) \neq 0$. *Let* $g = 1/\tilde{f}$ *be the reciprocal series of* $\tilde{f}$. *Then* (p,q) *is an* (m,n) *Padé form of* f *if and only if* $(q, z^{-k}p)$ *is an* $(n, m-k)$ *Padé form of* $g \in \mathcal{P}$.

Proof. Equation (1.24) for $(p,q) \in \tilde{\mathcal{P}}_m \times \mathcal{P}_n$ is equivalent to $g(z)z^{-k}p(z) - q(z) = O_+(z^{m-k+n+1})$ for $(z^{-k}p, q) \in \mathcal{P}_{m-k} \times \mathcal{P}_n$, since $f(z) \equiv O_+(z^k)$ implies $p(z) \equiv O_+(z^k)$. □

Since according to Theorem 1.12 (and its extension to deficient FOP1s) a FOP2 $Q_{l;n}$ is essentially the numerator of the $(n-1,n)$ Padé approximant of f^+, we can apply the lemma to $Q_{l;n}$ in order to get the following theorem.

THEOREM 1.15. *Assume that* $P_{l;n}$ $(n \geq 1)$ *is a* n*th* FOP1 *with respect to the functional* Φ_l *defined by* (1.1), *and that either* $\phi_l \neq 0$ *and* $k := l$ *or* $\phi_l = \phi_{l+1} = \cdots = \phi_{k-1} = 0$, $\phi_k \neq 0$. *Let*

$$g_k(z) = \psi_0^{(k)} + \psi_1^{(k)}z + \psi_2^{(k)}z^2 + \cdots \tag{1.36}$$

be the formal power series, which is reciprocal to $\phi_k + \phi_{k+1}z + \phi_{k+2}z^2 + \cdots$. *Then,* $Q_{l;n}$ *has exact degree* $n-k+l-1$ *and leading coefficient* $\phi_k = 1/\psi_0^{(k)}$, *and*

$$(z^{\sigma+n}P_{l;n}(z^{-1}),\ z^{\sigma+n-k+l-1}Q_{l;n}(z^{-1})), \tag{1.37}$$

with σ *given by* (1.27), *is an* $(n, n-k+l-1)$ *Padé form of* g_k. *Consequently, with respect to the linear functional* $\Psi_k : \mathcal{P} \to \mathbb{C}$ *defined by*

$$\Psi_k(z^i) = \psi_i^{(k)} \qquad (i \in \mathbb{N}), \tag{1.38}$$

$\phi_k^{-1}Q_{l;n}$ *is a normalized* $(n-k+l-1)$*th* FOP1.

The first pair of true FOP*s with index* $n > 0$ *is at* $n = n_1 := k-l+1$, *and there holds* $Q_{l;n_1}(z) \equiv \phi_k$. *In particular,* $n_1 = 1$ *if and only if* $\phi_l \neq 0$.

Moreover, for $n \geq n_1$, $Q_{l;n}$ is regular, singular, or deficient if and only if $P_{l;n}$ has the respective property.

Proof. What remains to be said is that, under the above assumptions, $f^+(z) = \phi_l + \phi_{l+1}z + \cdots \equiv O_+(z^{k-l})$, so that the numerator of the Padé fraction (1.34) has at 0 a zero of exact order $k-l$, which means that $Q_{l;n}$ has exact degree $n-k+l-1$. Consequently, in (1.37), which by Lemma 1.14 is an $(n, n-k+l-1)$ Padé form of g_k, the factor z^σ corresponds exactly to the one in (1.13). Hence, by Theorem 1.8, the polynomial $Q_{l;n}$, when renormalized to be monic, is a true FOP1 if and only if $\sigma = 0$. By inserting $z = 0$ in (1.36) and (1.37), it is seen that $Q_{l;n}$ has leading coefficient $\phi_k = 1/\psi_0^{(k)}$. Moreover, $P_{l;n}$ is unique (i.e., regular) if and only if $Q_{l;n}$ is unique. Finally, the statement about the first pair of true FOPs follows from the fact that $r^+ = 0 = 0/1$ if $l \leq m < k$, i.e., if $0 < n = m-l+1 < k-l+1$. □

Since the regular FOP1s $P_{l;n_j}$ (of respective exact degree n_j) are in a one-to-one correspondence with the regular FOP2s $Q_{l;n_j}$ (of respective exact degree $n_j - k + l - 1$), the gaps $h_j = n_{j+1} - n_j$ between the degrees are the same for both sequences; there is only a constant difference $k-l+1$ in the degrees. Note that $Q_{l;n}(z) \equiv 0$ for $n_0 = 1 \leq n < n_1$.

Later we will make the substitution $\zeta := z^{-1}$ and consider Padé approximants at $\zeta = \infty$ of the function

$$F(\zeta) := \zeta^{-1} f(\zeta^{-1}) = \sum_{k=0}^{\infty} \phi_k \zeta^{-k-1}. \tag{1.39}$$

We will call $r(1/\zeta)$ an (m,n) *Padé approximant of $F(\zeta)$ at ∞* if $r(z)$ is an (m,n) Padé approximant of $F(z^{-1}) = zf(z)$, at 0. Hence, instead of $f(z)$ we consider essentially $zf(z)$ whose Padé table is obtained by shifting the one of f down by one row, thus introducing a zero row at the top of the classical Padé table occupying the quadrant. While the FOPs generated by the functional Φ_0 yield the Padé approximants on the superdiagonal $\{(n-1,n)\,;\, n \in \mathbb{N}\}$ of the Padé table of f, they correspond to the Padé approximants on the main diagonal of the Padé table of $z(f(z))$, which we may also consider as the Padé table of F at ∞. Note, however, that $r(1/\zeta)$ (as a function of ζ) does not generally have type (m,n) if $m \neq n$. In §4 we will define the functional Φ_0 such that the corresponding FOP1s are the polynomials that are implicitly generated by the unsymmetric Lanczos process.

2. Orthogonality and recurrence formulas of formal orthogonal polynomials. Given the functional Φ_l of (1.1) for some fixed l, in §1 we have defined normalized formal orthogonal polynomials $P_n = P_{l;n}$ for each $n \in \mathbb{N}$. As in Theorem 1.10, we let $0 = n_0 < n_1 < n_2 < \cdots < n_J$ (where J can be finite or infinite) be the indices for which P_n is regular, i.e., unique. Then all the other P_n have the form (1.28) with W_{n-n_j} monic of degree $n - n_j$. Now, for simplicity, we choose a fixed sequence $\{W_m\}_{m=0}^{\infty}$ of monic polynomials $W_m \in \mathcal{P}_m$ to be used in the sequel for this purpose. Then (1.28) specifies one FOP1 P_n for each n. In this section we discuss the orthogonality properties of the sequence $\{P_n\}$ and deduce from them a recurrence formula, which also holds with different initial values for the sequence $\{Q_n\} := \{Q_{l;n}\}$ of associated FOP2s.

By definition, a true FOP1 is orthogonal to all polynomials of lower degree. It may, however, be orthogonal to itself. The following theorem shows that this happens whenever a FOP1 corresponds to a Padé approximant which lies in the Padé table of f (given in (1.25)) above the antidiagonal of its block. On the other hand, deficient

FOP1s, which lie below the antidiagonal, are not orthogonal to all polynomials of lower degree.

THEOREM 2.1 (orthogonality of FOP1s). *For a* FOP1 P_n *with* $n_j \le n < n_{j+1}$, *the following holds:*

(i) *If* $j < J$ *and either* $j > 0$ *or* $f(z) \neq O_+(z^{l+1})$, *i.e., if the* $(l+n_j-1, n_j)$ *entry of the Padé table of* f *lies on the upper or the left border of a finite square block, then*

$$\Phi_l(pP_n) = 0 \quad (\forall p \in \mathcal{P}_{\hat{n}-1}), \tag{2.1a}$$

$$\Phi_l(z^{\hat{n}}P_n) \neq 0, \tag{2.1b}$$

where

$$\hat{n} := 2n_j - n + h_j - 1 = n_j + n_{j+1} - n - 1. \tag{2.1c}$$

In particular, if $n = n_j$, *then*

$$\hat{n} = n_j + h_j - 1 = n_{j+1} - 1, \tag{2.2}$$

and therefore h_j *can be determined as*

$$h_j = \min\{k \in \mathbb{N}^+;\ \Phi_l(z^{n_j+k-1}P_{n_j}) \neq 0\} = \min\{k \in \mathbb{N}^+;\ \Phi_l(W_{k-1}(P_{n_j})^2) \neq 0\}. \tag{2.3}$$

(ii) *If* $j = J < \infty$, *i.e., if the* $(l+n_j-1, n_j)$ *entry is not the zero function and lies on the upper or left border of an infinite square block of the Padé table of* f, *then* $\Phi_l(pP_n) = 0$ *for any polynomial* p *and any* $n \ge n_j$.

(iii) *If the* $(l+n_j-1, n_j)$ *entry is the zero function, so that necessarily* $j = n_j = 0$, $P_{n_j}(z) \equiv 1$, *and* $f(z) \equiv O_+(z^k)$ *for* $k := n_1 + l > l$, *then the relations* (2.1a) *and* (2.1b) *hold with* $\hat{n} := n_1 - n - 1$, *which is in accordance with* (2.1c).

Remark. Since $n + \hat{n} = 2n_j + h_j - 1$ the degrees n and $\hat{n}$ belong to entries on the $(l+n-1, n)$ diagonal of the table that lie symmetrically about the antidiagonal of their block, which must be a finite or a zero block (cf. Fig. 2).

Proof. (i) Assume that (2.1) holds for $n = n_j$ and some $\hat{n}$, i.e., that for $P_{l;n} = P_{n_j}$ the orthogonality relation (1.2) holds up to the degree $\hat{n} - 1$, but not for $p(z) = z^{\hat{n}}$. Then (1.6) holds exactly up to $i = l + n_j + \hat{n} - 1$, and in (1.8a) the last term becomes $O_+(z^{l+n_j+\hat{n}})$. In view of (1.11) there holds therefore for the linearized error:

$$f(z)q_{l+n_j-1,n_j}(z) - p_{l+n_j-1,n_j}(z) \equiv O_+(z^{l+n_j+\hat{n}}). \tag{2.4}$$

Conversely, this "identity" obviously implies (2.1).

Because the $(l+n_j-1, n_j)$ entry lies in the first row or column of its block, there holds $\sigma = 0$ in (1.13), and when (2.4) is divided by q_{l+n_j-1,n_j}, the order of (2.4) is not changed. The comparison with (1.16) then yields

$$l + n_j + \hat{n} = \mu_j + \nu_j + \Delta_j + 1.$$

We claim that this is equal to $l + 2n_j + h_j - 1$, so that, under the assumption $n = n_j$, the formula coincides with (2.1c). Two cases must be distinguished (cf. Fig. 2): If the $(l + n_j - 1, n_j)$ entry lies in the first column of its block, then $n_j = \nu_j$ and $l + n_j - 1 + h_j = \mu_j + \Delta_j + 1$, while, if it lies in the first row, then $l + n_j - 1 = \mu_j$ and $n_j + h_j = \nu_j + \Delta_j + 1$. In both cases, adding the two respective equations yields the claimed identity.

Finally, if n increases from n_j to $n_{j+1}-1$ and P_n is defined by (1.28), it is clear from the linearity of Φ_l that $\hat{n}$, as implicitly defined by (2.1a) and (2.1b), decreased at the same rate. Hence, the formula (2.1c) for $\hat{n}$ is valid for all n.

(ii) If the $(l+n_j-1, n_j)$ entry lies on the border of an infinite block, then the left-hand side of (2.4) is identically zero, and by the arguments used at the beginning of the proof of part (i), (2.1a) holds for arbitrary $\hat{n}$ if $n = n_j$. If $n > n_j$, we split p into W_{n-n_j} times $\tilde{p}$, and obtain $\Phi_l(\tilde{p}P_n) = 0$ for arbitrary $\tilde{p}$.

(iii) In the case of a zero entry we must clearly have $j = n_j = 0$ and $n_1 = k-l+1$, and there holds for $n = n_0 = 0$

$$f(z)z^0 P_0(1/z) = f(z) \equiv O_+(z^k) \equiv O_+(z^{n_1+l-1}),$$

which is equivalent to (1.6) holding up to $i = n_1 + l - 2$ exactly and to (1.2) holding up to degree $n_1 - 2$. □

The following result, which can be found in [7, Propriété 1.17, p. 49] and [14, Thm. 2], is now an easy consequence of Theorem 2.1.

COROLLARY 2.2. *Under the assumptions of part* (i) *of Theorem* 2.1 *and the convention* $n_i \le n' < n_{i+1}$, $n_j \le n < n_{j+1}$ *holds*

$$\Phi_l(P_{n'}P_n) = 0 \quad \begin{array}{l} \text{if } i \ne j \\ \text{or } i = j \quad \text{and} \quad n' + n < n_j + n_{j+1} - 1, \end{array} \tag{2.5}$$

and, for some nonzero δ_j *independent of* $n - n_j$ *and* $n' - n_j$,

$$\Phi_l(P_{n'}P_n) =: \delta_j \ne 0 \quad \text{if} \quad i = j \quad \text{and} \quad n' + n = n_j + n_{j+1} - 1. \tag{2.6}$$

If $i = j$ *and* $n' + n \ge n_j + n_{j+1}$, *no statement can be made in general about the vanishing of* $\Phi_l(P_{n'}P_n)$.

Proof. Condition (2.5) follows directly from (2.1a) and (2.1c): If $i \ne j$, then either $n' < n_j \le n$ or $n < n_j \le n'$; if $i = j$, then $n' < \hat{n}$. Similarly, the nonvanishing of $\Phi_l(P_{n'}P_n)$ in (2.6) follows from (2.1b) and (2.1c). The fact that the value only depends on j is a consequence of all P_n being monic, (2.5), and the linearity of Φ_l:

$$\delta_j = \Phi_l(z^{n_{j+1}-1}P_{n_j}) = \Phi_l(z^{h_j-1}P_{n_j}^2). \tag{2.7}$$

□

In §3 we will prove that by choosing the polynomials W_m in (1.28) appropriately (namely, in a preliminary unknown way dependent on j), we can make $\Phi_l(P_{n'}, P_n) = 0$ when $i = j$ and $n' + n \ge n_j + n_{j+1}$.

Theorem 2.1 allows us to give a complete, direct proof of Corollary 1.13.

Proof of Corollary 1.13. By definition, Q_{n_j} is associated with P_{n_j}. If $n_j < n < n_{j+1}$, we obtain from (1.28) and (1.30)

$$\begin{aligned} Q_n(\zeta) &:= \Phi_l\left(\frac{W_{n-n_j}(\zeta)P_{n_j}(\zeta) - W_{n-n_j}(z)P_{n_j}(z)}{\zeta - z}\right) \\ &= W_{n-n_j}(\zeta)\Phi_l\left(\frac{P_{n_j}(\zeta) - P_{n_j}(z)}{\zeta - z}\right) + \Phi_l\left(\frac{W_{n-n_j}(\zeta) - W_{n-n_j}(z)}{\zeta - z}P_{n_j}(z)\right) \\ &= W_{n-n_j}(\zeta)Q_{n_j}(\zeta). \end{aligned}$$

In the last step we have used the fact that $p(z) := [W_{n-n_j}(\zeta) - W_{n-n_j}(z)]/(\zeta - z)$ is a polynomial of degree $n - n_j - 1 < n_{j+1} - n_j - 1 \le n_{j+1} - 1$, so that $\Phi_l(pP_{n_j}) = 0$ according to Theorem 2.1. □

In view of Theorem 1.15 the orthogonality results of Theorem 2.1 and Corollary 2.2 have an analogue for the associated FOP2s.

THEOREM 2.3 (orthogonality of FOP2s). *For $j = 1, \cdots, J$ let Q_{n_j} be the unnormalized* FOP2 *associated in the sense of* (1.30) *with the regular* FOP1 P_{n_j}. *For $n_j < n < n_{j+1}$ let P_n and Q_n be defined by* (1.28) *and* (1.35), *respectively. Define k and Ψ_k as in Theorem* 1.15. *Then for $j = 1, \cdots, J$ there holds in case* (i) *of Theorem* 2.1,

$$\Psi_k(pQ_n) = 0 \quad (\forall p \in \mathcal{P}_{\hat{n}-k+l-2}), \tag{2.8a}$$

$$\Psi_k(z^{\hat{n}-k+l-1}Q_n) \neq 0, \tag{2.8b}$$

where $\hat{n}$ is again defined by (2.1c), *while, in case* (ii) *of that theorem, $\Psi_k(pQ_n) = 0$ for any polynomial p and any $n \geq n_j$. (In both cases Q_n has exact degree $n-k+l-1$.)*

COROLLARY 2.4. *Except when $i = j = 0$ the statement of Corollary* 2.2 *remains valid if P_n and $P_{n'}$ are replaced by Q_n and $Q_{n'}$. (The constants δ_j have new values δ'_j obtained by replacing P_{n_j} by Q_{n_j} and Φ_l by Ψ_k in* (2.7).)

There are some other immediate conclusions from Corollaries 2.2 and 2.4 that may be worth formulating.

COROLLARY 2.5. (i) *A* FOP1 $P_n = P_{l;n}$ *is orthogonal to all polynomials of lower degree but not orthogonal to itself if and only if the corresponding $(l+n_j-1, n_j)$ entry lies in the Padé table of f on the antidiagonal of a finite or a zero block.*

(ii) *P_n is orthogonal to all polynomials of lower or equal degree if and only if the $(l+n_j-1, n_j)$ entry lies above the antidiagonal of a finite or a zero block or in an infinite nonzero block.*

(iii) *P_n is not orthogonal to all polynomials of lower degree if and only if the $(l+n_j-1, n_j)$ entry lies below the antidiagonal of a finite or a zero block.*

(iv) *Except when $n < n_1$, i.e., when the $(l+n_j-1, n_j)$ entry lies in a leftmost block, statements* (i)–(iii) *also hold for the* FOP2*s* Q_n *and formal orthogonality with respect to Ψ_k.*

COROLLARY 2.6. *For every $n \in \mathbb{N}$ there is a polynomial of degree at most n which is with respect to Φ_l orthogonal to all polynomials of degree less than n.*

At first sight, Corollary 2.6, which is due to Struble [44], seems to contradict the existence of deficient FOP1s. However, the point is that the degree of the polynomial may be less than n, which is not allowed for an nth FOP1. In fact, if we are in case (iii) of Corollary 2.5, then $\hat{n} < n$, and $P_{\hat{n}}$ is a polynomial fulfilling the claim of Corollary 2.6.

Now let us turn to the *recurrence formulas*. By definition (1.28) the polynomials P_n with $n_j < n < n_{j+1}$ are easily generated from the regular FOP1 P_{n_j}. Moreover, they can be computed recursively. For example, if $W_m(z) = z^m$, then

$$P_{n+1}(z) = zP_n(z), \qquad n_j \leq n \leq n_{j+1} - 2, \tag{2.9}$$

while, if the polynomials W_m are chosen to satisfy a three-term recurrence,

$$W_{m+1}(z) = (z - \alpha_m^W)\ W_m(z) - \beta_m^W\ W_{m-1}(z) \qquad (m \in \mathbb{N}) \tag{2.10}$$

(with $W_0(z) :\equiv 1$, $W_{-1}(z) :\equiv 0$, $\beta_0^W := 0$), then clearly

$$P_{n+1}(z) = (z - \alpha_{n-n_j}^W)\, P_n(z) - \beta_{n-n_j}^W\, P_{n-1}(z), \qquad n_j \leq n \leq n_{j+1} - 2. \tag{2.11}$$

There remains the question of whether the regular FOP1s P_{n_j} satisfy a similar three-term recurrence formula. A brief (somewhat sketchy) direct proof of such a formula, namely recurrence (2.17) below, was first given by Struble [44].[6] As we will see in a later part, its existence also follows readily from the P-fraction representation of f, which is due to Magnus [26], [27]. The result also appears in Nuttall and Singh [32, Lemma 3.5], Gragg and Lindquist [14, Thm. 2], and, in a less appealing formulation, in Draux [7, pp. 54–56]. Once the orthogonality relations (2.1a–c) have been established, the derivation of this three-term recurrence for $\{P_{n_j}\}_{j=0}^{J}$ $(J \leq \infty)$ is, in fact, quite easy.

First, these polynomials, since they are monic of respective degree n_j, clearly satisfy a recurrence

$$P_{n_{j+1}}(z) = (W_{h_j}(z) - t_{j,j}(z))P_{n_j}(z) - t_{j-1,j}(z)P_{n_{j-1}}(z) - \cdots - t_{0,j}(z)P_0(z)$$

with $t_{i,j} \in \mathcal{P}_{h_i-1}$, $i = 0, \cdots, j$. Let us multiply this relation by z^{n_i+k} $(k = 0, \cdots, h_i - 1;\ i = 0, \cdots, j)$ and then apply the functional Φ_l. Since $n_i + k \leq n_{i+1} - 1$, the left-hand side becomes 0, and we obtain for each pair (k, i) a linear equation for the polynomials $t_{i,j}$ (i.e., for their coefficients):

$$\text{(2.12)}\quad \sum_{s=0}^{j} \Phi_l(z^{n_i+k}\, t_{s,j}\, P_{n_s}) = \Phi_l(z^{n_i+k}\, W_{h_j}\, P_{n_j}) \quad (k = 0, \cdots, h_i - 1;\ \ i = 0, \cdots, j).$$

For the right-hand side there holds, in view of (2.1a–c) and (2.2),

$$\Phi_l(z^{n_i+k}\, W_{h_j}\, P_{n_j}) \begin{cases} = 0 & \text{if} \quad i < j-1, \\ = 0 & \text{if} \quad i = j-1 \quad \text{and} \quad k \leq h_{j-1} - 2, \\ \neq 0 & \text{if} \quad i = j-1 \quad \text{and} \quad k = h_{j-1} - 1, \end{cases}$$

and on the left-hand side we obtain likewise

$$\Phi_l(z^{n_i+k}\, t_{s,j}\, P_{n_s}) \begin{cases} = 0 & \text{if} \quad i < s, \\ = 0 & \text{if} \quad i = s \quad \text{and} \quad k + \partial t_{s,j} \leq h_s - 2, \\ \neq 0 & \text{if} \quad i = s \quad \text{and} \quad k + \partial t_{s,j} = h_s - 1. \end{cases}$$

Therefore, the system reduces actually to a coupled pair of systems, one homogeneous,

$$\text{(2.13)}\quad \sum_{s=0}^{i} \Phi_l(z^{n_i+k}\, t_{s,j}\, P_{n_s}) = 0 \quad \begin{array}{l}(k = 0, \cdots, h_i - 1;\ \ i = 0, \cdots, j-2 \\ \qquad \text{and } k = 0, \cdots, h_{j-1} - 2;\ \ i = j-1)\end{array}$$

consisting of $\sum_{i=0}^{j-1} h_i - 1 = n_j - 1$ equations for the j polynomials $\{t_{s,j}\}_{s=0}^{j-1}$ with total degree of freedom n_j, the other inhomogeneous,

$$\text{(2.14)}\quad \sum_{s=0}^{i} \Phi_l(z^{n_i+k} t_{s,j} P_{n_s}) = \Phi_l(z^{n_i+k} W_{h_j} P_{n_j}) \quad \begin{array}{l}(k = 0, \cdots, h_j - 1;\ \ i = j \\ \qquad \text{and } k = h_{j-1} - 1,\ \ i = j-1)\end{array}$$

consisting of $h_j + 1 = n_{j+1} - n_j + 1$ equations (giving a total of n_{j+1}) and containing additionally the polynomial $t_{j,j}$ with h_j free parameters (giving also a total of n_{j+1}).

The main point is now that the full system and its homogeneous part are triangular, and that therefore all but one of the unknowns appearing in the homogeneous

[6] The author is indebted to Walter Gautschi for this reference.

system are zero. In fact, first let $i = 0$, and assume that $t_{0,j} \neq 0$. Choose k such that $k + \partial t_{0,j} = h_0 - 1$. Then $\Phi_l(z^{n_0+k}\, t_{0,j}\, P_{n_0}) \neq 0$ according to (2.1a–c) and (2.2). But in (2.13) this contradicts the equation with index pair $(k, 0)$, hence, $t_{0,j} = 0$. Next make the induction assumption $t_{0,j} = t_{1,j} = \cdots = t_{i-1,j} = 0$ $(i \leq j - 1)$. Repeat the argument with k satisfying $k + \partial t_{i,j} = h_i - 1$ and with the (k, i) equation. Again it follows that $t_{i,j} = 0$, except that in the case $i = j - 1$ there is no equation for $k = h_{j-1} - 1$, so that $t_{j-1,j}$ could be a nonnull polynomial of degree 0, i.e., a constant. Conversely, for any such constant,

$$t_{0,j}(z) \equiv t_{1,j}(z) \equiv \cdots \equiv t_{j-2,j}(z) \equiv 0, \qquad t_{j-1,j}(z) \in \mathbb{C} \tag{2.15}$$

is a solution of the homogeneous system (2.13), and this must also be its general solution. Inserting this into (2.14) and replacing $t_{j-1,j}$ and $t_{j,j}$ by β_j (a constant) and a_j (a polynomial), respectively, reduces (2.14) finally to

$$\beta_j \Phi_l(z^{n_j-1} P_{n_{j-1}}) = \Phi_l(z^{n_j-1}\, W_{h_j}\, P_{n_j}), \tag{2.16a}$$

(2.16b)
$$\Phi_l(z^{n_j+k} a_j P_{n_j}) + \beta_j \Phi_l(z^{n_j+k} P_{n_{j-1}}) = \Phi_l(z^{n_j+k}\, W_{h_j}\, P_{n_j}), \qquad k = 0, \cdots, h_j - 1,$$

i.e., to an individual linear equation for the constant β_j and a system of h_j linear equations for the h_j coefficients of a_j. By (2.1a–c) and (2.2) the Φ_l-values in (2.16a) are not zero, hence β_j is uniquely determined and nonzero. If the polynomial a_j is written in ascending powers of z, the coefficient matrix in (2.16b) is in view of (2.1a–c) and (2.2) lower right triangular with a constant nonzero antidiagonal. Hence, the solution of (2.16b) is uniquely determined also. Summarizing, we get part (i) of the following theorem.

THEOREM 2.7 (recurrence formula). (i) *The regular* FOP1*s* P_{n_j}, $j = 0, \cdots, J$ $(\leq \infty)$, *satisfy a three-term recurrence*

$$P_{n_{j+1}}(z) = (W_{h_j}(z) - a_j(z)) P_{n_j}(z) - \beta_j P_{n_{j-1}}(z), \qquad j = 0, \cdots, J - 1, \tag{2.17}$$

with initial values $P_{n_{-1}}(z) := 0$, $P_{n_0}(z) := 1$, $\beta_0 := 0$, *where* $\{W_m\}_{m=0}^{\infty}$ *is an arbitrary prescribed sequence of monic polynomials of respective degree* m, $\{\beta_j\}_{j=1}^{J-1}$ *is a sequence of uniquely determined nonzero complex constants obtained by solving the single linear equation* (2.16a), *and* $\{a_j\}_{j=1}^{J-1}$ *is a sequence of complex polynomials of respective degree* $\partial a_j < h_j$, *which are the uniquely determined solutions of the linear system* (2.16b). $h_j := n_{j+1} - n_j$ *is determined by* (2.3).

(ii) *The regular* FOP2*s* Q_{n_j}, $j = 1, \cdots, J$ $(\leq \infty)$, *also satisfy the three-term recurrence* (2.17) *with the same coefficients* a_j *and* β_j, *but restricted to* $j > 0$ *and with the initial values* $Q_0(z) :\equiv 0$, $Q_1(z) :\equiv \phi_k$.

Proof of part (ii). Subtracting (2.17) from the same relation with z replaced by ζ, then dividing by $\zeta - z$ and applying Φ_l yields on the left-hand side $Q_{n_{j+1}}(z)$. On the right-hand side we apply the same trick as in the proof of Theorem 2.3. The most critical term is $\Phi_l(pP_{n_j})$ with $p(z) = [W_{h_j}(\zeta) - W_{h_j}(z)]/(\zeta - z)$, where $\partial p = h_j - 1 < n_j + h_j - 1 = \hat{n}$, since $n_j > 0$ because of the restriction to $j > 0$. □

For special choices of the basis $\{W_m\}$ additional information can often be given. We treat the case of practical interest where $\{W_m\}$ satisfies a three-term recurrence (2.10). Then (2.11) holds, and likewise

$$zP_{n_{j+1}-1}(z) = W_{h_j}(z) P_{n_j}(z) + \alpha^W_{h_j-1} P_{n_{j+1}-1} + \beta^W_{h_j-1} P_{n_{j+1}-2}\,. \tag{2.18}$$

After inserting the expression obtained from the recurrence relation (2.17) here for $W_{h_j}(z)P_{n_j}(z)$, and representing there a_j in terms of the polynomials W_m we obtain the following corollary.

COROLLARY 2.8. *Assume that the prescribed polynomials W_m satisfy the three-term recurrence* (2.10) *and let*

$$a_j(z) = \sum_{s=0}^{h_j-1} \alpha_{s,j} W_s(z). \tag{2.19}$$

Then, in terms of the FOP1*s P_n defined in* (1.28) *the recurrence* (2.17) *becomes*

$$\begin{aligned} P_{n_{j+1}}(z) = {} & (z - \alpha_{h_j-1,j} - \alpha^W_{h_j-1})P_{n_{j+1}-1}(z) - (\alpha_{h_j-2,j} + \beta^W_{h_j-1})P_{n_{j+1}-2}(z) \\ & -\alpha_{h_j-3,j}P_{n_{j+1}-3}(z) - \cdots - \alpha_{0,j}P_{n_j}(z) - \beta_j P_{n_{j-1}}(z), \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad j = 0, \cdots, J-1, \end{aligned} \tag{2.20}$$

while for $n_j < n < n_{j+1}$, $j = 0, \cdots, J-1$, the polynomials P_n satisfy the recurrence (2.11). *The same recurrences hold for $j = 1, \cdots, J-1$ for the associated* FOP2*s Q_n, which satisfy* (1.35).

Recall that Corollary 2.8 covers the basis $W_m(z) = z^m$ as the special case where $\alpha^W_m = \beta^W_m = 0$ $(\forall m \in \mathbb{N})$ in (2.10). In this case, the linear system (2.16b) for the coefficients of the polynomial a_j becomes a lower right triangular Hankel system.

The linear system (2.16) for β_j and the coefficients of a_j can be replaced by an equivalent one obtained by left multiplication with a nonsingular matrix. In fact, we will see later that the nongeneric Lanczos algorithm makes use of such an equivalent formulation. Recall that (2.16) just means that

$$\beta_j P_{n_{j-1}} - W_{h_j} P_{n_j} \perp z^{n_j-1}, \tag{2.21a}$$

$$a_j P_{n_j} + \beta_j P_{n_{j-1}} - W_{h_j} P_{n_j} \perp z^{n_j+k}, \qquad k = 0, \cdots, h_j - 1, \tag{2.21b}$$

where $\perp$ denotes (pseudo-)orthogonality with respect to the indefinite bilinear "inner product" $\Phi_l : (p, q) \to \Phi_l(pq)$. Since $z^{n_j-1} \perp P_{n_j}$, it follows from (2.21a) that (2.21b) also holds for $k = -1$. Moreover, by (2.1) and (2.2), the left-hand sides of (2.21a) and (2.21b) are orthogonal to any polynomial $p \in \mathcal{P}_{n_j-2}$ anyway. Hence, on the right-hand side, the monomials z^{n_j-1} and z^{n_j+k} $(k = 0, \cdots, h_j - 1)$ can be replaced by any other polynomials of the same degree. A natural choice, which is also made use of in the Lanczos process, is to replace them by P_{n_j-1} and P_{n_j+k} $(k = 0, \cdots, h_j - 1)$, respectively. This has in particular the effect that the terms with β_j in (2.16b) vanish. This modification can be understood as a left-multiplication of the system by a lower (right) triangular matrix and transforms the coefficient matrix of (2.16b) into another right triangular matrix:

$$\beta_j \Phi_l(P_{n_j-1} P_{n_{j-1}}) = \Phi_l(P_{n_j-1} W_{h_j} P_{n_j}), \tag{2.22a}$$

$$\Phi_l(a_j P_{n_j+k} P_{n_j}) = \Phi_l(P_{n_j+k} W_{h_j} P_{n_j}), \quad k = 0, \cdots, h_j - 1. \tag{2.22b}$$

Moreover, since $W_{h_j} P_{n_j}$ is a linear combination of $zP_{n_{j+1}-1}, P_{n_{j+1}-1}, P_{n_{j+1}-2}, \cdots, P_{n_j}$ and all these polynomials except the first one are orthogonal to P_{n_j-1}, we can

replace the right-hand side of (2.22a) by $\Phi_l(zP_{n_j-1}P_{n_{j+1}-1})$. In (2.22b) this simplification does not apply, but we can substitute $W_{h_j}P_{n_j}$ according to (2.18) if (2.10) holds.

THEOREM 2.9. *If* (2.10) *holds, the linear system* (2.16a, b) *for computing the polynomial* a_j *and the constant* β_j *for the recurrence* (2.17) *can be replaced by the equivalent linear system*

$$\beta_j\Phi_l(P_{n_j-1}P_{n_{j-1}}) = \Phi_l(zP_{n_j-1}P_{n_{j+1}-1}), \tag{2.23a}$$

$$\begin{aligned}\Phi_l(a_jP_{n_j+k}P_{n_j}) &= \Phi_l(zP_{n_j+k}P_{n_{j+1}-1}) - \alpha^W_{h_j-1}\Phi_l(P_{n_j+k}P_{n_{j+1}-1})\\ &\quad - \beta^W_{h_j-1}\Phi_l(P_{n_j+k}P_{n_{j+1}-2}), \qquad k = 0, \cdots, h_j - 1,\end{aligned} \tag{2.23b}$$

consisting also of a single equation for β_j *and, if* a_j *is expressed as in* (2.19), *of a right lower triangular system for the coefficients* $\alpha_{s,j}$ *of* a_j. *If* (2.10) *does not hold,* (2.23a) *and* (2.22b) *are valid.*

3. Matrix interpretations. As is well known, any three-term recurrence of the form (2.10) gives rise, after introducing the row vectors

$$\mathbf{w}_m := [W_0, W_1, \cdots, W_m] \qquad (m \in \mathbb{N}) \tag{3.1}$$

and the tridiagonal matrices

$$\mathbf{T}^W_m := \begin{bmatrix} \alpha^W_0 & \beta^W_1 & & & \\ 1 & \alpha^W_1 & \beta^W_2 & & \\ & 1 & \alpha^W_2 & \ddots & \\ & & \ddots & \ddots & \beta^W_m \\ & & & 1 & \alpha^W_m \end{bmatrix} \qquad (m \in \mathbb{N}), \tag{3.2}$$

to a sequence of vector-matrix relations

$$z\mathbf{w}_m(z) = \mathbf{w}_m(z)\mathbf{T}^W_m + [0, \cdots, 0, W_{m+1}(z)] \qquad (m \in \mathbb{N}), \tag{3.3}$$

from which it is seen that every zero of W_{m+1} is an eigenvalue of $\mathbf{T}^W_m$, and the vector $\mathbf{w}_m(z)$ evaluated at such a zero is a corresponding left eigenvector. In case of a zero of multiplicity $\mu > 1$ differentiation of (3.3) shows that the vectors $\mathbf{w}'_m(z), \cdots, \mathbf{w}^{(\mu-1)}_m(z)$ are corresponding principal vectors. Hence, $\mathbf{T}^W_m$ is nonderogatory, i.e., all its eigenvalues have unit geometric multiplicity [49, p. 15].

For the following it is important that these connections still hold if the recurrence includes further previous iterates and the tridiagonal matrix becomes an upper Hessenberg matrix [49, p. 426].

For those n where a regular FOP1 does not exist, we assume again that P_n is defined by (1.28), and that the prescribed sequence $\{W_m\}$ satisfies the three-term recurrence (2.10). (An arbitrary monic sequence $\{W_m\}$, for which a more general recurrence than (2.10) is valid, could also be used; the necessary modifications are straightforward.) Since (2.10) implies (2.11), which is valid for $n_j \le n \le n_{j+1} - 2$, there holds

$$z[P_{n_j}(z), \cdots, P_{n_{j+1}-2}(z)] = [P_{n_j}(z), \cdots, P_{n_{j+1}-2}(z)]\mathbf{T}^W_{h_j-2} + [0, \cdots, 0, P_{n_{j+1}-1}(z)]. \tag{3.4}$$

For $n = n_{j+1} - 1$, i.e., for computing $P_{n_{j+1}}$, we have the relation (2.20). Together these two relations lead immediately to the following result.

THEOREM 3.1. *For some fixed l, let $\{P_n\}$ and $\{Q_n\}$ be the* FOP1*s of* (1.28) *and the associated* FOP2*s, and let $\{W_m\}$ satisfy* (2.10) (*possibly with $\alpha_m^W = \beta_m^W = 0$, $\forall m \in \mathbb{N}$*). *Moreover, define the infinite row vectors*

$$\mathbf{p} := [P_0, P_1, \cdots], \qquad \mathbf{q} := [Q_{n_1}, Q_{n_1+1}, \cdots], \tag{3.5}$$

and the infinite unreduced block tridiagonal upper Hessenberg matrices

$$\mathbf{H} := \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_1 & & & \\ \mathbf{C}_0 & \mathbf{A}_1 & \mathbf{B}_2 & & \\ & \mathbf{C}_1 & \mathbf{A}_2 & \ddots & \\ & & \ddots & \ddots & \mathbf{B}_J \\ & & & \mathbf{C}_{J-1} & \mathbf{A}_J \end{bmatrix}, \quad \mathbf{H}' := \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_2 & & & \\ \mathbf{C}_1 & \mathbf{A}_2 & \ddots & & \\ & \ddots & \ddots & \mathbf{B}_J & \\ & & \mathbf{C}_{J-1} & \mathbf{A}_J & \end{bmatrix}, \tag{3.6a}$$

or

$$\mathbf{H} := \begin{bmatrix} \mathbf{A}_0 & \mathbf{B}_1 & & \\ \mathbf{C}_0 & \mathbf{A}_1 & \mathbf{B}_2 & \\ & \mathbf{C}_1 & \mathbf{A}_2 & \ddots \\ & & \ddots & \ddots \end{bmatrix}, \quad \mathbf{H}' := \begin{bmatrix} \mathbf{A}_1 & \mathbf{B}_2 & & \\ \mathbf{C}_1 & \mathbf{A}_2 & \ddots & \\ & \ddots & \ddots & \end{bmatrix}, \tag{3.6b}$$

in case $J < \infty$ or $J = \infty$, respectively. Here, for $0 \le i < J$,

$$\begin{aligned} \mathbf{A}_i :&= \begin{bmatrix} \alpha_0^W & \beta_1^W & & & \alpha_{0,i} \\ 1 & \alpha_1^W & \ddots & & \vdots \\ \ddots & \ddots & \ddots & \beta_{h_i-2}^W & \alpha_{h_i-3,i} \\ & & 1 & \alpha_{h_i-2}^W & \beta_{h_i-1}^W + \alpha_{h_i-2,i} \\ & & & 1 & \alpha_{h_i-1}^W + \alpha_{h_i-1,i} \end{bmatrix} \\ &= \mathbf{T}_{h_i-1}^W + \begin{bmatrix} \alpha_{0,i} \\ \vdots \\ \alpha_{h_i-3,i} \\ \alpha_{h_i-2,i} \\ \alpha_{h_i-1,i} \end{bmatrix} [0, \cdots, 0, 1] \end{aligned} \tag{3.7}$$

are square matrices of respective order h_i, which in the case where $h_i = 1$ reduce to $\mathbf{A}_i = [\alpha_{0,i}]$; moreover, for $0 < i \le j < J$,

$$\mathbf{B}_i := \begin{bmatrix} 0 & \cdots & 0 & \beta_i \\ 0 & \cdots & 0 & 0 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} = \begin{bmatrix} \beta_i \\ 0 \\ \vdots \\ 0 \end{bmatrix} [0, \cdots, 0, 1] \tag{3.8}$$

and

$$\mathbf{C}_{i-1} := \begin{bmatrix} 0 & \cdots & 0 & 1 \\ 0 & \cdots & 0 & 0 \\ \vdots & & \vdots & \vdots \\ 0 & \cdots & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} [0, \cdots, 0, 1] \tag{3.9}$$

are rectangular rank-one matrices of size $h_{i-1} \times h_i$ *and* $h_i \times h_{i-1}$, *respectively. For* $j = J < \infty$,

$$\mathbf{A}_J := \mathbf{T}^W := \begin{bmatrix} \alpha_0^W & \beta_1^W & & \\ 1 & \alpha_1^W & \beta_2^W & \\ & 1 & \alpha_2^W & \ddots \\ & \ddots & & \ddots \end{bmatrix} \tag{3.10}$$

is the infinite tridiagonal matrix of the three-term recurrence (2.10) *and* $\mathbf{B}_J$ *is the* $h_{J-1} \times \infty$ *zero matrix.*

Then,

$$z\mathbf{p}(z) = \mathbf{p}(z)\mathbf{H}\,, \tag{3.11}$$

and

$$z\mathbf{q}(z) = \mathbf{q}(z)\mathbf{H}'\,. \tag{3.11'}$$

Moreover, if

$$\begin{aligned} &\mathbf{p}_n := [P_0, P_1, \cdots, P_n] \qquad (n \in \mathbb{N}), \\ &\mathbf{q}_n := [Q_{n_1}, Q_{n_1+1}, \cdots, Q_n] \qquad (n \geq n_1), \end{aligned} \tag{3.12}$$

and if $\mathbf{H}_n$ *denotes the principal submatrix of order* $n+1$ *of* $\mathbf{H}$, *then there holds for* $n = 0, 1, \cdots$,

$$z\mathbf{p}_n(z) = \mathbf{p}_n(z)\mathbf{H}_n + [0, \cdots, 0, P_{n+1}(z)]\,, \tag{3.13}$$

while, if $\mathbf{H}'_n$ *is the principal submatrix of order* $n - n_1 + 1$ *of* $\mathbf{H}'$, *there holds for* $n = n_1, n_1 + 1, \cdots$,

$$z\mathbf{q}_n(z) = \mathbf{q}_n(z)\mathbf{H}'_n + [0, \cdots, 0, Q_{n+1}(z)]\,. \tag{3.13'}$$

The matrices $\mathbf{A}_i$ are so-called *comrade matrices* [2]. If the polynomials W_m do not satisfy a three-term recurrence, but a general recurrence involving $W_0, W_1, \cdots, W_{m-1}$, these matrices are instead general unit upper Hessenberg matrices (cf. [29], [30]).[7]

The relations (3.13) and (3.13') lead by the standard argument mentioned before (cf. [7, pp. 94–96]) to the following corollary.

COROLLARY 3.2. *The zeros of the polynomial* P_{n+1} *are eigenvalues of* $\mathbf{H}_n$, *and the vectors* $\mathbf{p}_n(z)$ *evaluated at these zeros are corresponding left eigenvectors. In case of a zero* z *of multiplicity* $\mu > 1$, *the vectors* $\mathbf{p}'_n(z), \cdots, \mathbf{p}^{(\mu-1)}(z)$ *are corresponding principle vectors. (Here, in* $\mathbf{p}'_n$, *the prime denotes differentiation.) The matrix* $\mathbf{H}_n$ *is nonderogatory, i.e., all its eigenvalues have unit geometric multiplicity.*

The analogue holds for Q_{n+1}, $\mathbf{H}'_n$, *and* $\mathbf{q}_n(z)$.

The following explicit formulas are valid:

$$P_n(z) = \det(z\mathbf{I}_n - \mathbf{H}_n) \qquad (n \in \mathbb{N}), \tag{3.14}$$

$$Q_n(z) = \phi_k \det(z\mathbf{I}_{n-n_1} - \mathbf{H}'_n) \qquad (n \geq n_1), \tag{3.14'}$$

[7] The author is indebted to Gene Golub and Mark Kent for these three references.

where $\mathbf{I}_n$ *denotes the unit matrix of order* $n+1$, *and where* $k = n_1 + l - 1$.

The formulas (3.11)–(3.13′) lead to interesting matrix relations if the polynomials P_n are represented in a basis, say, the monomial basis. We first let

$$\mathbf{z}(z) := (1,\, z,\, z^2,\, \cdots) \tag{3.15}$$

and

$$\mathbf{S} := \begin{bmatrix} 0 & & & \\ 1 & 0 & & \\ & 1 & 0 & \\ & & \ddots & \ddots \end{bmatrix}. \tag{3.16}$$

Still assuming l fixed, we define the infinite unit upper triangular matrices

$$\mathbf{P} := \begin{bmatrix} 1 & \pi_{l;0,1} & \pi_{l;0,2} & \cdots \\ & 1 & \pi_{l;1,2} & \\ & & 1 & \ddots \\ & & & \ddots \end{bmatrix}, \qquad \mathbf{P}' := \begin{bmatrix} 1 & \pi'_{k;0,1} & \pi'_{k;0,2} & \cdots \\ & 1 & \pi'_{k;1,2} & \\ & & 1 & \ddots \\ & & & \ddots \end{bmatrix}, \tag{3.17}$$

whose elements in the $(n+1)$th column are the coefficients of $P_n = P_{l;n}$ and $\phi_k^{-1} Q_{n-n_1} = \phi_k^{-1} Q_{l;n-n_1}$, respectively (cf. (1.4) and Theorem 1.15). The diagonal elements of $\mathbf{P}$ and $\mathbf{P}'$ are all 1 since the polynomials P_n and $\phi_k^{-1} Q_{n-n_1}$ are monic. Using this notation we can now write $z\mathbf{z}(z) = \mathbf{z}(z)\mathbf{S}$ and

$$\mathbf{p}(z) = \mathbf{z}(z)\mathbf{P}\,, \qquad z\mathbf{p}(z) = \mathbf{z}(z)\mathbf{S}\mathbf{P}\,, \tag{3.18}$$

$$\mathbf{q}(z) = \mathbf{z}(z)\mathbf{P}'\,, \qquad z\mathbf{q}(z) = \mathbf{z}(z)\mathbf{S}\mathbf{P}'\,, \tag{3.18′}$$

so that (3.11) and (3.11′) become

$$\mathbf{z}(z)\mathbf{S}\mathbf{P} = \mathbf{z}(z)\mathbf{P}\mathbf{H}\,, \qquad \mathbf{z}(z)\mathbf{S}\mathbf{P}' = \mathbf{z}(z)\mathbf{P}'\mathbf{H}'\,. \tag{3.19}$$

Each component of the row vectors that are claimed equal in these relations is a polynomial and hence the matrices containing the coefficients must be equal: $\mathbf{SP} = \mathbf{PH}$ and $\mathbf{SP}' = \mathbf{P}'\mathbf{H}'$. By looking at the principal submatrices we obtain relations corresponding to (3.13) and (3.13′). We summarize the result as the following corollary.

COROLLARY 3.3. *Under the assumptions of Theorem* 3.1 *there holds*

$$\mathbf{SP} = \mathbf{PH}\,, \qquad \mathbf{SP}' = \mathbf{P}'\mathbf{H}'\,. \tag{3.20}$$

Furthermore, if $\mathbf{S}_n$, $\mathbf{P}_n$, *and* $\mathbf{H}_n$ *are the principle submatrices of order* $n+1$ *of* $\mathbf{S}$, $\mathbf{P}$, *and* $\mathbf{H}$, *then*

$$\mathbf{S}_n\mathbf{P}_n = \mathbf{P}_n\mathbf{H}_n + \begin{bmatrix} \pi_{l;0,n+1} \\ \pi_{l;1,n+1} \\ \vdots \\ \pi_{l;n,n+1} \end{bmatrix} [0, \cdots, 0, 1]\,. \tag{3.21}$$

Likewise, if $\mathbf{P}'_n$ *and* $\mathbf{H}'_n$ *are the principle submatrices of order* $n - n_1 + 1$ *of* $\mathbf{P}'$ *and* $\mathbf{H}'$, *then*

$$\mathbf{S}_{n-n_1}\mathbf{P}'_n = \mathbf{P}'_n\mathbf{H}'_n + \begin{bmatrix} \pi'_{k;0,n-n_1+1} \\ \pi'_{k;1,n-n_1+1} \\ \vdots \\ \pi'_{k;n-n_1,n-n_1+1} \end{bmatrix} [0, \cdots, 0, 1]\,. \tag{3.21'}$$

Another important matrix interpretation is immediately obtained from Corollaries 2.2 and 2.4.

THEOREM 3.4. *Let* $\mathbf{p}$ *and* $\mathbf{q}$ *be defined by* (3.5), *and let* $\Phi_l(\mathbf{p}^T\mathbf{p})$ *and* $\Psi_k(\mathbf{q}^T\mathbf{q})$ *be the infinite matrices obtained by elementwise application of* Φ_l *and* Ψ_k, *respectively, to the infinite rank-one matrices* $\mathbf{p}^T\mathbf{p}$ *and* $\mathbf{q}^T\mathbf{q}$. *Then*

$$\Phi_l(\mathbf{p}^T\mathbf{p}) = \mathbf{D} \tag{3.22}$$

and

$$\Psi_k(\mathbf{q}^T\mathbf{q}) = \mathbf{D}'\,, \tag{3.22'}$$

where $\mathbf{D}$ *and* $\mathbf{D}'$ *are block diagonal matrices,*

$$\mathbf{D} := \text{block diag}\,[\mathbf{D}_0, \mathbf{D}_1, \cdots]\,, \qquad \mathbf{D}' := \text{block diag}\,[\mathbf{D}'_1, \mathbf{D}'_2, \cdots]\,, \tag{3.23}$$

with square blocks of size h_j ($j = 0, 1, \cdots, J$ *and* $j = 1, 2, \cdots, J$, *respectively*), *which are symmetric right lower triangular matrices of the form*

$$\mathbf{D}_j = \begin{bmatrix} & & & & \delta_j \\ & & & \delta_j & \star \\ & & \iddots & \iddots & \vdots \\ & \delta_j & \iddots & & \star \\ \delta_j & \star & \cdots & \star & \star \end{bmatrix}, \qquad \mathbf{D}'_j = \begin{bmatrix} & & & & \delta'_j \\ & & & \delta'_j & \star \\ & & \iddots & \iddots & \vdots \\ & \delta'_j & \iddots & & \star \\ \delta'_j & \star & \cdots & \star & \star \end{bmatrix}, \tag{3.24}$$

with $\delta_j := \Phi_l(P_{n_j}P_{n_{j+1}-1}) \neq 0$, $\delta'_j := \Psi_k(Q_{n_j}Q_{n_{j+1}-1}) \neq 0$, *and with the stars denoting possible nonnull entries. If* $W_m(z) = z^m$, $\mathbf{D}_j$ *and* $\mathbf{D}'_j$ *are Hankel matrices. If* $J < \infty$, $\mathbf{D}_J$ *and* $\mathbf{D}'_J$ *are the infinite zero matrix, and* $\delta_J = \delta'_J = 0$.

Moreover, let $\mathbf{P}$ *and* $\mathbf{P}'$ *be the unit triangular matrices of* (3.17), *and let* $\mathbf{M}$ *and* $\mathbf{M}'$ *be the infinite moment matrices of the functionals* Φ_l *and* Ψ_k,

$$\mathbf{M} := \begin{bmatrix} \phi_l & \phi_{l+1} & \phi_{l+2} & \cdots \\ \phi_{l+1} & \phi_{l+2} & & \\ \phi_{l+2} & & & \\ \vdots & & & \end{bmatrix}, \qquad \mathbf{M}' := \begin{bmatrix} \psi_0^{(k)} & \psi_1^{(k)} & \psi_2^{(k)} & \cdots \\ \psi_1^{(k)} & \psi_2^{(k)} & & \\ \psi_2^{(k)} & & & \\ \vdots & & & \end{bmatrix}. \tag{3.25}$$

(*The coefficients* $\psi_i^{(k)}$ *are defined in Theorem* 1.15.) *Then*

$$\mathbf{P}^T\mathbf{M}\mathbf{P} = \mathbf{D}\,, \tag{3.26}$$

and

$$(\mathbf{P}')^T\mathbf{M}'\mathbf{P}' = \mathbf{D}'\,. \tag{3.26'}$$

Hence, if $\mathbf{R} := \mathbf{P}^{-1}$ *and* $\mathbf{R}' := (\mathbf{P}')^{-1}$ *are the unit upper triangular matrices which contain in their columns the coefficients of the monomials when expressed as linear combinations of the* FOP1*s and the* FOP2*s, respectively, then*

$$\mathbf{M} = \mathbf{R}^T \mathbf{D} \mathbf{R} \tag{3.27}$$

and

$$\mathbf{M}' = (\mathbf{R}')^T \mathbf{D}' \mathbf{R}' \tag{3.27'}$$

are symmetric block LDU *decompositions of* $\mathbf{M}$ *and* $\mathbf{M}'$*, respectively.*

As we mentioned before, for the case $W_m(z) = z^m$, the recurrence for the FOP1s is due to Struble [44] (although he did not specify how to compute the coefficients), but the matrix decompositions of the left-hand side equation in (3.20) and (3.26) and the structure of the matrix $\mathbf{H}$ of (3.6a, b) are, for that case, first briefly mentioned without proof in Gragg [13, p. 222]. Ten years later, (3.26) was also given in the monograph of Draux [7, p. 92], who derived it too from his analogue of our Corollary 2.2. In view of Gragg's early discovery of these results, the following definition seems justified.

DEFINITION. A matrix of the form specified by (3.6)–(3.10) is called a *Gragg matrix.*

As we know there is some freedom in choosing the singular and the deficient FOP1s, namely, the polynomials W_m in (1.28) can be chosen arbitrarily. There is the question of how this freedom can be taken advantage of to achieve special properties. So far we have concentrated on the simplest choice $W_m(z) = z^m$ and on the one where a three-term recurrence holds. We now show that for fixed j these polynomials can be chosen in such a way that for $n_j + n_{j+1} \geq n + n'$, $n_j < n < n_{j+1}$, $n_j < n' < n_{j+1}$, the "inner products" $\Phi_l(P_n P_{n'}) = \Phi_l(P_{n'} P_n)$ take an arbitrary prescribed value, e.g., 0. Draux has a similar result, which, however, is unclearly stated [7, p. 75–76]. We start with a matrix theoretic lemma. Recall that a matrix $\mathbf{W}$ is called persymmetric if it is symmetric about its antidiagonal.

LEMMA 3.5. *Let* $\mathbf{D} = (\delta_{k,l})_{k,l=1}^{h}$ *be a nonsingular lower right triangular matrix of order* h*, and let* $\hat{\mathbf{D}}$ *denote the antidiagonal matrix with the same antidiagonal elements. Assume that no sum of any two of these antidiagonal elements vanishes. Moreover, let* $\mathbf{J}$ *be the antidiagonal unit matrix, so that for any* $h \times h$ *matrix* $\mathbf{A}$*, reflection at the antidiagonal yields* $(\mathbf{JAJ})^T = \mathbf{JA}^T\mathbf{J}$.

Then, there exists a uniquely determined unit upper triangular matrix $\mathbf{W} = (\omega_{k,l})_{k,l=1}^{h}$ *such that*

$$\mathbf{D} = (\mathbf{JWJ})\hat{\mathbf{D}}^T \mathbf{W} \tag{3.28}$$

is a decomposition of $\mathbf{D}$ *into a unit lower triangular, an antidiagonal, and a unit upper triangular matrix, the two triangular matrices being related by a rotation.*

If $\mathbf{D}$ *is symmetric,* $\mathbf{W}$ *is persymmetric, so that*

$$\mathbf{D} = \mathbf{W}^T \hat{\mathbf{D}} \mathbf{W}\,. \tag{3.29}$$

In particular, if $\mathbf{D}$ *is a Hankel matrix, then* $\mathbf{W}$ *is a Toeplitz matrix. More specifically, if*

$$\delta_{k,l} = \eta_{k+l-h-1}\,, \quad 1 \leq k \leq h, \quad 1 \leq l \leq h, \quad k + l > h, \tag{3.30a}$$

then

$$\omega_{k,l} = \chi_{l-k}\,, \qquad 1 \le k \le l \le h, \tag{3.30b}$$

and the two polynomials

$$y(z) := \sum_{k=0}^{h-1} \eta_k z^k, \qquad x(z) := \sum_{k=0}^{h-1} \chi_k z^k \tag{3.31}$$

are related by

$$\frac{y(z)}{y(0)} - [x(z)]^2 = O(z^h)\,, \tag{3.32}$$

which means that $x(z)$ is the partial sum of degree $h-1$ of the power series of $\sqrt{y(z)/y(0)}$.

Proof. The proof is constructive, namely, by establishing algorithms for computing $\mathbf{W}$ and $\mathbf{x}$. Equating the (m,l)-element in (3.28) yields

$$\delta_{m,l} = \sum_{k=1}^{h} \omega_{h+1-m,k}\,\delta_{h+1-k,k}\,\omega_{k,l} = \sum_{k=h+1-m}^{l} \omega_{h+1-m,k}\,\delta_{h+1-k,k}\,\omega_{k,l}\,. \tag{3.33}$$

For $m+l \le h$, both sides are clearly zero. For $m+l = h+1$, we obtain the condition $\delta_{h+1-l,l} = \omega_{l,l}^2\,\delta_{h+1-l,l}$, which is satisfied by choosing $\omega_{l,l} := 1$, $l = 1, \cdots, h$. Then, for $i = 1, \cdots, h-1$, there results the recurrence

$$\omega_{l-i,l} := \frac{1}{\delta_{h+1-l,l} + \delta_{h+1+i-l,l-i}} \left[\delta_{h+1+i-l,l} - \sum_{k=l-i+1}^{l-1} \omega_{l-i,k}\,\omega_{k,l}\,\delta_{h+1-k,k}\right], \tag{3.34}$$

$$l = i+1, \cdots, h\,,$$

which can be used to build up $\mathbf{W}$ codiagonal by codiagonal, since the denominators on the right-hand side cannot vanish by assumption.

If $\mathbf{D}$ is symmetric, it is seen that $\omega_{h+1-l,h+1+i-l} = \omega_{l-i,l}$, so that $\mathbf{JWJ} = \mathbf{W}^T$, which turns (3.28) into (3.29). Moreover, if $\mathbf{D}$ is Hankel and (3.30a, b) holds, and if we make the induction assumption that $\omega_{k,l} = \chi_{l-k}$ for $1 \le k \le l \le h$, $0 \le l-k \le i-1$, then (3.34) becomes

$$\omega_{l-i,l} := \frac{1}{2\eta_0}\left[\eta_i - \sum_{k=l-i+1}^{l-1} \chi_{k-l+i}\,\chi_{l-k}\,\eta_0\right] = \frac{1}{2}\left[\frac{\eta_i}{\eta_0} - \sum_{m=1}^{i-1} \chi_m\,\chi_{i-m}\right]. \tag{3.35}$$

This value only depends on i and can therefore be called χ_i. Consequently, since $\chi_0 = \omega_{l,l} = 1$, there holds for $i = 1, \cdots, h-1$,

$$\frac{\eta_i}{\eta_0} = 2\chi_i + \sum_{m=1}^{i-1} \chi_m\,\chi_{i-m} = \sum_{m=0}^{i} \chi_m\,\chi_{i-m}\,.$$

For the polynomials $y(z)$ and $x(z)$ of (3.31), this is exactly the coefficient relation equivalent to (3.32). □

Keeping l still fixed we now let $\{P_n\}$ be a particular sequence of (monic) FOP1s, say, the one defined by

$$P_n(z) := z^{n-n_j} P_{n_j}(z) \quad \text{if } n_j \le n < n_{j+1}\,. \tag{3.36}$$

(Recall that the regular FOP1s P_{n_j} are uniquely determined by Φ_l.) In addition, we consider another arbitrary sequence $\{\tilde{P}_n\}$ of FOP1s for the same functional Φ_l and the associated sequence $\{\tilde{Q}_n\}$ of FOP2s. They have the form

$$\tilde{P}_n(z) := w_n(z) P_{n_j}(z), \quad \tilde{Q}_n(z) := w_n(z) Q_{n_j}(z) \quad \text{if } n_j \le n < n_{j+1}\,, \tag{3.37}$$

where w_n is now an arbitrary monic polynomial of exact degree $n-n_j$. Consequently, if written in matrix-vector notation, the four sequences $\mathbf{p}, \tilde{\mathbf{p}}, \mathbf{q}, \tilde{\mathbf{q}}$ are related by

$$\tilde{\mathbf{p}} = \mathbf{p}\mathbf{W}, \qquad \tilde{\mathbf{q}} = \mathbf{q}\mathbf{W}' \tag{3.38}$$

with

$$\mathbf{W} := \text{ block diag } [\mathbf{W}_0, \mathbf{W}_1, \cdots], \qquad \mathbf{W}' := \text{ block diag } [\mathbf{W}_1, \mathbf{W}_2, \cdots], \tag{3.39}$$

where, for $j \in \mathbb{N}$, the $h_j \times h_j$ matrices $\mathbf{W}_j$ are unit upper triangular. They can be chosen arbitrarily, thus reflecting the freedom in choosing w_n. If we allow that the sequence $\{\tilde{P}_n\}$ does not necessarily consist of monic formal orthogonal polynomials, and we want to do that here in view of later applications, then the matrices $\mathbf{W}_j$ are arbitrary nonsingular upper triangular. If we denote by $\tilde{\mathbf{P}}, \tilde{\mathbf{P}}', \tilde{\mathbf{H}}, \tilde{\mathbf{H}}', \tilde{\mathbf{D}}, \tilde{\mathbf{D}}'$ the matrices belonging to the new sequences, we readily obtain the relations

$$\tilde{\mathbf{P}} = \mathbf{P}\mathbf{W}, \qquad \tilde{\mathbf{P}}' = \mathbf{P}'\mathbf{W}'\,, \tag{3.40}$$

$$\tilde{\mathbf{H}} = \mathbf{W}^{-1}\mathbf{H}\mathbf{W}, \qquad \tilde{\mathbf{H}}' = (\mathbf{W}')^{-1}\mathbf{H}'\mathbf{W}'\,, \tag{3.41}$$

$$\tilde{\mathbf{D}} = \mathbf{W}^{T}\mathbf{D}\mathbf{W}, \qquad \tilde{\mathbf{D}}' = (\mathbf{W}')^{T}\mathbf{D}'\mathbf{W}'\,. \tag{3.42}$$

For the blocks $\tilde{\mathbf{A}}_j, \tilde{\mathbf{B}}_j, \tilde{\mathbf{C}}_j$ in $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{H}}'$ we conclude in view of the special structure of $\mathbf{B}_j$ and $\mathbf{C}_j$:

$$\tilde{\mathbf{A}}_j = \mathbf{W}_j^{-1}\mathbf{A}_j\mathbf{W}_j\,, \quad \tilde{\mathbf{B}}_j = \frac{\omega_{n_j-1}}{\omega_{n_{j-1}}}\mathbf{B}_j\,, \quad \tilde{\mathbf{C}}_j = \frac{\omega_{n_{j+1}-1}}{\omega_{n_j}}\mathbf{C}_j\,, \tag{3.43}$$

where ω_n denotes now the leading coefficient of w_n. For the blocks $\tilde{\mathbf{D}}_j$ of $\tilde{\mathbf{D}}$ and $\tilde{\mathbf{D}}'$ we have simply

$$\tilde{\mathbf{D}}_j = \mathbf{W}_j^{T}\mathbf{D}\mathbf{W}_j\,. \tag{3.44}$$

Now we are ready to prove the following theorem.

THEOREM 3.6. *For $j = 0, 1, \cdots, J-1$ let $\tilde{\mathbf{D}}_j$ be any symmetric $h_j \times h_j$ matrix of the form given in* (3.24), *with $\delta_j := \Phi_l(P_{n_j}P_{n_{j+1}-1})$, and let $\tilde{\mathbf{D}}_J = \mathbf{O}$ if $J < \infty$. Then there is a monic sequence $\{\tilde{P}_n\}$ of* FOP1*s such that $\Phi_l(\tilde{\mathbf{p}}^T\tilde{\mathbf{p}}) = \tilde{\mathbf{D}} :=$* block diag *$[\tilde{\mathbf{D}}_0, \tilde{\mathbf{D}}_1, \cdots]$.*

In particular, there is such a sequence $\{\tilde{P}_n\}$ for which $\tilde{\mathbf{D}}_j$ is the antidiagonal matrix with elements δ_j $(j = 0, 1, \cdots, J-1)$, and the corresponding transformation

matrices $\mathbf{W}_j$ *in* (3.39) *are Toeplitz matrices whose elements can be computed according to* (3.30)–(3.32), *with*

$$\eta_m^{(j)} := \Phi_l(P_{n_j+m}P_{n_{j+1}-1}), \qquad 0 \le m < h_j\,, \tag{3.45}$$

where $\{P_n\}$ *is the sequence of* (3.36).

Proof. First let $\hat{\mathbf{D}}$ be the block diagonal matrix whose diagonal blocks $\hat{\mathbf{D}}_j$ are antidiagonal $h_j \times h_j$ matrices with elements δ_j, and let $\mathbf{D}$ be given by (3.22) with $\mathbf{p} = \{P_n\}$ defined by (3.36). The blocks $\mathbf{D}_j$ of $\mathbf{D}$ are Hankel and have the same constant antidiagonal as $\hat{\mathbf{D}}_j$. Hence, by Lemma 3.5, applied to each diagonal block, there is a set of unit upper triangular transformation matrices $\mathbf{W}_j$ such that $\mathbf{D}_j = \mathbf{W}_j^T\hat{\mathbf{D}}_j\mathbf{W}_j$, $j = 0, \cdots, J-1$. ($\mathbf{W}_J$ can be chosen as an arbitrary infinite unit upper triangular matrix.)

Likewise, if $\tilde{\mathbf{D}}$ satisfies the assumptions of the theorem, there is by Lemma 3.5 another set of unit upper triangular matrices $\tilde{\mathbf{W}}_j$, such that $\tilde{\mathbf{D}}_j = \tilde{\mathbf{W}}_j^T\hat{\mathbf{D}}_j\tilde{\mathbf{W}}_j$. The block diagonal matrix $\tilde{\mathbf{W}}^{-1}\mathbf{W}$, whose blocks are $\tilde{\mathbf{W}}_j^{-1}\mathbf{W}_j$, then yields $\tilde{\mathbf{p}} := \mathbf{p}\tilde{\mathbf{W}}^{-1}\mathbf{W}$ with the claimed property. □

4. The nongeneric Lanczos biorthogonalization algorithm and nongeneric BIORES. Let $\mathbf{A} : \mathcal{H} \to \mathcal{H}$ be a bounded linear operator that maps a separable real or complex Hilbert space $\mathcal{H}$ into itself. A real operator is treated here as a special case of a complex one, hence, we assume that in general the scalars are from $\mathbb{C}$. Then $\mathcal{H}$ is isomorphic to either l^2 or $\mathbb{C}^N$ (for some N), and if expressed in an orthogonal basis, $\mathcal{H}$ is represented by an infinite or a finite square complex matrix. We denote the standard inner product in $\mathcal{H}$ by $\langle .\,, .\rangle$ and the formal inner product based on an operator $\mathbf{B}$ (instead of the identity) by $\langle .\,, .\rangle_{\mathbf{B}}$, but besides that we use common matrix notation: $\|.\|$ for the norm in $\mathcal{H}$, $\mathbf{A}^H$ for the adjoint operator of $\mathbf{A}$, $\overline{\mathbf{A}}$ for the complex conjugate operator, $\mathbf{A}^T = (\overline{\mathbf{A}})^H$ for the transposed, and $\|\mathbf{A}\|$ for the spectral norm of $\mathbf{A}$.

Given $\mathbf{x}_0, \mathbf{y}_0 \in \mathcal{H}$, the *Schwarz constants* or *moments* of $\mathbf{A}$ with respect to $\mathbf{x}_0$ and $\mathbf{y}_0$ are normally defined by $\phi_k := \langle \mathbf{y}_0, \mathbf{A}^k\mathbf{x}_0\rangle$ ($k \in \mathbb{N}$). We use the slightly more general definition

$$\phi_k := \langle \mathbf{y}_0, \mathbf{A}^k\mathbf{x}_0\rangle_{\mathbf{B}} := \langle \mathbf{y}_0, \mathbf{B}\mathbf{A}^k\mathbf{x}_0\rangle \qquad (k \in \mathbb{N})\,, \tag{4.1}$$

where $\mathbf{B} : \mathcal{H} \to \mathcal{H}$ is a bounded linear operator that commutes with $\mathbf{A}$. (The three most relevant cases are $\mathbf{B} = \mathbf{I}$, $\mathbf{B} = \mathbf{A}$, and $\mathbf{B} = \mathbf{A}^{-1}$.) Since $\mathbf{A}$ and $\mathbf{B}$ are bounded, $|\phi_k| \le \|\mathbf{y}_0\|\,\|\mathbf{x}_0\|\,\|\mathbf{B}\|\,\|\mathbf{A}\|^k$; hence, the power series

$$F(\zeta) := \sum_{k=0}^{\infty} \phi_k \zeta^{-k-1} \tag{4.2}$$

in ζ^{-1} converges at least for $|\zeta| > \|\mathbf{A}\|$, while the series

$$f(z) := z^{-1}F(z^{-1}) = \sum_{k=0}^{\infty} \phi_k z^k \tag{4.3}$$

converges at least for $|z| < 1/\|\mathbf{A}\|$. A fortiori, F and f are analytic functions for $|\zeta| > \|\mathbf{A}\|$ (including ∞) and $|z| < 1/\|\mathbf{A}\|$, respectively. Inserting (4.1) into (4.2) and summing up the Neumann series

$$\mathbf{I} + \zeta^{-1}\mathbf{A} + \zeta^{-2}\mathbf{A}^2 + \cdots = \zeta(\zeta\mathbf{I} - \mathbf{A})^{-1}, \tag{4.4}$$

we obtain

$$F(\zeta) = \langle \mathbf{y}_0, (\zeta\mathbf{I} - \mathbf{A})^{-1}\mathbf{x}_0 \rangle_{\mathbf{B}}, \tag{4.5}$$

which means that $F(\zeta)$ is the first Schwarz constant for the *resolvent* $(\zeta\mathbf{I} - \mathbf{A})^{-1}$. According to (4.4) the resolvent exists and is bounded at least for $|\zeta| > \|\mathbf{A}\|$, while the spectrum $\lambda(\mathbf{A})$ of $\mathbf{A}$ is of course contained in the closed disk $|\zeta| \leq \|\mathbf{A}\|$.

If $\mathbf{A}$ has finite rank, then $F(\zeta)$ is a rational function of order $\nu_F \leq \operatorname{rank}(\mathbf{A})$ vanishing at ∞, i.e., a rational function of type $(\nu_F - 1, \nu_F)$. This holds in particular when $\mathbf{A}$ is a finite matrix. If $\mathbf{A}$, $\mathbf{x}_0$, and $\mathbf{y}_0$ are real, then the moments ϕ_k are real, and $F(\overline{\zeta}) = \overline{F(\zeta)}$.

Starting from two vectors $\mathbf{x}_0, \mathbf{y}_0 \in \mathcal{H}$ satisfying $\langle \mathbf{y}_0, \mathbf{x}_0 \rangle_{\mathbf{B}} \neq 0$, the classical (generic) *Lanczos biorthogonalization* (BO) *algorithm* [24], [25], [39], [18], [15] generates two sequences $\{\mathbf{x}_n\}_{n=0}^{\nu-1}$ and $\{\mathbf{y}_n\}_{n=0}^{\nu-1}$ such that for $n = 0, 1, \cdots, \nu - 1$,

$$\mathbf{x}_n \in \mathcal{K}_{n+1} := \operatorname{span}(\mathbf{x}_0, \mathbf{A}\mathbf{x}_0, \mathbf{A}^2\mathbf{x}_0, \cdots, \mathbf{A}^n\mathbf{x}_0), \tag{4.6a}$$

$$\mathbf{y}_n \in \mathcal{L}_{n+1} := \operatorname{span}(\mathbf{y}_0, \mathbf{A}^H\mathbf{y}_0, (\mathbf{A}^H)^2\mathbf{y}_0, \cdots, (\mathbf{A}^H)^n\mathbf{y}_0) \tag{4.6b}$$

and

$$\langle \mathbf{y}_m, \mathbf{x}_n \rangle_{\mathbf{B}} \begin{cases} = 0 & \text{if } m \neq n, \\ \neq 0 & \text{if } m = n. \end{cases} \tag{4.7}$$

Equations (4.6a, b) mean that $\mathbf{x}_n$ and $\mathbf{y}_n$ are chosen from two families of nested Krylov subspaces generated by $\mathbf{A}, \mathbf{x}_0$ and $\mathbf{A}^H, \mathbf{y}_0$, respectively; and according to (4.7) the sequences $\{\mathbf{x}_n\}$ and $\{\mathbf{y}_n\}$ are formally biorthogonal with respect to the formal inner product $\langle ., . \rangle_{\mathbf{B}}$, which is a true inner product if $\mathbf{B}$ is hermitian positive definite. For simplicity, we generally use the terms "orthogonal" and "biorthogonal," when we actually mean "formally B-orthogonal" and "formally B-biorthogonal."

The maximum length ν of the sequences depends on $\mathbf{A}, \mathbf{x}_0$, and $\mathbf{y}_0$. In fact, (4.6a, b) and (4.7) imply that

$$\mathbf{x}_n \in \mathcal{K}_{n+1} \backslash \mathcal{K}_n, \qquad \mathbf{y}_n \in \mathcal{L}_{n+1} \backslash \mathcal{L}_n \tag{4.8}$$

and

$$\mathbf{x}_n \perp_{\mathbf{B}} \mathcal{L}_n, \qquad \mathbf{y}_n \perp_{\mathbf{B}} \mathcal{K}_n. \tag{4.9}$$

Clearly, once $\mathcal{K}_n = \mathcal{K}_{n+1}$ or $\mathcal{L}_n = \mathcal{L}_{n+1}$, the process must terminate with $\nu = n$. In the first case, $\mathcal{K}_\nu$ is the maximum Krylov subspace of $\mathbf{A}$ generated by $\mathbf{x}_0$, and it is an invariant subspace of dimension ν of $\mathbf{A}$. In the other case, the analogue holds for $\mathbf{A}^H$ and $\mathbf{y}_0$. However, it is well known that when $\mathbf{A}$ is nonhermitian or when it is hermitian, but $\mathbf{y}_0 \neq \mathbf{x}_0$, the process may break down before one of these two situations occurs, because for some n all $\mathbf{x}_n$ and $\mathbf{y}_n$ satisfying (4.8) and (4.9) may be orthogonal to each other so that we cannot fulfill the second line of (4.7). It is this "serious breakdown" that is addressed, explained and overcome in this section by introduction of the *nongeneric* BO *algorithm*.

Every pair $\mathbf{x}_n$ and $\mathbf{y}_n$ satisfying (4.6a, b) can be represented as

$$\mathbf{x}_n = P_n(\mathbf{A})\mathbf{x}_0, \qquad \mathbf{y}_n = P_n^{\sim}(\mathbf{A}^H)\mathbf{y}_0, \tag{4.10}$$

where P_n and $P_n^{\sim}$ are polynomials of degree at most n. The degree is exactly n as long as (4.8) holds. But even beyond that, we may assume without restricting generality that the degree is exactly n, since when $\mathcal{K}_n = \mathcal{K}_{n+1}$, then $\mathbf{A}^n\mathbf{x}_0$ is a linear combination of $\mathbf{x}_0, \cdots, \mathbf{A}^{n-1}\mathbf{x}_0$. (This is clearly true for the first n where $\mathcal{K}_n = \mathcal{K}_{n+1}$, and the linear combination then carries over to all subsequent indices. The coefficients in this linear combination are in fact those of the minimal polynomial of the restriction of $\mathbf{A}$ to its invariant subspace $\mathcal{K}_n$.) Assuming (4.10) with polynomials of exact degree n we thus take care of (4.6a, b) and fulfill (4.8) as long as possible. Moreover, the condition (4.9) transforms into

$$\langle \mathbf{y}_0, \mathbf{A}^k P_n(\mathbf{A})\mathbf{x}_0\rangle_{\mathbf{B}} = 0\,, \qquad k = 0, \cdots, n-1\,, \tag{4.11a}$$

$$\langle \mathbf{y}_0, \mathbf{A}^k \overline{P_n^{\sim}}(\mathbf{A})\mathbf{x}_0\rangle_{\mathbf{B}} = 0\,, \qquad k = 0, \cdots, n-1\,, \tag{4.11b}$$

where the bar indicates complex conjugation of the coefficients of $P_n^{\sim}$.

In accordance with the situation in §1 let us now define on the space $\mathcal{P}$ of formal power series a linear functional Φ_0 by attributing to the monomials the moments

$$\Phi_0(z^k) := \phi_k = \langle \mathbf{y}_0, \mathbf{A}^k\mathbf{x}_0\rangle_{\mathbf{B}} \qquad (k \in \mathbb{N}) \tag{4.12}$$

as values of the functional. Then (4.11) turns into

$$\Phi_0(z^k P_n) = 0\,, \qquad k = 0, \cdots, n-1\,, \tag{4.13a}$$

$$\Phi_0(z^k \overline{P_n^{\sim}}) = 0\,, \qquad k = 0, \cdots, n-1\,. \tag{4.13b}$$

Hence, P_n and $\overline{P_n^{\sim}}$ are true formal orthogonal polynomials of the first kind with respect to the functional Φ_0. From §1 we know, however, that depending on the values ϕ_k we may not be able to satisfy (4.13) for all n, even when $\phi_k \neq 0$ $(\forall k)$. In fact, whenever the main diagonal of the Padé table of F at ∞ has more than one entry in common with a finite block, then there is a deficient FOP1 P_n on this diagonal and in this block, and for this P_n (4.13a) does not hold (cf. Fig. 1). On the other hand, if P_n belongs to the top row or the leftmost column of a block, then P_n and $\overline{P_n^{\sim}}$ are unique, and hence equal, up to a multiplicative constant. So far we have not yet imposed any scaling on $\mathbf{x}_n$ and $\mathbf{y}_n$. Even when uniqueness does not hold, we could always assume that $P_n = \overline{P_n^{\sim}}$ is an nth (monic) FOP1. However, in view of the practical implementation, we want to be able to choose scale factors for $\mathbf{x}_n$ and $\mathbf{y}_n$; we therefore write, instead of (4.10),

$$\mathbf{x}_n = P_n(\mathbf{A})\mathbf{x}_0\Gamma_n\,, \qquad \mathbf{y}_n = \overline{P_n}(\mathbf{A}^H)\mathbf{y}_0\bar{\Gamma}_n\,, \tag{4.14}$$

where Γ_n and $\bar{\Gamma}_n$ are the scale factors and P_n is still a monic FOP1 of degree n. ($\Gamma_0 := \bar{\Gamma}_0 := 1$.) $\bar{\Gamma}_n$ may, but need not, be the complex conjugate of Γ_n. The complex conjugate values will be denoted by $\overline{\Gamma_n}$ and $\overline{\bar{\Gamma}_n}$; but usually we choose real scale factors anyway. (An exception is the normalized BIORES algorithm, when applied to a complex system; see below.) We claim that even when (4.13) cannot be satisfied, (4.14) is in a certain sense the best possible choice for $\mathbf{x}_n$ and $\mathbf{y}_n$. In fact, from Theorem 2.1 it follows that (4.13) and hence also (4.11) hold then for k up to $\hat{n} - 1$, but not for $k = \hat{n}$, where

$$\hat{n} := n_j + n_{j+1} - n - 1 \quad \text{if } n_j \leq n < n_{j+1}\,. \tag{4.15}$$

Here $\{n_j\}_{j=0}^{J}$ $(J \leq \infty)$ still denotes the sequence of indices for which the FOP1 is regular (i.e., unique). For $n \neq n_j$ we may choose for P_n any FOP1 of degree n. All are equivalent with respect to the orthogonality properties, which is what matters here.

By (4.12) and (4.14), by the commutativity of $\mathbf{A}$ and $\mathbf{B}$, and by the linearity of Φ_0 we now obtain a general rule:

$$\langle \mathbf{y}_m, \mathbf{A}^k \mathbf{x}_n \rangle_{\mathbf{B}} = \overline{\bar{\Gamma}_m} \Gamma_n \langle \mathbf{y}_0, \mathbf{A}^k P_m(\mathbf{A}) P_n(\mathbf{A}) \mathbf{x}_0 \rangle_{\mathbf{B}} = \overline{\bar{\Gamma}_m} \Gamma_n \Phi_0(z^k P_m P_n) \tag{4.16}$$

for translating results from §§1–3 to the current situation.

After establishing this link to the formal orthogonal polynomials we can now easily draw conclusions from §§1–3. We only summarize the main results. First, concerning the orthogonality just discussed, we have the following theorem.

THEOREM 4.1. *Let $\mathbf{A} : \mathcal{H} \to \mathcal{H}$ be a bounded linear operator, and let $\mathbf{x}_0, \mathbf{y}_0 \in \mathcal{H}$ be two nonorthogonal initial vectors. Define two sequences $\{\mathbf{x}_n\}_{n=0}^{\infty}$, $\{\mathbf{y}_n\}_{n=0}^{\infty}$ by* (4.14), *where P_n is an nth* FOP1 *for the linear functional Φ_0 given by* (4.12), *and Γ_n and $\bar{\Gamma}_n$ are arbitrary nonzero scale factors. Let $\{n_j\}_{j=0}^{J}$ $(J \leq \infty)$ be the sequence of indices for which P_n is regular, i.e., for which the nth leading moment submatrix $\mathbf{M}_{0;n}$, defined in* (1.7), *is nonsingular. Then, for $n < n_J$ (or for all n if $J = \infty$),*

$$\mathbf{x}_n \perp_{\mathbf{B}} \mathcal{L}_{\hat{n}}\,, \qquad \mathbf{y}_n \perp_{\mathbf{B}} \mathcal{K}_{\hat{n}}\,, \tag{4.17a}$$

$$\langle \mathbf{y}_n, \mathbf{x}_{\hat{n}} \rangle_{\mathbf{B}} = \overline{\bar{\Gamma}_n} \Gamma_{\hat{n}} \delta_j \neq 0\,, \qquad \langle \mathbf{y}_{\hat{n}}, \mathbf{x}_n \rangle_{\mathbf{B}} = \overline{\bar{\Gamma}_{\hat{n}}} \Gamma_n \delta_j \neq 0\,, \tag{4.17b}$$

where n and $\hat{n}$ are linked by (4.15) *and where $\delta_j := \Phi_0(P_{\hat{n}} P_n)$. If $J < \infty$, then*

$$\mathbf{x}_n \perp_{\mathbf{B}} \mathcal{L}_m, \quad \mathbf{y}_n \perp_{\mathbf{B}} \mathcal{K}_m \quad (\forall n \geq n_J,\ \forall m \in \mathbb{N}^+). \tag{4.18}$$

Note that we do not exclude the possibility that $\mathbf{x}_n = \mathbf{0}$ or $\mathbf{y}_n = \mathbf{0}$ for $n \geq n_J$. On the other hand, it is conceivable and, as we will see, may in fact happen that (4.18) holds although $\mathbf{x}_n \neq \mathbf{0}$ and $\mathbf{y}_n \neq \mathbf{0}$.

Next we need to derive from the recurrence formulas for P_n, which were given in §2, an inductive algorithm for computing the sequences $\{\mathbf{x}_n\}$ and $\{\mathbf{y}_n\}$. We assume from now on that for those n where the FOP1 is not unique we choose as in Theorem 1.10 and §2, P_n according to (1.28), where $\{W_m\}$ is a fixed monic basis satisfying a three-term recurrence (2.10), so that (2.11) holds. This recurrence transforms then into

(4.19a)
$$\mathbf{x}_{n+1} = (\mathbf{A}\mathbf{x}_n - \mathbf{x}_n \alpha^W_{n-n_j})\,\gamma_{n+1,1} - \mathbf{x}_{n-1} \beta^W_{n-n_j} \gamma_{n+1,2}\,, \qquad n_j \leq n \leq n_{j+1} - 2\,,$$

(4.19b)
$$\mathbf{y}_{n+1} = (\mathbf{A}^H \mathbf{y}_n - \mathbf{y}_n \overline{\alpha^W_{n-n_j}})\,\bar{\gamma}_{n+1,1} - \mathbf{y}_{n-1} \overline{\beta^W_{n-n_j}}\, \bar{\gamma}_{n+1,2}\,, \qquad n_j \leq n \leq n_{j+1} - 2\,,$$

where

$$\gamma_{n,i} := \Gamma_n / \Gamma_{n-i}\,, \quad \bar{\gamma}_{n,i} := \bar{\Gamma}_n / \bar{\Gamma}_{n-i} \quad (n \in \mathbb{N}, i \in \mathbb{N}), \tag{4.20a}$$

so that in view of $\Gamma_0 = \bar{\Gamma}_0 = 1$,

$$\Gamma_n = \gamma_{1,1}\, \gamma_{2,1} \cdots \gamma_{n,1}\,, \qquad \gamma_{n,i} = \gamma_{n,1}\, \gamma_{n-1,1} \cdots \gamma_{n-i+1,1}\,, \tag{4.20b}$$

$$\bar{\Gamma}_n = \bar{\gamma}_{1,1}\, \bar{\gamma}_{2,1} \cdots \bar{\gamma}_{n,1}\,, \qquad \bar{\gamma}_{n,i} = \bar{\gamma}_{n,1}\, \bar{\gamma}_{n-1,1} \cdots \bar{\gamma}_{n-i+1,1}\,. \tag{4.20c}$$

(As mentioned in §2 the case $W_m(z) = z^m$ is included in (4.19), and, on the other hand, the general case where W_m need not satisfy a three-term recurrence or where

the recurrence depends on the block index j is obtained by a straightforward generalization; cf. (3.38).)

Since (2.11) is assumed, Corollary 2.8 and Theorem 2.9 apply. Replacing z by $\mathbf{A}$ or $\mathbf{A}^H$ in the recurrence formula (2.20), then applying both sides to $\mathbf{x}_0$ or $\mathbf{y}_0$, respectively, and taking care of the scale factors Γ_n and $\bar{\Gamma}_n$ yields for $j = 0, \cdots, J-1$:

$$\begin{aligned}\mathbf{x}_{n_{j+1}} &= [\mathbf{A}\mathbf{x}_{n_{j+1}-1} - \mathbf{x}_{n_{j+1}-1}(\alpha_{h_j-1,j} + \alpha^W_{h_j-1})]\,\gamma_{n_{j+1},1} \\ &\quad - \mathbf{x}_{n_{j+1}-2}(\alpha_{h_j-2,j} + \beta^W_{h_j-1})\,\gamma_{n_{j+1},2} - \mathbf{x}_{n_{j+1}-3}\alpha_{h_j-3,j}\,\gamma_{n_{j+1},3} - \cdots \\ &\quad - \mathbf{x}_{n_j}\alpha_{0,j}\,\gamma_{n_{j+1},h_j} - \mathbf{x}_{n_{j-1}}\beta_j\,\gamma_{n_{j+1},h_j+h_{j-1}}\,, \end{aligned} \tag{4.21a}$$

$$\begin{aligned}\mathbf{y}_{n_{j+1}} &= [\mathbf{A}^H\mathbf{y}_{n_{j+1}-1} - \mathbf{y}_{n_{j+1}-1}\overline{(\alpha_{h_j-1,j} + \alpha^W_{h_j-1})}]\,\bar{\gamma}_{n_{j+1},1} \\ &\quad - \mathbf{y}_{n_{j+1}-2}\overline{(\alpha_{h_j-2,j} + \beta^W_{h_j-1})}\,\bar{\gamma}_{n_{j+1},2} - \mathbf{y}_{n_{j+1}-3}\overline{\alpha_{h_j-3,j}}\,\bar{\gamma}_{n_{j+1},3} - \cdots \\ &\quad - \mathbf{y}_{n_j}\overline{\alpha_{0,j}}\,\bar{\gamma}_{n_{j+1},h_j} - \mathbf{y}_{n_{j-1}}\overline{\beta_j}\,\bar{\gamma}_{n_{j+1},h_j+h_{j-1}}\,. \end{aligned} \tag{4.21b}$$

In the case where $h_j = 1$ these two formulas simplify to:

$$\mathbf{x}_{n_j+1} = [\mathbf{A}\mathbf{x}_{n_j} - \mathbf{x}_{n_j}(\alpha_{0,j} + \alpha^W_0)]\,\gamma_{n_{j+1},1} - \mathbf{x}_{n_{j-1}}\beta_j\,\gamma_{n_{j+1},h_{j-1}+1}\,, \tag{4.22a}$$

$$\mathbf{y}_{n_j+1} = [\mathbf{A}^H\mathbf{y}_{n_j} - \mathbf{y}_{n_j}\overline{(\alpha_{0,j} + \alpha^W_0)}]\,\bar{\gamma}_{n_{j+1},1} - \mathbf{y}_{n_{j-1}}\overline{\beta_j}\,\bar{\gamma}_{n_{j+1},h_{j-1}+1}\,. \tag{4.22b}$$

Of course, the sequences $\{n_j\}$ and $\{h_j\}$ are not known in advance. But from (2.3) we recall that $h_j := n_{j+1} - n_j$ can be determined according to

$$h_j := 1 + \min\{k \in \mathbb{N};\, \langle \mathbf{y}_{n_j+k}, \mathbf{x}_{n_j}\rangle_{\mathbf{B}} \neq 0\}\,. \tag{4.23}$$

Note that $h_0 = 1$ since we have assumed that $\langle \mathbf{y}_0, \mathbf{x}_0\rangle_{\mathbf{B}} \neq 0$. (This is not at all important, however.)

For the determination of the coefficients in (4.21) we apply Theorem 2.9, which yields

$$\beta_j\,\gamma_{n_{j+1}-1,h_j+h_{j-1}-1}\langle \mathbf{y}_{n_j-1}, \mathbf{x}_{n_{j-1}}\rangle_{\mathbf{B}} = \langle \mathbf{y}_{n_j-1}, \mathbf{A}\mathbf{x}_{n_{j+1}-1}\rangle_{\mathbf{B}}\,, \tag{4.24a}$$

$$\begin{aligned}&\sum_{s=1}^{k+1} \alpha_{h_j-s,j}\,\gamma_{n_{j+1}-1,s-1}\langle \mathbf{y}_{n_j+k}, \mathbf{x}_{n_{j+1}-s}\rangle_{\mathbf{B}} \\ &\quad = \langle \mathbf{y}_{n_j+k}, (\mathbf{A}\mathbf{x}_{n_{j+1}-1} - \mathbf{x}_{n_{j+1}-1}\alpha^W_{h_j-1} - \mathbf{x}_{n_{j+1}-2}\beta^W_{h_j-1}\,\gamma_{n_{j+1}-1,1})\rangle_{\mathbf{B}}\,, \\ &\qquad\qquad k = 0, 1, \cdots, h_j - 1\,. \end{aligned} \tag{4.24b}$$

As in Theorem 2.9, these are a single linear equation (hence, an explicit formula) for the nonzero constant β_j and a lower triangular system (hence, a recursive formula) for the coefficients $\alpha_{s,j}$ of the polynomial $a_j(z)$. If $h_j = 1$, the latter system becomes a single equation, so that we have two explicit formulas:

$$\beta_j := \frac{1}{\gamma_{n_j,h_{j-1}}}\,\frac{\langle \mathbf{y}_{n_j-1}, \mathbf{A}\mathbf{x}_{n_j}\rangle_{\mathbf{B}}}{\langle \mathbf{y}_{n_j-1}, \mathbf{x}_{n_{j-1}}\rangle_{\mathbf{B}}}\,, \qquad \alpha_{0,j} + \alpha^W_0 := \frac{\langle \mathbf{y}_{n_j}, \mathbf{A}\mathbf{x}_{n_j}\rangle_{\mathbf{B}}}{\langle \mathbf{y}_{n_j}, \mathbf{x}_{n_j}\rangle_{\mathbf{B}}}\,. \tag{4.25}$$

These are the formulas of the classical Lanczos BO algorithm.

This completes the derivation of the algorithm.

ALGORITHM 1 (nongeneric BO algorithm). *The vector sequences* $\{\mathbf{x}_n\}_{n=0}^{\infty}$ *and* $\{\mathbf{y}_n\}_{n=0}^{\infty}$ *of Theorem* 4.1 *can be constructed by an inductive process, which for* $j = 0, 1, \cdots, J\,(\leq \infty)$ *consists of:*

(i) $\{\mathbf{x}_{n_j+k}\}_{k=1}^{h_j-1}, \{\mathbf{y}_{n_j+k}\}_{k=1}^{h_j-1}$, *and* h_j *are defined by concurrently executing* (4.19a), (4.19b), *and* (4.23); *if* $h_j = \infty$, *then* $j = J$; *in particular, if* $\mathbf{x}_{n_j} = \mathbf{0}$ *or* $\mathbf{y}_{n_j} = \mathbf{0}$, *then* $h_j = \infty$, $j = J$ *and* $\mathbf{x}_{n_J+k} = \mathbf{0}$ $(\forall k \geq 0)$ *or* $\mathbf{y}_{n_J+k} = \mathbf{0}$ $(\forall k \geq 0)$, *respectively, and the algorithm is said to terminate;*

(ii) *Once* h_j *has been determined, the nonzero constant* β_j *is given by* (4.24a) *and the coefficients* $\{\alpha_{s,j}\}_{s=1}^{h_j-1}$ *are obtained by solving the lower triangular linear system* (4.24b);

(iii) $\mathbf{x}_{n_{j+1}}$ *and* $\mathbf{y}_{n_{j+1}}$ *are then given by* (4.21a, b).

The nonvanishing constants $\gamma_{n,1}$, $\bar{\gamma}_{n,1}$ *and the recurrence coefficients* α_m^W *and* β_m^W *in* (4.19a, b) *can be chosen freely.* Γ_n *and* $\gamma_{n,i}$ *are linked to* $\gamma_{n,1}$ *by* (4.20b), $\bar{\Gamma}_n$ *and* $\bar{\gamma}_{n,i}$ *are linked to* $\bar{\gamma}_{n,1}$ *by* (4.20c).

If $h_j = \infty$ (and thus $j = J < \infty$) and $\mathbf{x}_{n_j} \neq \mathbf{0}$, $\mathbf{y}_{n_j} \neq \mathbf{0}$, then we conclude from the definitions of h_j and $\mathbf{y}_{n_j+k}$ that $\langle \mathbf{y}_{n_j}, \mathbf{A}^k \mathbf{x}_{n_j} \rangle_{\mathbf{B}} = 0$ $(\forall k \in \mathbb{N})$. Following Taylor [45] and Parlett, Taylor, and Liu [38] we call this an *incurable breakdown.* Theoretically, there is no reason to continue the computation, but in order to detect an incurable breakdown we have to compute $\mathbf{x}_{n_j+k}$ or $\mathbf{y}_{n_j+k}$ or $\mathbf{A}^k \mathbf{x}_{n_j}$ until $n_j + k$ is equal to the rank of $\mathbf{A}$, or, if the latter is unknown, equal to any known upper bound for it.

In the classical (generic) Lanczos algorithm a so-called *serious breakdown* occurs whenever the denominator $\langle \mathbf{y}_{n_j}, \mathbf{x}_{n_j} \rangle_{\mathbf{B}} = 0$ in the formula (4.25) for $\alpha_{0,j}$ vanishes, but $\mathbf{x}_{n_j} \neq \mathbf{0}$ and $\mathbf{y}_{n_j} \neq \mathbf{0}$. If, then, $h_j < \infty$, Algorithm 1 does not break down, and therefore the breakdown is called *curable.* It corresponds to a finite square block in the Padé table of $F(1/z)$, and the cure of Algorithm 1 consists of jumping to the next block. In contrast, an incurable (serious) breakdown or a termination mean that the infinite block has been reached. We will return to this relation between the Padé table and Algorithm 1 later.

Of course, the formulas of Algorithm 1 become simpler if we choose $\gamma_{n,1} := \bar{\gamma}_{n,1} := 1$ $(\forall n)$, so that $\Gamma_n := \gamma_{n,i} := \bar{\Gamma}_n := \bar{\gamma}_{n,i} := 1$ $(\forall n, \forall i)$, but it has been known for a long time that in practice this may cause overflow or underflow [39], and Taylor [45] pointed out that it does not suffice to have a common sequence of scale factors for both vector sequences.

To simplify matters, in practice we can redefine the formal inner product by $\langle \mathbf{y}, \mathbf{x} \rangle_{\mathbf{B}} := \mathbf{y}^T \mathbf{B} \mathbf{x}$, and then replace $\mathbf{A}^H$ by $\mathbf{A}^T$ and delete the conjugation bars in (4.21b) and (4.22b). However, from the theoretical point of view, it is nicer to use our formulation with $\mathbf{A}^H$ and $\mathbf{y}^H \mathbf{B} \mathbf{x}$, which reduces to the standard inner product if $\mathbf{B} = \mathbf{I}$. Furthermore, we may in practice redefine the coefficients $\alpha_{i,j}$, β_j, α_m^W, and β_m^W, so that the corresponding factors $\gamma_{n,i}$ and $\bar{\gamma}_{n,i}$ are incorporated. From the above formulas or from the corresponding matrix relations below we find expressions for these redefined coefficients, which are the elements of the matrices $\mathbf{H}_\Gamma$ and $\overline{\mathbf{H}}_{\bar{\Gamma}}$ defined below in (4.28). In [15] we have chosen this normalization, which is natural from the point of view of the Lanczos method, while here the basic recurrence coefficients are those of the (monic) FOP1s.

Next, we want to apply the matrix results of §3 to the particular situation we have now. In (3.11), $z\mathbf{p}(z) = \mathbf{p}(z)\mathbf{H}$, and in its finite-dimensional analogue (3.13), we must on the one hand substitute $\mathbf{A}$ for z and apply both sides to $\mathbf{x}_0$, and on the other hand, substitute $\mathbf{A}^T$ for z, take the complex conjugate, and apply both sides to $\mathbf{y}_0$. Finally, we must take into account the scale factors Γ_n and $\bar{\Gamma}_n$. For translating the orthogonality result (3.22), $\Phi_0(\mathbf{p}^T \mathbf{p}) = \mathbf{D}$, we only need to recall the rule (4.16). In summary we get the following theorem.

THEOREM 4.2. *Under the assumptions of Theorem* 4.1 *let*

$$\mathbf{X} := [\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \cdots], \qquad \mathbf{Y} := [\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \cdots], \tag{4.26}$$

$$\mathbf{\Gamma} := \operatorname{diag}[\Gamma_0, \Gamma_1, \Gamma_2, \cdots], \qquad \bar{\mathbf{\Gamma}} := \operatorname{diag}[\bar{\Gamma}_0, \bar{\Gamma}_1, \bar{\Gamma}_2, \cdots]; \tag{4.27}$$

let $\mathbf{H}$ *be the block tridiagonal matrix* (3.6a, b) *and* $\mathbf{D}$ *be the block diagonal matrix* (3.23). *Write the infinite matrix with* (m,n)*-element* $\langle \mathbf{y}_m, \mathbf{x}_n \rangle_{\mathbf{B}}$ *formally as* $\mathbf{Y}^H\mathbf{B}\mathbf{X}$, *and define diagonally scaled versions of* $\mathbf{H}$ *and* $\mathbf{D}$ *by*

$$\mathbf{H}_\Gamma := \mathbf{\Gamma}^{-1}\mathbf{H}\mathbf{\Gamma}, \quad \overline{\mathbf{H}}_{\bar{\Gamma}} := \bar{\mathbf{\Gamma}}^{-1}\overline{\mathbf{H}}\bar{\mathbf{\Gamma}}, \quad \mathbf{D}_\Gamma := \overline{\bar{\mathbf{\Gamma}}}\mathbf{D}\mathbf{\Gamma}. \tag{4.28}$$

Then

$$\mathbf{A}\mathbf{X} = \mathbf{X}\mathbf{H}_\Gamma, \qquad \mathbf{A}^H\mathbf{Y} = \mathbf{Y}\overline{\mathbf{H}}_{\bar{\Gamma}}, \tag{4.29}$$

and

$$\mathbf{Y}^H\mathbf{B}\mathbf{X} = \mathbf{D}_\Gamma. \tag{4.30}$$

Moreover, if for $n \in \mathbb{N}$,

$$\mathbf{X}_n := [\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_n], \qquad \mathbf{Y}_n := [\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_n], \tag{4.31}$$

if $\mathbf{H}_{\Gamma;n}$ *and* $\overline{\mathbf{H}}_{\bar{\Gamma};n}$ *denote the principal submatrix of order* $n+1$ *of* $\mathbf{H}_\Gamma$ *and* $\overline{\mathbf{H}}_{\bar{\Gamma}}$, *respectively, if* $\gamma_{\Gamma;n}$ *and* $\overline{\gamma}_{\bar{\Gamma};n}$ *are the* $(n+1,n)$ *elements of these two infinite matrices,*[8] *and if* $\mathbf{1}_n^T$ *is the last row of the identity matrix of the same order, then*

$$\mathbf{A}\mathbf{X}_n = \mathbf{X}_n\mathbf{H}_{\Gamma;n} + \mathbf{x}_{n+1}\gamma_{\Gamma;n}\mathbf{1}_n^T, \qquad \mathbf{A}^H\mathbf{Y}_n = \mathbf{Y}_n\overline{\mathbf{H}}_{\bar{\Gamma};n} + \mathbf{y}_{n+1}\overline{\gamma}_{\bar{\Gamma};n}\mathbf{1}_n^T. \tag{4.32}$$

Of course, $\mathbf{H}$, $\mathbf{H}_\Gamma$, and $\overline{\overline{\mathbf{H}}_{\bar{\Gamma}}} = \mathbf{H}_{\bar{\Gamma}}$ are diagonally similar to each other.

By noting that all finite diagonal blocks of $\mathbf{D}_\Gamma$ are nonsingular and, hence, $\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_{n_j}$ are linearly independent for every $j < J$, we finally come to the following conclusion, which exhibits the theoretical applicability of the BO process to eigenvalue computations.

COROLLARY 4.3. *If* $\mathbf{x}_{n_j} = \mathbf{0}$ (*and thus* $j = J$), *then the columns* $\mathbf{x}_0, \mathbf{x}_1, \cdots, \mathbf{x}_{n_j-1}$ *of* $\mathbf{X}_{n_j-1}$ *are linearly independent and span an invariant subspace of* $\mathbf{A}$; *hence,* $\mathbf{H}_{\Gamma;n_j-1}$, *which is nonderogatory, has the same spectrum as the restriction of* $\mathbf{A}$ *to this invariant subspace.*

Likewise, if $\mathbf{y}_{n_j} = \mathbf{0}$ (*and thus* $j = J$), *then the columns* $\mathbf{y}_0, \mathbf{y}_1, \cdots, \mathbf{y}_{n_j-1}$ *of* $\mathbf{Y}_{n_j-1}$ *are linearly independent and span an invariant subspace of* $\mathbf{A}^H$; *hence, the spectrum of* $\mathbf{H}_{\Gamma;n_j-1}$ *is complex conjugate to the one of the restriction of* $\mathbf{A}^H$ *to this invariant subspace.*

Theoretically, when this algorithm is applied, the aim is thus to terminate it either with $\mathbf{x}_{n_j} = \mathbf{0}$ or $\mathbf{y}_{n_j} = \mathbf{0}$. Although in practice roundoff spoils everything, the theoretical question is whether this termination can be guaranteed. Some insight is gained again by looking at the related Padé approximation problem. It must be emphasized here that the whole outcome of the Lanczos process (except for roundoff effects) only depends on $\mathbf{A}$ and the initial vectors $\mathbf{x}_0$ and $\mathbf{y}_0$. The influence of the

[8] The top left element of these matrices is considered as their (0,0) element.

initial vectors is twofold: On the one hand, they appear in (4.14) as the vectors on which $P_n(\mathbf{A})$ and $\overline{P_n}(\mathbf{A}^H)$ act; on the other hand, they also influence the set of FOP1s P_n; these FOP1s are, however, completely determined by the functions F of (4.5), or equivalently, by the Laurent coefficients of F, i.e., by the moments ϕ_k. Also the block tridiagonal Gragg matrix $\mathbf{H}$ depends only on the FOP1s; hence, all the important information is really contained in the function F. The process can provide exactly the eigenvalue information contained in F. If an eigenvalue leaves no trace in F, we cannot find it, neither by the classical Lanczos algorithm nor by the nongeneric BO algorithm; in particular, as was already pointed out in Lanczos' original work [24], there is no chance to detect the geometric multiplicity of an eigenvalue. (For the eigenvector information, the situation is more complicated, but the matrices $\mathbf{X}_n$ and $\mathbf{Y}_n$ that are implicitly generated are in practice not used for the eigenproblem; however, the vectors $\mathbf{x}_n$ play a role in the application of the process to solving linear systems of equations.)

For simplicity, let us assume for the moment that $N := \dim \mathcal{H} < \infty$ and that $\mathbf{A}$ and $\mathbf{B}$ are diagonalizable $N \times N$ matrices, which we still assume to commute, so that there exists a common nonsingular (but not necessarily unitary) $N \times N$ matrix $\mathbf{U}$ of eigenvectors such that

$$\mathbf{AU} = \mathbf{UD}_\lambda\,, \qquad \mathbf{BU} = \mathbf{UD}_\kappa\,, \tag{4.33}$$

where $\mathbf{D}_\lambda$ and $\mathbf{D}_\kappa$ are diagonal matrices containing the eigenvalues $\lambda_1, \cdots, \lambda_N$ of $\mathbf{A}$ and the eigenvalues $\kappa_1, \cdots, \kappa_N$ of $\mathbf{B}$, respectively. Then $\mathbf{A}^H\mathbf{U}^{-H} = \mathbf{U}^{-H}\mathbf{D}_\lambda^H$, i.e., the columns of $\mathbf{U}^{-H}$ are a set of eigenvectors of $\mathbf{A}^H$. Set

$$\mathbf{U} =: [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_N]\,, \qquad \mathbf{U}^{-H} =: \mathbf{V} =: [\mathbf{v}_1, \mathbf{v}_2, \cdots, \mathbf{v}_N]\,, \tag{4.34}$$

and represent $\mathbf{x}_0$ in the basis $\{\mathbf{u}_k\}$, $\mathbf{y}_0$ in the basis $\{\mathbf{v}_k\}$:

$$\mathbf{x}_0 =: \sum_{k=1}^{N} \mathbf{u}_k \xi_k =: \mathbf{U}\boldsymbol{\xi}\,, \qquad \mathbf{y}_0 =: \sum_{k=1}^{N} \mathbf{v}_k \eta_k =: \mathbf{V}\boldsymbol{\eta}\,, \tag{4.35}$$

where $\boldsymbol{\xi} := [\xi_1, \xi_2, \cdots, \xi_N]^T$, $\boldsymbol{\eta} := [\eta_1, \eta_2, \cdots, \eta_N]^T$. Then, in view of (4.14),

$$\mathbf{x}_n = \mathbf{U}P_n(\mathbf{D}_\lambda)\boldsymbol{\xi}\Gamma_n = \sum_{k=1}^{N} \mathbf{u}_k \Gamma_n P_n(\lambda_k)\xi_k\,, \tag{4.36a}$$

$$\mathbf{y}_n = \mathbf{V}\overline{P_n}(\mathbf{D}_\lambda^H)\boldsymbol{\eta}\bar{\Gamma}_n = \sum_{k=1}^{N} \mathbf{v}_k \bar{\Gamma}_n \overline{P_n(\lambda_k)}\eta_k\,, \tag{4.36b}$$

and, since $\mathbf{V}^H\mathbf{U} = \mathbf{I}$,

$$\langle \mathbf{y}_m, \mathbf{x}_n \rangle_{\mathbf{B}} = \overline{\bar{\Gamma}_m}\boldsymbol{\eta}^H P_m(\mathbf{D}_\lambda)\mathbf{D}_\kappa P_n(\mathbf{D}_\lambda)\boldsymbol{\xi}\bar{\Gamma}_n = \overline{\bar{\Gamma}_m}\Gamma_n \sum_{k=1}^{N} P_m(\lambda_k)P_n(\lambda_k)\kappa_k\xi_k\overline{\eta}_k\,, \tag{4.37}$$

which shows that the orthogonality of $\mathbf{y}_m$ and $\mathbf{x}_n$ is reflected by a formal orthogonality of P_m and P_n with respect to a discrete measure with the masses $\kappa_k\xi_k\overline{\eta}_k$ at the N points λ_k (which need not be distinct). In particular, the initial assumption that $\langle \mathbf{y}_0, \mathbf{x}_0 \rangle_{\mathbf{B}} \neq 0$ is equivalent to

$$\sum_{k=1}^{N} \kappa_k\xi_k\overline{\eta}_k \neq 0\,, \tag{4.38}$$

and the serious breakdown condition $\langle \mathbf{y}_n, \mathbf{x}_n \rangle_{\mathbf{B}} = 0$ can be expressed as

$$\sum_{k=1}^{N} P_n^2(\lambda_k)\kappa_k \xi_k \overline{\eta}_k = 0\,. \tag{4.39}$$

Moreover, the moments are given by

$$\phi_k = \langle \mathbf{y}_0, \mathbf{A}^k \mathbf{x}_0 \rangle_{\mathbf{B}} = \boldsymbol{\eta}^H \mathbf{D}_\kappa \mathbf{D}_\lambda^k \boldsymbol{\xi} = \sum_{k=1}^{N} \kappa_k \lambda_k^k \xi_k \overline{\eta}_k\,, \tag{4.40}$$

and the function F of (4.5) can be written as

$$F(\zeta) = \langle \mathbf{y}_0, (\zeta \mathbf{I} - \mathbf{A})^{-1} \mathbf{x}_0 \rangle_{\mathbf{B}} = \boldsymbol{\eta}^H \mathbf{D}_\kappa (\zeta \mathbf{I} - \mathbf{D}_\lambda)^{-1} \boldsymbol{\xi} = \sum_{k=1}^{N} \frac{\kappa_k \xi_k \overline{\eta}_k}{\zeta - \lambda_k}\,. \tag{4.41}$$

Note that if $\mathbf{A}$ is hermitian and $\mathbf{B}$ hermitian positive definite, and if we choose $\mathbf{y}_0 = \mathbf{x}_0$ (which one then normally does), then $\mathbf{v}_k = \mathbf{u}_k$ $(\forall k)$, $\eta_k = \xi_k$ $(\forall k)$, $\lambda_k \in \mathbb{R}$ $(\forall k)$, $\kappa_k > 0$ $(\forall k)$, and $\overline{P}_n = P_n$ $(\forall n)$, so that in (4.39), $P_n^2(\lambda_k)\kappa_k \xi_k \overline{\eta}_k = |P_n(\lambda_k)|^2 \kappa_k |\xi_k|^2$, and thus $\langle \mathbf{y}_n, \mathbf{x}_n \rangle_{\mathbf{B}} = 0$ implies $\mathbf{x}_n = \mathbf{0}$. (Actually, $\mathbf{y}_n$ is then equal to $\mathbf{x}_n$ up to scaling.) Consequently, there are no serious breakdowns in this case.

Formula (4.41) makes it clear that the denominator degree of F is equal to the actual number of terms in the sum after all terms belonging to the same pole have been taken together in the case of multiple eigenvalues. (Of course, all resulting terms with residual zero, i.e., vanishing numerator, are deleted.) Again let ν_F $(\leq N)$ denote this actual denominator degree of F. In view of $F(\infty) = 0$ and $\phi_0 = \langle \mathbf{y}_0, \mathbf{x}_0 \rangle_{\mathbf{B}} \neq 0$, F has exact type $(\nu_F - 1, \nu_F)$. If $\mathbf{A}$ is nonsingular, $\zeta = 0$ is not a pole of F. Consequently, the function $z \mapsto F(1/z) = zf(z)$ of $z = 1/\zeta$ has exact type (m, ν_F) with $m \leq \nu_F$. From part (ii) of the Block Structure Theorem (Corollary 1.6) we know that the Padé table of $F(1/z)$ has an infinite block whose first column lies at $n = \nu_F$ and whose first row is row m. If $\mathbf{A}$ is singular, F may have a pole at $\zeta = 0$, which for F(1/z) becomes a pole at ∞, and thus the denominator degree of $F(1/z) = zf(z)$ is then smaller than ν_F by the order of that pole, but the numerator degree is exactly ν_F. Hence, in this case the infinite block starts in row ν_F and in some column $n < \nu_F$. In both cases the main diagonal enters the block at column ν_F. Therefore, by Theorem 1.8, $\nu_F = n_J$ holds, and $P_{\nu_F} = P_{n_J}$ is the last regular FOP1 for the functional Φ_0. Hence, from now on, we can replace ν_F by n_J.

From (4.36) it is seen that $\mathbf{x}_n = \mathbf{0}$ requires that $P_n(\lambda_k)\xi_k = 0$ $(k = 1, \cdots, N)$, which just reflects the well-known fact that (under the assumption of diagonalizability) the so-called *grade of* $\mathbf{x}_0$ *with respect to* $\mathbf{A}$, i.e., the dimension of the maximum Krylov subspace generated by $\mathbf{A}$ and $\mathbf{x}_0$, is given by the actual number of components of $\mathbf{x}_0$ belonging to distinct eigenvalues, and that P_n is the minimal polynomial for the restriction of $\mathbf{A}$ to this subspace if $\mathbf{x}_n = \mathbf{0}$. Clearly it may happen that this grade is larger than n_J since in (4.41) the products $\xi_k \overline{\eta}_k$ appear, and we cannot guarantee that $\xi_k \neq 0$ implies $\eta_k \neq 0$. (Moreover, it is possible that, say, $\lambda_1 = \lambda_2 \neq \lambda_k$ $(k \geq 3)$ and $\kappa_1 \xi_1 \overline{\eta}_1 + \kappa_2 \xi_2 \overline{\eta}_2 = 0$, so that the two first terms cancel in (4.38)–(4.41).)

These arguments carry over with some minor modifications to the case where $\mathbf{A}$ is not diagonalizable. In fact, $\mathbf{x}_{n_j} = \mathbf{0}$ still means that $n_j = n_J$ is the grade of $\mathbf{x}_0$ with respect to $\mathbf{A}$ and implies that the main diagonal enters the infinite block of the Padé table of $F(1/z)$ in column n_J, which in turn means that $F(1/z)$ has exact

order n_J, and thus $F(\zeta)$, which has a zero at ∞, has exact denominator degree n_J. On the other hand, if the grade of $\mathbf{x}_0$ with respect to $\mathbf{A}$ is, say, n, then necessarily $\mathbf{x}_n = P_n(\mathbf{A})\mathbf{x}_0\Gamma_n = \mathbf{0}$ and therefore $n \geq n_J = \nu_F$. Hence, equality holds here if and only if $\mathbf{x}_{n_J} = \mathbf{0}$.

THEOREM 4.4. *For a finite rank operator* $\mathbf{A}$, *Algorithm* 1 *either terminates at step* n_J *or encounters an incurable breakdown there. The index* n_J *is bounded by the rank of* $\mathbf{A}$ *and is equal to the exact denominator degree* ν_F *of the rational function* F *defined in* (4.5).

Algorithm 1 *terminates with* $\mathbf{x}_{n_J} = \mathbf{0}$ (*or with* $\mathbf{y}_{n_J} = \mathbf{0}$) *if and only if the grade of* $\mathbf{x}_0$ *with respect to* $\mathbf{A}$ (*or the grade of* $\mathbf{y}_0$ *with respect to* $\mathbf{A}^H$, *respectively*) *is equal to* $\nu_F = n_J$.

By a classical result of Kronecker [9, Chap. XV, §10] ν_F is also equal to the rank of the infinite moment matrix $\mathbf{M}$ of (3.25) (with $l = 0$).

What generally matters for the grade of $\mathbf{x}_0$ with respect to $\mathbf{A}$ in addition to the distinct eigenvalues represented by components of $\mathbf{x}_0$ is the maximum grade of the components of $\mathbf{x}_0$ that belong to a particular eigenvalue, because it determines the multiplicity of this eigenvalue in the minimal polynomial. In fact, when it comes to details, the case where $\mathbf{A}$ is not diagonalizable requires some notationally awkward modifications. In (4.33) $\mathbf{D}_\lambda$ and $\mathbf{D}_\kappa$ then denote the (upper bidiagonal) Jordan canonical forms of $\mathbf{A}$ and $\mathbf{B}$. The relations (4.34) and (4.35) remain valid, and in (4.36a, b), (4.37), (4.40), and (4.41) only, the sums have to be modified. For future reference, we want to give these sums for (4.40) and (4.41), at least under the assumption that $\mathbf{B} = \mathbf{D}_\kappa = \mathbf{I}$. For eigenvalue computations this is by far the most important case. Other particular cases, like $\mathbf{B} = \mathbf{A}$ and $\mathbf{B} = \mathbf{A}^{-1}$, which play a role for the solution of linear systems of equations, could be treated analogously.

If $\mathbf{S} := \mathbf{S}_N$ denotes the $N \times N$ downshift matrix (cf. (3.16)) and if for an instant $\mathbf{D}_\lambda$ is assumed to consist of just one large Jordan block, so that $\mathbf{D}_\lambda = \lambda\mathbf{I} + \mathbf{S}^T$, then

$$\begin{aligned} \mathbf{D}_\lambda^k &= (\lambda\mathbf{I} + \mathbf{S}^T)^k = \sum_{i=0}^{k} \binom{k}{i} \lambda^{k-i} (\mathbf{S}^T)^i , \\ \boldsymbol{\eta}^H \mathbf{D}_\lambda^k \boldsymbol{\xi} &= \sum_{j=1}^{N} \sum_{i=0}^{\min\{N-j,k\}} \binom{k}{i} \lambda^{k-i} \xi_{j+i} \overline{\eta}_j , \end{aligned} \tag{4.42}$$

$$\begin{aligned} (\zeta\mathbf{I} - \mathbf{D}_\lambda)^{-1} &= ((\zeta - \lambda)\mathbf{I} - \mathbf{S}^T)^{-1} = \sum_{i=0}^{N-1} (\zeta - \lambda)^{-i-1} (\mathbf{S}^T)^i , \\ \boldsymbol{\eta}^H (\zeta\mathbf{I} - \mathbf{D}_\lambda)^{-1} \boldsymbol{\xi} &= \sum_{j=1}^{N} \sum_{i=0}^{N-j} \frac{\xi_{j+i}\overline{\eta}_j}{(\zeta - \lambda)^{i+1}} . \end{aligned} \tag{4.43}$$

For an arbitrary Jordan block structure of $\mathbf{A}$ we therefore get

$$\phi_k = \langle \mathbf{y}_0, \mathbf{A}^k \mathbf{x}_0 \rangle_{\mathbf{I}} = \boldsymbol{\eta}^H \mathbf{D}_\lambda^k \boldsymbol{\xi} = \sum_{j=1}^{N} \sum_{i=0}^{\min\{\partial(j),k\}} \binom{k}{i} \lambda^{k-i} \xi_{j+i} \overline{\eta}_j \tag{4.44}$$

and

$$F(\zeta) = \langle \mathbf{y}_0, (\zeta\mathbf{I} - \mathbf{A})^{-1} \mathbf{x}_0 \rangle_{\mathbf{I}} = \boldsymbol{\eta}^H (\zeta\mathbf{I} - \mathbf{D}_\lambda)^{-1} \boldsymbol{\xi} = \sum_{j=1}^{N} \sum_{i=0}^{\partial(j)} \frac{\xi_{j+i}\overline{\eta}_j}{(\zeta - \lambda_j)^{i+1}} , \tag{4.45}$$

where $\partial(j)$ is equal to the absolute difference of j and the column index of the last column that belongs in $\mathbf{D}_\lambda$ to the same Jordan subblock as column j.

It remains to investigate the *incurable* breakdown, where $\mathbf{x}_{n_J} \neq \mathbf{0}$, $\mathbf{y}_{n_J} \neq \mathbf{0}$, but $\langle \mathbf{y}_{n_J+k}, \mathbf{x}_{n_J} \rangle_{\mathbf{B}} = 0$ $(\forall k \in \mathbb{N})$. It was discussed in detail by Taylor [45], who proved the following, at first sight surprising, result, which was called the *Mismatch Theorem* in [38], and which is reexamined and brought into relation with the realization problem of system theory in Parlett [35].[9] The main message of the theorem is that although we no longer get a basis of an invariant subspace of $\mathbf{A}$, we do get with P_{n_J} the corresponding minimal polynomial and with $\mathbf{H}_{\Gamma;n_J-1}$ a block tridiagonal matrix whose characteristic polynomial is P_{n_J}.

THEOREM 4.5 (Mismatch Theorem). *Assume* $\mathbf{B} = \mathbf{I}$. *If incurable breakdown occurs, i.e., if*

$$\langle \mathbf{y}_{n_J+k}, \mathbf{x}_{n_J} \rangle_{\mathbf{I}} = 0 \quad (\forall k \in \mathbb{N}), \quad \mathbf{x}_{n_J} \neq \mathbf{0}, \quad \mathbf{y}_{n_J} \neq \mathbf{0}, \tag{4.46}$$

the FOP1 P_{n_J} *is the characteristic polynomial of the* n_J*th leading principal submatrix* $\mathbf{H}_{\Gamma;n_J-1}$ *of* $\mathbf{H}_\Gamma$ *and the minimal polynomial of an invariant subspace of* $\mathbf{A}$.

Proof. Consider the partial fraction decomposition of the function F. In the sum in (4.45) the terms with equal eigenvalues λ_j and equal power of $\zeta - \lambda_j$ have to be brought together:

$$F(\zeta) = \sum_{l=1}^{L} \sum_{i=1}^{\iota(l)} \frac{\tau_{l,i}}{(\zeta - \lambda_{j(l)})^i}. \tag{4.47}$$

Here the outer loop goes over the distinct poles $\lambda_{j(l)}$, $l = 1, \cdots, L$, of F, and the inner loop takes care of the order of the pole. This order is at most equal to the order $\iota(l)$ of the largest Jordan subblock for the particular eigenvalue; to stay in accordance with (4.45), we assume in (4.47) that this largest subblock is the one with column indices $j(l), \cdots, j(l+1) - 1$, i.e., we assume that for each distinct eigenvalue one largest subblock is among the first L subblocks of the Jordan form. The order of the subblock is $\iota(l) = j(l+1) - j(l)$, but the order of the pole may be smaller, namely,

$$\bar{\iota}(l) := \max\{i; \tau_{l,i} \neq 0\}, \tag{4.48}$$

and the order of F is therefore

$$n_J = \sum_{l=1}^{L} \bar{\iota}(l). \tag{4.49}$$

The theorem follows from Corollary 4.3 if we can find another pair of starting vectors $\mathbf{x}_0$, $\mathbf{y}_0$ which yield the same function F, but have the property that the grade of $\mathbf{x}_0$ with respect to $\mathbf{A}$ or the one of $\mathbf{y}_0$ with respect to $\mathbf{A}^H$ is equal to n_J, so that with these new starting vectors Algorithm 1 terminates with $\mathbf{x}_{n_J} = \mathbf{0}$ or $\mathbf{y}_{n_J} = \mathbf{0}$, respectively.

By comparing (4.47) with (4.45) it is seen that we may choose $\mathbf{x}_0$ and $\mathbf{y}_0$ such that in the representations (4.35)

$$\xi_k = 0, \quad \eta_k = 0 \quad \text{if } k \geq j(L+1), \tag{4.50}$$

[9] These paragraphs on the Mismatch Theorem were added in revision to provide a bridge to Parlett's paper and to give another interpretation and derivation of the theorem.

and

$$\sum_{j=j(l)}^{j(l)+\iota(l)-i} \xi_{j+i-1}\overline{\eta_j} = \tau_{l,i}\,, \quad i = 1,\cdots,\iota(l),\ \ l = 1,\cdots,L. \tag{4.51}$$

(If there are just L subblocks in $\mathbf{D}_\lambda$, we set $j(L+1) := N+1$.) Condition (4.50) means that it suffices to have components belonging to one maximally large subblock of each distinct eigenvalue, and (4.51) yields for each such subblock a set of equations for the corresponding components of $\mathbf{x}_0$ and $\mathbf{y}_0$. If the coordinates ξ_j were given, each of the L systems in (4.51) would be a linear left upper triangular Hankel system for the coordinates $\overline{\eta_j}$ that belong to the same subblock, and vice versa. Since neither $\mathbf{x}_0$ nor $\mathbf{y}_0$ is given, we can, for example, choose a solution with

$$\xi_{j(l)+i-1} = 0, \quad \eta_{j(l)+i-1} = 0 \quad \text{if } \bar{\iota}(l) < i \le \iota(l) \quad (l = 1,\cdots,L), \tag{4.52}$$

$$\xi_{j(l)+\bar{\iota}(l)-1} \neq 0 \qquad (l = 1,\cdots,L), \tag{4.53}$$

and with arbitrary $\xi_{j(l)},\cdots,\xi_{j(l)+\bar{\iota}(l)-2}$, so that for each l a regular left upper triangular Hankel subsystem remains to solve for $\overline{\eta}_{j(l)},\cdots,\overline{\eta}_{j(l)+\bar{\iota}(l)-1}$. From the last equation of this reduced system we obtain $\overline{\eta}_{j(l)} = \tau_{l,\bar{\iota}(l)}/\xi_{j(l)+\bar{\iota}(l)-1} \neq 0$. Forming $\mathbf{D}_\lambda^k\boldsymbol{\xi}$ and $\boldsymbol{\eta}\mathbf{D}_\lambda^k$ we see that $\mathbf{x}_0$ has grade n_J with respect to $\mathbf{A}$, so that $\mathbf{x}_{n_J} = \mathbf{0}$, while $\mathbf{y}_0$ has maximum grade with respect to $\mathbf{A}^H$, so that its grade is equal to the degree of the minimal polynomial of $\mathbf{A}$. □

There are in general many other ways to choose $\mathbf{x}_0$ and $\mathbf{y}_0$; in particular, condition (4.50) is not necessary. In the terminology of system theory the construction in the above proof yields just one possible *minimal realization* (cf. [23], [14], [35], [5]). Uniqueness holds when $n_J = N$, but for this it is necessary that $\mathbf{A}$ is itself nonderogatory. Our argumentation should make it clear that even when incurable breakdown occurs, the polynomials P_n constructed implicitly in the nongeneric BO algorithm are not "bad"; the same polynomials can be generated using another pair $\mathbf{x}_0$, $\mathbf{y}_0$ that leads to the most desirable version of termination $\mathbf{x}_{n_J} = \mathbf{0}$. This is surprising since (4.14) holds for $\mathbf{x}_{n_J}$ and $\mathbf{y}_{n_J}$. If $\mathbf{A}$ is diagonalizable, then $\iota(l) = \bar{\iota}(l) = 1$ $(l = 1,\cdots,L)$ in the above proof; the new pair $\mathbf{x}_0$, $\mathbf{y}_0$ that is constructed then yields even $\mathbf{x}_{n_J} = \mathbf{y}_{n_J} = \mathbf{0}$.

The case $\mathbf{B} = \mathbf{A}$ of the Mismatch Theorem, which has some relevance for the algorithm BIODIR introduced below, can be treated in the same way. However, our approach seems to become intractable in the general case of two arbitrary commuting matrices $\mathbf{A}$ and $\mathbf{B}$.

As in the generic case [16] the nongeneric BO process can also be applied to *solving a linear system of equations* $\mathbf{Az} = \mathbf{b}$. In the generic case the resulting algorithm has been called Lanczos/ORTHORES [19], but in [15] we introduce the briefer and more appropriate name BIORES, an abbreviation for BIORTHORES. In addition to the sequences $\{\mathbf{x}_n\}$ and $\{\mathbf{y}_n\}$ we normally generate a sequence $\{\mathbf{z}_n\}$ of approximate solutions in such a way that $\mathbf{x}_n$ is equal to the nth residual $\mathbf{b} - \mathbf{Az}_n$. At least theoretically, the aim is to reach $\mathbf{x}_n = \mathbf{0}$ for some n, in which case $\mathbf{z}_n$ is the solution of the system. In practice, the method must be viewed as an iterative method and the hope is that the residuals become sufficiently small. In the case of a nonhermitian matrix the convergence behavior is still not well understood. The method is a polynomial acceleration method since the nth residual can be expressed in the form (4.14) with P_n a polynomial of degree n. In any such method the recurrence for the residuals

can be expressed in matrix form as $\mathbf{AX} = \mathbf{XH}_\Gamma$, where $\mathbf{H}_\Gamma$ is an unreduced upper Hessenberg matrix with column sums 0. These two conditions mean that the nth residual polynomial has exact degree n and value 1 at 0. The column sum condition, which is equivalent to the normalization of the residual polynomials at 0, is the so-called *consistency condition.* Here the residual polynomial is $\Gamma_n P_n$, so that we need $P_n(0) = 1/\Gamma_n$ (or, equivalently, we need $\mathbf{H}_\Gamma$ to have column sums 0).

We first formulate the algorithm and then verify the properties just mentioned.

ALGORITHM 2 (normalized nongeneric BO algorithm for linear systems: normalized nongeneric BIORES). *For solving* $\mathbf{Az} = \mathbf{b}$ *choose an initial approximation* $\mathbf{z}_0$, *set* $\mathbf{x}_0 := \mathbf{b} - \mathbf{Az}_0$, *choose* $\mathbf{y}_0$ *with* $\langle \mathbf{y}_0, \mathbf{x}_0 \rangle_\mathbf{B} \neq 0$, *and apply Algorithm* 1 *with the following special settings:*

(i) *Choose for* (4.19a, b) *recurrence coefficients* $\{\alpha_m^W\}_{m=0}^\infty$ *and* $\{\beta_m^W\}_{m=1}^\infty$ *such that the recurrence*

$$\gamma_{m+1}^W := \frac{-1}{\alpha_m^W + \beta_m^W \gamma_m^W} \qquad (m \in \mathbb{N}) \tag{4.54}$$

(*started with* $\beta_0^W := 0$ *and* $\gamma_0^W := 1$) *is well defined, i.e.,* $\gamma_{m+1}^W \neq \infty$.

(ii) *For* $j = 0, 1, \cdots$ *set*

$$\gamma_{n,1} := \gamma_{n-n_j}^W \quad \textit{if } n_j < n \leq n_{j+1} - 1, \tag{4.55a}$$

$$\gamma_{n_{j+1},1} := -\left[\alpha_{h_j-1}^W + \beta_{h_j-1}^W \gamma_{n_{j+1}-1,1} + \sum_{i=1}^{h_j} \alpha_{h_j-i,j} \gamma_{n_{j+1}-1,i-1} + \beta_j \gamma_{n_{j+1}-1,h_j+h_{j-1}-1}\right]^{-1}. \tag{4.55b}$$

(iii) *Compute additionally for* $j = 0, 1, \cdots$,

$$\mathbf{z}_{n+1} := -\left[\mathbf{x}_n + \mathbf{z}_n \alpha_{n-n_j}^W\right] \gamma_{n+1,1} - \mathbf{z}_{n-1} \beta_{n-n_j}^W \gamma_{n+1,2} \quad \textit{if } n_j \leq n \leq n_{j+1} - 2, \tag{4.56a}$$

$$\begin{aligned} \mathbf{z}_{n_{j+1}} := &-\left[\mathbf{x}_{n_{j+1}-1} + \mathbf{z}_{n_{j+1}-1} (\alpha_{h_j-1,j} + \alpha_{h_j-1}^W)\right] \gamma_{n_{j+1},1} \\ &- \mathbf{z}_{n_{j+1}-2} (\alpha_{h_j-2,j} + \beta_{h_j-1}^W) \gamma_{n_{j+1},2} - \mathbf{z}_{n_{j+1}-3} \alpha_{h_j-3,j} \gamma_{n_{j+1},3} - \cdots \\ &- \mathbf{z}_{n_j} \alpha_{0,j} \gamma_{n_{j+1},h_j} - \mathbf{z}_{n_{j-1}} \beta_j \gamma_{n_{j+1},h_j+h_{j-1}}. \end{aligned} \tag{4.56b}$$

The algorithm terminates when $n_{j+1} = n_J$ *and* $\mathbf{x}_{n_J} = \mathbf{0}$. *Then* $\mathbf{z}_{n_J}$ *is the solution of* $\mathbf{Az} = \mathbf{b}$. *If* $n_{j+1} = n_J$ *but* $\mathbf{x}_{n_J} \neq \mathbf{0}$, *the solution cannot be found and a restart with another* $\mathbf{y}_0$ *is necessary. If, for some* j, *the denominator in the formula* (4.55b) *for* $\gamma_{n_{j+1},1}$ *vanishes, the algorithm breaks down.*

Note that γ_m^W and $\gamma_{n,1}$ have been chosen in such a way that with the definition

$$\Gamma_m^W := \gamma_1^W \gamma_2^W \cdots \gamma_m^W \tag{4.57}$$

and in view of (4.20b), the rescaled matrices

$$\operatorname{diag}\left[(\Gamma_0^W)^{-1}, (\Gamma_1^W)^{-1}, \cdots\right] \mathbf{T}^W \operatorname{diag}\left[\Gamma_0^W, \Gamma_1^W, \cdots\right] \tag{4.58}$$

and $\mathbf{H}_\Gamma$ (defined by (4.28)) have column sums 0. The scaling factors $\bar{\gamma}_{n,1}$ that determine the factors $\bar{\Gamma}_n$ for $\mathbf{y}_n$ can still be chosen independently.

In order to satisfy condition (4.54) we can choose the coefficients α_m^W and β_m^W for (4.19a) according to any standard two-step iterative method such as Chebyshev iteration or Frankel's "second-order Richardson iteration" (see, e.g., [41], [48], [50]), and we could adapt the method from time to time to match the spectral properties of $\mathbf{A}$ as far as they have emerged approximately from the spectrum of the initial sections $\mathbf{H}_n$, using techniques as in [11], [28], and [31]. As mentioned before, we can also use any other polynomial acceleration method for this "inner" or "local" iteration; then the diagonal blocks of $\mathbf{H}_\Gamma$ generally become Hessenberg matrices instead of comrade matrices, so that our formulas have to be modified in a straightforward way.

To prove that $\mathbf{x}_n$ is the residual at $\mathbf{z}_n$ we make the induction assumption that $\mathbf{x}_m = \mathbf{b} - \mathbf{A}\mathbf{z}_m$ for $m = n_{j-1}$ and $n_j \le m \le n$. Then, if $n_j \le n < n_{j+1} - 1$, it follows from (4.20b), (4.54), and (4.55a) that $\mathbf{b} = -\mathbf{b}\alpha_{n-n_j}^W \gamma_{n+1,1} - \mathbf{b}\beta_{n-n_j}^W \gamma_{n+1,2}$, and thus by (4.19a) and (4.56a), that

$$
\begin{aligned}
(4.59) \qquad (\mathbf{b} - \mathbf{A}\mathbf{z}_{n+1}) &= \mathbf{b} + [\mathbf{A}\mathbf{x}_n + \mathbf{A}\mathbf{z}_n \alpha_{n-n_j}^W]\,\gamma_{n+1,1} + \mathbf{A}\mathbf{z}_{n-1}\beta_{n-n_j}^W \gamma_{n+1,2} \\
&= [\mathbf{A}\mathbf{x}_n + \mathbf{x}_n \alpha_{n-n_j}^W]\,\gamma_{n+1,1} - \mathbf{x}_{n-1}\beta_{n-n_j}^W \gamma_{n+1,2} = \mathbf{x}_{n+1}\,.
\end{aligned}
$$

Similarly, for $n = n_{j+1} - 1$, (4.55b), (4.56b), and (4.21a) yield the same. Hence, by induction, it follows that $\mathbf{x}_n = \mathbf{b} - \mathbf{A}\mathbf{z}_n$ for all n. Note that the argument remains valid if $\mathbf{x}_n = \mathbf{0}$.

Note that aside from any numerical effects there are three causes for an unsatisfactory finish: An incurable breakdown due to (4.46), an early termination with $\mathbf{y}_{n_J} = \mathbf{0}$ and $\mathbf{x}_{n_J} \neq \mathbf{0}$ (in which case we will restart the algorithm with $\mathbf{x}_0 := \mathbf{x}_{n_J}$ and a new $\mathbf{y}_0$), and the possible vanishing of the denominator of $\gamma_{n_{j+1},1}$ in (4.55b). Hence, as in the generic case there is—compared to the BO algorithm—an additional danger of breakdown due to the last cause. It has nothing to do with any curable or incurable breakdown of the BO algorithm, but is instead just due to the fact that it is impossible to observe the consistency condition if $P_n(0) = 0$. (This, however, has to do with how the first subdiagonal in the Padé table crosses the blocks; cf. [15].) As in [15] for the generic case, we suggest here avoiding this kind of breakdown by using, instead of Algorithm 2, an unnormalized form of it, in which we generate the sequences $\{\mathbf{x}_n\}$ and $\{\mathbf{y}_n\}$ as in Algorithm 1 and compute additionally the values ρ_n of the polynomials $\Gamma_n P_n$ at 0 and a sequence $\{\mathbf{z}_n\}$, using for the latter the same formula as in Algorithm 2. Whenever $\|\mathbf{x}_n\|/|\Gamma_n P_n(0)|$ is sufficiently small, we then obtain in the form $\mathbf{z}_n/(\Gamma_n P_n(0))$ a solution with this small residual norm.

ALGORITHM 3 (unnormalized nongeneric BO algorithm for linear systems: unnormalized nongeneric BIORES). *For solving* $\mathbf{A}\mathbf{z} = \mathbf{b}$ *choose an initial approximation* $\mathbf{z}_0$, *set* $\mathbf{x}_0 := \mathbf{b} - \mathbf{A}\mathbf{z}_0$, *choose* $\mathbf{y}_0$ *with* $\langle \mathbf{y}_0, \mathbf{x}_0 \rangle_{\mathbf{B}} \neq 0$, *and apply Algorithm* 1 (*with arbitrary nonzero scale factors* $\gamma_{n,1}$, $\bar{\gamma}_{n,1}$), *computing additionally the vector sequence* $\{\mathbf{z}_n\}$ *according to* (4.56a, b) *and the scalar sequence* $\{\rho_n\}$ *according to*

$$
\begin{aligned}
(4.60a) \qquad & \rho_0 := 1\,, \\
(4.60b) \qquad & \rho_{n+1} := -\rho_n \alpha_{n-n_j}^W \gamma_{n+1,1} - \rho_{n-1}\beta_{n-n_j}^W \gamma_{n+1,2}\,, \qquad n_j \le n \le n_{j+1} - 2\,, \\
& \rho_{n_{j+1}} := -\rho_{n_{j+1}-1}\,(\alpha_{h_j-1,j} + \alpha_{h_j-1}^W)\,\gamma_{n_j+1,1} \\
(4.60c) \qquad & \qquad -\rho_{n_{j+1}-2}\,(\alpha_{h_j-2,j} + \beta_{h_j-1}^W)\,\gamma_{n_j+1,2} - \rho_{n_{j+1}-3}\,\alpha_{h_j-3,j}\,\gamma_{n_j+1,3} - \cdots \\
& \qquad -\rho_{n_j}\,\alpha_{0,j}\,\gamma_{n_{j+1},h_j} - \rho_{n_{j-1}}\beta_j\,\gamma_{n_{j+1},h_j+h_{j-1}}\,.
\end{aligned}
$$

The algorithm terminates when $\mathbf{x}_{n_{j+1}} = \mathbf{0}$ *(hence,* $n_{j+1} = n_J$*) and* $\rho_{n_{j+1}} \neq 0$. *Then* $\mathbf{z}_{n_J}/\rho_{n_J}$ *is the solution of* $\mathbf{Az} = \mathbf{b}$. *If* $n_{j+1} = n_J$ *but* $\mathbf{x}_{n_J} \neq \mathbf{0}$ *or* $\rho_{n_J} = 0$, *the solution cannot be found and a restart with another* $\mathbf{y}_0$ *is necessary.*

The relations between the sequences $\mathbf{x}_n$ and $\mathbf{z}_n$, which are relevant for the application of the BO process to linear systems, are summarized in the following theorem.

THEOREM 4.6. (i) *In Algorithm* 2

$$\mathbf{x}_n = \mathbf{b} - \mathbf{Az}_n, \quad n = 0, 1, 2, \cdots \tag{4.61}$$

holds.

(ii) *In Algorithm* 3

$$\mathbf{x}_n = \mathbf{b}\rho_n - \mathbf{Az}_n, \quad n = 0, 1, 2, \cdots \tag{4.62}$$

holds.

Proof. Equation (4.61) has already been verified. For (4.62) we have to modify the previous induction proof slightly by using (4.60a–c) instead of the relations (4.54)–(4.55a, b). □

It may be worth noting that Algorithms 2 and 3 are here formulated for systems of the form $\mathbf{Az} = \mathbf{b}$, while for systems given as $\mathbf{z} = \mathbf{Bz} + \mathbf{b}$ there exists a more appropriate alternative formulation.

Acknowledgments. The author is indebted to Adhemar Bultheel and Marc Van Barel for detecting an incorrect conclusion in a former proof of Theorem 4.5, and to Roland Freund for pointing out a number of other shortcomings of the manuscript.

REFERENCES

[1] G. A. BAKER, JR. AND P. GRAVES-MORRIS, *Padé Approximants. Part* I: *Basic Theory*, Addison-Wesley, Reading, MA, 1981.

[2] S. BARNETT, *A companion matrix analogue for orthogonal polynomials*, Linear Algebra Appl., 12 (1975), pp. 197–208.

[3] D. BOLEY, *Non-symmetric Lanczos and reconstructing indefinite weights*, manuscript, December 1989.

[4] D. BOLEY, S. ELHAY, G. H. GOLUB, AND M. H. GUTKNECHT, *Nonsymmetric Lanczos and finding orthogonal polynomials associated with indefinite weights*, Numer. Algorithms, 1 (1991), pp. 21–43.

[5] D. BOLEY AND G. GOLUB, *The nonsymmetric Lanczos algorithm and controllability*, Tech. Report NA-90-06, Computer Science Department, Stanford University, Stanford, CA, May 1990.

[6] C. BREZINSKI, *Padé-Type Approximants and General Orthogonal Polynomials*, Birkhäuser, Basel, 1980.

[7] A. DRAUX, *Polynômes Orthogonaux Formels—Applications*, Lecture Notes in Mathematics, 974, Springer-Verlag, Berlin, 1983.

[8] R. FLETCHER, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis, G. A. Watson, ed., Lecture Notes in Mathematics, 506, Springer-Verlag, Berlin, 1976, pp. 73–89.

[9] F. GANTMACHER, *The Theory of Matrices*, Vol. 2, Chelsea, New York, 1959.

[10] G. GOLUB AND M. GUTKNECHT, *Modified moments for indefinite weight functions*, Numer. Math., 57 (1990), pp. 607–624.

[11] G. H. GOLUB AND M. D. KENT, *Estimates of eigenvalues for iterative methods*, Math. Comp., 53 (1989), pp. 619–626.

[12] W. B. GRAGG, *The Padé table and its relation to certain algorithms of numerical analysis*, SIAM Rev., 14 (1972), pp. 1–62.

[13] ———, *Matrix interpretations and applications of the continued fraction algorithm*, Rocky Mountain J. Math., 4 (1974), pp. 213–225.

[14] W. B. GRAGG AND A. LINDQUIST, *On the partial realization problem*, Linear Algebra Appl., 50 (1983), pp. 277–319.

[15] M. H. GUTKNECHT, *The unsymmetric Lanczos algorithms and their relations to Padé approximation, continued fractions, and the qd algorithm*, Preliminary Proceedings of the Copper Mountain Conference on Iterative Methods (preliminary version).
[16] ———, *Continued fractions associated with the Newton–Padé table*, Numer. Math., 56 (1989), pp. 547–589.
[17] ———, *The rational interpolation problem revisited*, Rocky Mountain J. Math., 21 (1991), pp. 263–280.
[18] A. S. HOUSEHOLDER, *The Theory of Matrices in Numerical Analysis*, Dover, New York, 1964.
[19] K. C. JEA AND D. M. YOUNG, *On the simplification of generalized conjugate-gradient methods for nonsymmetrizable linear systems*, Linear Algebra Appl., 52 (1983), pp. 399–417.
[20] W. D. JOUBERT, *Lanczos methods for the solution of nonsymmetric systems of linear equations*, SIAM J. Matrix Anal. Appl., 13 (1992), to appear.
[21] ———, *Generalized conjugate gradient and Lanczos methods for the solution of nonsymmetric systems of linear equations*, Ph.D. thesis and Tech. Report CNA-238, Center for Numerical Analysis, University of Texas, Austin, TX, 1990.
[22] W. D. JOUBERT AND D. M. YOUNG, *Necessary and sufficient conditions for the simplification of generalized conjugate gradient algorithms*, Tech. Report CNA-204, Center for Numerical Analysis, University of Texas, Austin, TX, 1986.
[23] R. E. KALMAN, *On partial realizations, transfer functions, and canonical forms*, Acta Polytech. Scand. Math. Comput. Sci. Ser., 31 (1979), pp. 9–32.
[24] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45 (1950), pp. 255–281.
[25] ———, *Solution of systems of linear equations by minimized iterations*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 33–53.
[26] A. MAGNUS, *Certain continued fractions associated with the Padé table*, Math. Z., 78 (1962), pp. 361–374.
[27] ———, *Expansion of power series into P-fractions*, Math. Z., 80 (1962), pp. 209–216.
[28] T. A. MANTEUFFEL, *The Tchebyshev iteration for nonsymmetric linear systems*, Numer. Math., 28 (1977), pp. 307–327.
[29] J. MAROULAS AND S. BARNETT, *Polynomials with respect to a general basis. I. Theory*, J. Math. Anal. Appl., 72 (1979), pp. 177–194.
[30] ———, *Polynomials with respect to a general basis. II. Applications*, J. Math. Anal. Appl., 72 (1979), pp. 599–614.
[31] N. M. NACHTIGAL, L. REICHEL, AND L. N. TREFETHEN, *A hybrid* GMRES *algorithm for nonsymmetric linear systems*, SIAM J. Matrix Anal. Appl., 1993, 13 (1992), to appear.
[32] J. NUTTALL AND S. R. SINGH, *Orthogonal polynomials and Padé approximants associated with a system of arcs*, J. Approx. Theory, 21 (1977), pp. 1–42.
[33] C. C. PAIGE, *The computations of eigenvalues and eigenvectors of very large sparse matrices*, Ph.D. thesis, University of London, London, 1971.
[34] ———, *Error analysis of the Lanczos algorithms for tridiagonalizing a symmetric matrix*, J. Inst. Math. Appl., 18 (1976), pp. 341–349.
[35] B. N. PARLETT, *Reduction to tridiagonal form and minimal realizations*, SIAM J. Matrix Anal. Appl., this issue, pp. 567–593.
[36] ———, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.
[37] B. N. PARLETT AND B. NOUR-OMID, *Towards a black box Lanczos program*, Comput. Phys. Comm., 53 (1989), pp. 169–179.
[38] B. N. PARLETT, D. R. TAYLOR, AND Z. A. LIU, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp., 44 (1985), pp. 105–124.
[39] H. RUTISHAUSER, *Beiträge zur Kenntnis des Biorthogonalisierungs-Algorithmus von Lanczos*, Z. Angew. Math. Phys., 4 (1953), pp. 35–56.
[40] ———, *Der Quotienten-Differenzen-Algorithmus*, Mitt. Inst. Angew. Math. ETH, Nr. 7, Birkhäuser, Basel, 1957.
[41] ———, *Theory of gradient methods*, in Refined Iterative Methods for Computation of the Solution and the Eigenvalues of Self-Adjoint Boundary Value Problems, Mitt. Inst. Angew. Math. ETH Zürich, Nr. 8, Birkhäuser, Basel, 1959, pp. 24–49.
[42] H. STAHL, *Divergence of diagonal Padé approximants and the asymptotic behavior of orthogonal polynomials associated with nonpositive measures*, Constr. Approx., 1 (1985), pp. 249–270.
[43] ———, *Orthogonal polynomials with complex-valued weight function*, Constr. Approx., 2 (1986), pp. I: 225–240, II: 241–251.
[44] G. W. STRUBLE, *Orthogonal polynomials: Variable-signed weight functions*, Numer. Math., 5 (1963), pp. 88–94.

[45] D. R. TAYLOR, *Analysis of the look ahead Lanczos algorithm*, Ph.D. thesis, Department of Mathematics, University of California, Berkeley, CA, 1982.
[46] L. N. TREFETHEN, *Non-normal Matrices and Pseudospectra*, in preparation.
[47] L. N. TREFETHEN AND M. H. GUTKNECHT, *Padé, stable Padé, and Chebyshev-Padé approximation*, in Algorithms for Approximation, J. Mason and M. Cox, eds., IMA Conference Series, Vol. 10, Clarendon Press, Oxford, 1987, pp. 227–264.
[48] R. S. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.
[49] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.
[50] H. E. WRIGLEY, *Accelerating the Jacobi method for solving simultaneous equations by Chebyshev extrapolation when the eigenvalues of the iteration matrix are complex*, Comput. J., 6 (1963), pp. 169–176.
[51] W. B. GRAGG, Private communication, 1989.
[52] B. PARLETT, Private communication, 1988.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 640–654, April 1992

011

MATRICES WITH POSITIVE DEFINITE HERMITIAN PART: INEQUALITIES AND LINEAR SYSTEMS*

ROY MATHIAS†

Abstract. The Hermitian and skew-Hermitian parts of a square matrix A are defined by

$$H(A) \equiv (A + A^*)/2 \quad \text{and} \quad S(A) \equiv (A - A^*)/2.$$

It is shown that the function $f(A) = (H(A^{-1}))^{-1}$ is convex with respect to the Loewner partial order on the cone of matrices with positive definite Hermitian part. That is, for any matrices A and B with positive definite Hermitian part

$$\{f(A) + f(B)\}/2 - f(\{A + B\}/2) \quad \text{is positive semidefinite.}$$

Using this basic fact, this paper proves a variety of inequalities involving norms, Hadamard products and submatrices, and a perturbation result for the function f. These results are generalizations of results for positive definite matrices. Often the quantity

$$\kappa_H(A) \equiv \|H(A^{-1})^{-1}\|_2 \, \|H(A)^{-1}\|_2$$

plays the role that $\kappa_2(A) \equiv \|A\|_2\|A^{-1}\|_2$ plays in inequalities involving positive definite matrices. ($\|\cdot\|_2$ denotes the spectral norm.) Finally a bound is derived on the backward and forward error in $\hat{x}$, the solution to

$$Ax = b \quad \text{with } H(A) \text{ positive definite} \tag{1}$$

computed by Gaussian elimination without pivoting in finite precision. This result is analogous to Wilkinson's result for positive definite matrices and gives a rigorous criterion for deciding when it is numerically safe not to pivot when solving (1).

Key words. positive definite matrix, LU factorization, Loewner partial order, matrix convexity, Hadamard product, condition number

AMS(MOS) subject classifications. 15A48, 15A23, 15A45, 15A12

1. Introduction.

Let $M_n(C)$ (respectively, $M_n(R)$) denote the space of $n \times n$ complex (respectively, real) matrices. We call $A \in M_n(C)$ *positive definite* (respectively, *positive semidefinite*) if A is Hermitian and $x^*Ax > 0$ (respectively, $x^*Ax \geq 0$) for all nonzero $x \in C^n$. The Hermitian part of A is

$$H(A) \equiv (A + A^*)/2$$

and the skew-Hermitian part of A is

$$S(A) \equiv (A - A^*)/2.$$

Matrices with positive definite Hermitian part have many properties analogous to those of positive definite matrices. We discuss some of these in this paper.

In §§2 and 3 we derive a variety of inequalities for matrices with positive definite Hermitian part. Most of these involve principal submatrices, condition numbers, or Hadamard products and generalize well-known results for positive definite matrices. The two underlying facts are the formula for the Hermitian part of the inverse (Lemma 2.1) and a basic convexity result (Theorem 2.2). The results in §2 are applied in §4.

* Received by the editors August 16, 1990; accepted for publication November 1, 1990. This research was supported by an Eliezer Naddor postdoctoral fellowship in the Mathematical Sciences from the Johns Hopkins University during the year 1989–90 while the author was in residence at the Department of Computer Science at Cornell University.

† Department of Mathematics, The College of William and Mary, Williamsburg, Virginia 23187 (na.mathias@na-net.ornl.gov).

In §4 we derive error bounds for Gaussian elimination applied to a matrix with positive definite Hermitian part in finite precision arithmetic. The leading principal minors (in fact all the principal minors) of a matrix A with positive definite Hermitian part are positive and hence a linear system $Ax = b$ can be solved by Gaussian elimination without pivoting. This fact can be exploited in practical algorithms. However, Gaussian elimination without pivoting can lead to serious element growth, and in finite precision arithmetic this tends to result in an unacceptably inaccurate solution (see, e.g., [9, p. 87] for a simple example). In [9] the authors showed (under the reasonable assumption that $\| \, |L||U| \, \| \approx \| \, |\hat{L}||\hat{U}| \, \|$) that $\hat{x}$, the solution computed by Gaussian elimination without pivoting, satisfies $(A + E)\hat{x} = b$ where

$$\|E\|_F \leq unc_n \|H + S^T H^{-1} S\|_2,$$

u is machine precision, and c_n is a linear function of n. Using this result they argued that it is safe not to pivot when solving $Ax = b$ provided the ratio

$$\|H + S^T H^{-1} S\|_2 / \|A\|_2$$

is not large. (In Lemma 2.1 we show that this quantity is at least 1.) We show that it is not necessary to make the assumption $\| \, |L||U| \, \| \approx \| \, |\hat{L}||\hat{U}| \, \|$, and thereby give a sufficient a priori condition for the LU factorization in finite precision arithmetic (without pivoting) of a positive definite matrix to run to completion with positive pivots. These results are in §4.

All the results in this paper may be viewed as generalizations of results for positive definite matrices.

If A is Hermitian we use $\lambda_{\max}(A)$ (respectively, $\lambda_{\min}(A)$) to denote the algebraically largest (respectively, smallest) eigenvalue of A. The *spectral norm* ($\| \cdot \|_2$) and the *Frobenius norm* ($\| \cdot \|_F$) are defined on M_n by

$$\|A\|_2 \equiv \sqrt{\lambda_{\max}(A^*A)} = \max\{\|Ax\|_2 : \|x\|_2 = 1, x \in C^n\}$$

and

$$\|A\|_F \equiv \sqrt{\sum |a_{ij}|^2}.$$

We define $|A| \equiv [|a_{ij}|]$. We write $A \preceq B$ if $B - A$ is positive semidefinite. If T is positive definite then we use $T^{1/2}$ to denote the unique positive definite square root of T. We will frequently use the fact that for Hermitian matrices $A, B \in M_n(C)$

$$\lambda_{\min}(A + B) \geq \lambda_{\min}(A) + \lambda_{\min}(B) \geq \lambda_{\min}(A) - \|B\|_2,$$

and that for a positive definite matrix A

$$\|A\|_2 = [\lambda_{\min}(A^{-1})]^{-1}.$$

2. Matrices with positive definite Hermitian part. In this section we develop some of the properties of matrices with positive definite Hermitian part, in particular the properties of the Hermitian part of the inverse of such a matrix. Previous research on matrices with positive definite Hermitian or skew-Hermitian part [11], [5], [6], [13] has concentrated on the properties of AA^{-*}, especially interlacing inequalities for the arguments of the eigenvalues of AA^{-*}. (The eigenvalues of AA^{-*} all have unit modulus.)

We start by determining the Hermitian part of the inverse of a matrix.

LEMMA 2.1. *Let A have positive definite Hermitian part, and let $H = H(A)$ and $S = S(A)$. Then A is invertible and A^{-1} has positive definite Hermitian part given by*

$$H(A^{-1}) = (A^{-1} + A^{-*})/2 = (H + S^*H^{-1}S)^{-1}, \tag{2.1}$$

and we have the inequalities

$$\|A^{-1}\|_2 \le \|H^{-1}\|_2 \quad \textit{and} \quad \|A\|_2 \le \|H + S^*H^{-1}S\|_2. \tag{2.2}$$

The first inequality in (2.2) is Problem 4.2.3 in [8].

Proof. Let $A = H + S$ satisfy the conditions of the lemma. Since $X^{-1} + Y^{-1} = X^{-1}(X + Y)Y^{-1}$ for any nonsingular $X, Y \in M_n$ we have

$$\begin{aligned}\{(A^{-1} + A^{-*})/2\}^{-1} &= \{(H + S)^{-1}(2H)(H - S)^{-1}/2\}^{-1} \\ &= (H - S)H^{-1}(H + S) \\ &= H - SH^{-1}S \\ &= H + S^*H^{-1}S.\end{aligned}$$

Taking inverses now yields (2.1).

By (2.1) the first inequality in (2.2) applied to A^{-1} yields the second, so it suffices to prove the first. Using the fact that x^*Sx is imaginary for any $x \in C^n$ and that H is positive definite, we have

$$\begin{aligned}\|A^{-1}\|_2^{-1} &= \min\{\|Ax\|_2 : \|x\|_2 = 1\} \\ &\le \min\{|x^*Ax| : \|x\|_2 = 1\} \\ &= \min\{|x^*Hx + x^*Sx| : \|x\|_2 = 1\} \\ &\le \min\{|x^*Hx| : \|x\|_2 = 1\} \\ &\le \min\{x^*Hx : \|x\|_2 = 1\} \\ &= \|H^{-1}\|_2^{-1}.\end{aligned}$$

The inequality now follows by taking inverses. □

From (2.1) one can easily show that for A with positive definite Hermitian part,

$$H(A^{-1}) \preceq [H(A)]^{-1}. \tag{2.3}$$

We will refine this inequality in the next section. One can also use (2.1) to derive the bound

$$\lambda_{\min}(H(A^{-1})) \ge [\lambda_{\max}(H) + \|S\|^2/\lambda_{\min}(H)]^{-1} > 0,$$

which is used in [4] to prove the convergence of a variant of the conjugate gradient method for solving a linear system $Ax = b$ when A has positive definite Hermitian part.

Define the functions f and κ_H on the cone of positive definite matrices by

$$f(A) \equiv [(A^{-1} + A^{-*})/2]^{-1} = H + S^*H^{-1}S, \tag{2.4}$$

and

$$\kappa_H(A) \equiv \|H + S^*H^{-1}S\|_2 \, \|H^{-1}\|_2, \tag{2.5}$$

where $H = H(A)$ and $S = S(A)$. We define $\kappa_2(A) \equiv \|A\|_2\|A^{-1}\|_2$ for any nonsingular A. Note that $\kappa_2(A) = \kappa_H(A)$ if A is positive definite. In the next theorem we show that f is convex with respect to the partial order $\preceq$. Many of the following results are based on this fact.

THEOREM 2.2. *Let f be defined by (2.4) then f is convex with respect to the partial order $\preceq$. That is, for any $A_1, A_2 \in M_n$ with positive definite Hermitian part and any $t \in [0,1]$,*

$$f(tA_1 + (1-t)A_2) \preceq tf(A_1) + (1-t)f(A_2). \tag{2.6}$$

Furthermore, suppose that $A \in M_n$ has positive definite Hermitian part and is partitioned as

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix} \quad \text{with} \quad A_{11} \in M_k,\ A_{22} \in M_{n-k}, \tag{2.7}$$

and let $f(A)$ be partitioned in the same way. Then

(1) *$f(A)$ is positive definite and $f(A) = f(A^*)$.*
(2) *$f(XAX^*) = Xf(A)X^*$ for any nonsingular $X \in M_n$.*
(3) *$f(A_{22} - A_{21}A_{11}^{-1}A_{12}) = f(A)_{22}$.*
(4) *$\|f(A_{22} - A_{21}A_{11}^{-1}A_{12})\|_2 \le \|f(A)_{22}\|$.*
(5) *$f(A_{11} \oplus A_{22}) \preceq f(A)_{11} \oplus f(A)_{22}$.*
(6) *$f(A_{11}) \preceq f(A)_{11}$.*

Proof. To prove the convexity of f we will use the following fact, which is essentially Theorem 7.7.6 in [10]. Let

$$X = \begin{pmatrix} X_{11} & X_{12} \\ X_{12}^* & X_{22} \end{pmatrix}$$

be Hermitian with X_{11} positive definite. Then X is positive semidefinite if and only if

$$X_{12}^* X_{11}^{-1} X_{12} \preceq X_{22}.$$

Let $A_i = H_i + S_i \in M_n$ be given with H_i positive definite and S_i skew-Hermitian. Then

$$0 \preceq \begin{pmatrix} H_i^{1/2} & H_i^{-1/2}S \end{pmatrix}^* \begin{pmatrix} H_i^{1/2} & H_i^{-1/2}S \end{pmatrix} = \begin{pmatrix} H_i & S_i \\ S_i^* & S_i^* H_i^{-1} S_i \end{pmatrix}, \qquad i = 1,2,$$

and hence for $t \in [0,1]$

$$\begin{aligned} 0 &\preceq t \begin{pmatrix} H_1 & S_1 \\ S_1^* & S_1^* H_1^{-1} S_1 \end{pmatrix} + (1-t) \begin{pmatrix} H_2 & S_2 \\ S_2^* & S_2^* H_2^{-1} S_2 \end{pmatrix} \\ &= \begin{pmatrix} tH_1 + (1-t)H_2 & tS_1 + (1-t)S_2 \\ (tS_1 + (1-t)S_2)^* & tS_1^* H_1^{-1} S_1 + (1-t)S_2^* H_2^{-1} S_2 \end{pmatrix}. \end{aligned}$$

By the criterion above, the positive-semidefiniteness of this last matrix implies

$$[tS_1 + (1-t)S_2]^* \, [tH_1 + (1-t)H_2]^{-1} \, [tS_1 + (1-t)S_2] \preceq tS_1^* H_1^{-1} S_1 + (1-t)S_2^* H_2^{-1} S_2,$$

from which the assertion (2.6) follows.

The statements (1) and (2) follow immediately from the definitions. To prove (3) note that $(A_{22} - A_{12}A_{11}^{-1}A_{12})^{-1} = (A^{-1})_{22}$ (see, e.g., [10, §0.7.3]). The norm inequality in (4) follows from this. To prove (5) let Q be the nonsingular matrix $I_k \oplus (-I_{n-k})$ and note that

$$A_{11} \oplus A_{22} = (A + QAQ^*)/2 \quad \text{and} \quad f(A)_{11} \oplus f(A)_{22} = (f(A) + Qf(A)Q^*)/2.$$

Now use the convexity of f to obtain the desired inequality:

$$f(A_{11} \oplus A_{22}) = f((A + QAQ^*)/2) \preceq [f(A) + Qf(A)Q^*]/2 = f(A)_{11} \oplus f(A)_{22}.$$

(6) follows immediately from (5). □

We have shown that (2.6) implies (5). It is not hard to show that if f is any function from M_n to M_n such that $f(QAQ^*) = Qf(A)Q^*$ for any unitary $Q \in M_n$ then the convexity inequality (2.6) holds if and only if the submatrix inequalty (6) holds (see, e.g., [7]).

We collect several useful facts about κ_H in the following theorem. These results reduce to well-known results if one restricts A to be symmetric positive definite (in which case $\kappa_H(A) = \kappa_2(A)$).

THEOREM 2.3. *Let κ_H be defined by* (2.5) *and $A, B \in M_n$ be positive definite with A partitioned as in* (2.7). *Then*

(1) $\kappa_H(A) = \kappa_H(A^*) = \kappa_H(A^{-1}) = \kappa_H(cA)$ *for any* $c > 0$.
(2) $\kappa_H(A) = \kappa_H(QAQ^*)$ *for any unitary* $Q \in M_n$.
(3) $\kappa_H(A) \geq \|A\|_2\|A^{-1}\|_2 = \kappa_2(A) \geq 1$.
(4) $\kappa_H(tA + (1-t)B) \leq \max\{\kappa_H(A), \kappa_H(B)\}$ *for any* $t \in [0,1]$.
(5) $\kappa_H(A + I) \leq \kappa_H(A)$.
(6) $\kappa_H(A) \geq \kappa_H(A_{11} \oplus A_{22}) \geq \kappa_H(A_{11})$.
(7) $\kappa_H(A) \geq \kappa_H(A_{22} - A_{21}A_{11}^{-1}A_{12})$.

Proof. The statements in (1) and (2) follow immediately from the definitions. The inequalities in (2.2) imply (3). Since $\kappa_H(cX) = \kappa_H(X)$ for any $c > 0$ and any positive definite X, it suffices to prove (4) under the additional assumption

$$\lambda_{\min}((A + A^*)/2) = \lambda_{\min}((B + B^*)/2) = 1, \tag{2.8}$$

in which case

$$\kappa_H(A) = \|f(A)\|_2 \quad \text{and} \quad \kappa_H(B) = \|f(B)\|_2.$$

Let $C = A + B$. Then, by (2.8),

$$\lambda_{\min}((C + C^*)/2) \geq t\lambda_{\min}((A + A^*)/2) + (1-t)\lambda_{\min}((B + B^*)/2) = 1,$$

or, equivalently, $\|[(C + C^*)/2]^{-1}\|_2 \leq 1$. By Theorem 2.2 we have

$$f(C) \preceq tf(A) + (1-t)f(B).$$

Since $f(C)$ is positive definite, this implies

$$\begin{aligned}\|f(C)\|_2 &\leq \|tf(A) + (1-t)f(B)\|_2 \leq t\|f(A)\|_2 + (1-t)\|f(B)\|_2 \\ &\leq \max\{\kappa_H(A), \kappa_H(B)\}.\end{aligned}$$

Combining this with the bound on $\|[(C + C^*)/2]^{-1}\|_2$ gives the result.

The inequality in (5) is a special case of (4). The second inequality in (6) is immediate. To prove the first let Q be the unitary matrix $I_k \oplus (-I_{n-k})$ and note that

$$A_{11} \oplus A_{22} = (A + QAQ^*)/2.$$

The result now follows from (4) and (2). Finally, to show (7), let $(A^{-1})_{22}$ be the $(n-k)\times(n-k)$ submatrix in the bottom right corner of A^{-1}. Then $(A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1} = (A^{-1})_{22}$ [10, §0.7.3]. So by (1), then (6), and finally (1) again, we have

$$\begin{aligned}\kappa_H(A_{22} - A_{21}A_{11}^{-1}A_{12}) &= \kappa_H((A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}) \\ &= \kappa_H((A^{-1})_{22}) \le \kappa_H(A^{-1}) = \kappa_H(A).\end{aligned}$$ □

We will generalize (4) and (6) in the next section. Note that if A has positive definite Hermitian part and B is a principal submatrix of A then, combining (3) and (6), we have $\kappa_2(B) \le \kappa_H(B) \le \kappa_H(A)$. That is, we have a bound on the 2-norm condition number of any principal submatrix of a matrix with positive definite Hermitian part.

Finally we give a perturbation result for the function f.

LEMMA 2.4. *Let $A = T + S$ have positive definite Hermitian part and let E be such that*

$$\|H^{-1}\|_2\|E\|_2 \le \tfrac{1}{2}.$$

Then

$$\|f(A) - f(A+E)\| \le 6\|E\|_2\,\|H^{-1}\|_2\,\|T + S^*H^{-1}S\|_2 = 6\|E\|_2\,\kappa_H(A). \tag{2.9}$$

Proof. Let $\epsilon\|H^{-1}\|_2 \le \frac{1}{2}$; then by standard arguments we have

$$\begin{aligned}(T - \epsilon I)^{-1} &= H^{-1/2}(I - \epsilon H^{-1})^{-1}H^{-1/2} \\ &\preceq H^{-1/2}\,(I - \epsilon\|H^{-1}\|_2 I)^{-1}\,H^{-1/2} \\ &\preceq H^{-1/2}\,[(1 + 2\epsilon\|H^{-1}\|_2)I]\,H^{-1/2} \\ &= (1 + 2\epsilon\|H^{-1}\|_2)\,H^{-1}.\end{aligned}$$

Let A and E satisfy the conditions of the lemma and set $\epsilon = \|E\|_2$. Then $\epsilon\|H^{-1}\|_2 \le \frac{1}{2}$. Define

$$F = (E + E^*)/2, \qquad G = (E - E^*)/2.$$

Then $\|F\|_2 \le \epsilon$ and $\|G\|_2 \le \epsilon$. Furthermore, the condition $\epsilon\|H^{-1}\|_2 \le \frac{1}{2}$ implies

$$1 + 2\epsilon\|H^{-1}\|_2 \le 2 \quad \text{and} \quad (2G^*H^{-1}G - \epsilon I) \preceq 0.$$

So

$$\begin{aligned}f(A+E) &= (T+F) \;+\; (S+G)^*(T+F)^{-1}(S+G) \\ &\preceq (T+\epsilon I) \;+\; (S+G)^*(T-\epsilon I)^{-1}(S+G) \\ &\preceq (T+\epsilon I) \;+\; (1+2\epsilon\|H^{-1}\|_2)(S+G)^*H^{-1}(S+G) \\ &= T + S^*H^{-1}S \;+\; 2\epsilon TH^{-1} - \epsilon I \;+\; 2\epsilon\|H^{-1}\|_2 S^*H^{-1}S \\ &\quad +(1+2\epsilon\|H^{-1}\|_2)\{S^*H^{-1}G + G^*H^{-1}S + G^*H^{-1}G\}\end{aligned}$$

$$\begin{aligned}
&\preceq T + S^*H^{-1}S \ + \ 2\epsilon\|H^{-1}\|_2\{T + S^*H^{-1}S\} - \epsilon I \\
&\quad +2\{S^*H^{-1}G + G^*H^{-1}S\} + 2G^*H^{-1}G \\
&= f(A) \ + \ 2\epsilon\|H^{-1}\|_2 f(A) + 2\{S^*H^{-1}G + G^*H^{-1}S\} \\
&\quad +(2G^*H^{-1}G - \epsilon I) \\
&\preceq f(A) \ + \ 2\epsilon\|H^{-1}\|_2 f(A) + 2\{S^*H^{-1}G + G^*H^{-1}S\}.
\end{aligned}$$

Now we will bound the norm of the final term in this expression.

$$\begin{aligned}
\|S^*H^{-1}G\|_2 &\leq \ \|S^*H^{-1/2}H^{-1/2}\|_2\|G\|_2 \\
&\leq \epsilon\|S^*H^{-1/2}\|_2\|H^{-1/2}\|_2 \\
&= \epsilon\sqrt{\|S^*H^{-1}S\|_2\|H^{-1}\|_2} \\
&\leq \epsilon\sqrt{\|T + S^*H^{-1}S\|_2\|H^{-1}\|_2} \\
&\leq \epsilon\|T + S^*H^{-1}S\|_2\|H^{-1}\|_2.
\end{aligned}$$

(For the final inequality we have used the fact that $\|T + S^*H^{-1}S\|_2\|H^{-1}\|_2 \geq 1$.) Thus,

$$\begin{aligned}
\lambda_{\max}(f(A+E) - f(A)) &\leq 2\epsilon\|H^{-1}\|\|H + S^TH^{-1}S\|_2 + 2\|S^*H^{-1}G + G^*H^{-1}S\|_2 \\
&= 6\epsilon\|H^{-1}\|\|H + S^TH^{-1}S\|_2.
\end{aligned}$$

A similar argument shows that

$$\lambda_{\min}(f(A+E) - f(E)) \ \geq \ -6\epsilon\|H^{-1}\|\|H + S^TH^{-1}S\|_2.$$

Combining these two we have the inequality in (2.9). The equality in (2.9) follows from the definition of $\kappa_H(A)$. □

A simpler approach to bounding $\|f(A+E) - f(A)\|_2$ is to bound $\|A^{-1} - (A+E)^{-1}\|_2$, then use the fact that $f(A)^{-1} = (A^{-1} + A^{-*})/2$. However, this gives an inequality of the form (2.9), but with $\kappa_H(A)$ replaced by $\kappa_H^2(A)$. Note that if we restrict A and E to be Hermitian, then we have a result which is stronger than (2.9):

$$\|f(A+E) - f(A)\|_2 = \|E\|_2,$$

regardless of the value of $\kappa_H(A)$. However, the bound (2.9) is quite satisfactory for our purposes since our results in Theorem 4.1, when restricted to Hermitian matrices, reduce to the bounds proved for Hermitian matrices in [14] (up to a constant).

3. Further inequalities. In this section we prove some additional inequalities that will not be used in §4. The first is a refinement of (2.3).

COROLLARY 3.1. *Let $A = T + S$ have positive definite Hermitian part. Then*

$$\alpha[H(A)]^{-1} \ \preceq \ H(A^{-1}) \ \preceq \ \beta[H(A)]^{-1}, \tag{3.1}$$

if and only if

$$\begin{aligned}
&\alpha \leq (1 + \max\{|\lambda| : \lambda \ \textit{is an eigenvalue of} \ H(A)^{-1}S(A)\})^{-1}, \\
&\beta \geq (1 + \min\{|\lambda| : \lambda \ \textit{is an eigenvalue of} \ H(A)^{-1}S(A)\})^{-1}.
\end{aligned} \tag{3.2}$$

Proof. By (2.1) the first inequality is equivalent to

$$\alpha H(A)^{-1} \preceq (H(A) + S(A)H(A)^{-1}S(A))^{-1}. \tag{3.3}$$

It is known that for positive definite $X, Y \in M_n$ we have $X \preceq Y$ if and only if the spectral radius of $X^{-1}Y$ is less than or equal to 1 [10, Thm. 7.7.3]. Using this fact and elementary manipulations one can show that (3.3) holds if and only if α is less than or equal to the right-hand side of (3.2). The proof of the second inequality is similar. □

The first inequality in (3.1) is Theorem 2 in [11]. However, the proof there depended on the fact that A was real; here it does not.

Recall that the *Hadamard product* of $A = [a_{ij}] \in M_n$ and $B = [b_{ij}] \in M_n$ is $A \circ B = [a_{ij}b_{ij}]$. Thus, the results in Theorem 2.2 (5) and Theorem 2.3 (6) may be stated as

$$f(A \circ B) \preceq f(A) \circ B \quad \forall\, A \in M_n(C), \qquad \text{with } H(A) \text{ positive definite,} \tag{3.4}$$

$$\kappa_H(A \circ B) \leq \kappa_H(A) \quad \forall\, A \in M_n(C), \qquad \text{with } H(A) \text{ positive definite} \tag{3.5}$$

where $B = J_k \oplus J_{n-k}$ (J_i is the $i \times i$ matrix of ones). In fact, (3.4) and (3.5) are true more generally, as is Theorem 2.3 (4). First we will provide some preliminary facts and definitions.

We call a norm $\|\cdot\|$ on M_n *monotone* if $\|A\| \leq \|B\|$ whenever A and B are positive semidefinite matrices with $A \preceq B$. We call a norm $\|\cdot\|$ *unitarily invariant* if $\|A\| = \|UAV\|$ if for any $A \in M_n$ and unitary $U, V \in M_n$. A unitarily invariant norm must be monotone. However, the monotone norm $\|A\| = \max |a_{ij}|$ is not unitarily invariant. Let $\|\cdot\|$ be a norm on M_n. Then we define $\kappa_{H\|\cdot\|}$ on the cone of $n \times n$ matrices with positive definite Hermitian part by

$$\kappa_{H\|\cdot\|}(A) = \|T + S^*H^{-1}S\|\ \|H^{-1}\|, \quad \text{where } H = H(A),\ S = S(A). \tag{3.6}$$

For $x \in R^n$ let $D_x \in M_n$ denote the diagonal matrix with i,i entry x_i. Let $A \in M_n$ and $x, y \in R^n$, then $A \circ (xy^*) = D_xAD_y$. We call a matrix $B \in M_n$ a *correlation matrix* if it is positive definite and its main diagonal entries are all 1. If $B \in M_n$ is a correlation matrix and $\|\cdot\|$ is a unitarily invariant norm then, by [2, eq. (33)] or [3, Cor. 2], one can show that for any $A \in M_n(C)$

$$\|A \circ B\| \leq \|A\| \tag{3.7}$$

and, by [1, Lem. 2], it follows that for any positive definite $H \in M_n$

$$(H \circ B)^{-1} \preceq H^{-1} \circ B. \tag{3.8}$$

We will extend the definition of f to

$$\mathcal{P}_n = \{A \in M_n :\ \text{range } S(A) \subset \text{ range } H(A), \text{ and } H(A) \text{ is positive semidefinite}\}$$

by

$$f(A) = H(A) + S(A)^*H(A)^{\dagger}S(A). \tag{3.9}$$

($A^{\dagger}$ denotes the Moore–Penrose inverse of A (see, e.g., [10, p. 421]).) Note that even though the function f is not continuous on $\mathcal{P}_n$ we do have

$$f(A) = \lim_{\epsilon \downarrow 0} f(A + \epsilon I) \quad \text{for all } A \in \mathcal{P}_n.$$

With this extended definition of f we no longer have Theorem 2.3, but we do have the inequality

$$f(XAX^*) \preceq Xf(A)X^* \qquad \text{for all } X \in M_n. \tag{3.10}$$

If one restricts A and B to be positive definite, then the following result is Theorem 10.C.3 in [12].

THEOREM 3.2. *Let A, $B \in M_n$ have positive definite Hermitian part and let $\|\cdot\|$ be a monotone norm on M_n. Then*

$$\kappa_{H\|\cdot\|}(A+B) \le \max\{\kappa_{H\|\cdot\|}(A), \kappa_{H\|\cdot\|}(B)\}.$$

Proof. The proof is essentially the same as that of Theorem 2.3(4) except that one uses the matrix convexity of the function $g(H) = H^{-1}$ (see, e.g., [12, pp. 470–471]) on the positive definite matrices to obtain the bound on $\|[(C+C^*)/2]^{-1}\|$. □

THEOREM 3.3. *Let $A \in M_n$ have positive definite Hermitian part and $B \in M_n$ be positive semidefinite. Then*

$$0 \preceq f(A \circ B) \preceq f(A) \circ B. \tag{3.11}$$

Proof. Let A, B satisfy the conditions of the theorem. The left-hand inequality is immediate. Because B is positive semidefinite we may write $B = \sum_{i=1}^n \lambda_i x_i x_i^*$ with $\lambda_i \ge 0$. So now by the convexity and homogeneity of f and the inequality (3.10)

$$\begin{aligned}
f(A \circ B) &\preceq f\left(A \circ \sum_{i=1}^n \lambda_i x_i x_i^*\right) \\
&= f\left(\sum_{i=1}^n \lambda_i A \circ (x_i x_i^*)\right) \\
&= f\left(\sum_{i=1}^n \lambda_i D_{x_i} A D_{x_i}\right) \\
&\preceq \sum_{i=1}^n \lambda_i f(D_{x_i} A D_{x_i}) \\
&\preceq \sum_{i=1}^n \lambda_i D_{x_i} f(A) D_{x_i} \\
&= f(A) \circ B.
\end{aligned}$$

□

It appears that the next result has not been observed except in the case where A is positive definite and B is the correlation matrix $J_k \oplus J_{n-k}$ [12, Thm. 10.D.2].

THEOREM 3.4. *Let $A \in M_n$ have positive definite Hermitian part, $B \in M_n$ be a correlation matrix, and $\|\cdot\|$ be any unitarily invariant norm on M_n. Then,*

$$\kappa_{H\|\cdot\|}(A \circ B) \le \kappa_{H\|\cdot\|}(A). \tag{3.12}$$

Proof. Let A and B satisfy the conditions of the theorem and let $\|\cdot\|$ be a unitarily invariant norm. Then, since by the previous result $f(A \circ B) \preceq f(A) \circ B$, taking norms and applying (3.7) we have

$$\|f(A \circ B)\| \leq \|f(A) \circ B)\| \leq \|f(A)\|.$$

Let $T = (A + A^*)/2$. So, by (3.8) and (3.7),

$$\|(T \circ B)^{-1}\| \leq \|H^{-1} \circ B\| \leq \|H^{-1}\|.$$

Combining these two inequalities gives (3.12). □

4. Computation of the LU factorization. In this section we analyze the backward stability of the outer product LU factorization algorithm without pivoting (described below) when applied to a matrix with positive definite Hermitian part using finite precision arithmetic. We will assume that all matrices are real in this section. (In this case, $(A + A^T)/2$, the *symmetric part* of A, is the same as the Hermitian part of A.) These results generalize those in [14] for positive definite matrices and the bounds for the *exact* LU factors of a matrix with positive definite Hermitian part in [9]. We assume the model of floating point arithmetic in [8, §2.4] and let u denote unit roundoff.

There are many reasons to avoid pivoting. We will only mention two; see [8] for a more complete discussion. First, block algorithms [8, § 3.2.11] perform better without pivoting. Second, pivoting will usually destroy sparsity.

Although we consider the outer product LU factorization algorithm, the gaxpy LU factorization algorithm, with the computations organized in the natural way (e.g., [8, Algorithm 3.2.4]), computes *exactly* the same LU factors in floating point arithmetic as the outer product algorithm. So the results are valid for the gaxpy algorithm also. The gaxpy algorithm is often preferred in practice (see [8, § 1.4.8]) for a discussion of some of the issues. Block LU factorization algorithms (see, e.g., [8, Algorithms 3.2.5, 3.2.6]) typically will not produce exactly the same computed LU factorization as (4.1), but one may expect the error analysis to produce similar conclusions since we have shown in §2 that $\kappa_2(B) \leq \kappa_H(A)$ for any submatrix of a positive definite matrix A.

The outer product algorithm that we will consider is

```
for k = 1 to n − 1
    for j = k + 1 to n
        a_jk = a_jk/a_kk
        for i = k + 1 to n
            a_ij = a_ij − a_jk a_ki
        end
    end
end
```

(4.1)

It runs to completion provided that at the kth stage $a_{kk} \neq 0$. The algorithm overwrites A with U and the strictly lower triangular part of L.

THEOREM 4.1. *Let $A \in M_n(R)$ have positive definite Hermitian part, and let $H = H(A)$ and $S = S(A)$. Then L and U, the exact* LU *factors of A, satisfy*

$$\| \, |L| \, |U| \, \|_F \leq n\|H + S^T H^{-1} S\|_2. \tag{4.2}$$

Let u be machine precision. If

$$24n^{3/2}\kappa_H(A)u \le 1 \tag{4.3}$$

then the LU *factorization algorithm* (4.1) *runs to completion and the computed factors, $\hat{L}$ and $\hat{U}$, satisfy*

$$\|\hat{L}\hat{U} - A\|_F \le 7un^{3/2}\|H + S^T H^{-1} S\|_2. \tag{4.4}$$

Proof. Let A be positive definite and partitioned as

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix}, \qquad A_{22} \in M_{n-1} \tag{4.5}$$

and let $\tilde{A} = A_{22} - A_{21}A_{11}^{-1}A_{12}$. Given a matrix X, let X_1 denote the first column of X, and $X_{(1)}$ denote the first row of X (note $X_{(1)}$ is $1 \times n$).

First we will prove the following inequalities, which will be used several times:

$$\|A_1\|_2^2 \le A_{11}\|H + S^T H^{-1} S\|_2 \quad \text{and} \quad \|A_{(1)}\|_2^2 \le A_{11}\|H + S^T H^{-1} S\|_2. \tag{4.6}$$

It suffices to prove the first inequality as the second inequality is the first with A replaced by A^T. Note that

$$A_1 = (H + S)_1 = [(H^{1/2} + SH^{-1/2})(H^{1/2})]_1 = (H^{1/2} + SH^{-1/2})(H^{1/2})_1.$$

Also, because S is skew-symmetric, we have $A_{11} = H_{11} + S_{11} = H_{11}$ and hence

$$\|H_1^{1/2}\|_2^2 = (H^{1/2}H^{1/2})_{11} = H_{11} = A_{11}.$$

So, combining these two results, and using the skew-symmetry of S for the final equality, we have

$$\begin{aligned} \|A_1\|_2^2 &= \|(H^{1/2} + SH^{-1/2})(H^{1/2})_1\|_2^2 \\ &\le \|H^{1/2} + SH^{-1/2}\|_2^2 \, \|H_1^{1/2}\|_2^2 \\ &= \|(H^{1/2} + SH^{-1/2})(H^{1/2} + SH^{-1/2})^*\|_2 A_{11} \\ &= A_{11}\|H + S^T H^{-1} S\|_2. \end{aligned}$$

We prove (4.2) by induction on n (it is also proved in [9] in another way). Let L and U be partitioned in the same way as A.

$$\begin{aligned} \| \, |L| \, |U| \, \|_F &= \| \, |L_1| \, |U_{(1)}| \, + \, |L_{22}| \, |U_{22}| \, \|_F \\ &\le \| \, |L_1| \, |U_{(1)}| \, \|_F \, + \, \| \, |L_{22}| \, |U_{22}| \, \|_F \\ &\le \| \, |L_1| \, \|_2 \| \, |U_{(1)}| \, \|_2 \, + \, (n-1)\|f(\tilde{A})\|_2 \\ &\le \|A_1/A_{11}\|_2 \, \|A_{(1)}\|_2 + (n-1)\|f(A)\|_2 \\ &\le \|H + S^T H^{-1} S\|_2 + (n-1)\|H + S^T H^{-1} S\|_2 \\ &= n\|T + S^* H^{-1} S\|_2. \end{aligned}$$

We have used the induction hypothesis for the second inequality, Theorem 2.2 for the next, and (4.6) for the last.

We now consider floating point arithmetic with precision u. Let $\hat{L}$ and $\hat{U}$ be the computed LU factors. Again we will use induction on the order of the matrices. It is clear that the assertions are true when $n = 1$. After one step of the LU factorization we have computed

$$\hat{L}_1 = fl(A_1/A_{11}) = L_1 + F, \quad \hat{U}_{(1)} = U_{(1)}, \quad \hat{\tilde{A}} = fl(A_{22} - \hat{L}_{21}\hat{U}_{21}) = \tilde{A} + E,$$

where E and F satisfy the componentwise bounds

$$|E| \le u(|A_{22}| + |A_{21}A_{12}/A_{11}|) \; + \; 2u|A_{21}A_{12}/A_{11}|, \tag{4.7}$$

$$|F| \le u|L_1| = u|A_1|/A_{11}. \tag{4.8}$$

(We have used $u^2 \le u$ to obtain the bound on $|E|$.) Note that by (4.6) and the fact that A_{21} is a submatrix of A_1 we have

$$\|A_{21}/\sqrt{A_{11}}\|_2^2 \le \|A_1/\sqrt{A_{11}}\|_2^2 \le \|H + S^T H^{-1} S\|_2.$$

Similarly,

$$\|A_{12}/\sqrt{A_{11}}\|_2^2 \le \|H + S^T H^{-1} S\|_2.$$

Thus

$$\begin{aligned} \|E\|_F &\le u\| \, |A_{22}| \, \|_F + 3u\| \, |A_{21}A_{12}/A_{11}| \, \|_F \\ &= u\|A_{22}\|_F + 3u\|A_{21}A_{12}/A_{11}\|_F \\ &\le u\sqrt{n}\|H + S^T H^{-1} S\|_2 + 3u\|H + S^T H^{-1} S\|_2 \\ &\le 4u\sqrt{n}\|H + S^T H^{-1} S\|_2, \end{aligned}$$

which implies

$$\|E\|_2 \; \le \; \|E\|_F \; \le \; 4u\sqrt{n}\|H + S^T H^{-1} S\|_2. \tag{4.9}$$

It is immediate from (4.8) and (4.6) that

$$\|F\|_2 \; \le \; u\|H + S^T H^{-1} S\|_2. \tag{4.10}$$

First we will show that if $H(A) \succeq 0$ and the condition (4.3), then the LU factorization runs to completion with positive pivots. Our proof is by induction. The case $n = 1$ is immediate. Assume that $A \in M_n(R)$ has $H(A) \succeq 0$ and that A satisfies (4.3). We will show that $\tilde{A} + E \in M_{n-1}$ has positive definite Hermitian part and also satisfies (4.3). To do this we must compute a bound on $\kappa_H(\tilde{A} + E)$.

$$\begin{aligned} \lambda_{\min}([(\tilde{A} + E) + (\tilde{A} + E)]/2) &\ge \lambda_{\min}([\tilde{A} + \tilde{A}]/2) - \|E\|_2 \\ &\ge \lambda_{\min}([\tilde{A} + \tilde{A}]/2) - 4u\sqrt{n}\|H + S^T H^{-1} S\|_2 \\ &= \lambda_{\min}([\tilde{A} + \tilde{A}]/2)(1 - 4u\sqrt{n}\kappa_H(\tilde{A})). \end{aligned}$$

Now using Theorem 2.2(4) and Lemma 2.4 for the second inequality, and the bound on $\|E\|_2$ for the third, we have

$$\begin{aligned} \|f(\tilde{A} + E)\|_2 &\le \|f(\tilde{A})\|_2 + \|f(\tilde{A} + E) - f(\tilde{A})\|_2 \\ &\le \|f(\tilde{A})\|_2 + 6\|E\|_2\kappa_H(\tilde{A}) \\ &\le \|f(\tilde{A})\|_2(1 + 12u\sqrt{n}\kappa_H(\tilde{A})) \; . \end{aligned} \tag{4.11}$$

Combining these two bounds, and then using the fact $\kappa_H(\tilde{A}) \leq \kappa_H(A)$ (Theorem 2.3) yields

$$\begin{aligned}\kappa_H(\tilde{A}+E) &\leq \kappa_H(\tilde{A})(1+12u\sqrt{n}\kappa_H(\tilde{A}))(1-4u\sqrt{n}\kappa_H(\tilde{A}))^{-1}\\ &\leq \kappa_H(A)(1+12u\sqrt{n}\kappa_H(A))(1-4u\sqrt{n}\kappa_H(A))^{-1}.\end{aligned}$$

We can now show that $\tilde{A}+E$ satisfies the condition (4.3):

$$\begin{aligned}24u\kappa_H(\tilde{A}+E) &\leq 24u(n-1)^{3/2}\kappa_H(A)\frac{1+12u\sqrt{n}\kappa_H(A)}{1-4u\sqrt{n}\kappa_H(A)}\\ &\leq 24u(n-1)^{3/2}\kappa_H(A)\frac{1}{(1-12u\sqrt{n}\kappa_H(A))(1-4u\sqrt{n}\kappa_H(A))}\\ &\leq 24u(n-1)^{3/2}\kappa_H(A)\frac{1}{1-16u\sqrt{n}\kappa_H(A)}\\ &= 24un^{3/2}\kappa_H(A)\sqrt{(n-1)/n}\frac{n-1}{n}\frac{1}{1-16u\sqrt{n}\kappa_H(A)}\\ &\leq \sqrt{(n-1)/n}\,\frac{n-1}{n-16un\sqrt{n}\kappa_H(A)}\\ &\leq \frac{n-1}{n-16un^{3/2}\kappa_H(A)}\\ &\leq 1.\end{aligned}$$

Thus we have shown that if A satisfies (4.3), then so does $\tilde{A}+E \in M_{n-1}$, and hence by induction the finite precision LU factorization will run to completion with positive pivots.

Finally, we show that under the same condition (4.3), we have (4.4). Using the inductive hypothesis, the bound (4.11), and the condition (4.3) (for the third inequality), we have

$$\begin{aligned}\|\hat{L}_{22}\hat{U}_{22}-(\tilde{A}+E)\|_F &\leq 5u(n-1)^{3/2}\|f(\tilde{A}+E)\|_2\\ &\leq 5u(n-1)^{3/2}(1+12\sqrt{n}u\kappa_H(A))\|H+S^TH^{-1}S\|_2\\ &\leq 5u(n-1)^{3/2}(1+1/2n)\|H+S^TH^{-1}S\|_2\\ &= 5u(n-1)\sqrt{n}\,\sqrt{(n-1)/n}\,(1+1/2n)\|H+S^TH^{-1}S\|_2\\ &\leq 5u(n-1)\sqrt{n}(1-1/2n)(1+1/2n)\|H+S^TH^{-1}S\|_2\\ &\leq 5u(n-1)\sqrt{n}\|H+S^TH^{-1}S\|_2.\end{aligned}$$

Also,

$$\begin{aligned}\|\hat{L}_1\hat{U}_{(1)}-L_1U_{(1)}\|_F &= \|FU_{(1)}\|_F\\ &\leq \|F\|_2\|U_{(1)}\|_2\\ &\leq u\|H+S^TH^{-1}S\|_2.\end{aligned}$$

We now combine these bounds to obtain (4.4):

$$\begin{aligned}\|\hat{L}\hat{U}-A\|_F &= \|\hat{L}\hat{U}-LU\|_F\\ &= \|\hat{L}_1\hat{U}_{(1)}+\hat{L}_{22}\hat{U}_{22}-L_1U_{(1)}-L_{22}U_{22}\|_F\\ &\leq \|\hat{L}_1\hat{U}_{(1)}-L_1U_{(1)}\|_F+\|\hat{L}_{22}\hat{U}_{22}-L_{22}U_{22}\|_F\end{aligned}$$

$$
\begin{aligned}
&= \|\hat{L}_1\hat{U}_{(1)} - L_1U_{(1)}\|_F + \|\hat{L}_{22}\hat{U}_{22} - \tilde{A}\|_F \\
&\le \|\hat{L}_1\hat{U}_{(1)} - L_1U_{(1)}\|_F + \|\hat{L}_{22}\hat{U}_{22} - (\tilde{A}+E)\|_F + \|E\|_F \\
&\le \sqrt{n}u\|H + S^TH^{-1}S\|_2 + 5\sqrt{n}(n-1)\|H + S^TH^{-1}S\|_2 \\
&\qquad + 4u\sqrt{n}\|H + S^TH^{-1}S\|_2 \\
&\le 5n^{3/2}u\|H + S^TH^{-1}S\|_2. \qquad \square
\end{aligned}
$$

COROLLARY 4.2. *Let $A \in M_n(R)$ have positive definite Hermitian part and suppose that the condition* (4.3) *holds. Then the computed* LU *factors of A satisfy*

$$
\| \, |\hat{L}| \, |\hat{U}| \, \|_F \;\le\; n[1 + 30un^{3/2}\kappa_H(A)] \, \|H + S^TH^{-1}S\|_2. \tag{4.12}
$$

Proof. By Theorem 4.1 we have $\hat{L}\hat{U} = A + E$ and

$$
\|E\|_2 \le \|E\|_F \le 5un^{3/2}\|H + S^TH^{-1}S\|_2.
$$

So, by the first part of Theorem 4.1, and then Lemma 2.4, we have the desired bound:

$$
\begin{aligned}
\| \, |\hat{L}| \, |\hat{U}| \, \|_F &\le \| \, |\hat{L}| \, \|_2 \, \| \, |\hat{U}| \, \|_F \\
&\le \|\hat{L}\|_F \, \|\hat{U}\|_F \\
&\le n\|f(A+E)\|_2 \\
&\le n\|f(A)\|_2 \;+\; n\|f(A+E) - f(A)\|_2 \\
&\le n\|H + S^TH^{-1}S\|_2 \;+\; 6n\|E\|_2\kappa_H(A) \\
&\le n\|H + S^TH^{-1}S\|_2 \;+\; 6n(5un^{3/2}\|H + S^TH^{-1}S\|_2)\kappa_H(A) \\
&\le n(1 + 30un^{3/2}\kappa_H(A)) \, \|H + S^TH^{-1}S\|_2. \qquad \square
\end{aligned}
$$

Now consider a linear system $Ax = b$ with $H(A)$ positive definite. Let $\hat{L}$ and $\hat{U}$ be the LU factors computed by algorithm (4.1), let $\hat{y}$ be the computed solution to $\hat{U}y = b$, and let $\hat{x}$ the solution to $\hat{L}z = \hat{y}$. Then combining the bound (4.12) with [8, Thm. 3.3.2] we know that $(A + E)\hat{x} = b$ where

$$
\begin{aligned}
\|E\|_2 &\le nu(3 + 5n + 150un^{3/2}\kappa_H(A))\|H + S^TH^{-1}S\|_2 + O(u^2) \\
&= \alpha_{n,u,\kappa_H(A)}\|H + S^TH^{-1}S\|_2 + O(u^2).
\end{aligned} \tag{4.13}
$$

Thus we have a rigorous a priori upper bound on the backward error of the solution computed without pivoting. Using the fact that if $Ax = b$ and $(A + F)y = b$, then (see, e.g., [10, eq. (5.8.7)])

$$
\frac{\|x - y\|_2}{\|x\|_2} \le \frac{\|E\|_2\|A^{-1}\|_2}{1 - \|E\|_2\|A^{-1}\|_2}, \tag{4.14}
$$

one can derive an a priori upper bound on the relative error in the computed solution $\hat{x}$.

Now let us compare $\hat{x}$ with $\hat{x}_{\text{piv}}$, the solution computed by Gaussian elimination with pivoting, in order to decide when it is worth pivoting. From [8, (3.4.3)] we have

$$
\|E_{piv}\|_2 \le nu(3\sqrt{n} + 5n^2\rho)\|A\| + O(u^2) = nu\beta_n\|A\|_2 + O(u^2), \tag{4.15}
$$

where ρ is the growth factor. Ignoring the factors $\alpha_{n,u,\kappa_H(A)}$ and $\beta_{n,\rho}$ the ratio of the bounds in (4.13) and (4.15) is $\|H + S^TH^{-1}S\|_2/\|A\|_2 \ge 1$. Thus if the ratio

$\|H+S^TH^{-1}S\|_2/\|A\|_2$ is not large it is reasonable to expect that $\hat{x}$ is not significantly worse that $\hat{x}_{piv}$ from the standpoint of backward error or relative error (in view of (4.14)). This is of course a heuristic because the inequalities (4.13) and (4.15) are worst case bounds. First we will show that $\|A\| \approx \|H+S^TH^{-1}S\|$ does not imply that one will obtain as accurate an answer without pivoting as with pivoting. Consider the contrived example

$$A = \begin{pmatrix} 1+1/\epsilon^2 & 0 & 0 \\ 0 & 1 & 1/\epsilon \\ 0 & -1/\epsilon & 1 \end{pmatrix}.$$

Then $\|A\|_2 = \|H+S^TH^{-1}S\|_2$, but for small $\epsilon > 0$ solving $Ax = b$ with pivoting will, in general, produce a considerably more accurate solution than without pivoting, while, on the other hand, a large value of $\|H + S^TH^{-1}S\|_2/\|A\|_2$ does not imply that Gaussian elimination without pivoting will give significantly worse results. For one thing we only use $\|H + S^TH^{-1}S\|_2$ to bound $\| \, |\hat{L}| \, |\hat{U}| \, \|_F$, but the fact that $\|H+S^TH^{-1}S\|_2$ is large does not imply that $\| \, |\hat{L}| \, |\hat{U}| \, \|_F$ is large. This is in contrast to the case when A is positive definite when we have

$$\|A\|_2 \leq \| \, |L| \, |U| \, \|_F \leq n\|A\|_2.$$

Also, a large value of $\| \, |\hat{L}| \, |\hat{U}| \, \|_F$ need not imply a large relative error. Both these points are illustrated by the numerical example in §3 of [9].

Acknowledgment. I am grateful to Izchak Lewkowicz for pointing out an error in the proof of Lemma 2.1.

REFERENCES

[1] T. Ando, *Concavity of certain maps on positive definite matrices and applications to Hadamard products*, Linear Algebra Appl., 26 (1979), pp. 203–241.

[2] T. Ando, R. A. Horn, and C. R. Johnson, *The singular values of a Hadamard product: A basic inequality*, Linear and Multilinear Algebra, 21 (1987), pp. 345–365.

[3] R. B. Bapat and V. S. Sunder, *On majorization and Schur products*, Linear Algebra Appl., 72 (1985), pp. 107–117.

[4] S. C. Eisenstat, H. C. Elman, and M. H. Schultz, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.

[5] K. Fan, *On real matrices with positive definite symmetric component*, Linear and Multilinear Algebra, 1 (1973), pp. 1–4.

[6] ———, *On strictly dissipative matrices*, Linear Algebra Appl., 9 (1974), pp. 223–241.

[7] S. Friedland and M. Katz, *On a matrix inequality*, Linear Algebra Appl., 85 (1987), pp. 185–190.

[8] G. Golub and C. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[9] ———, *Unsymmetric positive definite linear systems*, Linear Algebra Appl., 28 (1979), pp. 85–97.

[10] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, New York, 1985.

[11] C. R. Johnson, *An inequality for matrices whose symmetric part is positive definite*, Linear Algebra Appl., 6 (1973), pp. 13–18.

[12] A. W. Marshall and I. Olkin, *Inequalities: Theory of Majorization and its Applications*, Academic Press, London, 1979.

[13] R. C. Thompson, *Dissipative matrices and related results*, Linear Algebra Appl., 11 (1975), pp. 255–269.

[14] J. H. Wilkinson, *A priori error analysis of algebraic processes*, in Proc. Internat. Congress of Mathematicians, Moscow, pp. 629–640, 1968.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 655–658, April 1992

012

QR-LIKE ALGORITHMS FOR SYMMETRIC ARROW MATRICES*

PETER ARBENZ† AND GENE H. GOLUB‡

Abstract. It is shown that no QR-like algorithm exists for symmetric arrow matrices, i.e., for matrices whose elements vanish, except those on the diagonal and in the first row and column.

Key words. QR algorithm, arrow matrix, bordered diagonal matrix

AMS(MOS) subject classification. 65F15

1. Introduction.

The QR algorithm

$$\begin{aligned} T_{k-1} - \sigma_k I &=: Q_k R_k \quad \text{(QR decomposition)}, \\ T_k &:= R_k Q_k + \sigma_k I \end{aligned} \tag{1.1}$$

for computing the spectral decomposition of a symmetric tridiagonal matrix $T = T_0$ is well known [3], [6]. The algorithm computes a sequence $\{T_k\}_{k=1}^{\infty}$ of symmetric tridiagonal matrices similar to T converging to a diagonal matrix. Appropriately chosen *shifts* σ_k can increase the rate of convergence from linear to cubic [6, p.154].

It is also well known that for symmetric matrices, the band structure of a matrix is preserved by the QR algorithm. It remains an open question as to which matrix structures are preserved using the QR method described by (1.1).

A matrix A is said to be an *arrow* (or bordered diagonal [9, p.95]) matrix if it has the form

$$A = \begin{pmatrix} \times & \times & \cdots & \times \\ \times & \times & & \\ \vdots & & \ddots & \\ \times & & & \times \end{pmatrix}, \qquad a_{ij} = a_{ji} = 0 \quad \text{for } 1 < i < j.$$

In this paper, we show that there is in general *no* QR-like algorithm of the form described by (1.1) which generates a sequence $\{T_k\}$ of arrow matrices. By QR-like algorithm, we mean R_k need not be triangular but Q_k must be an orthogonal matrix which is the product of only *finitely* many Givens rotations. In the classical QR algorithm for symmetric tridiagonal matrices, Q_k is the product of $n-1$ Givens rotations. One step of any *practical* QR-like algorithm for symmetric arrow matrices must have complexity $O(n)$, because it is possible to transform a symmetric arrow matrix into a similar tridiagonal matrix in $O(n^2)$ flops [4].

Arrow matrices occur, e.g., in the course of the Lanczos method for solving the symmetric eigenvalue problem $A\mathbf{x} = \lambda\mathbf{x}$ [6]. In the Lanczos algorithm a series of

* Received by the editors July 16, 1990; accepted for publication (in revised form) November 20, 1990.

† Eidgenössiche Technische Hochschule, Institut für Wissenschaftliches Rechnen, 8092 Zürich, Switzerland (arbenz@inf.ethz.ch).

‡ Department of Computer Science, Stanford University, Stanford, California 94305 (na.golub@na-net.ornl.gov). The work of this author was supported in part by Army Research Office grant DAAL03-90-G-0105.

tridiagonal matrices T_j, $j \geq 1$, is computed. The eigenvalues of

$$T_j = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_{j-1} \\ & & \beta_{j-1} & \alpha_j \end{pmatrix} \in \mathbb{R}^{j \times j}$$

tend to the eigenvalues of A as j increases (cf. [6] for a discussion). With the last component of the eigenvectors of T_j the convergence of the algorithm can be controlled. Therefore, the eigenvalues and last components of the eigenvectors of $T_j\mathbf{x} = \lambda\mathbf{x}$ are to be computed for all j. Let $T_j = S_j\Theta_j S_j^T$ be the spectral decomposition of T_j. To make use of this knowledge in the computation of the spectral decomposition of T_{j+1}, we transform T_{j+1} similarly into (reversed) arrow form

$$A_{j+1} := (S_j \oplus 1)^T T_{j+1}(S_j \oplus 1) = \begin{pmatrix} \Theta_j & \beta_j \mathbf{s}_j^{(j)} \\ \beta_j \mathbf{s}_j^{(j)T} & \alpha_{j+1} \end{pmatrix}.$$

Here, $\mathbf{s}_j^{(j)T}$ is the last *row* of S_j. The components of the vector $\beta_j \mathbf{s}_j^{(j)}$ are used to check for convergence. Parlett and Nour-Omid [7] carefully investigate the solution of $A_{j+1}\mathbf{x} = \lambda\mathbf{x}$ in connection with the Lanczos algorithm.

O'Leary and Stewart [5] have described a similar algorithm to solve the eigenvalue problem for arrow matrices which arose in an application in molecular physics.

2. Analysis. Our analysis is based on the fact that T_k and T_{k-1} are similar by virtue of $T_k = Q_k^T T_{k-1} Q_k$. So, for our investigation, the form of R_k does not matter at all.

Let $A \in \mathbb{R}^{n \times n}$ be a symmetric and Q an orthogonal matrix such that

$$T := Q^T A Q \tag{2.1}$$

is arrow. Let H be the Householder reflector that maps the first column $\mathbf{q}_1$ of Q on a multiple of the first coordinate vector $\mathbf{e}_1$,

$$H\mathbf{q}_1 = \pm\|\mathbf{q}_1\|_2\mathbf{e}_1 = \pm\mathbf{e}_1, \qquad \mathbf{q}_1 = Q\mathbf{e}_1. \tag{2.2}$$

For notational convenience we choose the positive sign here. Then $Q_1 := H^T Q = HQ$ has the form

$$Q_1 = 1 \oplus \bar{Q}_1. \tag{2.3}$$

Note that $H\mathbf{e}_1 = \mathbf{q}_1$. Since the arrow matrix $T = Q_1^T HAHQ_1$, the columns of $\bar{Q}_1$ are the eigenvectors of the $(n-1) \times (n-1)$ matrix $\bar{A}$ obtained by deleting the first row and column from HAH. Therefore we have the following proposition.

PROPOSITION 2.1. *The set* $\{t_{22}, \cdots, t_{nn}\}$ *is uniquely determined by the first column* $\mathbf{q}_1$ *of* Q. □

The eigenvalues of $\bar{A}$ can be arranged arbitrarily on the diagonal of T.

Now let A itself be an arrow matrix:

$$A = \begin{pmatrix} \alpha & \mathbf{a}^T \\ \mathbf{a} & D \end{pmatrix}, \qquad D = \operatorname{diag}(d_1, \cdots, d_{n-1}).$$

We assume that A is *irreducible*, meaning that $d_i \neq d_j$, $i \neq j$, and that $\mathbf{a}$ has no zero component. Any arrow matrix can easily be transformed into irreducible form by a deflation procedure analogous to the one used in rank-one modified eigenvalue problems for symmetric tridiagonal matrices [1], [5]. The determinant of $A - \lambda I$ satisfies the equation [9, p. 95]

$$\frac{\det(A - \lambda I)}{\det(D - \lambda I)} = \alpha - \lambda - \sum_{i=1}^{n-1} \frac{a_i^2}{d_i - \lambda},$$

from which the interlacing properties

$$\lambda_i < d_i < \lambda_{i+1}, \qquad 1 \leq i < n, \tag{2.4}$$

for the simple eigenvalues $\lambda_1 < \cdots < \lambda_n$ of A follow immediately. (Here we have assumed that the d_i are arranged in increasing order.)

Now, let Q be orthogonal such that

$$\tilde{A} := Q^T A Q = \begin{pmatrix} \tilde{\alpha} & \tilde{\mathbf{a}}^T \\ \tilde{\mathbf{a}} & \tilde{D} \end{pmatrix}, \qquad \tilde{D} = \operatorname{diag}(\tilde{d}_1, \cdots, \tilde{d}_{n-1}), \tag{2.5}$$

is again arrow. If $\tilde{A}$ is irreducible and $\tilde{d}_1 < \cdots < \tilde{d}_{n-1}$, we have

$$\lambda_i < \tilde{d}_i < \lambda_{i+1}, \qquad 1 \leq i < n. \tag{2.6}$$

Combining (2.4) and (2.6) and setting $d_0 = -\infty$, $d_n = +\infty$, we get the inequalities

$$d_{i-1} < \tilde{d}_i < d_{i+1}, \qquad 1 \leq i \leq n-1.$$

From (2.5) we have that $AQ = Q\tilde{A}$, so that $A\mathbf{q}_{i+1} = \tilde{d}_i \mathbf{q}_{i+1} + \tilde{a}_i \mathbf{q}_1$. If $\tilde{a}_i = 0$ for some $i < n$, then $\mathbf{q}_{i+1}$ is an eigenvector of A.

Further, if $\mathbf{q}_1$ is orthogonal to an eigenvector $\mathbf{u}$ of A corresponding to λ, then $\lambda = \tilde{d}_i$ and $\mathbf{u} = \mathbf{q}_{i+1}$ for some i. In the extreme case where $\mathbf{q}_1$ itself is an eigenvector of A, $\tilde{\mathbf{a}} = \mathbf{0}$ and $\tilde{A}$ becomes diagonal.

We now turn to the actual point of interest. To that end, we assume that Q in (2.1) is the product of a *finite* number of Givens transformations,

$$Q = \prod_{k=1}^{m} J(p_k, q_k, \vartheta_k), \qquad p_k < q_k.$$

Then, because

$$J(i,j,\vartheta)J(1,k,\varphi) = \begin{cases} J(1,k,\varphi)J(i,j,\vartheta), & k \neq i < j \neq k, \\ J(1,i,\alpha_1)J(1,j,\alpha_2)J(i,j,\alpha_3), & k \in \{i,j\}, i < j, \end{cases}$$

we can rewrite Q in the form

$$Q = \prod_{k=1}^{m'} J(1, q_k', \vartheta_k') \prod_{k=m'+1}^{m''} J(p_k', q_k', \vartheta_k'), \qquad p_k' > 1.$$

The angles $\alpha_1, \alpha_2, \alpha_3$ are obtained by computing the Givens QR decomposition of a 3×3 matrix [3, p. 214]. As

$$J(1,j,\vartheta)J(1,k,\varphi) = \begin{cases} J(1,j,\vartheta)J(1,k,\varphi), & j < k, \\ J(1,j,\varphi+\vartheta), & j = k, \\ J(1,k,\alpha_1)J(1,j,\alpha_2)J(k,j,\alpha_3), & j > k, \end{cases}$$

we can proceed in a similar way to get the representation

$$Q = \prod_{k=1}^{n-1} J(1, k+1, \vartheta_k^*) \prod_{k=n}^{m^*} J(p_k^*, q_k^*, \vartheta_k^*), \qquad 1 < p_k^* < q_k^*.$$

The first product determines the first column $\mathbf{q}_1$ of Q. As the Householder matrix H in (2.2) has the same first column as Q, we have

$$H = \prod_{k=1}^{n-1} J(1, k+1, \vartheta_k^*) \prod_{k=n}^{m^{**}} J(p_k^{**}, q_k^{**}, \vartheta_k^{**}), \qquad 1 < p_k^{**} < q_k^{**}.$$

Thus

$$Q = H \left(\prod_{k=n}^{m^{**}} J(p_k^{**}, q_k^{**}, \vartheta_k^{**}) \right)^T \left(\prod_{k=n}^{m^*} J(p_k^*, q_k^*, \vartheta_k^*) \right) =: HQ_1.$$

Q_1 has the form (2.3), $Q_1 = 1 \oplus \bar{Q}$, where $\bar{Q}$ diagonalizes the matrix $\bar{A}$ obtained by deleting the first row and column from HAH. That means that $\bar{A}$ with *unknown* eigenvalues has been diagonalized with finitely many Givens rotations or equivalently that the $n-1$ unknown roots of the characteristic polynomial of $\bar{A}$ have been computed with a finite number of rational and root operations. According to Galois' theory, this is possible if and only if $n-1 \leq 4$ [8, p. 190]. Therefore we have the following theorem.

THEOREM 2.2. *Let $A \in \mathbb{R}^{n \times n}$, $n > 5$, with unknown spectrum. Let $\mathbf{q}_1 \neq \mathbf{e}_1$, $\|\mathbf{q}_1\|_2 = 1$, be a normalized vector. Then it is not possible to construct by finitely many Givens rotations an orthogonal matrix Q with first column $\mathbf{q}_1$ such that $Q^T AQ$ is arrow.* □

As a consequence of this theorem and our initial considerations, we have the following corollary.

COROLLARY 2.3. *There is no QR-like algorithm for irreducible arrow matrices of order greater than* 5. □

REFERENCES

[1] J. J. M. CUPPEN, *A divide and conquer method for the symmetric tridiagonal eigenproblem*, Numer. Math., 36 (1981), pp. 177–195.

[2] G. H. GOLUB, *Some modified matrix eigenvalue problems*, SIAM Rev., 15 (1973), pp. 318–334.

[3] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[4] W. B. GRAGG AND W. J. HARROD, *The numerically stable reconstruction of Jacobi matrices from spectral data*, Numer. Math., 44 (1984), pp. 317–335.

[5] D. O'LEARY AND G. W. STEWART, *Computing the eigenvalues and eigenvectors of symmetric arrowhead matrices*, J. Comput. Phys., 90 (1990), pp. 497–505.

[6] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[7] B. N. PARLETT AND B. NOUR-OMID, *The use of a refined error bound when updating eigenvalues of tridiagonals*, Linear Algebra Appl., 68 (1985), pp. 179–219.

[8] B. L. VAN DER WAERDEN, *Algebra* I, Eighth Edition, Springer-Verlag, Berlin, 1971.

[9] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 659–662, April 1992

013

A SHERMAN–MORRISON–WOODBURY IDENTITY FOR RANK AUGMENTING MATRICES WITH APPLICATION TO CENTERING*

KURT S. RIEDEL†

Abstract. Matrices of the form $\mathbf{A} + (\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}(\mathbf{V}_2 + \mathbf{W}_2)^*$ are considered where $\mathbf{A}$ is a *singular* $\ell \times \ell$ matrix and $\mathbf{G}$ is a nonsingular $k \times k$ matrix, $k \leq \ell$. Let the columns of $\mathbf{V}_1$ be in the column space of $\mathbf{A}$ and the columns of $\mathbf{W}_1$ be orthogonal to $\mathbf{A}$. Similarly, let the columns of $\mathbf{V}_2$ be in the column space of $\mathbf{A}^*$ and the columns of $\mathbf{W}_2$ be orthogonal to $\mathbf{A}^*$. An explicit expression for the inverse is given, provided that $\mathbf{W}_i^*\mathbf{W}_i$ has rank k. An application to centering covariance matrices about the mean is given.

Key words. linear algebra, Schur matrices, generalized inverses

AMS(MOS) subject classifications. 65R10, 33A65, 35K05, 62G20, 65P05

The well-known Sherman–Morrison–Woodbury matrix identity [1]:

$$(\mathbf{A} + \mathbf{X}_1\mathbf{G}\mathbf{X}_2^T)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{X}_1(\mathbf{G}^{-1} + \mathbf{X}_2^T\mathbf{A}^{-1}\mathbf{X}_1)^{-1}\mathbf{X}_2^T\mathbf{A}^{-1} \tag{1}$$

is widely used.[1] Several excellent review articles have appeared recently [2]–[4]. However, (1) is only valid when $\mathbf{A}$ is nonsingular. In this article, we consider matrix inverses of the form $\mathbf{A} + \mathbf{X}_1\mathbf{G}\mathbf{X}_2^T$ where the rank of $\mathbf{A} + \mathbf{X}_1\mathbf{G}\mathbf{X}_2^T$ is larger than the rank of $\mathbf{A}$.

We decompose the matrix $\mathbf{X}_1$ into $\mathbf{V}_1 + \mathbf{W}_1$, where the columns of $\mathbf{V}_1$ are contained in the column space of $\mathbf{A}$ and the columns of $\mathbf{W}_1$ are orthogonal to it. We denote the column space of $\mathbf{A}$ by $M(\mathbf{A})$. Similarly, we decompose $\mathbf{X}_2$ into $\mathbf{V}_2 + \mathbf{W}_2$, where the columns of $\mathbf{V}_2$ are contained in the column space of $\mathbf{A}^*$ and the columns of $\mathbf{W}_2$ are orthogonal to $M(\mathbf{A}^*)$. The Moore–Penrose generalized inverse will be denoted by the superscript $^+$. We denote the $k \times k$ matrix $\mathbf{W}_i^*\mathbf{W}_i$ by $\mathbf{B}_i$ and define $\mathbf{C}_i \equiv \mathbf{W}_i(\mathbf{W}_i^*\mathbf{W}_i)^{-1}$. We will require $\mathbf{B}_i$ to be nonsingular. However, the rank of the perturbation k can be significantly less than the size of the original matrix. We note that $\mathbf{V}_i^*\mathbf{W}_i = 0$ and $\mathbf{W}_i^*\mathbf{C}_i = \mathbf{I}_k$. Finally, the projection operator onto the column space of $\mathbf{W}$ satisfies $\mathbf{W}_i\mathbf{B}_i^{-1}\mathbf{W}_i^* = \mathbf{W}_1\mathbf{C}_1^* = \mathbf{C}_2\mathbf{W}_2^*$.

THEOREM 1. *Let $\mathbf{A}$ be an $\ell \times \ell$ matrix of rank ℓ_1, $\ell_1 < \ell$, $\mathbf{V}_i$ and $\mathbf{W}_i$ be $\ell \times k$ matrices and $\mathbf{G}$ be a $k \times k$ nonsingular matrix. Let the columns of $\mathbf{V}_1 \in M(\mathbf{A})$ and the columns of $\mathbf{W}_1$ be orthogonal to $M(\mathbf{A})$. Similarly, let the columns of $\mathbf{V}_2 \in M(\mathbf{A}^*)$ and the columns of $\mathbf{W}_2$ be orthogonal to $M(\mathbf{A}^*)$. Let $\mathbf{B}_i \equiv \mathbf{W}_i^*\mathbf{W}_i$ have rank k. $M(\mathbf{W}_1) = M(\mathbf{W}_2)$. The matrix,*

$$\Omega \equiv \mathbf{A} + (\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}(\mathbf{V}_2 + \mathbf{W}_2)^*$$

has the following Moore–Penrose generalized inverse:

$$\Omega^+ = \mathbf{A}^+ - \mathbf{C}_2\mathbf{V}_2^*\mathbf{A}^+ - \mathbf{A}^+\mathbf{V}_1\mathbf{C}_1^* + \mathbf{C}_2(\mathbf{G}^+ + \mathbf{V}_2^*\mathbf{A}^+\mathbf{V}_1)\mathbf{C}_1^*. \tag{2}$$

* Received by the editors July 12, 1990; accepted for publication (in revised form) February 19, 1991. This work was supported by U.S. Department of Energy grant DE-FG02-86ER53223.

† Courant Institute of Mathematical Sciences, New York University, New York, New York 10012 (reidel@mfdd4.nyu.edu).

[1] We denote the transpose of a matrix $\mathbf{A}$ by $\mathbf{A}^T$ and the hermitian or conjugate transpose by $\mathbf{A}^*$.

Proof. We recall that the Moore–Penrose inverse is the unique generalized inverse which satisfies the following four conditions [5, p. 26]:

(a) $\Omega\Omega^{+}\Omega = \Omega$,

(b) $\Omega^{+}\Omega\Omega^{+} = \Omega^{+}$,

(c) $(\Omega\Omega^{+})^{*} = \Omega\Omega^{+}$,

(d) $(\Omega^{+}\Omega)^{*} = \Omega^{+}\Omega$.

The identity is verified by direct computation,

$$\begin{aligned}\Omega\Omega^{+} \equiv{}& \mathbf{A}\mathbf{A}^{+} - \mathbf{A}\mathbf{C}_2\mathbf{V}_2^{*}\mathbf{A}^{+} - \mathbf{A}\mathbf{A}^{+}\mathbf{V}_1\mathbf{C}_1^{*} + \mathbf{A}\mathbf{C}_2(\mathbf{G}^{+} + \mathbf{V}_2^{*}\mathbf{A}^{+}\mathbf{V}_1)\mathbf{C}_1^{*}\\ &+(\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}(\mathbf{V}_2 + \mathbf{W}_2)^{*}\mathbf{A}^{+} - (\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}(\mathbf{V}_2 + \mathbf{W}_2)^{*}\mathbf{C}_2\mathbf{V}_2^{*}\mathbf{A}^{+}\\ &-(\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}\ (\mathbf{V}_2 + \mathbf{W}_2)^{*}\mathbf{A}^{+}\mathbf{V}_1\mathbf{C}_1^{*}\\ &+(\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}(\mathbf{V}_2 + \mathbf{W}_2)^{*}\mathbf{C}_2(\mathbf{V}_2^{*}\mathbf{A}^{+}\mathbf{V}_1)\mathbf{C}_1^{*}\\ &+(\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}(\mathbf{V}_2 + \mathbf{W}_2)^{*}\mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1^{*}.\end{aligned}$$

Since $\mathbf{W}_2$ is orthogonal to $\mathbf{A}^{*}$, we have $\mathbf{A}\mathbf{W}_2 = 0$, $\mathbf{W}_2^{*}\mathbf{A}^{+} = 0$, and $\mathbf{V}_2^{*}\mathbf{W}_2 = 0$, which simplifies the previous expression to

$$\begin{aligned}\Omega\Omega^{+} \equiv{}& \mathbf{A}\mathbf{A}^{+} - \mathbf{A}\mathbf{A}^{+}\mathbf{V}_1\mathbf{C}_1^{*} + +(\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}\mathbf{V}_2^{*}\mathbf{A}^{+}\\ &- (\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{V}_2^{*}\mathbf{A}^{+} - (\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}\ \ \mathbf{V}_2^{*}\mathbf{A}^{+}\mathbf{V}_1\mathbf{C}_1^{*}\\ &+(\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{V}_2^{*}\mathbf{A}^{+}\mathbf{V}_1\mathbf{C}_1^{*} + (\mathbf{V}_1 + \mathbf{W}_1)\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1^{*}.\end{aligned}$$

This expression may be simplified using $\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1^{*} = \mathbf{C}_1^{*}$, $\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{V}_2^{*} = \mathbf{G}\mathbf{V}_2^{*}$, and $\mathbf{A}\mathbf{A}^{+}\mathbf{V}_1 = \mathbf{V}_1$ to

$$\Omega\Omega^{+} \equiv \mathbf{A}\mathbf{A}^{+} + \mathbf{W}_1\mathbf{C}_1^{*},$$

and clearly condition (c) is satisfied.

The corresponding identity for $\Omega^{+}\Omega \equiv \mathbf{A}^{+}\mathbf{A} + \mathbf{C}_2\mathbf{W}_2^{*}$ requires the decomposition to satisfy $\mathbf{A}^{+}\mathbf{W}_1 = 0$, $\mathbf{W}_1^{*}\mathbf{A} = 0$, $\mathbf{V}_1^{*}\mathbf{W}_1 = 0$, and $\mathbf{V}_2\mathbf{A}^{+}\mathbf{A} = \mathbf{V}_2$. In addition, the matrix $\mathbf{G}$ must satisfy $\mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1^{*}\mathbf{W}_1\mathbf{G} = \mathbf{C}_2$ and $\mathbf{V}_1\mathbf{C}_1^{*}\mathbf{W}_1\mathbf{G} = \mathbf{V}_1\mathbf{G}$. These requirements guarantee that conditions (a), (b), and (d) are also satisfied. □

Remark. The conditions that $\mathbf{G}$ and $\mathbf{W}_i^{*}\mathbf{W}_i$ have rank k may be replaced by the somewhat weaker but more complicated conditions that $\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1^{*} = \mathbf{C}_1^{*}$, $\mathbf{G}\mathbf{W}_2^{*}\mathbf{C}_2\mathbf{V}_2^{*} = \mathbf{G}\mathbf{V}_2^{*}$, $\mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1^{*}\mathbf{W}_1\mathbf{G} = \mathbf{C}_2$ and $\mathbf{V}_1\mathbf{C}_1^{*}\mathbf{W}_1\mathbf{G} = \mathbf{V}_1\mathbf{G}$.

Note that the generalized inverse in (2) is singular and tends to infinity as $\mathbf{W}_i$ approaches zero. Thus (2) does not reduce to the (1) as the perturbation tends to zero. When the perturbation of the column space of $\mathbf{A}$ is zero, i.e., $\mathbf{V} \equiv 0$, Theorem 1 simplifies to

$$\Omega^{+} = \mathbf{A}^{+} + \mathbf{C}_2\mathbf{G}^{+}\mathbf{C}_1. \tag{3}$$

When $\mathbf{A}$ is a symmetric matrix, the column spaces of $\mathbf{A}$ and $\mathbf{A}^{*}$ are identical. Thus, for the case of symmetric $\mathbf{A}$ and Ω, Theorem 1 reduces to Theorem 2.

THEOREM 2. *Let $\mathbf{A}$ be a symmetric $\ell \times \ell$ matrix of rank ℓ_1, $\ell_1 < \ell$, $\mathbf{V}$ and $\mathbf{W}$ be $\ell \times k$ matrices, and $\mathbf{G}$ be a $k \times k$ nonsingular matrix. Let $\mathbf{V} \in M(\mathbf{A})$ and the columns of $\mathbf{W}$ be orthogonal to $M(\mathbf{A})$. Let $\mathbf{B} \equiv \mathbf{W}^{*}\mathbf{W}$ have rank k. The matrix*

$$\Omega \equiv \mathbf{A} + (\mathbf{V} + \mathbf{W})\mathbf{G}(\mathbf{V} + \mathbf{W})^{*}$$

has the following Moore–Penrose generalized inverse:

$$\Omega^+ = \mathbf{A}^+ - \mathbf{C}\mathbf{V}^*\mathbf{A}^+ - \mathbf{A}^+\mathbf{V}\mathbf{C}^* + \mathbf{C}(\mathbf{G}^+ + \mathbf{V}^*\mathbf{A}^+\mathbf{V})\mathbf{C}^*. \tag{4}$$

For concreteness, we specialize the preceding identities to the case of rank one perturbations. In this special case, $k \equiv 1$, and $\mathbf{V}_i$ and $\mathbf{W}_i$ reduce to ℓ vectors v_i and w_i. In the nonsingular case, (1) reduces to Bartlett's identity [6]. It states for an arbitrary nonsingular $\ell \times \ell$ matrix $\mathbf{A}$ and ℓ vectors v_i,

$$(\mathbf{A} + v_1 v_2^*)^{-1} = \mathbf{A}^{-1} - \frac{(\mathbf{A}^{-1}v_1)(v_2^*\mathbf{A}^{-1})}{(1 + v_2^*\mathbf{A}^{-1}v_1)}. \tag{5}$$

In this case, Theorem 1 reduces to the analogous result for an arbitrary singular matrix $\mathbf{A}$ with a rank one perturbation which contains a component perpendicular to the column space of $\mathbf{A}$. Noting that $\mathbf{G} \equiv 1$ and $\mathbf{C}_i \equiv w_i/|w_i|^2$, Theorem 1 simplifies to the following result.

THEOREM 3. *Let $\mathbf{A}$ be an $\ell \times \ell$ matrix of rank ℓ_1, $\ell_1 < \ell$, and v_i, w_i, $i = 1, 2$ be ℓ vectors. Let $v_1 \in M(\mathbf{A})$ and w_1 be orthogonal to $M(\mathbf{A})$, and $v_2 \in M(\mathbf{A}^*)$ and w_2 be orthogonal to $M(\mathbf{A}^*)$. Assume w_2 is parallel to w_1 and $w_i \neq 0$. Let*

$$\Omega \equiv \mathbf{A} + (v_1 + w_1)(v_2 + w_2)^*.$$

The Moore–Penrose generalized inverse is

$$\Omega^+ = \mathbf{A}^+ - \frac{w_2 v_2^* \mathbf{A}^+}{|w_2|^2} - \frac{\mathbf{A}^+ v_1 w_1^*}{|w_1|^2} + (1 + v_2^*\mathbf{A}^+ v_1)\frac{w_2 w_1^*}{|w_1|^2|w_2|^2}. \tag{6}$$

This generalized inverse is singular and tends to infinity as $1/|w_1||w_2|$, as w_i approaches zero. Thus (6) does not reduce to Bartlett's identity.

The projection operator onto the row space of Ω is

$$P_{X_T} = \mathbf{A}^+\mathbf{A} + \frac{w_i w_i^*}{|w_i|^2}.$$

The symmetric version of Theorem 3 was originally developed and applied by the author in his statistical analysis of magnetic fusion data [7]. To estimate the regression parameters in ordinary least squares regression, the sum of the squares and products (SSP) matrix needs to be inverted. We apply Theorem 3 to determine the inverse of the SSP matrix in terms of the inverse of the covariance matrix of the covariates.

We decompose the independent variable vector x into a mean value vector $\bar{x}$ and a fluctuating part $\tilde{x}$. Thus the ith individual observation has the form

$$x_i = \bar{x} + \tilde{x}_i.$$

Let $\mathbf{X}$ denote the $n \times \ell$ data matrix whose rows consist of x_i^T and let $\tilde{\mathbf{X}}$ be the centered data matrix whose rows consist of $\tilde{x}_i^T$.

We assume that some of the independent variables x_k have not been varied. Thus $\tilde{\mathbf{X}}^*\tilde{\mathbf{X}}$ is singular.

The inverse of the uncentered sum of squares and crossproducts matrix $\mathbf{X}^*\mathbf{X}$ can now be expressed in terms of the Moore–Penrose generalized inverse of the centered covariance matrix $\tilde{\mathbf{X}}^*\tilde{\mathbf{X}}$. We decompose a multiple of the mean value vector $\sqrt{n}\bar{x}$ into $v + w$, where $v \in M(\tilde{\mathbf{X}}^*\tilde{\mathbf{X}})$ and $w \perp M(\tilde{\mathbf{X}}^*\tilde{\mathbf{X}})$. The data matrix has the form

$$\mathbf{X}^*\mathbf{X} = \tilde{\mathbf{X}}^*\tilde{\mathbf{X}} + n\bar{x}\bar{x}^T = \tilde{\mathbf{X}}^*\tilde{\mathbf{X}} + (v + w)(v + w)^*.$$

Thus we have rewritten $\mathbf{X}^*\mathbf{X}$ in a form appropriate to the application of Theorem 3.

In conclusion, the application of these matrix identities requires the decomposition of $\mathbf{X}_i$ into the orthogonal components $\mathbf{V}_i$ and $\mathbf{W}_i$. Thus our theorems are most useful in situations where the decomposition is trivial.

Acknowledgments. The helpful comments of the referees are gratefully acknowledged.

Note added in proof. A first order approximation to the matrix identity given in Theorem 1 in the limit of small perturbing matrices is given in equation (3.24) of [8].

REFERENCES

[1] W. J. DUNCAN, *Some devices for the solution of large sets of simultaneous equations (with an appendix on the reciprocation of partitioned matrices)*, The London, Edinburgh and Dublin Philosophical Magazine and Journal of Science, Seventh Series, 35 (1944), pp. 660–670.

[2] H. V. HENDERSON AND S. R. SEARLE, *On deriving the inverse of a sum of matrices*, SIAM Rev., 23 (1981), pp. 53–60.

[3] D. V. OUELLETE, *Schur complements and statistics*, J. Linear Algebra, 36 (1981), pp. 187–295.

[4] W. W. HAGER, *Updating the inverse of a matrix*, SIAM Rev., 31 (1989), pp. 221–239.

[5] C. R. RAO, *Linear Statistical Inference and Its Applications*, John Wiley and Sons, New York, 1973.

[6] M. S. BARTLETT, *An inverse matrix adjustment arising in discriminant analysis*, Ann. Math. Statist., 22 (1951), pp. 107-111.

[7] K. S. RIEDEL, *Advanced statistics for Tokamak transport: Collinearity and Tokamak to Tokamak variation*, New York University Report MF-118, National Technical Information Service Document no. DOE/ER/53223-98, 1989.

[8] G. W. STEWART, *On the perturbation of pseudo-inverses, projections and linear least squares*, SIAM Rev., 19 (1977), pp. 634–663.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 663–670, April 1992

014

ON THE RELATIONSHIP BETWEEN OVERLAPPING AND NONOVERLAPPING DOMAIN DECOMPOSITION METHODS*

TONY F. CHAN† AND DANNY GOOVAERTS‡

Abstract. It is proven that the two apparently different approaches in domain decomposition, namely the Schwarz-type overlapping domain algorithms and the Schur complement-type nonoverlapping algorithms, are essentially the same: for any given Schwarz algorithm there corresponds a Schur complement algorithm, with a particular preconditioner, which produces the same iterates on the interfaces. This observation was first made by Bjørstad and Widlund [*SIAM J. Sci. Statist. Comput.*, 10 (1989), pp. 1053–1061], who showed that a result of Chan for Schur complement-type preconditioners [T. F. Chan and D. Resasco, *Analysis of domain decomposition preconditioners on irregular regions*, in Advances in Computer Methods for Partial Differential Equations,VI, R. Vichnevetsky and R. Stepleman, eds., IMACS, 1987, pp. 317–322] can be applied to a related Schwarz-type iteration. This paper gives a different proof using a new characterization of the two algorithms as two different methods for solving the reduced interface problem, which also allows immediate generalizations to other more complicated domains with more than two interior interfaces.

Key words. domain decomposition, Schwarz alternating procedure, Schur complement preconditioners, overlapping domains, nonoverlapping domains

AMS(MOS) subject classification. 65F10

1. Introduction.

Domain decomposition for solving elliptic partial differential equations has been a very active area of research in the past several years [10], [5]. A main motivation has been the potential of parallelism. This type of algorithm is also ideally suited for local adaptive mesh refinement, exploiting fast solvers on regular subdomains and coupling different mathematical models in different subdomains.

There are two main approaches to decomposing the domain. The first is to use overlapping subdomains that cover the original domain. Variants of the Schwarz alternating procedure are then used to compute the solution iteratively via solving subdomain problems [13], [14], [15], [16]. The second approach is to decompose the domain into nonoverlapping subdomains by interfaces. The original problem is then reduced to an equivalent one defined on the interfaces, and solved by some iterative procedure (such as fixed point iteration and the preconditioned conjugate gradient method), usually with some sort of preconditioning [1], [3], [4], [9], [11], [12].

At first sight, these two approaches do not seem to be related. In fact their research directions have been proceeding in a parallel but noninteracting fashion. One of the main purposes of this paper is to bring to light the surprising fact that the two approaches are *identical* under certain conditions. To the best of our knowledge, this insightful observation was first made by Bjørstad and Widlund [2], when they mentioned to Chan that a result of Chan and Resasco [7] concerning the convergence rate of a Schur complement-type iteration for domains with two interfaces also applies

* Received by the editors August 19, 1988; accepted for publication (in revised form) December 10, 1990. This research was supported in part by Department of Energy contract DE-FG03-87ER25037, National Science Foundation contract NSF-DMS87-14612 and Army Research Office contract DAAL03-88-K-0085.

† Department of Mathematics, University of California, Los Angeles, California 90024 (chan@math.ucla.edu or na.tchan@na-net.stanford.edu).

‡ Alcatel Bell, F. Wellesplein 1, B-2018 Antwerp, Belgium. This research was sponsored by a Senior Research Assistantship of the National Fund for Scientific Research of Belgium (NFWO/FNRS). Part of this work was performed when the author was visiting the Department of Mathematics at University of California, Los Angeles, California 90024 in 1988 with a travel grant from the National Fund for Scientific Research of Belgium (NFWO/FNRS).

to a corresponding Schwarz-type iteration. Subsequently, Bjørstad showed Chan a sketch of a proof of the equivalence of the two iterations for this decomposition. In this paper, we give a different proof based on a new characterization of the two algorithms as two different methods for solving the reduced problem for the interfaces. More important, the new characterization allows the immediate generalization of the equivalence of these two types of iterations to more complicated domains with more than two interior interfaces (such as T-shaped and C-shaped ones). Some of these generalizations will be discussed in this paper.

For simplicity we shall first restrict our attention to cases where the original domain Ω is decomposed into overlapping subdomains with two artificial interior interfaces, say, Γ_4 and Γ_5, such as the three examples in Fig. 1. Note that the domain in Fig. 1(c) is L-shaped. Now consider reducing the original problem on Ω to an equivalent one on Γ_4 and Γ_5 (in the discrete case this can be accomplished by formally performing block Gaussian elimination to eliminate all the other interior unknowns). We can think of this reduced problem as a block 2×2 system with the unknowns u_4 on Γ_4 and u_5 on Γ_5. Our proof is based on three simple observations:

1. Schwarz's (overlapping) procedure corresponds to a block Gauss–Seidel iterative procedure for solving this interface problem.
2. The Schur complement (nonoverlapping) approach corresponds to further eliminating one of the unknowns (say, u_4) and solving the reduced problem in u_5 alone by a fixed point iteration with an appropriate diagonal block splitting (or preconditioner).
3. A simple lemma in linear algebra that says that the two ways of solving the block 2×2 system in (1) and (2) above produce identical iterates.

2. Formulation for two interfaces. We shall follow the formulation and notation used in [7]. Consider the linear system arising from a discretization of a linear elliptic problem $Lu = f$ on any one of the domains Ω in Fig. 1, where L is a linear elliptic second-order partial differential operator. We assume that the interfaces decouple the unknowns in the interior of neighboring subdomains (for large stencils the interfaces may have to consist of more than one grid line). If we order the unknowns corresponding to Ω_1, Ω_2, and Ω_3 before those corresponding to Γ_4 and Γ_5, we arrive at a linear system

$$Au = f,$$

where

$$A = \begin{pmatrix} A_{11} & 0 & 0 & A_{14} & 0 \\ 0 & A_{22} & 0 & A_{24} & A_{25} \\ 0 & 0 & A_{33} & 0 & A_{35} \\ A_{41} & A_{42} & 0 & A_{44} & A_{45} \\ 0 & A_{52} & A_{53} & A_{54} & A_{55} \end{pmatrix}, \tag{1}$$

and $u = (u_1, u_2, u_3, u_4, u_5)^T, f = (f_1, f_2, f_3, f_4, f_5)^T$. By eliminating the unknowns u_1, u_2, and u_3 in (1), we obtain the following reduced problem in u_4 and u_5:

$$\begin{pmatrix} M_4 & A_{45} - A_{42}A_{22}^{-1}A_{25} \\ A_{54} - A_{52}A_{22}^{-1}A_{24} & M_5 \end{pmatrix} \begin{pmatrix} u_4 \\ u_5 \end{pmatrix} = \begin{pmatrix} g_4 \\ g_5 \end{pmatrix}, \tag{2}$$

where

$$M_4 = A_{44} - A_{41}A_{11}^{-1}A_{14} - A_{42}A_{22}^{-1}A_{24},$$

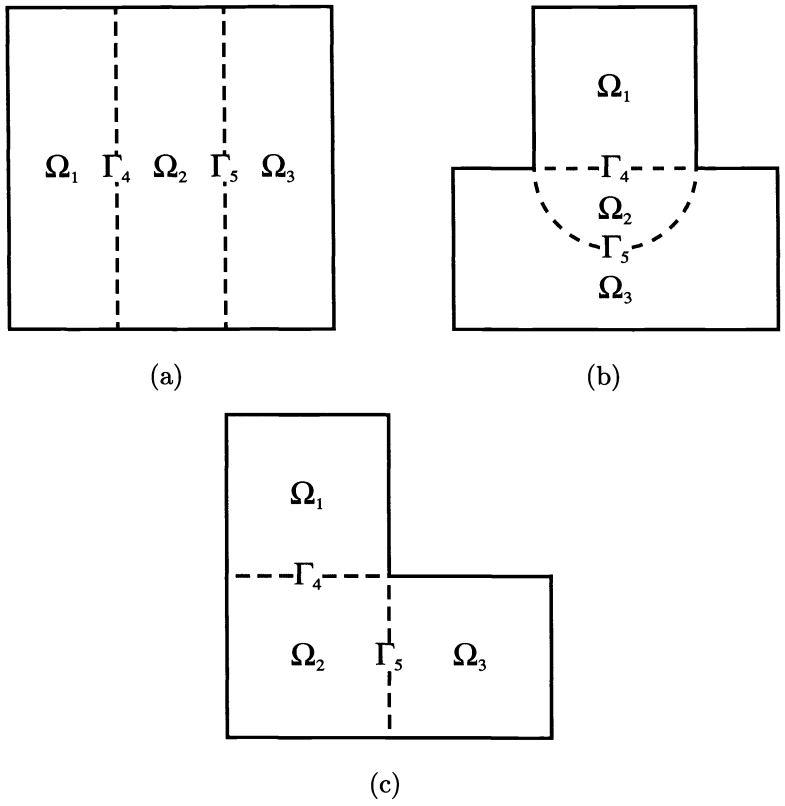


FIG. 1. *Domain decomposition with two interior interfaces.*

$$M_5 = A_{55} - A_{52}A_{22}^{-1}A_{25} - A_{53}A_{33}^{-1}A_{35},$$
$$g_4 = f_4 - A_{41}A_{11}^{-1}f_1 - A_{42}A_{22}^{-1}f_2,$$
$$g_5 = f_5 - A_{52}A_{22}^{-1}f_2 - A_{53}A_{33}^{-1}f_3.$$

As we shall show in the next two sections, the Schur complement and Schwarz algorithms can be viewed as two different ways of solving (2).

3. Schur complement domain decomposition. In this section we briefly describe the Schur complement algorithm for solving (1). The approach is to solve a reduced problem on one interface (say, Γ_5), which divides the domain Ω into two subdomains (in this case $\Omega_{12} \equiv \Omega_1 \cup \Gamma_4 \cup \Omega_2$ and Ω_3).[1] This reduced problem, which we shall denote by

$$C_5 u_5 = p_5, \tag{3}$$

is in fact the system obtained from (1) by taking the Schur complement of A_{55} in A. For a given vector v defined on Γ_5, the matrix-vector product $C_5 v$ can be computed via solutions of the original problem on Ω_{12} and Ω_3. Given a preconditioner Q for C_5 (or equivalently a splitting $C_5 = Q + (C_5 - Q)$), we can then define the following

[1] We shall use the notation $\Omega_{ij\ldots k}$ to denote the union of $\Omega_i \cup \Omega_j \cup \cdots \Omega_k$ and all the interfaces between the subdomains.

iteration for solving (3):

$$Qw \leftarrow (Q - C_5)w + p_5.$$

The particular preconditioner Q of interest here is that proposed in [4], which takes Q to be the *exact* interface operator on Γ_5 for the domain $\Omega_{23} \equiv \Omega_2 \cup \Gamma_5 \cup \Omega_3$. In our notation this corresponds to $Q = M_5$. We emphasize that if Ω_{23} is rectangular and L has constant coefficients, then M_5 can be inverted efficiently using fast Fourier transforms on Γ_5 [9].

Note that (3) can also be derived by eliminating the unknown u_4 in (2), which yields

$$C_5 = M_5 - (A_{54} - A_{52}A_{22}^{-1}A_{24})M_4^{-1}(A_{45} - A_{42}A_{22}^{-1}A_{25}).$$

Thus the Schur complement method described above can also be viewed as a method for solving the interface system (2), which proceeds by first forming the Schur complement system for u_5 and then performing an iteration on this system with the diagonal block M_5 as a preconditioner. This is precisely the formulation used in [7] and it will be used later to relate to the Schwarz iteration.

4. Schwarz domain decomposition. In this section we shall briefly describe the Schwarz algorithm and relate it to a particular block Gauss–Seidel iteration for solving (2).

The classical Schwarz algorithm can be summarized as follows:

1. Start with an initial guess on one of the interfaces, say, u_5^0 on Γ_5.
2. Solve the problem $Lu = f$ on Ω_{12} using the boundary conditions $u = u_5^0$ on Γ_5 together with the original boundary condition on the rest of the boundary.
3. As part of the solution from step (2), obtain a guess u_4^0 on Γ_4.
4. Solve the problem $Lu = f$ on Ω_{23} using the boundary conditions $u = u_4^0$ on Γ_4 together with the original boundary condition on the rest of the boundary.
5. As part of the solution in step (4), obtain a new guess for u_5^1, which completes one iteration of the method.

It can be easily verified that u_4^0 is obtained by solving the following subproblem of (1):

$$\begin{pmatrix} A_{11} & 0 & A_{14} \\ 0 & A_{22} & A_{24} \\ A_{41} & A_{42} & A_{44} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ u_4^0 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 - A_{25}u_5^0 \\ f_4 - A_{45}u_5^0 \end{pmatrix}. \tag{4}$$

Thus u_4^0 is the solution to the Schur complement system of (4):

$$M_4 u_4^0 = f_4 - A_{45}u_5^0 - A_{41}A_{11}^{-1}f_1 - A_{42}A_{22}^{-1}(f_2 - A_{25}u_5^0).$$

Similarly, u_5^1 is obtained from

$$M_5 u_5^1 = f_5 - A_{54}u_4^0 - A_{53}A_{33}^{-1}f_3 - A_{52}A_{22}^{-1}(f_2 - A_{24}u_4^0).$$

Rearranging the above two equations, we have

$$M_4 u_4^0 + (A_{45} - A_{42}A_{22}^{-1}A_{25})u_5^0 = g_4,$$

$$(A_{54} - A_{52}A_{22}^{-1}A_{24})u_4^0 + M_5 u_5^1 = g_5.$$

Referring to (2), we can see that the Schwarz iteration is precisely a block Gauss–Seidel iteration for the 2×2 two-interface system.

5. The equivalence of Schur complement and Schwarz. We have now shown that the Schur complement algorithm and the Schwarz algorithm are two different iterative methods for solving the interface system (2). We shall next give a simple lemma in linear algebra, which states that these two apparently different iterations produce identical iterates.

LEMMA 5.1. *Consider the block* 2×2 *linear system*

$$\begin{pmatrix} A & B \\ C & D \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}$$

and its solution by the following two iterations with the same initial guess $y = y^0$*:*

1. (*Schur complement*) $y^1 \leftarrow D^{-1}(CA^{-1}By^0 + g - CA^{-1}f)$,
2. (*Block Gauss–Seidel*) $x^0 \leftarrow A^{-1}(f - By^0), y^1 \leftarrow D^{-1}(g - Cx^0)$.

Then the two iterations produce exactly the same iterates for y.

Proof. Eliminate x^0 in the second algorithm. □

We have therefore proved the following theorem.

THEOREM 5.2. *Starting with the same initial guess on* Γ_5*, the Schwarz algorithm (as defined in* §4*) and the Schur complement algorithm with the preconditioner* M_5 *(as defined in* §3*) produce the same iterates.*

Remarks. 1. The choice of Γ_5 over Γ_4 for the Schur complement algorithm is arbitrary. A similar equivalence result would hold if Γ_4 had been chosen. The equivalent Schwarz iteration is the same as before except now we start with an initial guess on Γ_4. Moreover, since the ordering of the blocks does not affect the convergence behavior of the block Gauss–Seidel iteration for the 2×2 system (2), the two choices have effectively the same convergence rate (more precisely, for an iteration of one starting with a given initial guess, there corresponds an initial guess for the other iteration such that the same iterates are produced). In other words, for the Schur complement algorithm, it does not matter which of Γ_4 and Γ_5 we use to decompose the domain. This slightly sharpens and generalizes a similar result in [7].

2. When L is self-adjoint, the Schur complements C_5 and M_5 are symmetric and positive definite and the iteration (4) can be accelerated using conjugate gradients. From the equivalence result, it follows immediately that the Schwarz iterative procedure, considered as an iterative method for u_5, can be accelerated in the same way and will yield identical results.

3. While the convergence behavior of the two algorithms is the same, their costs per iteration could be different. For the Schwarz algorithm, each iteration requires solves on two *overlapping* subdomains, which implies two solves on the overlapped region in each iteration step, whereas for the Schur complement algorithm only two *nonoverlapping* solves are required. The cost of implementing the preconditioner is extra for the Schur complement algorithm. However, for a large class of elliptic operators on rectangular domains, with appropriate boundary conditions, this only costs two FFTs on the interface [9]. Therefore, for these problems the Schur complement algorithms are more efficient. In effect, the analytical information built into the preconditioner alleviates the need for solving the problem over the overlapped region twice.

4. Since the two algorithms are the same, theoretical convergence results for one immediately apply to the other. For example, the shape and mesh size independent convergence results for the Schur complement algorithm on L-shaped domains established in [7] automatically apply to the corresponding Schwarz iteration. On the other hand, the many nice characterizations (variational and via the maximum principle)

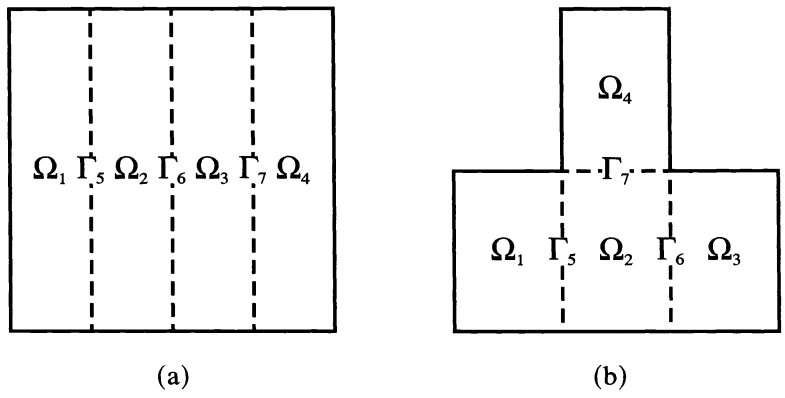


FIG. 2. *Domain decomposition with three interior interfaces.*

for the Schwarz iteration [13], [14] often allow an easy derivation of the theoretical results [6].

5. The type of boundary conditions for the original problem on Ω does not matter since the equivalence result is proved for any discretization of the problem having the form (1).

6. Extensions to geometries with more interfaces. The characterization of the Schwarz and Schur complement algorithms in terms of two different iterations for solving the interface system (2) allows immediate generalizations of the equivalence result to geometries with more than two interior interfaces, such as T-shaped and C-shaped domains. We shall consider a few examples here.

Consider the domains in Fig. 2, with overlapping decompositions resulting in three interior interfaces. In particular, consider the T-shaped domain in Fig. 2(b). Proceeding as before, we reduce the problem to a block 3×3 system coupling the three interior interfaces Γ_5, Γ_6, and Γ_7. Treating two of the three interfaces as a block, we can view this as a block 2×2 system to which Lemma 5.1 can be applied. For example, if we group Γ_5 and Γ_6 together, then a block Gauss–Seidel iteration starting with an iterate on Γ_7 would correspond to a Schwarz iteration involving the overlapping domains Ω_{123} and Ω_{24}. The equivalent Schur complement algorithm would correspond to using Γ_7 to decompose Ω into two subdomains Ω_4 and Ω_{123} and using as preconditioner on Γ_7 the *exact* Schur complement of Γ_7 in Ω_{24}. This particular preconditioner was used in [9]. Note that the Schur complement algorithm requires only solves on Ω_4 and Ω_{123}.

By interchanging the role of Γ_7 and $\Gamma_5 \cup \Gamma_6$ above, we can obtain another set of algorithms. The Schwarz algorithm is defined on the same overlapping subdomains as before except that now we start the iteration on $\Gamma_5 \cup \Gamma_6$. The equivalent Schur complement algorithm solves the reduced problem for $\Gamma_5 \cup \Gamma_6$ with a preconditioner corresponding to the exact Schur complement of $\Gamma_5 \cup \Gamma_6$ in Ω_{123} (such preconditioners have been derived in [8]). This particular Schur complement algorithm requires solves on Ω_{24}, Ω_1, and Ω_3.

Altogether, there are six sets of possible algorithms for T-shaped domains corresponding to three ways of grouping the 3×3 interface system into 2×2 ones and two orderings for performing the Schwarz iteration.

There are still more possibilities. For example, we can perform a block Gauss–

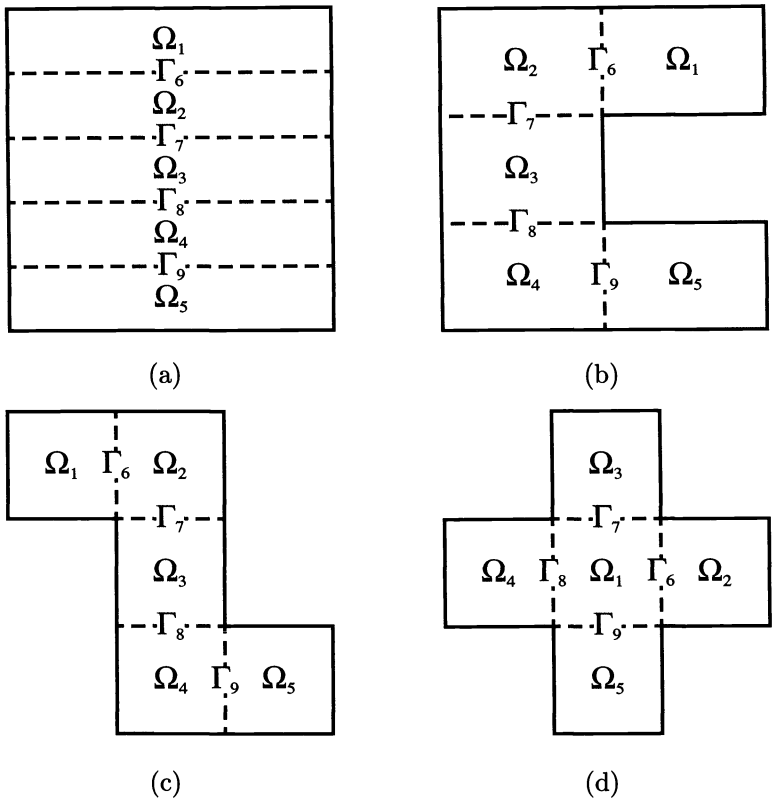


FIG. 3. *Domain decomposition with four interior interfaces.*

Seidel iteration on the 3×3 interface system directly without first regrouping it into a block 2×2 one. Starting with iterates on $\Gamma_6 \cup \Gamma_7$, for example, we can update in sequence the iterates on Γ_5, Γ_6, and Γ_7. The corresponding Schwarz iteration first does a subdomain solve on Ω_{12}, followed by a solve on Ω_{23}, and finally a solve on Ω_{24}. The equivalent Schur complement algorithm corresponds to solving the reduced interface problem on $\Gamma_6 \cup \Gamma_7$ with a preconditioner corresponding to a block Gauss–Seidel splitting of the interface system. This requires the inversion of the exact Schur complements of Γ_6 in Ω_{23} and Γ_7 in Ω_{24}.

Generalizations to overlapping decompositions with more interior interfaces are similar. Figure 3 gives some examples of domains with four interfaces. Consider in particular the domains in Figs. 3(a), 3(b), and 3(c). The reduced interface problem is now block 4×4 because there are four interior interfaces. Regrouping this into a block 2×2 system, say, with $\Gamma_6 \cup \Gamma_9$ in one group and $\Gamma_7 \cup \Gamma_8$ in another group, we obtain a Schwarz iteration involving the overlapping subdomains Ω_{234} and $\Omega_{12} \cup \Omega_{45}$. The equivalent Schur complement algorithm divides Ω by $\Gamma_6 \cup \Gamma_9$ (or $\Gamma_7 \cup \Gamma_8$) and solves the reduced interface system for $\Gamma_6 \cup \Gamma_9$ with a block diagonal preconditioner, one block corresponding to the exact Schur complement of Γ_6 in Ω_{12} and the other block corresponding to the exact Schur complement of Γ_9 in Ω_{45}. This particular Schur complement algorithm was also considered in [7]. There are many other variants, such as block red/black ordering for grouping the interfaces, but we shall not discuss this further here.

Finally, we emphasize that the remarks in §5 also apply to decompositions with more than two interfaces.

Acknowledgment. The first author acknowledges helpful discussions with Petter Bjørstad and Olof Widlund.

REFERENCES

[1] P. E. BJØRSTAD AND O. B. WIDLUND, *Iterative methods for the solution of elliptic problems on regions partitioned into substructures*, SIAM J. Numer. Anal., 23 (1986), pp. 1097–1120.

[2] ———, Jan. 1988, private communication.

[3] J. H. BRAMBLE, J. E. PASCIAK, AND A. H. SCHATZ, *An iterative method for elliptic problems on regions partitioned into substructures*, Math. Comp., 46 (1986), pp. 361–369.

[4] T. F. CHAN, *Analysis of preconditioners for domain decomposition*, SIAM J. Numer. Anal., 24 (1987), pp. 382–390.

[5] T. F. CHAN, R. GLOWINSKY, J. PERIAUX, AND O. WIDLUND, EDS., *Proc. Second International Symposium on Domain Decomposition Methods, University of California, Los Angeles, CA, Jan.* 14–16, 1988, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1989.

[6] T. F. CHAN, T. Y. HOU, AND P. L. LIONS, *Geometry related convergence results for domain decomposition algorithms*, SIAM J. Numer. Anal. , 28 (1991), pp. 378–391.

[7] T. F. CHAN AND D. RESASCO, *Analysis of domain decomposition preconditioners on irregular regions*, in Advances in Computer Methods for Partial Differential Equations,VI, R. Vichnevetsky and R. Stepleman, eds., IMACS, New Brunswick, NJ, 1987, pp. 317–322.

[8] ———, *A domain decomposed fast Poisson solver on a rectangle*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. s14–s26.

[9] ———, *A framework for the analysis and the construction of domain decomposition preconditioners*, in Glowinsky et al. [10], pp. 217–230.

[10] R. GLOWINSKY, G. H. GOLUB, G. A. MEURANT, AND J. PERIAUX, EDS., *Proc. First International Symposium on Domain Decomposition Methods, Jan.* 1987, *Paris, France*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1988.

[11] G. H. GOLUB AND D. MAYERS, *The use of pre-conditioning over irregular regions*, in Proc. Sixth Internat. Conference on Computational Methods in Science and Engineering, Versailles, France, December 12–16, 1983.

[12] D. E. KEYES AND W. GROPP, *A comparison of domain decomposition techniques for elliptic partial differential equations*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. s166–s202.

[13] P. L. LIONS, *On the Schwarz alternating method.* I, in Glowinsky et al. [10], pp. 1–42.

[14] ———, *On the Schwarz alternating method* II*: Stochastic interpretation and order properties*, in Chan et al. [5], pp. 47–70.

[15] H. A. SCHWARZ, *Über eine Grenzübergang durch alternirendes Verfahren*, in Gesammelte Mathematische Abhandlungen 2, Springer-Verlag, Berlin, 1890, pp. 133–143.

[16] W.-P. TANG, *Schwarz splittings and template operators*, Ph.D. thesis, Department of Computer Science, Stanford University, Stanford, CA, July 1987.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 2, pp. 671–679, April 1992

015

TWO-STAGE AND MULTISPLITTING METHODS FOR THE PARALLEL SOLUTION OF LINEAR SYSTEMS*

DANIEL B. SZYLD† AND MARK T. JONES‡

Dedicated to Gene Golub on the occasion of his 60th birthday

Abstract. Two-stage and multisplitting methods for the parallel solution of linear systems are studied. A two-stage multisplitting method is presented that reduces to each of the others in particular cases. Conditions for its convergence are given. In the particular case of a multisplitting method related to block Jacobi, it is shown that it is equivalent to a two-stage method with only one inner iteration per outer iteration. A fixed number of iterations of this method, say, p, is compared with a two-stage method with p inner iterations. The asymptotic rate of convergence of the first method is faster, but, depending on the structure of the matrix and the parallel architecture, it takes more time to converge. This is illustrated with numerical experiments.

Key words. solution of linear systems, block iterative methods, parallel methods

AMS(MOS) subject classification. 65F15

1. Introduction and preliminaries.

O'Leary and White [9] introduced the multisplitting method for the parallel solution of linear systems of equations of the form

$$Ax = b,$$

where A is an $n \times n$ matrix and x and b are vectors. Lanzkron, Rose, and Szyld [6] analyzed the convergence of two-stage methods for the solution of the same system. In this paper, we study the relationship between these two methods, also analyzing rates of convergence and relative costs in specific examples. Numerical experiments that illustrate our findings are presented. Further, for illustrative purposes, an algorithm combining the two methods is given.

A matrix T is nonnegative, denoted $T \geq 0$, if it has nonnegative elements. The representation $A = M - N$ is called a splitting of A if M is nonsingular; it is called a regular splitting if $M^{-1} \geq 0$ and $N \geq 0$ [11]; and it is called a weak regular splitting if $M^{-1} \geq 0$ and $M^{-1}N \geq 0$ [10]. A splitting $A = M - N$ is convergent if $\rho(M^{-1}N) < 1$, where $\rho(T)$ denotes the spectral radius of T. Thus, if the splitting $A = M - N$ is convergent, then iterative methods of the form

$$x_{i+1} = M^{-1}Nx_i + M^{-1}b \tag{1}$$

are convergent for any initial vector x_0. It is well known that if $A^{-1} \geq 0$, every weak regular splitting of A is convergent [1]. We denote by I the identity matrix.

DEFINITION 1.1 [9]. Let A, M_k, N_k, and D_k $(k = 1, \cdots, K)$ be $n \times n$ matrices. Let $D_k \geq 0$ be diagonal. Then (M_k, N_k, D_k) $(k = 1, \cdots, K)$ is called a multisplitting of A if

* Received by the editors August 8, 1990; accepted for publication (in revised form) March 9, 1991.

† Department of Mathematics, Temple University, Philadelphia, Pennsylvania 19122-2585. The work of this author was supported by National Science Foundation grant DMS-8807338 (szyld@euclid.math.temple.edu).

‡ Mathematics and Computer Science Division, Argonne National Laboratory, 9700 South Cass Avenue, Argonne, Illinois 60439-4844 (mjones@mcs.anl.gov). This author received support from the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under contract W-31-109-Eng-38.

(i) $A = M_k - N_k$, $k = 1, \cdots, K$, is a splitting of A;
(ii) $\sum_k D_k = I$.

Given an initial vector x_0 and a multisplitting of A, the multisplitting iterative method is defined by

$$x_{i+1} = \sum_k D_k M_k^{-1} N_k x_i + \sum_k D_k M_k^{-1} b. \tag{2}$$

The solution of the different linear systems

$$M_k y = v \tag{3}$$

in (2) can be solved by different processors in a parallel environment. This fact makes these methods particularly attractive.

O'Leary and White [9] have shown that if for $k = 1, \cdots, K$, $A = M_k - N_k$ is a weak regular splitting of A satisfying $A^{-1} \geq 0$, then the multisplitting method converges for any initial vector x_0; see also Neumann and Plemmons [8]. Examples of multisplittings can be found in these references and in White [13].

If the diagonal (masking) matrices D_k have only zero and one entries, the method (2) corresponds to the (parallel) block Jacobi method [11]. The splittings $A = M_k - N_k$ are then the same as $A = M - N$, since only the nonzeros corresponding to nonzeros of D_k are relevant. We are mainly concerned about this case in this paper; see also §3. We note that there is then no overlap between the groups of variables that are modified in the fractional steps of the algorithm; see (3). In §4 we make some observations for some cases when there is overlap. The dichotomy between overlap and nonoverlap is similar to that of the Schwarz additive algorithm in the context of domain decomposition methods; see, e.g., Dryja [3] or Dryja and Widlund [4] and the references given therein.

Thus, in the nonoverlap case, the processors have to be synchronized between the formation of iterates x_i and x_{i+1}. A discussion of chaotic multisplitting methods (with overlap), where synchronization is not required, can be found in [2].

Two-stage methods, also called inner/outer methods, consist of using a splitting within a splitting; i.e., given a splitting $A = M - N$, let $M = B - C$ and perform, say, p "inner" iterations [6]. Thus the resulting method is

$$x_{i+1} = (B^{-1}C)^p x_i + \sum_{j=0}^{p-1} (B^{-1}C)^j B^{-1}(Nx_i + b). \tag{4}$$

2. Two-stage multisplitting. A simple unification of the two methods described in §1 is considered where each splitting $A = M_k - N_k$ of the multisplitting method can itself be a two-stage method. The discussion in this section is general in the sense that no assumption is made on the form of the matrices D_k.

DEFINITION 2.1. Let A, M_k, B_k, C_k, N_k, and D_k be $n \times n$ matrices. Let $D_k \geq 0$ be diagonal. Then $(M_k, B_k, C_k, N_k, D_k)$ $(k = 1, \cdots, K)$ is called a two-stage multisplitting if

(i) $A = M_k - N_k$, $k = 1, \cdots, K$, is a splitting of A;
(ii) $\sum_k D_k = I$;
(iii) $M_k = B_k - C_k$, $k = 1, \cdots, K$, is a splitting of M_k.

The two-stage multisplitting algorithm is as follows:

Given the initial vector x_0.
For $i = 0, 1, 2, \cdots$, until convergence.
  For $k = 1$ to K
    $y_{k,0} = x_i$
    For $j = 0$ to $p - 1$
      $B_k y_{k,j+1} = C_k y_{k,j} + N_k x_i + b$
  $x_{i+1} = \sum_k D_k y_{k,p}$

Clearly, this algorithm reduces to the multisplitting method (2) when $p = 1$ and to the two-stage method (4) when $K = 1$. In the analysis of the convergence of the two-stage multisplitting algorithm, we use the following

LEMMA 2.2 [6]. *Given a nonsingular matrix A and T such that $(I-T)^{-1}$ exists, there exists a unique pair of matrices M, N, such that $T = M^{-1}N$ and $A = M - N$, where M is nonsingular. These matrices are $M_T = A(I - T)^{-1}$ and $N_T = M_T - A$. The splitting $A = M_T - N_T$ is the (unique) splitting induced by T.*

Lanzkron, Rose, and Szyld [6] have shown that if the "outer" splitting $A = M - N$ is regular and the "inner" splitting $M = B - C$ is weak regular, then the two-stage method (4) is convergent for any initial vector x_0 and for any value of p. Moreover, under these conditions, the splitting induced by the iteration matrix

$$T_p = I - (I - (B^{-1}C)^p)(I - M^{-1}N) \tag{5}$$

is a weak regular splitting. They also present an example where the outer splitting is weak regular and there is no convergence for certain values of p.

THEOREM 2.3. *Let $A^{-1} \geq 0$. Let $A = M_k - N_k$ be a regular splitting, and let $M_k = B_k - C_k$ be a weak regular splitting for $k = 1, \cdots, K$. Then, the two-stage multisplitting method is convergent for any initial vector x_0 and any value of p.*

Proof. For a fixed k, and for any value of p, the splittings $A = M_k - N_k$ and $M_k = B_k - C_k$ define a (convergent) two-stage method and induce weak regular splittings $A = M_{k,p} - N_{k,p}$. The two-stage multisplitting method reduces then to the multisplitting method (2), where this collection of induced weak regular splittings is used in Definition 1.1(i), and is thus convergent for any initial vector x_0. □

We remark that the preceding discussion can be easily generalized to allow the inclusion of the nested iterative methods described in [6].

3. A comparison of some block methods. Consider a partition of the n variables as $\{1, \cdots, n\} = S = \cup_{k=1}^{K} S_k$, where S_k has n_k elements. Let the diagonal matrices $D_k \geq 0$ $(k = 1, \cdots, K)$ be such that $D_{k,i} = 1$ if $i \in S_k$, and zero otherwise. This implies in particular that $\sum_k D_k = I$. Let $A_k = D_k A D_k$, and let $\bar{A}_k$ be the $n_k \times n_k$ matrix with nonzero entries coinciding with those of A_k. If A is an M-matrix [1], [11], the splitting $A = M - N$, where $M = \sum_k A_k$, is a regular splitting. This is a standard device; and with this splitting, the method (1) is the block Jacobi method, with its inherent parallelism. Let $\bar{A}_k = \bar{B}_k - \bar{C}_k$ $(k = 1, \cdots, K)$ be weak regular splittings. Using the structure of the matrices D_k, we can "assemble" the matrices $\bar{B}_k$ and $\bar{C}_k$ in the obvious way and obtain a weak regular splitting of $M = B - C$, where $B_k = D_k B D_k$ has the same nonzeros as $\bar{B}_k$.

The two-stage method (4) with these matrices corresponds to a block method in which the same number, say, p, of "inner" iterations is performed in each block (the case of different values of p for different blocks has been studied in [6]). If $p = 1$, this two-stage method is equivalent to the multisplitting method (2), where $M_k = B_k$ and $N_k = C_k + N$ $(k = 1, \cdots, K)$. The special case of $\bar{A}_k = \bar{B}_k - \bar{C}_k$ being the splitting

corresponding to the SOR method was treated (with and without overlap) by White [12, §4].

In a parallel implementation of these methods, it is assumed that a different processor solves for the variables in different sets S_k. In the absence of communication delays between processors, p (outer) iterations of the multisplitting method take more work than one iteration of the two-stage method with p "inner" iterations. The difference is $p-1$ matrix times vector products with the matrix N; see (4). It is therefore natural to compare the convergence properties of these two methods. In the next section we study their asymptotic rate of convergence, while in §3.2 we report on some numerical experiments. We show that if T_p is the iteration matrix of the two-stage method with p "inner" iterations, then $\rho(T_p) \geq \rho(T_1)^p$, i.e., this multisplitting method (one inner iteration) is asymptotically faster. Nevertheless, we show that there are problems for which, because of the communications delay and the sparsity structure of N, $p > 1$ will give faster convergence.

3.1. Asymptotic convergence. In the comparison of the asymptotic convergence of the block methods just described, we use the following result, proofs of which can be found in [6] or [7]; conditions for strict inequalities of the spectral radius are also presented in the latter reference.

THEOREM 3.1. *Let $A = M - N = \tilde{M} - \tilde{N}$ be convergent weak regular splittings such that $\tilde{M}^{-1} \geq M^{-1}$, and let x and z be the nonnegative Frobenius eigenvectors of $T = M^{-1}N$ and $\tilde{T} = \tilde{M}^{-1}\tilde{N}$, respectively. If*

$$\tilde{N}z \geq 0 \tag{6}$$

or if $Nx \geq 0$ with $x > 0$, then $\rho(T) \geq \rho(\tilde{T})$.

We present now our main theoretical result. It applies to any two-stage method, and in particular to the case where $M = \sum_k A_k$, $A_k = D_k A D_k$, and $D_k \geq 0$ ($k = 1, \cdots, K$) are diagonal matrices with zeros and ones such that $\sum_k D_k = I$, i.e., to the multisplitting in question.

THEOREM 3.2. *Let $A^{-1} \geq 0$. Let $A = M - N$ be a regular splitting and let $M = B - C$ be a weak regular splitting. Let T_p be the iteration matrix of the two-stage method with p "inner" iterations. Then $\rho(T_p) \geq \rho(T_1)^p$.*

Proof. By Lemma 2.2 the matrix T_p in (5) induces the splitting $A = M_{T_p} - N_{T_p}$, where

$$M_{T_p} = M(I - (B^{-1}C)^p)^{-1} = B(I - B^{-1}C)(I - (B^{-1}C)^p)^{-1}. \tag{7}$$

The iteration matrix of the multisplitting method in question is

$$H = T_1^p = [B^{-1}(C+N)]^p.$$

It induces the splitting $A = M_H - N_H$, where

$$M_H = A(I-H)^{-1} = B(I - B^{-1}(C+N))(I - (B^{-1}(C+N))^p)^{-1}. \tag{8}$$

We will use Theorem 3.1. As shown in [6, Thm. 4], condition (6) is directly satisfied for N_{T_p}. It remains only to show that $M_H^{-1} \geq M_{T_p}^{-1}$. We have that

$$M_{T_p}^{-1} = (I - (B^{-1}C)^p)(I - B^{-1}C)^{-1}B^{-1} = \sum_{i=0}^{p-1}(B^{-1}C)^i B^{-1},$$

$$M_H^{-1} = (I - (B^{-1}(C+N))^p)(I - B^{-1}(C+N))^{-1}B^{-1}$$
$$= \sum_{i=0}^{p-1}(B^{-1}(C+N))^i B^{-1},$$

where we have used the identity

$$(I - R^p)(I - R)^{-1} = \sum_{i=0}^{p-1} R^i.$$

Since $B^{-1} \geq 0$ and $N \geq 0$, $B^{-1}N \geq 0$. Therefore, $B^{-1}(C+N) \geq B^{-1}C$ and the theorem follows. □

3.2. Practical considerations. The two-stage methods with $p > 1$ do have a significant advantage on parallel architectures with nonuniform memory access times: the inner iterations can be accomplished with references exclusively to memory that is "local" to that processor. Examples of architectures that fall into this class are the BBN GP1000, the BBN TC2000, Intel hypercube computers, and the CRAY-2. Parallel computers with cache memory that is local to a processor can also take advantage of these two-stage splitting methods if the data necessary to accomplish the inner iterations can be stored in the cache. Examples of architectures that fall into this class are the Encore Multimax and Sequent Symmetry computers. The two-stage methods can gain an advantage from this data locality only if the number of outer iterations necessary to converge to a solution decreases as p increases. As we have shown in Theorem 3.2, asymptotically, the two-stage method is slower. Therefore, there must be some "optimal" number of inner iterations p, which would depend on the architecture and the matrix in question.

To briefly demonstrate the practical advantages of using $p > 1$ inner iterations on parallel processors with nonuniform memory access times, we ran two experiments on the BBN TC2000 multiprocessor. To perform the experiments, we modified a parallel block iterative code to use the block Jacobi method as the outer iteration, inducing the splitting $A = M - N$, and the point Gauss–Seidel method as the inner iteration, inducing the splitting $M = B - C$. In each experiment the number of processors varied from 1 to 32 and the number of inner iterations was increased until no further decrease in execution time was observed. The iterations were stopped when the ratio of the norms of the residual and the right-hand side was less than 10^{-6}.

In the first experiment, a matrix generated with a nine-point cross stencil on a 64×64 grid was used.[1] All the unknowns in a grid row were grouped into a block, resulting in 64 blocks of size 64×64. The right-hand side was constructed by setting the right boundary of the grid to 1s and the other boundaries to 0s. An examination of the results in Fig. 1 shows that a clear advantage is gained when p is increased. We have included in the same graph the number of outer iterations needed for convergence. Of course the number of outer iterations is independent of the number of processors.

In the second experiment, a matrix of order 1280 with a semi-bandwidth of 20 was used. The off-diagonals inside the band were -1, and the elements in the ith diagonal position were $(\sum_{j \neq i} \mid a_{ij} \mid) + 2$. The matrix was partitioned into 64 20×20

[1] Note: In order to significantly reduce the number of outer iterations as p increases, as many of the larger elements in A as possible must be included in B. To simulate this, we used the following stencil on the grid: $A(i,i) = 9.02$, $A(i,i+1) = A(i,i+2) = A(i+1,i) = A(i+2,i) = -2.24$, $A(i,i+gridsize) = A(i,i+gridsize*2) = A(i+gridsize,i) = A(i+gridsize*2,i) = -0.01$.

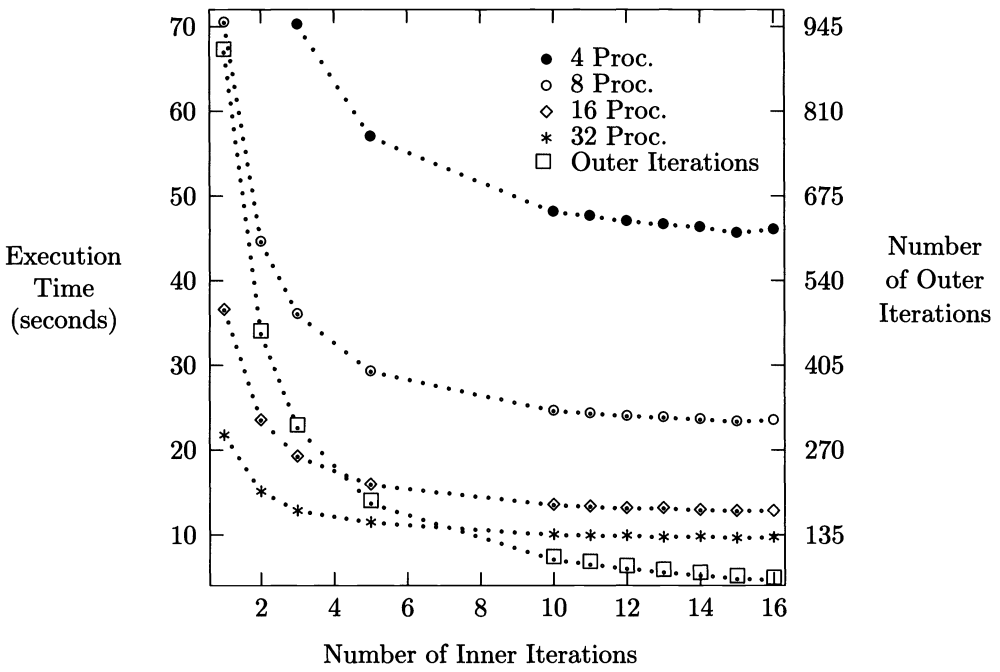


FIG. 1. *Results from experiment on grid problem.*

blocks. The right-hand side consisted of 1s in every 64th position and 0s in every other position. In this experiment, the reduction in the number of outer iterations as p increases was not large; therefore, the possible advantage to be gained from increasing p, if any, was small. The results are given in Fig. 2.

4. The effect of overlap. The asymptotic results of Theorem 3.2 apply only to a specific class of multisplittings. The same theory does not extend easily to other cases. The question arises then if the conclusions of §3.2 can also be observed in more general multisplittings. In particular, is it true that when there is overlap, $p > 1$ inner iterations is still more advantageous? Also, if this is the case, how does the order of the variables affect the efficiency of the method? This last question was studied in a recent paper by White [13].

The experiments in this section were devised to explore these questions. The matrix used corresponds to a nine-point discretization on a 256×256 grid ($n = 65536$). The coefficients associated with a row on the grid are: $-1/6$, $-1/3$, 1, $-1/3$, $-1/6$. We have considered blocks of size 2048 corresponding to 8 rows of 256 nodes, and an overlap of 256 nodes, i.e., one row of overlap.

TABLE 1
Effect of the order of the variables.

Inner Iterations	Ordering 0	Ordering 1	Ordering 2
1	320 (531)	314 (509)	339 (551)
2	211 (276)	211 (267)	223 (283)
4	165 (152)	169 (149)	174 (153)
8	162 (94)	172 (94)	172 (94)
16	207 (69)	219 (68)	222 (69)

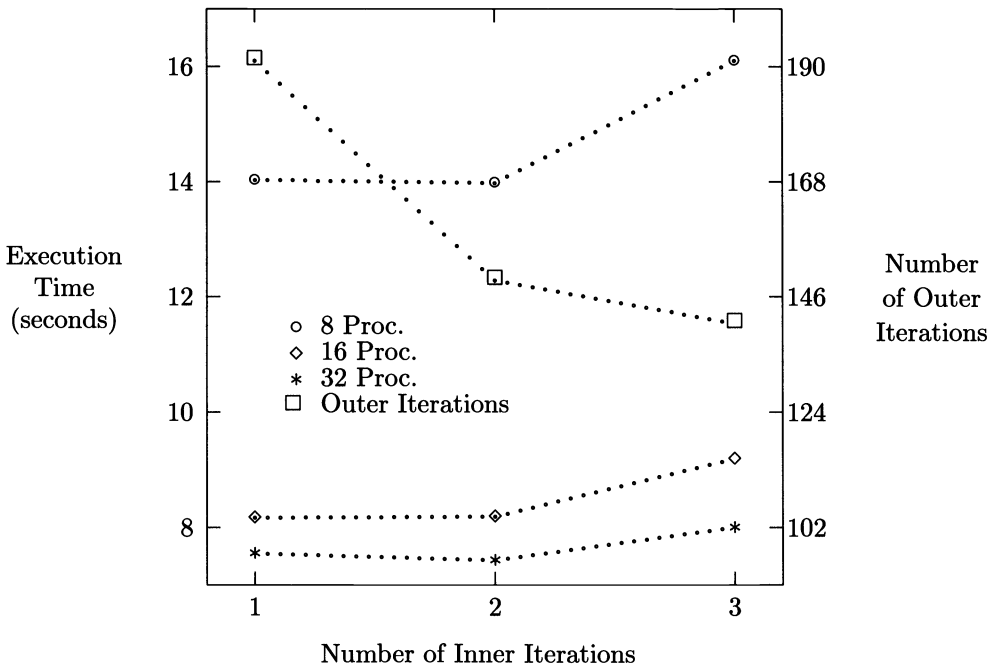


FIG. 2. *Results from experiment on band problem.*

The experiment reported in Table 1 illustrates the effect of different orderings when overlap is used. We have used 16 processors of the BBN TC2000. In each column, the time in seconds is given first and then in parentheses is the number of (outer) iterations. Ordering 0 is the natural ordering. Ordering 1 is numbering the overlapping rows first. Ordering 2 is numbering the overlapping rows last. It can be readily observed that (a) $p > 1$ can indeed give better results, as in the case described in §3, (b) fewer outer iterations are needed when numbering the overlap first, as in the case studied by White [13], and (c) the effect of the ordering diminishes dramatically as the number of inner iterations is increased. We should point out that the timings for orderings 1 and 2 can be improved by using more sophisticated code for the reordering portion of it.

In Fig. 3 we report on an experiment with the same matrix, using ordering 1, with different number of processors, with and without overlap. It is interesting to observe that for p less than the "optimal" value, the nonoverlap case is faster while the situation is reverse for p larger. At the same time, there is little difference between the two cases near the "optimal" value. For completeness we report that the number of outer iterations in the nonoverlap case are: 549, 303, 186, 135, and 114, for 1, 2, 4, 8, and 16 inner iterations, respectively. Those for the overlap case are given in Table 1.

5. Concluding remarks. We have compared, in the case of (outer) block Jacobi, a two-stage method and a corresponding multisplitting method for the parallel solution of linear systems. Based on the asymptotic rate of convergence, the first method appears to be slower. On the other hand, the savings in communication between processors makes the two-stage method more competitive for a certain range of p, the number of inner iterations. The same situation was observed in two-stage

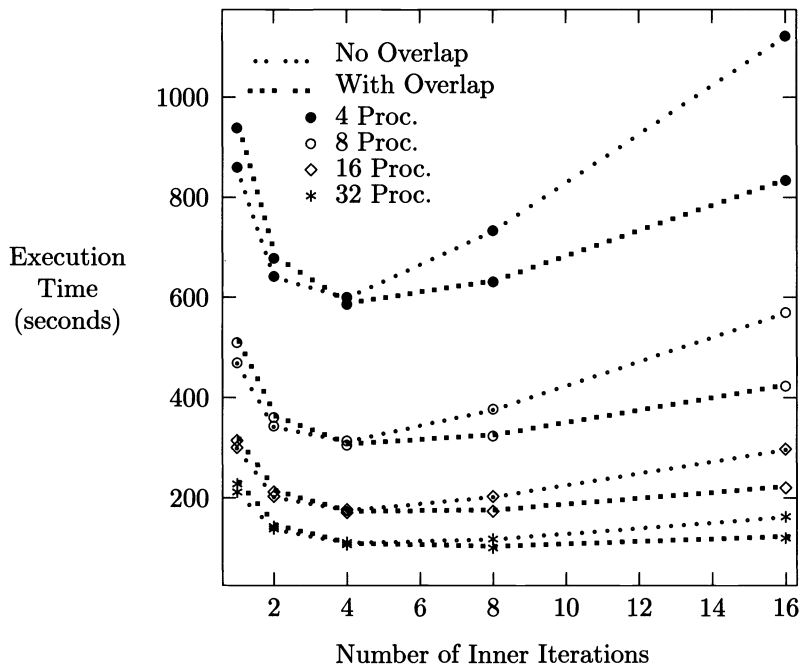


FIG. 3. *Time in seconds with and without overlap.*

multisplittings with overlap.

Acknowledgments. We extend thanks to Julio César Díaz and Michael Neumann for asking us about the relationship between two-stage and multisplitting methods, to Paul Lanzkron for an interesting discussion about his computational experience in [5], and to James Wilkes for making available his code for parallel block iterative methods. We also thank a referee for useful comments which led us to write §4.

REFERENCES

[1] A. BERMAN AND R. J. PLEMMONS, *Nonnegative Matrices in the Mathematical Sciences*, Academic Press, New York, 1979.

[2] R. BRU, L. ELSNER, AND M. NEUMANN, *Models of parallel chaotic iteration methods*, Linear Algebra Appl., 103 (1988), pp. 175–192.

[3] M. DRYJA, *An additive Schwarz algorithm for two- and three-dimensional finite element problems*, in Second International Symposium on Domain Decomposition Methods for Partial Differential Equations, T. Chan, R. Glowinski, G. Meurant, J. Pèriaux, and O. Widlund, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1989, pp. 168–172.

[4] M. DRYJA AND O. B. WIDLUND, *Some domain decomposition algorithms for elliptic problems*, in Iterative Methods for Large Linear Systems, Academic Press, San Diego, CA, 1989, pp. 273–291 (Proceeding of the Conference on Iterative Methods for Large Linear Systems held in Austin, TX, October 19–21, 1988, to celebrate the sixty-fifth birthday of David M. Young, Jr.).

[5] P. J. LANZKRON, *Nested iterative methods for linear systems*, Ph.D. thesis, Department of Computer Science, Duke University, Durham, NC, August 1989.

[6] P. J. LANZKRON, D. J. ROSE, AND D. B. SZYLD, *Convergence of nested classical iterative methods for linear systems*, Numer. Math., 58 (1991), pp. 685–702.

[7] I. MAREK AND D. B. SZYLD, *Comparison theorems for weak splittings of bounded operators*, Numer. Math., 58 (1990), pp. 387–397.

[8] M. NEUMANN AND R. J. PLEMMONS, *Convergence of parallel multisplitting methods for M-matrices*, Linear Algebra Appl., 88-89 (1987), pp. 559–574.
[9] D. P. O'LEARY AND R. E. WHITE, *Multi-splittings of matrices and parallel solution of linear systems*, SIAM J. Algebraic Discrete Meth., 6 (1985), pp. 630–640.
[10] J. M. ORTEGA AND W. C. RHEINBOLDT, *Iterative Solution of Nonlinear Equations in Several Variables*, Academic Press, New York, London, 1970.
[11] R. S. VARGA, *Matrix Iterative Analysis*, Prentice-Hall, Englewood Cliffs, NJ, 1962.
[12] R. E. WHITE, *Multisplittings and parallel iterative methods*, Comput. Meth. Appl. Mech. Engrg., 64 (1987), pp. 567–577.
[13] ———, *Multisplitting of a symmetric positive definite matrix*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 69–82.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 681–687, July 1992

STABILITY OF A METHOD FOR MULTIPLYING COMPLEX MATRICES WITH THREE REAL MATRIX MULTIPLICATIONS*

NICHOLAS J. HIGHAM†

Abstract. By use of a simple identity, the product of two complex matrices can be formed with three real matrix multiplications and five real matrix additions, instead of the four real matrix multiplications and two real matrix additions required by the conventional approach. This alternative method reduces the number of arithmetic operations, even for small dimensions, achieving a saving of up to 25 percent. The numerical stability of the method is investigated. The method is found to be less stable than conventional multiplication but stable enough to warrant practical use. Issues involved in the choice of method for complex matrix multiplication are discussed, including the relative efficiency of real and complex arithmetic and the backward stability of block algorithms.

Key words. matrix multiplication, complex matrix, Strassen's method, Winograd's identity, numerical stability, error analysis, level-3 BLAS

AMS(MOS) subject classifications. 65F05, 65G05

1. Introduction.

How many real multiplications are required to multiply two complex numbers? In view of the familiar identity

$$z = (a + ib)(c + id) = ac - bd + i(ad + bc),$$

the answer might appear to be four. However, it is possible to make do with three multiplications, because

$$z = ac - bd + i\big[(a + b)(c + d) - ac - bd\big]. \tag{1.1}$$

This formula was suggested by Peter Ungar in 1963, according to Knuth [14, p. 647]. That three multiplications (or divisions) are *necessary* for evaluating z was proved by Winograd [17].

Ungar's formula does not rely on commutativity, so it can be generalized to matrix multiplication, as noted by Fam [10]. Let $A = A_1 + iA_2$ and $B = B_1 + iB_2$, where $A_j, B_j \in \mathbb{R}^{n\times n}$, and define $C = C_1 + iC_2 = AB$. (We concentrate on square matrices, although everything we say extends easily to rectangular matrices.) Then C can be formed using three real matrix multiplications as

$$\begin{aligned} &T_1 = A_1B_1, \qquad T_2 = A_2B_2, \\ &C_1 = T_1 - T_2, \\ &C_2 = (A_1 + A_2)(B_1 + B_2) - T_1 - T_2, \end{aligned} \tag{1.2}$$

which we will refer to as the "3M method." This computation involves $3n^3$ scalar multiplications and $3n^3 + 2n^2$ scalar additions. Straightforward evaluation of the conventional formula $C = A_1B_1 - A_2B_2 + i(A_1B_2 + A_2B_1)$ requires $4n^3$ multiplications and $4n^3 - 2n^2$ additions. Thus, the 3M method requires strictly less arithmetic operations than the conventional means of multiplying complex matrices for $n \geq 3$, and it achieves a saving of about 25 percent for $n \geq 30$ (say). Similar savings occur in the important special case where A or B is triangular.

* Received by the editors February 12, 1990; accepted for publication November 2, 1990.
† Department of Mathematics, University of Manchester, Manchester, M13 9PL, England (na.nhigham@na-net.ornl.gov).

It is rare in matrix computations to be able to produce such a clear-cut computational saving over a standard technique, and the 3M method therefore deserves careful consideration for practical use.

Since an increase in speed is often accompanied by a loss of numerical stability, it is important to investigate the behaviour of the 3M method in the presence of rounding errors. Doubts about the stability are raised by the comment of Knuth [14, p. 647] on a variation of (1.1) (see (2.7)): "Beware numerical instability." We investigate the stability in §2 and show that the 3M method is stable in a certain sense, although it does not match the stability properties of conventional multiplication.

In §3 we offer some guidance on the choice of method for multiplying complex matrices.

This work was motivated by the knowledge that the 3M method is being used in Fortran routines `CGEMMS` and `ZGEMMS` in IBM's ESSL library. The 3M method is also used by routines of the same name in Cray's UNICOS library [5]. All these routines form complex matrix products by using Strassen's fast matrix multiplication method [15] to evaluate the real matrix products in (1.2). Although the ESSL documentation warns about potential instability of Strassen's method [13, p. 344], it contains no comment on the stability of the 3M method itself.

2. Numerical stability. A simple example reveals a fundamental weakness of the 3M method. Consider the computation of the scalar

$$z = x + iy = (\theta + i/\theta)^2 = \theta^2 - 1/\theta^2 + 2i.$$

Suppose we use floating point arithmetic with unit roundoff u. If y is computed the usual way, as $y = \theta(1/\theta) + (1/\theta)\theta$, then no cancellation occurs and the computed $\widehat{y}$ has high relative accuracy: $|\widehat{y} - y|/|y| = O(u)$. The 3M method computes

$$y = \left(\theta + \frac{1}{\theta}\right)\left(\theta + \frac{1}{\theta}\right) - \theta^2 - \frac{1}{\theta^2}.$$

If $|\theta|$ is large this formula expresses a number of order 1 as the difference of large numbers. The computed $\widehat{y}$ will almost certainly be contaminated by rounding errors of order $u\theta^2$, in which case the relative error is large: $|\widehat{y}-y|/|y| = O(u\theta^2)$. However, if we measure the error in $\widehat{y}$ relative to z, then it is acceptably small: $|\widehat{y}-y|/|z| = O(u)$.

This example suggests that the 3M method may be stable in a weaker sense than conventional multiplication. In the rest of this section we establish the stability properties in a precise form, for general n.

For the error analysis we assume that the floating point arithmetic obeys the model

$$\begin{aligned} fl(x \operatorname{op} y) &= (x \operatorname{op} y)(1+\delta), \quad |\delta| \le u, \quad \operatorname{op} = *, /, \\ fl(x \pm y) &= x(1+\alpha) \pm y(1+\beta), \qquad |\alpha|, |\beta| \le u, \end{aligned}$$

where the latter equation allows for possible lack of a guard digit in addition and subtraction. Standard analysis (analogous to that in [11, p. 66], for example) shows that if $A, B \in \mathbb{R}^{n\times n}$ and we compute $\widehat{C} = fl(AB)$ by conventional multiplication, then

$$|\widehat{C} - AB| \le nu|A||B| + O(u^2). \tag{2.1}$$

Here, $|\cdot|$ denotes the operation of replacing each matrix element by its absolute value, and the matrix inequality is interpreted componentwise.

Now we consider the product $C_1 + iC_2 = (A_1 + iA_2)(B_1 + iB_2)$ for $n \times n$ complex matrices, as in §1. Using (2.1) we find that the computed product from conventional multiplication,

$$\widehat{C} = fl\big(A_1B_1 - A_2B_2 + i(A_1B_2 + A_2B_1)\big),$$

satisfies

$$|\widehat{C}_1 - C_1| \le (n+1)u\big(|A_1||B_1| + |A_2||B_2|\big) + O(u^2), \tag{2.2}$$

$$|\widehat{C}_2 - C_2| \le (n+1)u\big(|A_1||B_2| + |A_2||B_1|\big) + O(u^2). \tag{2.3}$$

It is easy to verify that, apart from the factors $n+1$, these bounds reflect the sensitivity of the product AB to perturbations in A and B of the form $A_j \to A_j + \Delta A_j$, where $|\Delta A_j| \le u|A_j|$.

For the 3M method C_1 is computed in the conventional way, and so (2.2) holds. It is straightforward but tedious to show that $\widehat{C}_2$ satisfies

$$\begin{aligned}|\widehat{C}_2 - C_2| \le (n+4)u\big[&(|A_1| + |A_2|)(|B_1| + |B_2|) \\ &+ |A_1||B_1| + |A_2||B_2|\big] + O(u^2).\end{aligned} \tag{2.4}$$

Two notable features of the bound (2.4) are as follows. First, it is of a different and weaker form than (2.3); in fact, it exceeds the sum of the bounds (2.2) and (2.3). Second and more pleasing, it retains the property of (2.2) and (2.3) of being invariant under diagonal scalings

$$C = AB \to D_1AD_2 \cdot D_2^{-1}BD_3 = D_1CD_3, \quad D_j \text{ diagonal},$$

in the sense that the upper bound ΔC_2 in (2.4) scales also according to $D_1\Delta C_2 D_3$. (The "hidden" second-order terms in (2.2)–(2.4) are invariant under these diagonal scalings.)

The disparity between (2.3) and (2.4) is, in part, a consequence of the differing numerical cancellation properties of the two methods. It is easy to show that there are always subtractions of like-signed numbers in the 3M method, whereas if A_1, A_2, B_1, and B_2 have nonnegative elements (for example), then no numerical cancellation takes place in conventional multiplication.

We can define a measure of stability with respect to which the 3M method matches conventional multiplication by taking norms in (2.3) and (2.4). We obtain the weaker bounds

$$\|\widehat{C}_2 - C_2\|_\infty \le 2(n+1)u\|A\|_\infty\|B\|_\infty + O(u^2), \tag{2.5}$$

$$\|\widehat{C}_2 - C_2\|_\infty \le 4(n+4)u\|A\|_\infty\|B\|_\infty + O(u^2). \tag{2.6}$$

Combining these with an analogous weakening of (2.2), we find that for both conventional multiplication and the 3M method, the computed complex matrix $\widehat{C}$ satisfies

$$\|\widehat{C} - C\|_\infty \le c_n u\|A\|_\infty\|B\|_\infty + O(u^2),$$

where $c_n = O(n)$.

Our findings can be summarised as follows. The 3M method produces a computed product $\widehat{C}$ whose imaginary part may be contaminated by relative errors much larger than those for conventional multiplication (or equivalently, much larger than can be accounted for by small componentwise perturbations in the data A and B). However, if the errors are measured relative to $\|A\|_\infty\|B\|_\infty$, which is a natural quantity to use for comparison when employing matrix norms, then they are just as small as for conventional multiplication.

We conclude this section with several further comments.

(1) It does not seem possible to improve the stability of the 3M method by "tinkering" with the basic formula. The symmetric formula

$$z = a(c+d) - (a+b)d + i\big[a(c+d) + (b-a)c\big], \tag{2.7}$$

mentioned by Knuth [14, p. 647] as an alternative to (1.1), is "worse" in the sense that either of the real and imaginary parts can be relatively inaccurate. Of course, by adapting formula (1.1) we can arrange that only the real part be of questionable accuracy.

(2) The 3M formula resembles Winograd's identity for computing the inner product of vectors $x, y \in \mathbb{R}^n$. The identity is [16], for even n,

$$x^T y = \sum_{i=1}^{n/2}(x_{2i-1} + y_{2i})(x_{2i} + y_{2i-1}) - \sum_{i=1}^{n/2} x_{2i-1}x_{2i} - \sum_{i=1}^{n/2} y_{2i-1}y_{2i}. \tag{2.8}$$

Setting $n = 2$, we have

$$x^T y = (x_1 + y_2)(x_2 + y_1) - x_1x_2 - y_1y_2. \tag{2.9}$$

For comparison, the 3M formula (1.1) uses the identity

$$x^T y = (x_1 + x_2)(y_1 + y_2) - x_1y_2 - x_2y_1 \tag{2.10}$$

to compute the imaginary part of z. Although they look similar, formulas (2.9) and (2.10) have quite different stability properties, because (2.9) exploits commutativity ($y_2x_2 = x_2y_2$ and $y_2y_1 = y_1y_2$), while (2.10) does not. Thus only (2.10) permits the generalisation where the x_j and y_j are matrices. On the other hand, Winograd's identity can be used to trade half the multiplications for additions in a matrix product AB (this being the main application of Winograd's identity), but the analogue of (2.10) for n-vectors cannot be employed in this fashion.

A further difference is that (2.9), and more generally (2.8), is numerically unstable in the sense that the best available normwise error bound is of the form

$$|fl(x^T y) - x^T y| \le c_n u\big(\|x\|_\infty + \|y\|_\infty\big)^2 + O(u^2), \tag{2.11}$$

which can be arbitrarily weaker than the bound

$$|fl(x^T y) - x^T y| \le c_n u\|x\|_\infty\|y\|_\infty + O(u^2),$$

which holds for conventional multiplication (and for (2.10)).

The instability of Winograd's identity was first pointed out by Brent [4], who proves a bound of the form (2.11). He shows that the instability can be overcome by scaling x and y so that $\|x\|_\infty \approx \|y\|_\infty$ before applying the identity.

(3) To put the stability of the 3M method into perspective it is worth noting that it is at least as stable as Strassen's method. The best available bound for Strassen's method for forming $C = AB$, where $A, B \in \mathbb{R}^{n\times n}$, is [12]

$$\|\widehat{C} - C\|_\infty \leq f(n, n_0)u\|A\|_\infty\|B\|_\infty + O(u^2),$$

where $f(n, n_0) \approx nn_0^2(n/n_0)^{3.6}$ and where $n_0 \leq n$ is the threshhold such that conventional multiplication is used for matrices of dimension n_0 or less. Thus Strassen's method satisfies a normwise bound only, and has a potentially much larger constant in the bound than the 3M method.

(4) It is straightforward to show that if the 3M method is implemented using Strassen's method to form the real matrix products, then the computed complex $\widehat{C}$ satisfies

$$\|\widehat{C} - C\|_\infty \leq 6(f(n, n_0) + 4)u\|A\|_\infty\|B\|_\infty + O(u^2). \tag{2.12}$$

In other words, the 3M method combined with Strassen's method has the same stability properties as Strassen's method alone.

(5) We have done numerical experiments in MATLAB to confirm the theoretical analysis. Our experience is that for "random" matrices the 3M method is quite likely to produce a computed answer of similar quality to that from conventional multiplication. (The same is true for Strassen's method; see [12].) However, it is easy to generate examples where instability occurs—for example, by generalizing the example at the beginning of this section.

3. Practical considerations. What method should we use to multiply complex matrices? If the best possible accuracy is required, or if execution time is not a primary concern, then the multiplication should be done in the conventional manner. When implementing conventional matrix multiplication in Fortran, we have the choice of splitting the computation into its real and imaginary parts at the beginning, as is necessary to apply the 3M method, or of using "complex arithmetic," which effectively means resorting to real arithmetic only at the scalar level. These two approaches carry out the same (real) arithmetic operations in different orders, and so satisfy the same error bounds (2.2) and (2.3). The choice of which approach to use can therefore be guided by considerations other than accuracy, such as the relative efficiency of real and complex arithmetic implementations, which depends on various factors, including memory reference time, the overhead of invoking complex arithmetic routines, and the intrinsic costs of real and complex arithmetic. The relative efficiency can vary greatly between machines and compilers. The LINPACK manual [8, p. 1.25, Appendix B] reports the execution times of `CGEFA` (complex LU factorization) and `SGEFA` (real LU factorization) for the LINPACK test sites (21 computing environments). For $n = 100$, the ratio "`CGEFA/SGEFA`" varies between 1.64 and 8.98, with an average of 4.31.

If a faster multiplication is desired, the most promising possibilities involve the 3M method and Strassen's method. Recent experience with Strassen's method on real matrices has shown that on certain machines it can produce useful speedups for n in the hundreds [1], [2]. If the computing environment is such that complex arithmetic is implemented very efficiently, it may be best to use Strassen's method alone in complex arithmetic. For example, in experiments in Algol-W on an IBM 360/67, Brent [3] found that a complex matrix multiplication took less than three times as long as a real matrix multiplication, for both the conventional method and Strassen's method. Thus it is probably not worth using the 3M method in this environment.

The evidence quoted above from the LINPACK manual suggests that in many Fortran environments complex arithmetic will exceed real arithmetic in cost by a factor of more than three. In such situations it is appropriate to use the 3M method in conjunction with Strassen's method, as is done in the ESSL library [13] and the UNICOS library [5], and as is discussed at the end of §2.

A prominent source of Fortran 77 matrix multiplication routines is the level-3 basic linear algebra subprograms (BLAS3) [9]. The BLAS3 specifications define what each routine must do but not how it must do it. Thus there is freedom of implementation, subject to the requirement of retaining numerical stability. One of the main uses of the BLAS3 is as modules in block algorithms for solving linear equation and eigenvalue problems, for example, in LAPACK [6]. Two important questions in this context are whether the block algorithms remain backward stable when they are built upon BLAS3 operations satisfying bounds of the form (2.12), and, if they do, whether the backward error results are sufficiently strong for a given application. In joint work with Demmel [7], we have shown that a wide class of block algorithms satisfy a backward error bound of the form (2.12) if the BLAS3 themselves satisfy (2.12). In combination with the work here, this provides motivation for preparing complex BLAS3 routines based on the 3M method combined with Strassen's method.

Acknowledgement. I thank Des Higham for his helpful comments on the manuscript and Phuong Vu for pointing out reference [10].

REFERENCES

[1] D. H. Bailey, *Extra high speed matrix multiplication on the Cray-2*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 603–607.

[2] D. H. Bailey, K. Lee, and H. D. Simon, *Using Strassen's algorithm to accelerate the solution of linear systems*, J. Supercomputing, 4 (1991), pp. 357–371.

[3] R. P. Brent, *Algorithms for matrix multiplication*, Tech. Report CS 157, Computer Science Department, Stanford University, Stanford, CA, 1970.

[4] ———, *Error analysis of algorithms for matrix multiplication and triangular decomposition using Winograd's identity*, Numer. Math., 16 (1970), pp. 145–156.

[5] Cray Research Inc., UNICOS *Math and Scientific Library Reference Manual*, Number SR-2081, Version 5.0, March 1989, Eagan, MN.

[6] J. W. Demmel, J. J. Dongarra, J. J. Du Croz, A. Greenbaum, S. J. Hammarling, and D. C. Sorensen, *Prospectus for the development of a linear algebra library for high-performance computers*, Tech. Memorandum No. 97, Mathematics and Computer Science Division, Argonne National Laboratory, Argonne, IL, 1987.

[7] J. W. Demmel and N. J. Higham, *Stability of block algorithms with fast level* 3 BLAS, LAPACK Working Note #22 and Numerical Analysis Report No. 188, University of Manchester, Manchester, England, 1990; to appear in ACM Trans. Math. Software.

[8] J. J. Dongarra, J. R. Bunch, C. B. Moler, and G. W. Stewart, LINPACK *Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[9] J. J. Dongarra, J. J. Du Croz, I. S. Duff, and S. J. Hammarling, *A set of level* 3 *basic linear algebra subprograms*, ACM Trans. Math. Software, 16 (1990), pp. 1–17.

[10] A. T. Fam, *Efficient complex matrix multiplication*, IEEE Trans. Comput., C-37 (1988), pp. 877–879.

[11] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[12] N. J. Higham, *Exploiting fast matrix multiplication within the level* 3 BLAS, ACM Trans. Math. Software, 16 (1990), pp. 352–368.

[13] IBM, *Engineering and Scientific Subroutine Library, Guide and Reference, Release* 3, Fourth Edition (Program Number 5668-863), 1988.

[14] D. E. Knuth, *The Art of Computer Programming, Volume* 2, *Seminumerical Algorithms*, Second Edition, Addison–Wesley, Reading, MA, 1981.

[15] V. STRASSEN, *Gaussian elimination is not optimal*, Numer. Math., 13 (1969), pp. 354–356.
[16] S. WINOGRAD, *A new algorithm for inner product*, IEEE Trans. Comput., C-18 (1968), pp. 693–694.
[17] ———, *On multiplication of* 2×2 *matrices*, Linear Algebra Appl., 4 (1971), pp. 381–388.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 688–706, July 1992

002

ON SCALING NEWTON'S METHOD FOR POLAR DECOMPOSITION AND THE MATRIX SIGN FUNCTION*

CHARLES KENNEY† AND ALAN J. LAUB†

Abstract. A tight bound is given on the speed of convergence of Newton's method with optimal scaling for the polar decomposition of a nonsingular complex matrix. Necessary and sufficient conditions are then derived that tell when an approximation to the optimal scaling value will give better results than the unscaled Newton method. For the related matrix sign problem, it is shown that optimal scaling requires complete knowledge of the eigenvalues of the original matrix. Because this is impractical, a family of scaling methods that are optimal with respect to partial eigenvalue information is derived. This family includes optimal scaling as well as a "semioptimal" scaling method based on the dominant eigenvalues of the matrix and its inverse. Semioptimal scaling can be implemented using the power method and it gives nearly optimal performance on a set of test problems. These test problems also show that a variety of other commonly used scaling strategies, including spectral scaling, determinantal scaling, and 2-norm scaling, can result in unduly slow convergence.

Key words. polar decomposition, matrix sign function, Newton's method, optimal scaling

AMS(MOS) subject classifications. 65F35, 65F30, 15A18

1. Introduction.

The polar decomposition of a nonsingular complex matrix A of order m is a matrix pair (U, H) such that U is unitary, H is Hermitian and positive definite, and $A = UH$. If A has a singular value decomposition, $A = P\Sigma Q^H$, where P and Q are unitary and $\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_m)$ with $0 < \sigma_m \leq \cdots \leq \sigma_1$, then [7]

$$U = PQ^H, \qquad H = Q\Sigma Q^H. \tag{1}$$

However, it is more efficient to compute the polar decomposition using scaled Newton recursions of the form

$$A_{n+1} = \tfrac{1}{2}(\gamma_n A_n + (\gamma_n A_n^H)^{-1}), \quad A_0 = A, \quad \gamma_n > 0. \tag{2}$$

For γ_n suitably chosen [7], [10], $A_n \to U$, and H can be found from $H = U^H A$. In the next section we give convergence bounds supporting the empirical observation that Newton's method with optimal scaling,

$$\gamma_n = \sqrt{\|A_n^{-1}\|_2 / \|A_n\|_2},$$

converges very rapidly, even when A is ill conditioned with respect to inversion. Here $\|\cdot\|_2$ denotes the matrix 2-norm.

In practice, the optimal scaling factor γ_n must be approximated and this raises the question: how bad can the estimate be before it is better to use the unscaled ($\gamma = 1$) Newton method? A simple analysis (Theorem 2.6) shows that this question has an elegant answer: it is better to use γ_n^{est} if and only if $\gamma_n^2 \leq \gamma_n^{\text{est}} \leq 1$. Following this, we consider three scaling methods; the first uses power method estimates of $\|A_n\|_2$ and $\|A_n^{-1}\|_2$ based on the work in [5]. This method gives good results but its inherent uncertainty has led to other methods such as the one suggested by Higham

* Received by the editors June 5, 1989; accepted for publication (in revised form) December 10, 1990. This research was supported by National Science Foundation (and Air Force Office of Scientific Research) grant ECS87-18897 and the Air Force Office of Scientific Research contract AFOSR-91-0240.

† Department of Electrical and Computer Engineering, University of California, Santa Barbara, California 93106-9560 (laub@ece.ucsb.edu).

[7] based on approximating the 2-norm by the geometric average of the 1-norm and the ∞-norm. This method works well and is closely related to the third method based on a Frobenius norm approximation of the 2-norm. This "Frobenius" scaling always gives better results than the unscaled Newton method (Lemma 2.7). Numerical tests on all three of these methods suggest that they are nearly optimal for most problems.

Analysis of Newton's method for the polar decomposition uses the fact that the matrix recursion (2) can be reduced to scalar recursions for the singular values of A_n. The convergence of the singular values to 1 then determines the rate of convergence of A_n to the polar factor U. For the matrix sign function the same is true of the eigenvalues of A_n and their convergence to ± 1, but the conditioning of the associated eigenspaces can affect convergence at least for the first few iterations.

If A has no eigenvalues on the imaginary axis, the sign of A, denoted by $\operatorname{sgn}(A)$, is the limit of the sequence

$$A_{n+1} = \tfrac{1}{2}(A_n + A_n^{-1}), \qquad A_0 = A. \tag{3}$$

By using nilpotency arguments, we show in Lemma 3.3 that once the eigenvalues of A_n have converged, the overall convergence of A_n to $\operatorname{sgn}(A)$ will be complete in at most $\log_2(m) + 1$ more steps where m is the order of A. This places a well-defined limit on the effects of ill-conditioning and allows us to concentrate on the problem of choosing the scaling factors for the matrix recursion,

$$A_{n+1} = \tfrac{1}{2}(\gamma_n A_n + (\gamma_n A_n)^{-1}), \qquad A_0 = A, \quad \gamma_n > 0, \tag{4}$$

to optimize the convergence of the eigenvalues to ± 1 (see [1]).

An optimal scaling stategy is described in Theorem 3.6, but requires complete knowledge of the eigenvalues of A_n. Since this is impractical, we have developed a family of scaling methods that are optimal with respect to incomplete eigenvalue information. The simplest of these methods uses only the two eigenvalues corresponding to the dominant eigenvalues of A_n and its inverse. This method, which we refer to as "semioptimal" scaling, gave nearly optimal performance in numerical tests.

For the purposes of completeness, we also considered other suboptimal methods, including "spectral" scaling,

$$\gamma_n \equiv \sqrt{\rho(A_n^{-1})/\rho(A_n)}\,, \tag{5}$$

where the spectral radius $\rho(M)$ of a square matrix M, is the maximum modulus of its eigenvalues:

$$\rho(M) \equiv \max\{|\lambda| : \lambda \in \Lambda(M)\}. \tag{6}$$

Spectral scaling is asymptotically optimal as the eigenvalues of A_n approach the real axis (Theorem 3.4). However, if the eigenvalues of A_n are near the imaginary axis, then this method can result in unduly slow convergence (see Example 1).

Another commonly used method is "determinantal" scaling

$$\gamma_n^{\det} \equiv 1/|\det(A_n)|^{1/m}, \tag{7}$$

which has the tendency to bring the eigenvalues to the unit circle and then to the real axis (see the comments prior to Example 1). Unfortunately, determinantal scaling is not asymptotically optimal. This is clearly seen in Example 2. We also tested a

variety of other scaling strategies, including the 2-norm scaling method of Barraud [2] and a scaling method based on the Frobenius norm.

In the following, we will assume for the purposes of analysis that all arithmetic operations are performed exactly; this allows us to separate convergence effects from those of roundoff.

2. Polar decomposition. For $A = P\Sigma Q^H$, the sequence generated by (2) satisfies $A_n = P\Sigma_n Q^H$, where $\Sigma_n = \operatorname{diag}(\sigma_1^{(n)}, \cdots, \sigma_m^{(n)})$ and

$$\sigma_j^{(n+1)} = \tfrac{1}{2}(\gamma_n \sigma_j^{(n)} + (\gamma_n \sigma_j^{(n)})^{-1}), \qquad 1 \le j \le m. \tag{8}$$

This means that the rate of convergence of A_n to U is determined by how quickly the singular values of A_n converge to 1:

$$\|A_n - U\|_2 = \|P\Sigma_n Q^H - PQ^H\|_2 = \|\Sigma_n - I\|_2. \tag{9}$$

Although $\sigma_1^{(n)}, \cdots, \sigma_m^{(n)}$ are the singular values of A_n , their ordering may be scrambled by (8). Because of this, we will use the special notation,

$$s_1^{(n)} \equiv \sigma_{\max}(A_n), \qquad s_m^{(n)} \equiv \sigma_{\min}(A_n). \tag{10}$$

Define γ_n by

$$\gamma_n \equiv \sqrt{\|A_n^{-1}\|_2 / \|A_n\|_2} = 1/\sqrt{s_1^{(n)} s_m^{(n)}}. \tag{11}$$

We now need three elementary facts about Newton's method. For x in $(0, +\infty)$ let

$$f(x) = \tfrac{1}{2}(x + 1/x). \tag{12}$$

Then

$$f : (0, +\infty) \to [1, +\infty), \tag{13}$$

$$f(x) = f(1/x), \tag{14}$$

$$0 \le x_2 \le x_1 \le 1 \quad \text{or} \quad 1 \le x_1 \le x_2 \quad \Rightarrow \quad 1 \le f(x_1) \le f(x_2). \tag{15}$$

An immediate consequence of (9) and (13) is that

$$\|A_n - U\|_2 = s_1^{(n)} - 1, \qquad n \ge 1. \tag{16}$$

This gives the following practical error bound.

LEMMA 2.1. *For A_n defined by* (2) *and $n \ge 1$,*

$$\|A_n - U\|_2 \le \|A_n - A_n^{-H}\|_2. \tag{17}$$

Proof. By (13), all of the singular values of A_n are greater than or equal to 1. Combining this with (16) gives

$$\begin{aligned} \|A_n - U\|_2 &= s_1^{(n)} - 1 \\ &\le s_1^{(n)} - 1/s_1^{(n)} \\ &= \|\Sigma_n - \Sigma_n^{-1}\|_2. \\ &= \|A_n - A_n^{-H}\|_2. \end{aligned}$$

□

Remark 1. Since A_n and A_n^{-H} are available at each step, the bound in Lemma 2.1 is very convenient. It also provides a dependable assessment of the error because

$$\frac{s_1^{(n)} - 1/s_1^{(n)}}{s_1^{(n)} - 1} = 1 + 1/s_1^{(n)} \to 2, \tag{18}$$

which implies that

$$\frac{\|A_n - A_n^{-H}\|_2}{\|A_n - U\|_2} \to 2. \tag{19}$$

The next result shows that if A is of order m, then the optimally scaled Newton method converges in at most m steps. (See [10] for a discussion of the optimality of the scaling value given by (11) for polar decomposition.)

LEMMA 2.2. *For γ_n defined by* (11), $A_k = U$ *where k is the number of distinct singular values of A.*

Proof. The effect of the scaling is to make the largest and smallest singular values reciprocals of each other:

$$\sigma_{\max}(\gamma_n A_n) = \sqrt{s_1^{(n)}/s_m^{(n)}}, \qquad \sigma_{\min}(\gamma_n A_n) = \sqrt{s_m^{(n)}/s_1^{(n)}},$$

with the other singular values between these extremes. These reciprocal singular values then map to the same value by (14), which is also the largest singular value of A_{n+1} by (15). Thus the multiplicity of the largest singular value increases by 1 at each step until it is equal to k, which happens at the end of step $k-1$. At the start of the kth step, all of the singular values are equal and scaling takes them collectively to 1, which completes the convergence by (16). □

2.1. Speed of convergence bounds. Lemma 2.2 is a nice result, but its dependence on the dimension of A obscures the fact that convergence is rapid no matter what the dimension of A is. This speed results from the relation

$$s_1^{(n+1)} \cong \frac{1}{2}\sqrt{s_1^{(n)}}$$

for $s_1^{(n)}$ large, which moves $s_1^{(n)}$ quickly to a neighborhood of 1 where the quadratic convergence of Newton's method comes into play.

By (16), we only need to measure how quickly the largest singular value $s_1^{(n)}$ goes to 1. Assume that A has been prescaled so that $\|A^{-1}\|_2 = 1$; we may do this without loss of generality because the sequence A_n generated by (2) and (11) is invariant with respect to such prescaling. From (13), we know that the smallest singular value of A_n is greater than or equal to one, so $\gamma_n = (s_1^{(n)} s_m^{(n)})^{-1/2} \le (s_1^{(n)})^{-1/2}$. Thus,

$$\gamma_n s_1^{(n)} \le \sqrt{s_1^{(n)}}$$

and

$$s_1^{(n+1)} = f\left(\gamma_n s_1^{(n)}\right) \le f\left(\sqrt{s_1^{(n)}}\right) = \frac{1}{2}\left(\sqrt{s_1^{(n)}} + 1/\sqrt{s_1^{(n)}}\right)$$

by (15). We can conclude that if

$$x_{n+1} = \tfrac{1}{2}(\sqrt{x_n} + 1/\sqrt{x_n}), \qquad x_0 = s_1^{(0)}, \tag{20}$$

then

$$1 \leq s_1^{(n)} \leq x_n. \tag{21}$$

In this section, we obtain bounds on how quickly x_n in (20) converges to 1. The main idea is to work with numbers $\alpha = \alpha(n) > 1$, which map into $1 + 1/\alpha$ after n steps of (20):

$$x_n = 1 + 1/\alpha, \qquad x_0 = \alpha. \tag{22}$$

For example, $\alpha(1) = 4$ since $x_1 = \frac{1}{2}(\sqrt{4} + 1/\sqrt{4}) = 1 + \frac{1}{4}$. If we let $S_n = \{x > 1 : x_n \leq 1 + 1/x\}$, where x_n is given by (20) with $x_0 = x$, then it is not hard to show that $\alpha(n) = \sup\{x : x \in S_n\}$.

THEOREM 2.3. *Let $\alpha = \alpha(n)$ satisfy (22) and suppose that $\|A\|_2\|A^{-1}\|_2 \leq \alpha$. Then in n steps the error is less than or equal to $1/\alpha$:*

$$\|A_n - U\|_2 \leq 1/\alpha. \tag{23}$$

Proof. As in the proof of Lemma 2.1, prescale A so that $s_1^{(0)} = \|A\|_2\|A^{-1}\|_2$. By (21) and (22), $s_1^{(n)} \leq x_n = 1 + 1/\alpha$. Hence, $\|A_n - U\|_2 = s_1^{(n)} - 1 \leq 1/\alpha$ by (16). □

The proof of this theorem is short because all of the difficulty has been hidden in the numbers $\alpha(n)$ satisfying (22). In particular, (22) is an implicit relation on α that is rather complicated; α can be found numerically, but this does not give us much insight into the problem. Instead, the next theorem gives a very sharp lower bound on $\alpha(n)$.

THEOREM 2.4. *For $n \geq 3$, $\alpha(n) \geq (8^{2^{n/2}}/4)$.*

Proof. See Appendix 1 for the proof. □

Table 1 illustrates this bound.

TABLE 1
Lower bounds on $\alpha(n)$.

n	$\frac{8^{2^{(n/2)}}}{4}$	$\alpha(n)$
3	89.59	95.33
5	3.2×10^4	5.1×10^4
7	4.1×10^9	1.5×10^{10}

Because Theorem 2.4 does not depend on the dimension of A, it also applies to infinite-dimensional operators. Let W be a Hilbert space with inner product $\langle \cdot, \cdot \rangle$, and let $\{q_i\}$ and $\{p_i\}$ be two complete orthonormal bases for W. For any linear operator L on W, define the norm of L by $\|L\|_2 = \sup\{\|Lx\|_2 : \|x\|_2 = 1\}$, where the vector norm is induced by the inner product, i.e., $\|x\|_2^2 = \langle x, x \rangle$ for any $x \in W$. Let A be a linear operator on W such that $Aq_i = \sigma_i p_i$ where $0 < \sigma_{\min} \equiv \inf\{\sigma_i\} \leq \sigma_{\max} \equiv \sup\{\sigma_i\} < +\infty$. For any such A define the scaling factor $\gamma(A) = (\sigma_{\max}\sigma_{\min})^{-1/2}$. Then the sequence

$$A_{n+1} = \tfrac{1}{2}(\gamma(A_n)A_n + (\gamma(A_n)A_n^H)^{-1})$$

is well defined and converges to U where $Uq_i = p_i$.

THEOREM 2.5. *If A has only k distinct singular values then $A_k = U$. In any case, if*

$$\|A\|_2\|A^{-1}\|_2 \leq \frac{8^{2^{(n/2)}}}{4},$$

then

$$\|A_n - U\|_2 \leq \frac{4}{8^{2^{(n/2)}}} \quad \text{for } n \geq 3.$$

Proof. The first assertion has the same proof as Lemma 2.1 and the second follows from Theorems 2.3 and 2.4. □

2.2. Approximate optimal scaling. In practice, the optimal scaling value $\gamma_n = (\|A_n^{-1}\|_2/\|A_n\|_2)^{1/2}$ is not known exactly and some approximation must be used, which leads us to ask how scaling inaccuracies affect convergence. There are two distinct cases for this problem, depending on whether or not some of the singular values are far from 1.

As per the remarks following Lemma 2.2, the principal benefit of optimal scaling occurs in the "initial phase" in which any large singular values are rapidly brought to a neighborhood of 1. Fortunately, the effect of scaling errors in this phase is minimal (see the discussion following Theorem 2.6 below). After this, in the "terminal phase," quadratic convergence to 1 takes place but the increase in speed over the unscaled Newton method is only marginal. At the same time, the sensitivity of the convergence to small scaling errors increases, as seen by Theorem 2.6 below. Because of this, the standard approach to handling this problem is to switch to the unscaled Newton method ($\gamma = 1$) when A_n is sufficiently close to U; for example, see the algorithm of Higham in [7]. This "switching" problem can be stated in the following form. Which scaling values γ give better convergence than the unscaled Newton method? The next theorem answers this question.

THEOREM 2.6. *Let $A_{n+1}(\gamma) = (\gamma A_n + (\gamma A_n)^{-H})/2$. Then for $\gamma > 0$ and $n \geq 1$,*

$$\|A_{n+1}(\gamma) - U\|_2 \leq \|A_{n+1}(1) - U\|_2 \tag{24}$$

if and only if

$$\gamma_n^2 \leq \gamma \leq 1, \tag{25}$$

where γ_n is the optimal scaling factor given by (11).

Proof. See Appendix 1 for the proof. □

Because $\gamma_n \to 1$, inequality (25) shows that the margin of acceptable error in the approximate scaling factor narrows, in both an absolute and relative sense, as the singular values of A_n approach 1 in the terminal phase.

In order to illustrate the above, let us suppose that we can estimate the optimal scaling value to within a multiplicative factor f:

$$\frac{1}{f}\gamma_n \leq \gamma \leq f\gamma_n. \tag{26}$$

In this case, we should use the approximate scaling value in preference to $\gamma = 1$ as long as $f \leq 1/\gamma_n$ because this is equivalent to (25). In this case, the speed of convergence

in the initial phase (in which we use the estimated optimal scaling value) is controlled by the sequence

$$x_{n+1} = \tfrac{1}{2}(f\sqrt{x_n} + 1/(f\sqrt{x_n})), \qquad x_0 = s_1^{(0)}. \tag{27}$$

Subsequently, in the terminal phase when $f \geq 1/\gamma_n$, the speed of convergence is just that of the regular unscaled Newton method. An analysis of this combined method can be given as in Theorem 2.4, but for brevity we will simply consider a representative example. Under optimal scaling, if $\|A_0\|_2\|A_0^{-1}\|_2 = 10^{10}$, then convergence is complete in 7 steps as per the results in Table 1. For approximate optimal scaling, in which the scaling factor is determined to within, say, 50 percent (i.e., $f = 2$), 10 steps are needed. This can be seen by using the sequence generated initially by (27) with $x_0 = 10^{10}$ and then switching to the unscaled Newton method when $f = 2 \geq \sqrt{x_n}$. This is encouraging because an increase from 7 to 10 steps is negligible compared to the 38 steps that the unscaled Newton method requires for this problem. (For all numerical problems considered in this paper the iterations were terminated after the error, as measured by Lemmas 2.1 and 3.1, was less than 10^{-10}.)

We now examine aspects of convergence for three practical scaling strategies. The first involves estimating the 2-norms of A_n and A_n^{-1} by the power method [5]. For a square matrix M and a starting vector v_0, let $v_k = M^H M v_{k-1}$ and set $P_k(M) = (\|v_k\|_2/\|v_{k-1}\|_2)^{1/2}$. Then

$$\sigma_{\min}(M) \leq P_k(M) \leq \sigma_{\max}(M), \qquad k \geq 1. \tag{28}$$

Moreover, $P_k(M) \to \|M\|_2$ as k increases provided v_0 is not perpendicular to the singular vector space of M corresponding to the largest singular value of M. Because of roundoff effects we may assume that this condition is satisfied; see [5] for a discussion of this point. This suggests that we define a sequence of estimates of γ_n by

$$\gamma_n^k = (P_k(A_n^{-1})/P_k(A_n))^{1/2}. \tag{29}$$

For $n \geq 1$, the singular values of A_n are greater than or equal to 1, which when combined with (28) gives

$$\|A_n^{-1}\|/\|A_n\| = \gamma_n^2 \leq 1/\|A_n\| = (\sigma_{\min}(A_n^{-1})/\|A_n\|)^{1/2} \leq \gamma_n^k. \tag{30}$$

Thus if we agree to let $\tilde{\gamma}_n^k = \min\{1, \gamma_n^k\}$, then the $\tilde{\gamma}_n^k$ satisfies (25) and so must give better convergence than the unscaled Newton method.

Based on the theory of the power method [14], the asymptotic convergence of γ_n^k to γ_n should be linear. However the indeterminancy in this procedure has led to the search for other methods of approximating γ_n. For example, Higham [7] suggests using the scaling factor

$$\gamma_n^{1,\infty} = \left(\frac{\|A_n^{-1}\|_1\|A_n^{-1}\|_\infty}{\|A_n\|_1\|A_n\|_\infty}\right)^{1/4} \tag{31}$$

and gives the bounds

$$1/m^{1/4} \leq \gamma_n^{1,\infty}/\gamma_n \leq m^{1/4}. \tag{32}$$

If $m^{1/2} \leq 1/\gamma_n^{1,\infty}$, then by (25), $\gamma_n^{1,\infty}$ should be used in preference to $\gamma = 1$. However, (25) may be satisfied when this rather restrictive requirement is not met. Higham

[7] has found empirically that scaling with $\gamma_n^{1,\infty}$ is effective until A_n is within, say, 1 percent of U.

This brings us to the third scaling method, which is closely related to Higham's scaling procedure but is easier to analyze with regard to the switching problem. To motivate this method, consider the problem of finding $\gamma = \gamma_n^F$, which minimizes $\|\gamma A_n - (\gamma A_n)^{-H}\|_F$ where $\|\cdot\|_F$ denotes the Frobenius matrix norm [6]. (Compare with the 2-norm error bound in Lemma 2.1.) By using the relation $\|M\|_F^2 = Tr(M^H M)$ we find that

$$\gamma_n^F = \left(\frac{\|A_n^{-1}\|_F}{\|A_n\|_F} \right)^{1/2}. \tag{33}$$

From $\|M\|_F/\sqrt{m} \le \|M\|_2 \le \|M\|_F$, we have $1/m^{1/4} \le \gamma_n^F/\gamma_n \le m^{1/4}$, which has the same form as (32). The next lemma shows that this "Frobenius" scaling method is always better than the unscaled Newton method.

LEMMA 2.7. *For $n \ge 1$, $\gamma_n^2 \le \gamma_n^F \le 1$.*

Proof. Since $\|M\|_F^2 = \sigma_1^2(M) + \cdots + \sigma_m^2(M)$, we have

$$\gamma_n^F = \left(\frac{\frac{1}{\sigma_1^2} + \cdots + \frac{1}{\sigma_m^2}}{\sigma_1^2 + \cdots + \sigma_m^2} \right)^{1/4}, \tag{34}$$

where σ_j denotes the jth singular value of A_n. This gives $\gamma_n^F \le 1$ because $1 \le \sigma_m \le \cdots \le \sigma_1$. Also,

$$\gamma_n^2 = 1/(\sigma_1\sigma_m) \le 1/\sigma_1 = \left(\frac{m/\sigma_1^2}{m\sigma_1^2} \right)^{1/4} \le \gamma_n^F, \tag{35}$$

which completes the proof. □

Numerical tests on a wide variety of problems indicate that there is little difference between the performance of optimal scaling, the power method, Higham's scaling, and Frobenius scaling. Even for extreme problems with only two distinct singular values, the spread between these methods is small compared to the unscaled Newton method. For such an example, with $\|A\|_2\|A^{-1}\|_2 = 10^{12}$ and $m = 20$, optimal scaling took two steps, the power method took four, while Higham's method and Frobenius scaling required six and seven steps, respectively. The unscaled Newton method took 45 steps. This also illustrates our general finding that Higham's method is as good as or slightly better than Frobenius scaling, which leads us to suspect that an analogue of Lemma 2.7 may be true for $\gamma_n^{1,\infty}$ in (31).

3. Matrix sign function. For polar decomposition, the convergence of Newton's method is determined by the convergence of the singular values to 1 as in (16). For the matrix sign problem, the same can be said of the convergence of the eigenvalues to ± 1. However, the conditioning of the associated eigenspaces also affects convergence, at least in the initial stages of the iteration. In order to give more substance to these remarks, assume that A has no eigenvalues on the imaginary axis and define $S = \operatorname{sgn}(A)$ to be the limit of the unscaled Newton sequence (3). Let A have Schur form $A = Q\tilde{A}Q^H$ where Q is unitary and the tilde over a matrix is used to indicate that the matrix is upper triangular. Since the inverse of an upper triangular matrix is again upper triangular, the sequence A_n in (3) has Schur form $A_n = Q\tilde{A}_n Q^H$. The same applies to the scaled sequence (4).

The use of the upper triangular tilde notation is convenient because the equivalent Schur form of formulas such as (3) or (4) can be found by simply inserting tildes over the appropriate matrices. From this we see that the eigenvalues $\{\lambda_1^{(n)}, \cdots, \lambda_m^{(n)}\}$ of A_n are the main diagonal entries of $\tilde{A}_n$ and satisfy the scalar recursions,

$$\lambda_j^{(n+1)} = \tfrac{1}{2}(\gamma_n \lambda_j^{(n)} + (\gamma_n \lambda_j^{(n)})^{-1}), \qquad 1 \le j \le m. \tag{36}$$

For γ_n real and positive, it can be shown that $\lambda_j^{(n)} \to \operatorname{sgn}(\lambda_j^{(0)})$ if and only if $\gamma_n \to 1$, where $\operatorname{sgn}(z) = 1$ if $\operatorname{Re}(z) > 0$ and $\operatorname{sgn}(z) = -1$ if $\operatorname{Re}(z) < 0$. The decoupling of the main diagonal elements in (36) means that the eigenvalue convergence is independent of the upper triangular entries of $\tilde{A}_n$. These entries reflect the nonnormality [6] of A_n. This raises the question of how deviation from normality affects the convergence of A_n to S and whether overall convergence will occur at the same rate as that of the eigenvalues. The next three lemmas explore this problem; the first gives practical error estimates similar to those of Lemma 2.1 for the polar decomposition.

LEMMA 3.1. *For the scaled Newton method* (4), *let* $A_n = S + E_n$. *If* A_n *is close enough to* S *so that* $\|E_n\|_2 \|S\|_2 \le \frac{1}{2}$, *then*

$$\|A_n - S\|_2 \le \|A_n - A_n^{-1}\|_2 \tag{37}$$

and

$$\frac{\|A_n - S\|_2}{\|S\|_2} \le \|A_n^2 - I\|_2. \tag{38}$$

Proof. From the definition of S, we have that $S^2 = I$ and $S^{-1} = S$. Moreover, S commutes with $A = A_0$ so it must also commute with each A_n in (4). Using this and some algebra we find that

$$E_n = (A_n - A_n^{-1})(I + E_n S)(2I + E_n S)^{-1}. \tag{39}$$

Thus for $\|E_n\|_2 \|S\|_2 = \epsilon \le \frac{1}{2}$,

$$\|E_n\|_2 \le \left(\frac{1+\epsilon}{2-\epsilon}\right) \|A_n - A_n^{-1}\|_2 \le \|A_n - A_n^{-1}\|_2, \tag{40}$$

which proves (37). The relative error bound (38) can be proved in a similar fashion (see [11]). □

Remark 2. As n increases, the error bound in (37) gives a reliable estimate of the true error because

$$\frac{\|A_n - S\|_2}{\|A_n - A_n^{-1}\|_2} \to \frac{1}{2} \tag{41}$$

by (39). The fourth and fifth columns of Table 2 give a comparison of the exact error and the estimate in (37) for a particular problem.

In order to illustrate the effect that eigenspace ill-conditioning can have on the convergence of the unscaled Newton method, consider an example suggested by an anonymous reviewer of this paper. If A_n is a Jordan block with ones on the main diagonal as well as the first superdiagonal, then the eigenvalues have converged to 1 but overall convergence is not complete because $\operatorname{sgn}(A_n) = I \ne A_n$. Problems of this

type were considered by Barraud [2], who showed that the asymptotic convergence is quadratic. However, a more general approach to the ill-conditioning problem allows us to conclude that any delays in overall convergence due to ill-conditioning are minor (i.e., the number of extra steps is at most logarithmic in the order of the largest Jordan block).

LEMMA 3.2. *For the unscaled Newton method* (3), *let* $A_n = S + E_n$. *Then*

$$A_{n+1} = S + A_n^{-1}E_n^2/2 \tag{42}$$

and

$$A_{n+1}^2 = I + E_n^2 - A_n^{-1}E_n^3 + (A_n^{-1}E_n^2/2)^2. \tag{43}$$

Proof. Use (3) and the fact that A_n, E_n, and S all commute for the proof. □

Equation (42) shows the quadratic convergence of the unscaled Newton sequence with possible ill-conditioning entering via the term A_n^{-1}. Since $A_n^{-1} \to S^{-1} = S$, the norm of S can affect the asymptotic convergence. (See [8] for a discussion of how the norm of S determines the "asymptotic" conditioning of the matrix sign function.) It is interesting to note that the term A_n^{-1} is missing from the principal part of the error term in (43); this proves useful when we turn to the problem of estimating the dominant eigenvalues of A_n and A_n^{-1}. Another consequence of this lemma is that once the eigenvalues of A_n have converged, the overall convergence of A_n to S will be complete in no more than $\log_2(m) + 1$ extra steps as shown below. This places a well-defined limit on the convergence effects of ill-conditioning.

LEMMA 3.3. *For the unscaled Newton sequence* (3), *if* A_n *has eigenvalues* ± 1 *for some value of* n, *then* $A_{n+p} = S$ *for* $2^p \geq m$.

Proof. Since the matrix 2-norm is invariant with respect to unitary similarity transformations, we need only consider the convergence of the upper triangular Schur sequence $\tilde{A}_n$ where $A_n = Q\tilde{A}_nQ^H$. Now if $\tilde{A}_n = \tilde{S} + \tilde{E}_n$ and the eigenvalues of $\tilde{A}_n$ have converged, then $\tilde{E}_n$ must be strictly upper triangular and hence is nilpotent of order m, i.e., $\tilde{E}_n^m = 0$. This means that $\tilde{E}_n^2$ is nilpotent of order $m/2$, or the next highest integer if m is not divisible by 2. The same is true for $\tilde{E}_{n+1}$ because by (42), $\tilde{E}_{n+1} = \tilde{A}_n^{-1}\tilde{E}_n^2/2$. (Note that $\tilde{A}_n^{-1}$ and $\tilde{E}_n$ commute since $\tilde{A}_n$ and $\tilde{S}$ commute.) Continuing in this way shows that $\tilde{E}_{n+p}$ is nilpotent of order $m/2^p$ or the next higher integer if m is not divisible by 2^p. Thus for $2^p \geq m$, $\tilde{E}_{n+p}$ is nilpotent of order 1 and so must be 0. □

Remark 3. By working with the Jordan form instead of the Schur form we can actually show that in Lemma 3.3, $A_{n+p} = S$ for $2^p \geq \tilde{m}$ where $\tilde{m}$ is the order of the largest Jordan block of A. However, the bound in Lemma 3.3 is more reliable numerically because of the sensitivity of the Jordan structure with respect to small perturbations.

In order to illustrate Lemma 3.3, suppose that A_0 is an 80×80 Jordan block matrix with ones on the main diagonal, as discussed above. Then $A_7 = S$ because $2^7 \geq 80$; by actual computation we also have $A_6 \neq S$. This shows that the bound in Lemma 3.3 can be attained. However, for most problems the effect of even large ill-conditioning is not as pronounced as this because once the eigenvalues get close to ± 1, then $\tilde{E}_n$ is "nearly" nilpotent of order $\tilde{m} \leq m$. Thus the reduction of $\tilde{E}_{n+i}$ proceeds in a manner similar to the above proof at the same time that the eigenvalues are completing their convergence. This overlap means that by the time the eigenvalue convergence is complete, the norm of the error should also be near 0 and very few

additional steps should be needed. In order to test this numerically we set up a series of matrices of the form

$$A = D + T, \tag{44}$$

where $D = \operatorname{diag}(\lambda_1, \cdots, \lambda_m)$ with T strictly upper triangular. This allowed us to see how convergence in the normal case $T = 0$, compared to the nonnormal case $T \neq 0$. For example, Table 2 gives the results of taking $m = 80$ and letting the eigenvalues λ_j be chosen at random in balls about ± 1, respectively, of radius 0.75. The upper entries of T were then taken randomly in the interval [-1,1], and subsequently rescaled to attain the desired ill-conditioning as measured by the norm of S. Tests of this type support the idea that, except for extreme cases such as the large Jordan blocks discussed above, the number of steps needed for convergence is largely determined by the initial distribution of the eigenvalues with the conditioning of the associated eigenspaces playing a secondary role. Note in that Table 2 the final convergence of the ill-conditioned problems (columns 3 and 4) is much faster than just quadratic, which reflects the nilpotency reduction described above. The last column of Table 2 gives the values of $\|A_n - A_n^{-1}\|_2$ and was included to test the error estimate (37) in Lemma 3.1. Comparing columns 4 and 5 we see that this estimate is excellent even when the assumption $\|E_n\|_2\|S\|_2 \leq \frac{1}{2}$ of Lemma 3.1 is not satisfied.

TABLE 2
The effect of ill-conditioning on convergence for matrices with the same initial eigenvalues.

Iteration number	Error for $\|S\|_2 = 1$	Error for $\|S\|_2 = 10^3$	Error for $\|S\|_2 = 10^6$	Est. error for $\|S\|_2 = 10^6$
1	3.8×10^{0}	2.6×10^{3}	2.7×10^{6}	3.1×10^{7}
2	1.0×10^{0}	9.8×10^{5}	1.3×10^{7}	1.5×10^{7}
3	1.0×10^{-1}	4.9×10^{5}	5.5×10^{6}	9.8×10^{6}
4	1.8×10^{-3}	2.6×10^{5}	1.0×10^{6}	2.0×10^{6}
5	9.2×10^{-7}	3.9×10^{2}	8.8×10^{2}	2.0×10^{3}
6	3.9×10^{-13}	2.4×10^{-4}	5.0×10^{-4}	1.0×10^{-3}
7	$\leq 10^{-16}$	$\leq 10^{-16}$	$\leq 10^{-16}$	$\leq 10^{-16}$

If the eigenvalues of A_n are all real, then there is an immediate correspondence between selecting the best scaling constant for the matrix sign problem and selecting the best scaling constant for the polar decomposition. Barraud [2] has shown that in this case, the eigenvalues converge in a finite number of steps under spectral scaling. Combining this with Lemma 3.2 and Theorem 2.4 gives the following.

THEOREM 3.4. *If the eigenvalues of A are real and nonzero then under spectral scaling, $A_{k+p} = \operatorname{sgn}(A)$, where k is the number of distinct eigenvalues of A and p is any integer greater than or equal to $\log_2(m)$. In any case,*

$$\text{if} \quad n \geq 3 \quad \text{and} \quad \rho(A)\rho(A^{-1}) \leq \frac{8^{2^{(n/2)}}}{4}, \quad \text{then } \rho(A_n - \operatorname{sgn}(A)) \leq \frac{4}{8^{2^{(n/2)}}}.$$

Proof. Under the assumption that the eigenvalues of A are real, the scalar relation (36) subject to (5) has exactly the same form as the relation (8) subject to (11), except that the eigenvalues may be negative whereas the singular values are always positive. However, this has no effect on the convergence arguments used to prove Theorem 2.5, so those arguments can be applied to the matrix sign function, as long as we take care

to replace the matrix 2-norm with the spectral radius. This establishes the spectral radius bound and shows that the eigenvalues of the sequence in (4) converge in k steps. By Lemma 3.2, the overall convergence of A_n to $\operatorname{sgn}(A)$ is complete in p more steps because (5) reduces to the unscaled Newton method when all of the eigenvalues are at ± 1. □

It is somewhat surprising that this theorem can be used to show that with the right scaling, convergence for arbitary real matrices need never take more than $m+\log_2 m+1$ steps.

THEOREM 3.5. *Let A be a real $m\times m$ matrix with no eigenvalues on the imaginary axis. Then there exist scaling constants for the sequence* (4) *such that $A_p = \operatorname{sgn}(A)$ for $p \le m + \log_2 m + 1$.*

Proof. Suppose that A has p_1 real eigenvalues and p_2 complex conjugate eigenvalue pairs, where $p_1 + 2p_2 = m$. Let the first scaling factor be $\gamma = 1/\rho$ where one of the complex conjugate eigenvalue pairs is of the form $\rho e^{\pm\theta i}$. After scaling and taking one Newton step, this pair is mapped to the real eigenvalue $\cos(\theta)$ and has multiplicity at least 2. Repeating this procedure shows that in at most p_2 steps all of the eigenvalues can be made real and the number of distinct eigenvalues is then at most $k = p_1 + p_2$ because of the doubling up of multiplicity on the previously complex eigenvalues. By Theorem 3.4, $p_1 + p_2 + \log_2 m + 1$ more steps will give convergence under spectral scaling. Thus the total number of steps needed is at most $p_1 + 2p_2 + \log_2 m + 1 = m + \log_2 m + 1$. □

Remark 4. The same proof shows that if A is complex, then with the proper scaling, convergence need not take more than $2m+\log_2 m+1$ steps. See also [9], which discusses rational eigenvalue assignment techniques that can be used to map all of the eigenvalues of A to ± 1 in one step provided that the eigenvalues are known. In this case the total number of steps for overall convergence would be at most $\log_2(m) + 2$.

3.1. Optimal and suboptimal scaling. Since the overall convergence is determined in the main by the convergence of the eigenvalues, we now turn to the problem of selecting the scaling constants γ_n in (4) to maximize the rate of this "spectral" convergence. If $d(x)$ is the distance from x to $\operatorname{sgn}(x)$, then we define the optimal scaling constant γ_n^* to be that positive real number that minimizes $\max_j d(\lambda_j^{(n+1)})$, where $\lambda_j^{(n+1)}$ is given by (36). This optimal scaling constant depends on how the distance function d is defined. In general, the Euclidean metric $d_e(x) = |x - \operatorname{sgn}(x)|$ is not the best choice because $d(x_0) \neq d(1/x_0)$ unless $x_0 = \pm 1$, whereas both x_0 and $1/x_0$ map to the same point, $x_1 = (x_0 + 1/x_0)/2$ under Newton's method, and so are essentially equidistant from $\operatorname{sgn}(x_0)$. (See [1] for a discussion of this point.)

A more appropriate metric can be derived from the observation that if $s = \operatorname{sgn}(x_0)$ and $x_1 = (x_0 + 1/x_0)/2$, then

$$\frac{x_1 - s}{x_1 + s} = \left(\frac{x_0 - s}{x_0 + s}\right)^2 .$$

This suggests that we work with the "Cayley" metric, which is the modulus of the appropriate Cayley transform of x:

$$d(x) = \begin{cases} |x-1|/|x+1|, & \operatorname{Re}(x) > 0, \\ |x+1|/|x-1|, & \operatorname{Re}(x) < 0. \end{cases}$$

In this case, the reciprocal values x_0 and $1/x_0$ are equidistant from $\operatorname{sgn}(x_0)$ and $d(x_1) = d^2(x_0)$. The problem of finding the optimal scaling constant γ_n^* when A_n has complex eigenvalues is addressed in the following theorem.

THEOREM 3.6. *Let* $\lambda_j^{(n)} = \pm\rho_j e^{i\theta_j}$, *where* $\rho_j > 0$, $|\theta_j| < \pi/2$; $1 \le j \le m$. *Then*

$$(\gamma_n^*)^2 = \frac{1}{\rho_i\rho_j}\left(\frac{\rho_i\cos\theta_i - \rho_j\cos\theta_j}{\rho_i\cos\theta_j - \rho_j\cos\theta_i}\right) \tag{45}$$

for some pair i *and* j. *If* $i = j$, *then we interpret* (45) *as* $\gamma_n^* = 1/\rho_i$.

Proof. See Appendix 2. □

This description of the optimal scaling factor is somewhat frustrating for several reasons. First, it requires that all of the eigenvalues of A_n be known, which is almost never the case. Second, even if we did know all of the eigenvalues, it is not clear whether a simple procedure exists for finding the indices i and j such that (45) holds. (A procedure based on exhaustive testing over the index pairs was used for the numerical examples below.) Finally, the lack of a simple expression for the optimal scaling value means that it is much harder to make precise statements about how scaling inaccuracies affect convergence, such as when an approximate optimal scaling value will give better convergence than the unscaled Newton method. However, when all of the eigenvalues are real (which is approximately the asymptotic case), direct analogues of Theorem 2.6 and Lemma 2.7 apply with the 2-norm being replaced by the spectral radius and the singular values replaced by the moduli of the eigenvalues.

If only some of the eigenvalues of A_n are known, then the best scaling factor that can be determined from this limited knowledge will have the form of (45) except that the indices i and j vary only over the known eigenvalues. This is easily seen by applying Theorem 3.6 to a matrix of smaller order whose spectrum consists of the known eigenvalues. Thus we have a family of scaling methods that includes at one extreme optimal scaling and at the other a semioptimal scaling method that utilizes only two eigenvalues of A_n. This latter method consists of estimating the dominant eigenvalues of A_n and A_n^{-1} via the power method [12]–[14] and then selecting the best scaling value as in Theorem 3.6. From the remarks following Lemma 3.2, we see that it is generally better to use the power method on A_n^2 and $(A_n^{-1})^2$ rather than A_n and A_n^{-1} because both A_n^2 and $(A_n^{-1})^2$ approach $S^2 = (S^{-1})^2 = I$ as n increases. This approach to normality means that the power methods of [12] are especially appropriate; see Appendix 2 for MATLAB routines that implement these techniques. As an illustration, suppose that

$$A_n = \begin{bmatrix} 1 & x \\ 0 & -1 \end{bmatrix}. \tag{46}$$

Then applying the power method to just A_n can work poorly when x is large whereas using A_n^2 works well since $A_n^2 = I$.

Remark 5. By estimating the eigenvalues in this way and then keeping track of them from step to step via (36), we could optimize over an increasing set of eigenvalues until we were working with the entire spectrum. However, it is not clear how small inaccuracies in the eigenvalue estimates propagate under (36) when the eigenvalues are originally near the imaginary axis. Moreover, as the number of eigenvalues in the optimizing set increases, the work involved in finding the optimal scaling factor goes up rapidly.

For completeness in the following, we compare semioptimal scaling with other suboptimal scaling methods such as spectral scaling (5). For this method and all the others except optimal scaling, when the estimated error $\|A_n - A_n^{-1}\|_F$ was less than 0.01 the unscaled Newton method was used.

Since the eigenvalues of A_n approach the real axis as n increases, spectral scaling is asymptotically optimal. The following theorem shows that when the eigenvalues of A are reasonably near the real axis this scaling strategy works well.

THEOREM 3.7. *Assume that the eigenvalues of A lie within an angle of $\theta = \pi/4$ of the (positive or negative) real axis. For spectral scaling,*

$$\text{if} \quad \rho(A)\rho(A^{-1}) \leq \frac{3^{2^{(n/2)}}}{2}, \quad \text{then } \rho(A_n - \operatorname{sgn}(A)) \leq \frac{2}{3^{2^{(n/2)}}}.$$

Proof. A more general version of this theorem can be proved by first defining $\alpha = \alpha(n, \theta)$ such that if all the eigenvalues lie within an angle θ of the real axis and if $\rho(A)\rho(A^{-1}) \leq \alpha$, then $\rho(A_n - \operatorname{sgn}(A)) \leq 1/\alpha$. We can then proceed in much the same manner as in the proof of Theorem 2.4, but we will omit the details. □

In order to compare this with Theorem 3.4, write

$$\frac{3^{2^{(n+2/2)}}}{2} = \frac{\left(3^2\right)^{2^{(n/2)}}}{2} = \frac{9^{2^{(n/2)}}}{2} > \frac{8^{2^{(n/2)}}}{4}.$$

Thus two extra steps in Theorem 3.7 give the same effect as that seen in Theorem 3.4.

In spite of the above, if the eigenvalues of A cluster along the imaginary axis, spectral scaling can actually be slower than the unscaled Newton method even in the normal case, as seen in Example 1 below. Because of this, we also consider the commonly used determinantal scaling method given by (7), which has been suggested by several authors [1], [4] in various forms. The rationale behind this method is that since the determinant of a matrix is equal to the product of its eigenvalues, the geometric average of the eigenvalues of $\gamma_n^{\det} A_n$ is equal to one. This is significant because if z_0 has modulus 1, say $z_0 = e^{i\theta}$, then one step of Newton's method takes z_0 to the real axis, $z_1 = (e^{i\theta} + 1/e^{i\theta})/2 = \cos\theta$. Thus, determinantal scaling has a tendency to move eigenvalues to the unit circle and then to the real axis. This effect is seen in Example 1, in which determinantal scaling is nearly optimal. Moreover, $\gamma_n^{\det}$ is easily computed during the formation of A_n^{-1}.

Unfortunately, determinantal scaling is not asymptotically optimal, i.e., $\gamma_n^{\det}$ in (7) is not equal to the value in (5) as the eigenvalues approach the real axis. This defect is clearly evident in Example 2 below, in which all of the initial eigenvalues are clustered at ± 1 except for one outlier x. If we take m large enough so that $|x|^{1/m} = |\det A_n|^{1/m}$ is near 1, then determinantal scaling is effectively the same as Newton's method with no scaling, and convergence will be slow.

In addition, we also tested Higham scaling (31), Frobenius scaling (33), and the 2-norm scaling suggested by Barraud [2], [3],

$$\gamma_n^B = \left(\frac{\|A_n^{-1}\|_2}{\|A_n\|_2}\right)^{1/2}. \tag{47}$$

Since these three methods all gave results that were similar, we will report only on the Barraud 2-norm scaling for the following examples.

Example 1. Let $A = D + T$ as in (44) with $\lambda_j = \pm 1 \pm 1000(j/m)i$ for $1 \leq j \leq m$ with $i = \sqrt{-1}$. Table 3 gives the number of steps needed to ensure that $\|A_n - A_n^{-1}\|_2 \leq 10^{-10}$ for six different scaling methods with the upper entries of T uniformly and randomly distributed in the interval $[-1, 1]$.

TABLE 3
Steps to convergence for Example 1 *with eigenvalues parallel to the imaginary axis.*

Order of A	Opt. scaling	Semioptimal scaling	Determinantal scaling	Spectral scaling	2-norm scaling	No scaling
10	15	15	16	18	18	25
20	15	16	16	24	22	25
40	15	16	16	26	28	25

Example 2. Let $A = D + T$ as in (44) with $\lambda_1 = x$ and $\lambda_j = 1$ for $j = 2, 3, \cdots, m$. For $m = 20$ and $x = 1000$, the determinantal scaling factor is near one as described above and gives slow convergence compared to the other methods. For complex outliers, this pattern is even more pronounced, as seen in Table 4.

TABLE 4
Steps to convergence for Example 2 *with eigenvalues clustered at* ± 1 *with one outlier at* x.

Outlier x	T	Optimal scaling	Semioptimal scaling	Determinantal scaling	Spectral scaling	2-norm scaling	No scaling
1000	0	2	2	13	2	2	15
1000	Random	7	8	13	8	9	15
1+1000i	0	3	3	22	8	8	25
1+1000i	Random	7	9	22	15	15	25

TABLE 5
Steps to convergence for Example 3 *with random eigenvalues.*

Order of A	Optimal scaling	Semioptimal scaling	Determinantal scaling	Spectral scaling	2-norm scaling	No scaling
10	7	8	9	8	8	13
20	9	9	10	9	9	14
40	9	10	10	10	10	16

The next example shows that for random matrices, all of the scaling methods considered here are nearly optimal, with the exception of the unscaled Newton method.

Example 3. Let $A = D + T$ as in (44) with eigenvalues $\lambda = \pm x \pm iy$ where x and y are randomly and uniformly distributed in the interval $[0, 100]$. The results are given in Table 5 with the upper entries of T uniformly and randomly distributed in $[-1, 1]$.

4. Conclusion. Optimally scaled Newton's method for the polar decomposition converges very rapidly, regardless of the dimension of the problem (Theorem 2.5). It has been shown (Theorem 2.6) that approximate optimal scaling strategies provide better results than the unscaled Newton's method if and only if the approximate scaling factor lies between the square of the optimal factor and 1. This condition is always satisfied by optimal scaling approximations based on the power method as well as by the Frobenius scaling method (Lemma 2.7).

For the matrix sign function, optimal scaling is more complicated and complete knowledge of the eigenvalues of A is required (Theorem 3.6). Because this is impractical, a family of scaling methods has been derived that is optimal with respect to partial eigenvalue knowledge. This family includes optimal scaling and a semioptimal scaling method based on the dominant eigenvalues of the matrix and its inverse.

For the set of test problems considered, semioptimal scaling gave nearly optimal performance but analytical bounds relating the convergence under semioptimal scaling with that under optimal scaling are still needed. The test problems also show that commonly used methods based on determinantal scaling and spectral scaling can be unduly slow.

5. Appendix 1. In this section we give the proofs of Theorems 2.4, 2.6, and 3.6. We first need a technical lemma.

LEMMA 5.1. *For* $n \geq 1$, $\alpha(n+2) \geq 4\alpha^2(n) - 3$.

Proof. Let β map into $\alpha = \alpha(n)$ in one step of (20). That is, $\alpha = \frac{1}{2}(\sqrt{\beta} + 1/\sqrt{\beta})$, and

$$4\alpha^2 \geq \beta = 2\alpha^2 + 2\alpha^2\sqrt{1 - 1/\alpha^2} - 1 \geq 4\alpha^2 - 3 \,. \tag{48}$$

Let $x_0 = \beta$ and define $x_1, x_2, \cdots$ by (20). Since $x_1 = \alpha$, n more steps of (20) gives $x_{n+1} = 1 + 1/\alpha$ by (22). One more step gives

$$x_{n+2} = \tfrac{1}{2}(\sqrt{1 + 1/\alpha} + 1/\sqrt{1 + 1/\alpha}) \equiv r(1/\alpha).$$

Standard calculus arguments show that

$$r(z) = r(0) + r'(0)z + r''(\xi)z^2/2 \leq 1 + z^2/8,$$

for $0 < z < 1$ and some $0 \leq \xi \leq z$. Thus $x_{n+2} \leq 1 + 1/(8\alpha^2) \leq 1 + 1/\beta$ by (48). Since $x_0 = \beta$, this implies that $\beta \in S_{n+2}$. By the remarks following (22) and (48), $4\alpha^2(n) - 3 \leq \beta \leq \alpha(n+2)$. □

Proof of Theorem 2.4. We have already seen that $\alpha(1) = 4$. If $x_0 = 15$ then $x_2 < 1 + 1/(15)$, which means that $15 < \alpha(2)$. Similarly, $\alpha(3) > 95$. By the preceding lemma,

$$\alpha(n_0 + 2) \geq 4\alpha^2(n_0) - 3 = 3.99\alpha^2(n_0) + (\alpha^2(n_0)/100 - 3) \geq 3.99\alpha^2(n_0) \,,$$

for $n_0 \geq 3$. Applying this twice gives $\alpha(n_0 + 4) \geq (3.99)^3\alpha^4(n_0)$, and in general,

$$\alpha(n_0 + 2k) \geq \frac{1}{3.99}(3.99\alpha(n_0))^{2^k}.$$

Let $n = n_0 + 2k$. Then

$$\alpha(n) \geq \frac{1}{3.99}(3.99\alpha(n_0))^{2^{(n-n_0)/2}}.$$

For odd $n \geq 3$, use $n_0 = 3$ and $\alpha(3) \geq 95$ to get

$$\alpha(n) \geq \frac{1}{3.99}(379.05)^{2^{(n-3)/2}} = \frac{1}{3.99}((379.05)^{2^{-1.5}})^{2^{n/2}} \geq \frac{1}{3.99}(8.16)^{2^{n/2}} \geq \frac{8^{2^{n/2}}}{4} \,.$$

For even $n \geq 3$, use $n_0 = 4$ and $\alpha(4) \geq 1286$ to get

$$\alpha(n) \geq \frac{1}{3.99}(5131.14)^{2^{(n-4)/2}} = \frac{1}{3.99}((5131.14)^{2^{-2}})^{2^{n/2}} \geq \frac{1}{3.99}(8.46)^{2^{n/2}} \geq \frac{8^{2^{n/2}}}{4} \,.$$

This completes the proof of Theorem 2.4. □

Proof of Theorem 2.6. Since $n \geq 1$, the singular values of A_n satisfy $1 \leq s_m^{(n)} \leq \cdots \leq s_1^{(n)}$. For convenience we will omit the superscripts on these singular values. Denoting the largest singular values of $A_{n+1}(\gamma)$ by $\tilde{s}_1$ we find that

$$\tilde{s}_1(\gamma) = (\gamma s_1 + 1/(\gamma s_1))/2 \quad \text{if } 1/(\gamma s_m) \leq \gamma s_1, \tag{49}$$

and

$$\tilde{s}_1(\gamma) = (\gamma s_m + 1/(\gamma s_m))/2 \quad \text{otherwise.} \tag{50}$$

For the unscaled Newton method, the largest singular value of $A_{n+1}(1)$ is $\tilde{s}_1(1) = (s_1 + 1/s_1)/2$. By (17), $\|A_{n+1}(\gamma) - U\|_2 = \tilde{s}_1(\gamma) - 1$, so we only need to show that (25) is equivalent to

$$\tilde{s}_1(\gamma) \leq \tilde{s}_1(1). \tag{51}$$

Since $\gamma_n = (\|A_n^{-1}\|_2/\|A_n\|_2)^{1/2}$, we have that $\tilde{s}_1(\gamma)$ is given by (49) if $\gamma_n \leq \gamma$ and by (50) otherwise.

Case 1. Suppose that $\gamma_n \leq \gamma$. Then $\gamma s_1 \geq \gamma_n s_1 = (s_1/s_m)^{1/2} \geq 1$. For the Newton function f in (11), $\tilde{s}_1(\gamma) = f(\gamma s_1)$ and $\tilde{s}_1(1) = f(s_1)$. Thus, by (14) we may conclude that for $\gamma_n \leq \gamma$ the inequality (51) is equivalent to $\gamma \leq 1$.

Case 2. Suppose that $\gamma \leq \gamma_n$. Then by (50), $1/(\gamma s_m) \geq \gamma_n s_1 = (s_1/s_m)^{1/2} \geq 1$. This gives $\tilde{s}_1(\gamma) = f(1/(\gamma s_m))$. By (14) this means that for $\gamma \leq \gamma_n$ the inequality (51) is equivalent to $1/(\gamma s_m) \leq s_1$, which in turn is the same as $\gamma_n^2 = 1/(s_1 s_m) \leq \gamma$. Joining the results of these two cases shows that (51) is equivalent to (25). This completes the proof of Theorem 2.6. □

We now turn to the proof of Theorem 3.6. First we need two technical lemmas.

LEMMA 5.2. *Let* $z = \pm\rho e^{i\theta}$ *for* $\rho > 0$, $|\theta| < \pi/2$, *and set* $f(\gamma, z) = d^2(\gamma z)$. *Then*

$$f(\gamma, z) = \frac{\gamma^2\rho^2 - 2\gamma\rho\cos\theta + 1}{\gamma^2\rho^2 + 2\gamma\rho\cos\theta + 1}, \qquad \frac{\partial f}{\partial \gamma} = \frac{4\rho\cos\theta(\gamma^2\rho^2 - 1)}{(\gamma^2\rho^2 + 2\gamma\rho\cos\theta + 1)^2}. \tag{52}$$

Also, if $f(\gamma, z_1) = f(\gamma, z_2)$ *for* $\gamma > 0$, *then either* $z_1 = \pm z_2$ *or*

$$\gamma^2 = \frac{1}{\rho_1\rho_2}\left(\frac{\rho_2\cos\theta_2 - \rho_1\cos\theta_1}{\rho_2\cos\theta_1 - \rho_1\cos\theta_2}\right). \tag{53}$$

Proof. The proof is obtained through standard calculus. □

LEMMA 5.3. *For* $1 \leq j \leq m$, *let* $f_j = f_j(\gamma)$ *be nonnegative continuously differentiable functions over* $0 \leq \gamma \leq \Gamma$. *Let* γ^* *be the value of* $\gamma \in [0, \Gamma]$ *that minimizes* $\max_j f_j(\gamma)$. *If* $\gamma^* \neq 0$ *or* Γ *then either* $f_j'(\gamma^*) = 0$ *for some* j *or* $f_j(\gamma^*) = f_i(\gamma^*)$ *for some pair* $i \neq j$.

Proof. If $f_j(\gamma^*) = f_i(\gamma^*)$ for some $i \neq j$, then we are done. If not, then by rearranging if necessary, suppose that $f_1(\gamma^*) = \min_\gamma \max_j f_j(\gamma)$, with $f_1(\gamma^*) > f_j(\gamma^*)$ for $j > 1$. By continuity, $f_1(\gamma^*+\delta) > f_j(\gamma^*+\delta)$ for all δ sufficiently small. Thus, if $f_1'(\gamma^*) \neq 0$, then there exists a δ such that $f_1(\gamma^*) > f_1(\gamma^* + \delta) \geq \max_j f_j(\gamma^* + \delta)$, which is a contradiction. □

Proof of Theorem 3.6. For $\lambda_j^{(n+1)}$ given by (36), set $f_j(\gamma) = d(\lambda_j^{(n+1)})$ for the Cayley metric, so that $f_j(\gamma) = d^2(\gamma\lambda_j^{(n)})$. Say $1/\rho_1 = \max_j(1/\rho_j)$. By Lemma 3.1, $f_j'(\gamma) > 0$ if $\gamma > 1/\rho_1$, so $\gamma^* \leq 1/\rho_1$. If $\gamma^* = 1/\rho_1$ then (45) holds with $i = j = 1$. Moreover, $f_j(0) = 1 \geq f_j(1/\rho_1)$, so $0 < \gamma^*$. Now apply Lemma 3.2 to get (45). This completes the proof of Theorem 3.6. □

6. Appendix 2. This appendix gives MATLAB routines for computing the sign of a matrix using semioptimal scaling. For the numerical problems tested in this paper, we used the values tol=1.0d-10, ns=100, and pm=2.

```
function [S,numiter,reldiff]=matsign(A,n,tol,ns,pm)
% This function computes the sign (S) of an nxn matrix A using
% Newton's method with semioptimal scaling.  The accuracy of the
% solution is controlled by the parameters:
%   tol = relative error tolerance,
%   ns  = maximum number of Newton steps, and
%   pm  = the number of iterations in the power method.
% On return, numiter = number of Newton steps taken and
% reldiff is an estimate of the relative error in S.
  S=A; reldiff=tol+1; imethod=0;
  for numiter=1:ns,
% Invert S and calculate the semioptimal scaling factor.
    W=inv(S);
    if(imethod==0), factor=scale(S,W,n,pm); S=S*factor; W=W/factor; end
% Check relative error.
    reldiff=norm(S-W,'fro')/(1+norm(S,'fro'));
% Check switch to unscaled Newton's method (imethod=1).
    if(reldiff<0.01), imethod=1; end
    S=(S+W)/2;
    if(reldiff<=tol), break, end
  end

function [factor]=scale(S,W,n,pm)
% This function finds the semioptimal scaling factor.
% Get estimates of the dominant eigenvalues of S and W = inv(S).
  v=rand(n,1)+rand(n,1)*sqrt(-1);
  for i=1:pm, [d1,v]=power(S,v); end
  v=rand(n,1)+rand(n,1)*sqrt(-1);
  for i=1:pm, [d2,v]=power(W,v); end
% Calculate the scaling factor.
  d2=1/d2; r1=abs(d1); r2=abs(d2); r=r1;
  c1=real(d1)/r1; c2=real(d2)/r2;
  z=(r1/r2+r2/r1)/2; g=max(c2/c1,c1/c2);
  if(c2<c1), r=r2; end
  if(z<=g),
    factor=1/r;
    else
    factor=sqrt((r1*c1-r2*c2)/((r1*c2-r2*c1)*(r1*r2)));
  end

function [e,v]=power(A,v)
% This function uses a variant of the power method to estimate
% the dominant eigenvalue of A.
  v0=A*(A*v); v1=A*(A*v0); v2=A*(A*v1);
  b=[v0'*v0,v0'*v1;v1'*v0,v1'*v1];
```

```
c=-[v0'*v2;v1'*v2]; c=pinv(b)*c; v=v2/norm(v2);
e=sqrt((-c(2)+sqrt(c(2)^2-4*c(1)))/2);
f=sqrt((-c(2)-sqrt(c(2)^2-4*c(1)))/2);
if(abs(f)>abs(e)), e=f; end
```

REFERENCES

[1] L. A. BALZER, *Accelerated convergence of the matrix sign function method of solving Lyapunov, Riccati, and other matrix equations*, Internat. J. Control, 32 (1980), pp. 1057–1078.

[2] A. Y. BARRAUD, *Investigations autour de la fonction signe d'une matrice—application à l'Équation de Riccati*, R.A.I.R.O. Automat. Syst. Anal. Control, 13 (1979), pp. 335–368.

[3] ———, *The numerical solution of* $A^TQ + QA = -C$, IEEE Trans. Automat. Control, AC-24 (1979), pp. 671–672.

[4] R. BYERS, *Solving the algebraic Riccati equation with the matrix sign function*, Linear Algebra Appl., 85 (1987), pp. 267–279.

[5] A. K. CLINE, C. B. MOLER, G. W. STEWART, AND J. H. WILKINSON, *An estimate for the condition number of a matrix*, SIAM J. Numer. Anal., 16 (1979), pp. 368–375.

[6] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, 1983. (Second Edition, 1989.)

[7] N. J. HIGHAM, *Computing the polar decomposition—with applications*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1160–1174.

[8] C. KENNEY AND A. J. LAUB, *Polar decomposition and matrix sign function condition estimates*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 488–504.

[9] ———, *Rational iterative methods for the matrix sign function*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 273–291.

[10] Z. KOVARIK, *Some iterative methods for improving orthonormality*, SIAM J. Numer. Anal., 7 (1970), pp. 386–389.

[11] P. PANDEY, C. KENNEY, AND A. J. LAUB, *A parallel algorithm for the matrix sign function*, Internat. J. High Speed Comput., 2 (1990), pp. 181–191.

[12] A. SIDI, *On extensions of the power method for normal operators*, Linear Algebra Appl., 120 (1989), pp. 207–224.

[13] A. SIDI AND J. BRIDGER, *Convergence and stability analyses for some vector extrapolation methods in the presence of defective iteration matrices*, J. Comput. Appl. Math., 22 (1988), pp. 35–61.

[14] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, Oxford, U.K., 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 707–728, July 1992

003

A REVIEW ON THE INVERSE OF SYMMETRIC TRIDIAGONAL AND BLOCK TRIDIAGONAL MATRICES*

GÉRARD MEURANT†

Dedicated to Gene H. Golub on the occasion of his 60th birthday

Abstract. In this paper some results are reviewed concerning the characterization of inverses of symmetric tridiagonal and block tridiagonal matrices as well as results concerning the decay of the elements of the inverses. These results are obtained by relating the elements of inverses to elements of the Cholesky decompositions of these matrices. This gives explicit formulas for the elements of the inverse and gives rise to stable algorithms to compute them. These expressions also lead to bounds for the decay of the elements of the inverse for problems arising from discretization schemes.

Key words. block tridiagonal matrices, decay of elements

AMS(MOS) subject classifications. 15A09, 65F50

1. Introduction.

When solving elliptic or parabolic partial differential equations (pde's) with finite difference methods, we have to consider tridiagonal (for one-dimensional (1D) problems) or block tridiagonal (for higher dimensions) matrices. For developing and studying preconditioners for iterative methods like the conjugate gradient method, it is often of interest to know the properties of the inverse, for instance, how the elements of the inverse decay along a row or a column; see [13], [14], [17].

Inverses of tridiagonal matrices have been extensively studied in the past, although it seems that most of the results that have been obtained were unrelated and that many of the authors did not know each others' results. To mention just a few, let us cite [1], [2], [5], [6], [18], [19], [24], [26], [29], and [34], where formulas are given for inverses of tridiagonal matrices and [3], [9], [10], [24], [29], [31], and [32], where extensions to block tridiagonal or banded matrices are provided.

Closed form explicit formulas for elements of the inverses can only be given for special matrices, e.g., Toeplitz tridiagonal matrices [19] corresponding, for instance, to constant coefficients 1D partial differential elliptic equations, or for block matrices arising from separable 2D elliptic pde's [3]. We recall that a Toeplitz matrix is a matrix with constant diagonals.

Basically there are two kinds of papers: the first gives analytic formulas for special cases; the second gives characterizations of matrices whose inverse has certain properties, e.g., being tridiagonal or banded.

Historically, the oldest paper we found considering the explicit inverse of matrices is that of Moskovitz [29] from 1944, in which analytic expressions are given for 1D and 2D Poisson model problems. A very important paper for the inverses of band matrices is the seminal 1959 work by Asplund [1] in which conditions under which the inverse of a matrix is banded were given. In the 1960 paper by Bickley and McNamee [10], formulas were given for the 2D problem and separable equations. In 1969, Fischer and Usmani [19] gave a general analytical formula for symmetric Toeplitz tridiagonal matrices, i.e., for 1D model problems. In 1971, Baranger and Duc–Jacquet [5] considered symmetric factorizable matrices (whose elements are $a_i b_j$ for $i \leq j$) and

*Received by the editors March 5, 1990; accepted for publication December 20, 1990.

†Département de Mathématiques Appliquées, Centre d'Etudes de Limeil-Valenton, 94195 Villeneuve St Georges Cedex, France (meurant@romanee.inria.fr).

proved that the inverse is tridiagonal (this is Asplund's result) and conversely. In 1973, Bukhberger and Emel'yanenko [11] gave formulas based on Cholesky factorization for the inverse of a symmetric tridiagonal matrix. Two 1977 papers, by Bank and Rose [3] and Bank [4], gave analytic formulas for the inverse of block tridiagonal matrices arising from separable problems. The 1979 Ikebe paper [24] studied inverses of Hessenberg matrices; specialization of this result to tridiagonal matrices gave Asplund's results. This result is extended to block tridiagonal matrices when the outer blocks are nonsingular. In 1979, Barrett [6] introduced the "triangle property" (a matrix R has this property if $R_{ij} = (R_{ik}R_{kj})/R_{kk}$); a matrix having the "triangle property" and nonzero diagonal elements has a tridiagonal inverse and vice versa). This result is not strictly equivalent to Asplund's result. In one theorem there is a restriction on the diagonal elements and in the other there is a restriction on the nondiagonal elements of the inverse. Also in 1979, Yamamoto and Ikebe [34] obtained formulas for the inverses of banded matrices. Fadeev [18] gave another proof of Ikebe's result for Hessenberg matrices in 1981. Another important paper is that by Barrett and Feinsilver [7]. It established a correspondence between the vanishing of a certain set of minors of a matrix and the vanishing of a related set of minors of the inverse. This gave a characterization of inverses of banded matrices; for tridiagonal matrices this reduces to the "triangle property." In 1984 Barrett and Johnson [8] generalized the work of Barrett and Feinsilver. Also in 1984, Rizvi [32] generalized the "triangle property" to block matrices and gave expressions for inverses of block tridiagonal matrices. The 1987 paper by Rózsa [31] generalized Asplund's work. In 1986, Romani [30] studied the additive structure of the inverses of banded matrices, namely, that the inverse of a $2k+1$ diagonal symmetric banded matrix can be expressed as a sum of k symmetric matrices belonging to the class of inverses of symmetric irreducible tridiagonal matrices. In 1988, Bevilacqua, Codenotti, and Romani [9] gave formulas for block Hessenberg and block tridiagonal matrices with nonsingular outer blocks.

Regarding the decay of the elements of inverses the most interesting papers are those by Demko [15], in which results are proved for particular banded matrices, and by Demko, Moss, and Smith [16], which presents results for positive definite banded matrices. In 1987, Greengard [22] studied the decrease of Green's functions which is equivalent to studying the inverse of the 2D and 3D Poisson problems. Eijkhout and Polman [17] in 1988 exhibited bounds for the inverses of M-matrices, the Cholesky factors of which are bounded by diagonally dominant Toeplitz matrices. These matrices arise in the design of block preconditioners (cf. [13]). Also in 1988, Kuznetsov [25] gave results on the decay of the elements of the inverse for symmetric positive definite matrices that are used in a domain decomposition method (cf. Meurant [28]).

When no explicit solutions for the elements of the inverse can be found, they are usually given in terms of solutions of second-order linear recurrences [5], [9], [14]. However, as it was shown in Concus and Meurant [14] for tridiagonal and pentadiagonal matrices, these recurrences can be numerically unstable and can lead to trouble for large problems. In this paper we obtain most of the previously known results as well as new ones using a unified framework. Simple relationships between elements of the inverse and Cholesky or block Cholesky decompositions are obtained. This allows us to obtain analytic formulas and to compute elements of the inverse in a very stable way, at least when the matrix is symmetric and positive definite. We also provide estimates of decays of the elements of the inverse. It is clear that most of our results can be easily extended to nonsymmetric matrices with straightforward modifications.

The outline of the paper is as follows: in §2, we study the tridiagonal case corresponding to one-dimensional pde problems, in particular, simple but precise formulas

for the decay of the elements of the inverse are given. Section 3 is devoted to block tridiagonal matrices. We solve the general problem and as a consequence we easily get formulas for separable two-dimensional elliptic problems. In §4, results about the decay of the inverse are recalled and it is shown how to obtain estimates for two-dimensional pde problems.

Throughout the paper, it is supposed that the matrices under consideration are nonsingular and that their Cholesky decompositions exist. So, the principal minors of the matrices are also nonsingular.

2. Tridiagonal matrices. We are interested in finding formulas for the inverse of a symmetric tridiagonal matrix T of order n,

$$T = \begin{pmatrix} a_1 & -b_2 & & & \\ -b_2 & a_2 & -b_3 & & \\ & \ddots & \ddots & \ddots & \\ & & -b_{n-1} & a_{n-1} & -b_n \\ & & & -b_n & a_n \end{pmatrix}.$$

As a particularly interesting case for pde's, the example of a tridiagonal Toeplitz matrix will be considered:

$$T_a = \begin{pmatrix} a & -1 & & & \\ -1 & a & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & a & -1 \\ & & & -1 & a \end{pmatrix}.$$

It should be noted that this latter case has been previously studied by several authors; see, for instance, [19] and [29]. A good reference for numerical methods for solving Toeplitz linear systems is [12].

2.1. The general case. It is natural to suppose that $b_i \neq 0$, for all $i \geq 2$ (that is, T is irreducible) as if one of the b_i's is 0, then the problem can be reduced to two smaller subproblems (for a discussion of this issue, see [6]). Here, the $-$ sign is just a technical convenience and has no specific significance, unless otherwise stated. From [1], [5], and [18] it is known that there exist two sequences $\{u_i\}, \{v_i\}, i = 1, n$ such that

$$T^{-1} = \begin{pmatrix} u_1v_1 & u_1v_2 & u_1v_3 & \dots & u_1v_n \\ u_1v_2 & u_2v_2 & u_2v_3 & \dots & u_2v_n \\ u_1v_3 & u_2v_3 & u_3v_3 & \dots & u_3v_n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ u_1v_n & u_2v_n & u_3v_n & \dots & u_nv_n \end{pmatrix}.$$

This result can also be easily proved with the techniques used in §3. Moreover, every nonsingular matrix of the previous form (the matrices of this class have been called "matrices factorisables" in [5]) is the inverse of an irreducible tridiagonal matrix. It means that to know all the elements of T^{-1}, it is enough to compute its first and last columns. In fact, it is enough to know $2n-1$ quantities as u_1 can be chosen arbitrarily (note that $2n-1$ is the number of nonzero terms determining T). The second-order recurrences for computing u_i and v_i given in [13] can be unstable and can lead to trouble for large systems, but this problem was already solved in [14]. However, much simpler formulas can be obtained if $\{u_i\}, \{v_i\}$ are computed in the following way: let

us first compute v. The $\{u_i\}, \{v_i\}$ are only defined up to a multiplicative constant. So, for instance, u_1 can be chosen as $u_1 = 1$. Then let $v = (v_1, \cdots, v_n)^T$; as $u_1 = 1$ the first column of T^{-1} is v, so

$$Tv = e_1$$

where $e_1 = (1, 0, \cdots, 0)^T$.

Because of the special structure of the right-hand side, it is natural to consider a UL decomposition of T:

$$T = UD_U^{-1}U^T,$$

with

$$U = \begin{pmatrix} d_1 & -b_2 & & & \\ & d_2 & -b_3 & & \\ & & \ddots & \ddots & \\ & & & d_{n-1} & -b_n \\ & & & & d_n \end{pmatrix}, \qquad D_U = \begin{pmatrix} d_1 & & & & \\ & d_2 & & & \\ & & \ddots & & \\ & & & d_{n-1} & \\ & & & & d_n \end{pmatrix}.$$

By inspection, it is easily seen that

$$d_n = a_n, \quad d_i = a_i - \frac{b_{i+1}^2}{d_{i+1}}, \quad i = n-1, \cdots, 1.$$

With the help of the UL decomposition the linear system for v can be easily solved.

PROPOSITION 2.1.

$$v_1 = \frac{1}{d_1}, \quad v_i = \frac{b_2 \cdots b_i}{d_1 \cdots d_{i-1} d_i}, \quad i = 2, \cdots, n.$$

Proof. It is clear that solving $Tv = e_1$ is equivalent to solving

$$D_U^{-1}U^T v = \frac{1}{d_1}\, e_1,$$

and the proposition follows. □

Let $u = (u_1, \cdots, u_n)^T$; the last column of T^{-1} is $v_n u$ and therefore

$$v_n T u = e_n,$$

where $e_n = (0, \cdots, 0, 1)^T$. To solve this system, because of the structure of the right-hand side, it is easier to use an LU decomposition of T:

$$T = LD_L^{-1}L^T,$$

with

$$L = \begin{pmatrix} \delta_1 & & & & \\ -b_2 & \delta_2 & & & \\ & \ddots & \ddots & & \\ & & -b_{n-1} & \delta_{n-1} & \\ & & & -b_n & \delta_n \end{pmatrix}, \qquad D_L = \begin{pmatrix} \delta_1 & & & & \\ & \delta_2 & & & \\ & & \ddots & & \\ & & & \delta_{n-1} & \\ & & & & \delta_n \end{pmatrix}.$$

By inspection,

$$\delta_1 = a_1, \quad \delta_i = a_i - \frac{b_i^2}{\delta_{i-1}}, \quad i = 2, \cdots, n.$$

PROPOSITION 2.2.

$$u_n = \frac{1}{\delta_n v_n}, \quad u_{n-i} = \frac{b_{n-i+1} \cdots b_n}{\delta_{n-i} \cdots \delta_n v_n}, \quad i = 1, \cdots, n-1.$$

Proof. Clearly,

$$D_L L^T u = \frac{1}{\delta_n v_n} e_n.$$

Solving for u gives the result. □

Note that

$$u_1 = \frac{b_2 \cdots b_n}{\delta_1 \cdots \delta_n v_n} = \frac{d_1 \cdots d_n}{\delta_1 \cdots \delta_n},$$

but $d_1 \cdots d_n = \delta_1 \cdots \delta_n = \det T$, so $u_1 = 1$, as the values of $\{v_i\}$ were computed with this scaling.

Together, the preceding results prove the following theorem.

THEOREM 2.3. *For the general case,*

$$(T^{-1})_{i,j} = u_i v_j = b_{i+1} \cdots b_j \frac{d_{j+1} \cdots d_n}{\delta_i \cdots \delta_n} \quad \forall i, \quad \forall j > i,$$

$$(T^{-1})_{i,i} = u_i v_i = \frac{d_{i+1} \cdots d_n}{\delta_i \cdots \delta_n} \quad \forall i.$$

In these products, terms that have indices greater than n must be taken equal to 1.

This gives a computationally stable and simple algorithm for computing elements of the inverse of T as it involves only Cholesky decompositions that are proved to be stable when the matrix T possesses enough properties as to be diagonally dominant.

We are also interested in characterizing the decrease of the elements of T^{-1} along a row or column starting from the diagonal element. In [13], it is proved that if T is strictly diagonally dominant, then the sequence $\{u_i\}$ is strictly increasing and the sequence $\{v_i\}$ is strictly decreasing. From Theorem 2.3, we have

$$\frac{(T^{-1})_{i,j}}{(T^{-1})_{i,j+1}} = \frac{d_{j+1}}{b_{j+1}},$$

and, therefore,

$$(T^{-1})_{i,j} = \frac{d_{j+1} \cdots d_{j+l}}{b_{j+1} \cdots b_{j+l}} T^{-1}_{i,j+l}.$$

By induction, the following result is proved.

THEOREM 2.4. *If T is strictly diagonally dominant ($a_i > b_i + b_{i+1}$, for all i) then the sequence d_i is such that*

$$d_i > b_i.$$

Hence, the sequence $T^{-1}_{i,j}$ is a strictly decreasing function of j, for $j > i$. Similarly, we have $\delta_i > b_{i+1}$.

Proof. We have

$$d_n = a_n > b_n.$$

Suppose $d_{i+1} > b_{i+1}$; then

$$d_i = a_i - \frac{b_{i+1}^2}{d_{i+1}} > a_i - b_{i+1} > b_i. \qquad \square$$

Remark that the theorem can be proved under a weaker hypothesis. Namely, we can only suppose that T is diagonally dominant ($a_i \geq b_i + b_{i+1}$) and $a_n > b_n$, $a_1 > b_2$. This result was already proven in [13], although not in the same way.

2.2. The Toeplitz case. Here, as an example, the Toeplitz tridiagonal matrix T_a that was defined in the introduction of §2 is considered. The interesting thing is that we are then able to analytically solve the recurrences arising in the Cholesky decompositions. This is given in the following lemma.

LEMMA 2.5. *Let*

$$\alpha_1 = a, \quad \alpha_i = a - \frac{1}{\alpha_{i-1}}, \quad i = 2, \cdots, n.$$

Then, if $a \neq 2$,

$$\alpha_i = \frac{r_+^{i+1} - r_-^{i+1}}{r_+^i - r_-^i},$$

where

$$r_\pm = \frac{a \pm \sqrt{a^2 - 4}}{2}$$

are the two solutions of the quadratic equation $r^2 - ar + 1 = 0$. *If* $a = 2$, *then* $\alpha_i = (i+1)/i$.

Proof. We set

$$\alpha_i = \frac{\beta_i}{\beta_{i-1}}.$$

Therefore, we now have a recurrence on β_i:

$$\beta_i - a\beta_{i-1} + \beta_{i-2} = 0, \quad \beta_0 = 1, \quad \beta_1 = a.$$

The solution of this linear second-order difference equation is well known (see, for instance, [23]):

$$\beta_i = c_0 r_+^{i+1} + c_1 r_-^{i+1}.$$

From the initial conditions we have $c_0 + c_1 = 0$. Hence, the solution can be written as

$$\beta_i = c_0 (r_+^{i+1} - r_-^{i+1});$$

when $a = 2$, it is easy to see that $\beta_i = i + 1$, and the result follows. $\square$

This proof explains the difficulties that arise when using the second-order recurrences for u_i and v_i. As $r_+ > 1$, $\beta_i \to \infty$ when $i \to \infty$. On the contrary, α_i remains bounded.

Remark. α_i can be written in the other form,

$$\alpha_i = \frac{\sinh((i+1)\psi)}{\sinh(i\psi)}, \quad \text{where } \cosh(\psi) = \frac{a}{2} \quad \text{if } a > 2,$$

$$\alpha_i = \frac{\sin((i+1)\psi)}{\sin(i\psi)}, \quad \text{where } \cos(\psi) = \frac{a}{2} \quad \text{if } a < 2.$$

From this lemma, the solutions of the recurrences involved in the Cholesky decompositions of T_a can be deduced. When $a \neq 2$, we have

$$d_{n-i+1} = \frac{r_+^{i+1} - r_-^{i+1}}{r_+^i - r_-^i}.$$

Solving for v the following result is obtained.

PROPOSITION 2.6. *For the sequence v_i in T_a^{-1},*

$$v_i = \frac{r_+^{n-i+1} - r_-^{n-i+1}}{r_+^{n+1} - r_-^{n+1}} \quad \forall i.$$

Note in particular, that

$$v_n = \frac{r_+ - r_-}{r_+^{n+1} - r_-^{n+1}}.$$

It is obvious that for the Toeplitz case, we have the relation

$$\delta_i = d_{n-i+1}.$$

Solving for u, the following result is obtained.

PROPOSITION 2.7. *For the sequence u_i in T_a^{-1},*

$$u_i = \frac{r_+^i - r_-^i}{r_+ - r_-}, \qquad i = 1, \cdots, n.$$

With these two last results, the expression of the elements of the inverse can be computed.

THEOREM 2.8. *For $j \geq i$ and when $a \neq 2$,*

$$({T_a}^{-1})_{i,j} = u_i v_j = \frac{(r_+^i - r_-^i)(r_+^{n-j+1} - r_-^{n-j+1})}{(r_+ - r_-)(r_+^{n+1} - r_-^{n+1})},$$

where $r_\pm$ are the two solutions of the quadratic equation $r^2 - ar + 1 = 0$. This can also be written (for $a > 2$) as

$$({T_a}^{-1})_{i,j} = \frac{\sinh(i\psi)\ \sinh((n-j+1)\psi)}{\sinh(\psi)\ \sinh((n+1)\psi)}, \quad \textit{with } \cosh(\psi) = \frac{a}{2};$$

for $a = 2$, we have

$$(T_a)_{i,j}^{-1} = i\ \frac{n-j+1}{n+1}.$$

These formulas are similar to the ones in [19], where they were obtained with a different method.

Regarding the decay of the elements of T_a^{-1}, in this simple case we can obtain useful bounds. Suppose that $a > 2$. Then we have

$$\frac{u_i v_j}{u_i v_{j+1}} = \frac{r_+^{n-j+1} - r_-^{n-j+1}}{r_+^{n-j} - r_-^{n-j}} = \frac{r_+^{n-j+1}}{r_+^{n-j}} \left(\frac{1 - r^{n-j+1}}{1 - r^{n-j}} \right) > r_+ > 1$$

and

$$u_i v_j < r_+^{i-j-1} \frac{(1-r^i)(1-r^{n-j+1})}{(1-r)(1-r^{n+1})}, \qquad j \geq i+1,$$

where $r = (r_-/r_+) < 1$.

From this, the following result can be deduced.

THEOREM 2.9. *If $a > 2$, we have the bound*

$$(T_a^{-1})_{i,j} < (r_-)^{j-i} (T_a^{-1})_{i,i} \quad \forall\, i, \quad \forall j \geq i,$$

$$(T_a^{-1})_{i,j} < \frac{r_-^{j-i+1}}{1-r} \quad \forall\, i, \quad \forall j \geq i+1.$$

This implies that the following estimate holds: let $\epsilon_1 > 0$ and $\epsilon_2 > 0$ be given:

$$\frac{(T_a^{-1})_{i,j}}{(T_a^{-1})_{i,i}} \leq \epsilon_1 \quad \text{if } j - i \geq \frac{\log \epsilon_1^{-1}}{\log r_+},$$

$$(T_a^{-1})_{i,j} \leq \epsilon_2 \quad \text{if } j - i + 1 \geq \frac{\log [\epsilon_2 (1-r)]^{-1}}{\log r_+}.$$

3. Block tridiagonal matrices. In this section we consider the symmetric block tridiagonal matrix

$$A = \begin{pmatrix} D_1 & -A_2^T & & & \\ -A_2 & D_2 & -A_3^T & & \\ & \ddots & \ddots & \ddots & \\ & & -A_{n-1} & D_{n-1} & -A_n^T \\ & & & -A_n & D_n \end{pmatrix}.$$

Each block is of order n, although this is not essential for our results.

In the two-dimensional partial differential applications we have in mind, the matrices D_i will be tridiagonal and the matrices A_i will be diagonal, but this does not influence the method and the results that will be described in this section.

As an interesting example, the following problem will be considered:

$$A_T = \begin{pmatrix} T & -I & & & \\ -I & T & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & T & -I \\ & & & -I & T \end{pmatrix},$$

T being a Toeplitz tridiagonal matrix. This example arises, for instance, from the discretization of the Poisson equation in a square.

3.1. The general case. To obtain the formulas for the inverse, three different block factorizations will be used: LU, UL, and a twisted factorization. Let us first give formulas for the block LU and UL factorizations. Denote by L the block lower part of A. Then,

$$A = (\Delta + L)\, \Delta^{-1}\, (\Delta + L^T) = (\Sigma + L^T)\, \Sigma^{-1}\, (\Sigma + L),$$

where Δ and Σ are block diagonal matrices whose diagonal blocks are denoted by Δ_i and Σ_i and are given by block recurrences

$$\begin{cases} \Delta_1 = D_1, \\ \Delta_i = D_i - A_i\ (\Delta_{i-1})^{-1}\ (A_i)^T, \end{cases} \qquad \begin{cases} \Sigma_n = D_n, \\ \Sigma_i = D_i - (A_{i+1})^T\ (\Sigma_{i+1})^{-1}\ A_{i+1}. \end{cases}$$

A twisted factorization can be defined for each $j = 2, \cdots, n-1$ as

$$A = (\Phi + \mathcal{L})\Phi^{-1}(\Phi + \mathcal{L}^T)$$

where Φ is a block diagonal matrix and $\mathcal{L}$ has the following twisted block structure:

$$\mathcal{L} = \begin{pmatrix} 0 & & & & & & \\ -A_2 & 0 & & & & & \\ & \ddots & \ddots & & & & \\ & & -A_j & 0 & -A_{j+1}^T & & \\ & & & & \ddots & \ddots & \\ & & & & & 0 & -A_n^T \\ & & & & & & 0 \end{pmatrix},$$

where the row with two nonzero terms is the jth block row. By inspection, we have

$$\begin{aligned} \Phi_i &= \Delta_i, \qquad i = 1, \cdots, j-1, \\ \Phi_i &= \Sigma_i, \qquad i = n, \cdots, j+1, \\ \Phi_j &= D_j - A_j \Delta_{j-1}^{-1} A_j^T - A_{j+1}^T \Sigma_{j+1}^{-1} A_{j+1}. \end{aligned}$$

It should be noted that when we know the LU and UL decompositions, we know the twisted factorizations for all j's. So, these twisted factorizations are only a convenience to obtain simpler formulas.

With the twisted factorization at hand, the block jth column X of the inverse can be computed in a straightforward way.

THEOREM 3.1. *The jth block column X of A^{-1} is given by*

$$\begin{aligned} X_j &= \Phi_j^{-1}, \\ X_{j-l} &= \Delta_{j-l}^{-1}\ A_{j-l+1}^T\ \Delta_{j-l+1}^{-1} \cdots \Delta_{j-1}^{-1}\ A_j^T\ \Phi_j^{-1}, \qquad l = 1, \cdots, j-1, \\ X_{j+l} &= \Sigma_{j+l}^{-1}\ A_{j+l}\ \Sigma_{j+l-1}^{-1} \cdots \Sigma_{j+1}^{-1}\ A_{j+1}\ \Phi_j^{-1}, \qquad l = 1, \cdots, n-j. \end{aligned}$$

These expressions are valid for any block tridiagonal matrix that satisfies our hypothesis. When matrices A_i are nonsingular, A is said to be *proper* (cf. [9]); in this case, the formulas can be simplified. Using the uniqueness of the inverse, we can prove the following.

PROPOSITION 3.2. *If A is proper, then*

$$\begin{aligned} \Phi_j^{-1} &= A_{j+1}^{-1}\ \Sigma_{j+1} \cdots A_n^{-1}\ \Sigma_n\ \Delta_n^{-1}\ A_n \cdots \Delta_{j+1}^{-1}\ A_{j+1}\ \Delta_j^{-1} \\ &= A_j^{-T}\ \Delta_{j-1} \cdots A_2^{-T}\ \Delta_1\ \Sigma_1^{-1}\ A_2^T \cdots \Sigma_{j-1}^{-1}\ A_j^T\ \Sigma_j^{-1}. \end{aligned}$$

From these relations, we deduce alternate formulas for the other elements of the inverse.

THEOREM 3.3. *If A is proper,*

$$X_{j-l} = (A_{j-l}^{-T}\Delta_{j-l-1}\cdots A_2^{-T}\Delta_1)(\Sigma_1^{-1}A_2^T\cdots A_j^T\Sigma_j^{-1}), \qquad l = 1,\cdots,j-1,$$
$$X_{j+l} = (A_{j+l+1}^{-1}\Sigma_{j+l+1}\cdots A_n^{-1}\Sigma_n)(\Delta_n^{-1}A_n\cdots\Delta_{j+1}^{-1}A_{j+1}\Delta_j^{-1}), \qquad l = 1,\cdots,n-j.$$

As before, the elements of the inverse can be computed in a stable way using block Cholesky decomposition when the matrix is diagonally dominant or positive definite. These formulas are the block counterpart of the ones for tridiagonal matrices in Theorem 2.3. They give a characterization of the inverse of a proper block tridiagonal matrix.

THEOREM 3.4. *If A is proper, there exist two (nonunique) sequences of matrices $\{U_i\}, \{V_i\}$ such that for $j \geq i$*

$$(A^{-1})_{ij} = U_i V_j^T,$$

with $U_i = A_i^{-T}\Delta_{i-1}\cdots A_2^{-T}\Delta_1$ and $V_j^T = \Sigma_1^{-1}A_2^T\cdots A_j^T\Sigma_j^{-1}$.

In other words, A^{-1} can be written as

$$A^{-1} = \begin{pmatrix} U_1V_1^T & U_1V_2^T & U_1V_3^T & \cdots & U_1V_n^T \\ V_2U_1^T & U_2V_2^T & U_2V_3^T & \cdots & U_2V_n^T \\ V_3U_1^T & V_3U_2^T & U_3V_3^T & \cdots & U_3V_n^T \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ V_nU_1^T & V_nU_2^T & V_nU_3^T & \cdots & U_nV_n^T \end{pmatrix}.$$

This result was proven using different methods in [9].

If we denote by E_n the matrix

$$E_n = \begin{pmatrix} 1 & 1 & \cdots & 1 \\ 0 & 1 & \cdots & 1 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{pmatrix},$$

and $L_n = E_n^T - I$, $U = (U_1,\cdots,U_n)^T$, $V^T = (V_1^T,\cdots,V_n^T)$, we can write the result of Theorem 3.4 as

$$A^{-1} = UV^T \circ E_n + VU^T \circ L_n,$$

where $\circ$ denotes the Hadamard (element by element) product. If we denote by u_i the columns of U and by v_i those of V, this can also be written as

$$A^{-1} = \sum_{j=1}^{n} \left(u_iv_i^T \circ E_n + v_iu_i^T \circ L_n\right).$$

This result about the additive structure of the inverse was proved for banded matrices in [30].

3.2. Separable problems. Here we consider the matrix A_T defined at the beginning of this section. This is not the most general problem that can be considered, but for this case explicit formulas can be given. Because the matrix is persymmetric, we have

$$\Delta_i = \Sigma_{n-i+1}, \qquad i = 1,\cdots,n,$$

and

$$X_{j-l} = (\Delta_{j-l+1} \cdots \Delta_1)(\Delta_n^{-1} \cdots \Delta_{n+j+1}^{-1}), \qquad l = 1, \cdots, j-1.$$

With the same methods as in [27], it can be proved that all the Δ_i's have the same eigenvectors as T and hence they commute. In this case, the block recursion for the diagonal elements in the Cholesky decomposition can be solved:

$$\Delta_1 = T,$$
$$\Delta_i = T - (\Delta_{i-1})^{-1}.$$

Let $\Lambda = (\lambda^j)$ be the diagonal matrix of the eigenvalues of T and Λ_i the corresponding one for Δ_i. Then the following propositions hold.

PROPOSITION 3.5. *The following relation holds*:

$$\Lambda_1 = \Lambda,$$
$$\Lambda_i = \Lambda - (\Lambda_{i-1})^{-1},$$

which can be written elementwise as

$$\lambda_1^j = \lambda^j,$$
$$\lambda_i^j = \lambda^j - \frac{1}{\lambda_{i-1}^j}.$$

From this last result and Lemma 2.5, we can compute the values of λ_i^j.

PROPOSITION 3.6.

$$\lambda_i^j = \frac{(r(j)_+)^{i+1} - (r(j)_-)^{i+1}}{(r(j)_+)^i - (r(j)_-)^i},$$

where $r(j)_\pm$ *are the roots of* $r^2 + \lambda^j r - 1 = 0$. *If* $\lambda^j > 2$ (*which is the case for the Poisson problem*), *this can be written*

$$\lambda_i^j = \frac{\sinh((i+1)\psi_j)}{\sinh(i\psi_j)}, \qquad \cosh(\psi_j) = \frac{\lambda^j}{2}.$$

Now let Λ_+ and Λ_- be the diagonal matrices whose diagonal elements are $r(j)_+$ and $r(j)_-$. From Proposition 3.6, we have

$$\Lambda_i = (\Lambda_+^{i+1} - \Lambda_-^{i+1})(\Lambda_+^i - \Lambda_-^i)^{-1}.$$

Let Q be the matrix of the eigenvectors of all the matrices. Denote

$$T_+ = Q\Lambda_+ Q^T, \qquad T_- = Q\Lambda_- Q^T;$$

then

$$\Delta_i = (T_+{}^{i+1} - T_-{}^{i+1})(T_+^i - T_-^i)^{-1}.$$

Along the same lines as what was done for a tridiagonal matrix, we have the following theorem.

THEOREM 3.7. *The (block) elements of the inverse are given by*

$$(A_T^{-1})_{i,j} = ({T_+}^{i+1} - {T_-}^{i+1})({T_+}^{n-j+1} - {T_-}^{n-j+1})({T_+}^{n+1} - {T_-}^{n+1})^{-1}(T_+ - T_-)^{-1}.$$

From these results, it can easily be seen that

$$(A_T^{-1})_{i,j} = S_n^{-1}(T)\ S_{i-1}(T)\ S_{n-j}(T) \quad \text{for } j \geq i,$$

where $S_n(x)$ is the shifted Chebyshev polynomial of the second kind, that is, defined for $x > 2$ as

$$S_i(x) = \frac{\sinh((i+1)\psi)}{\sinh(\psi)} \quad \text{with } \cosh(\psi) = x/2.$$

This expression for the inverse was given in [3]. Now we establish relations that will be useful in the next section. The (simple) roots of the Chebyshev polynomial S_n are $\mu_l = 2\cos(l\pi/(n+1)), \quad l = 1, \cdots, n$. Therefore,

$$S_n(x) = \prod_{l=1}^{n}(x - \mu_l).$$

As in [20], remark that for $j \geq i$, $S_n^{-1}(x)\ S_{i-1}(x)\ S_{n-j}(x)$ is a rational function in x, so it can be developed in elementary fractions. We write

$$S_n^{-1}(x)\ S_{i-1}(x)\ S_{n-j}(x) = \sum_{l=1}^{n} \frac{\alpha_{ij}^l}{x - \mu_l}.$$

It can easily be seen that

$$\alpha_{ij}^l = \frac{S_{i-1}(\mu_l)\ S_{n-j}(\mu_l)}{S_n'(\mu_l)}.$$

From this expansion, we get an expression for the elements of the inverse in terms of the zeros of Chebyshev polynomials.

THEOREM 3.8. *The (block) elements of the inverse of A_T are given by:*

$$(A_T^{-1})_{i,j} = \sum_{l=1}^{n} \alpha_{ij}^l\ (T - \mu_l I)^{-1}, \qquad j \geq i.$$

These results can be extended to more general separable problems, although in these cases the roots of the involved polynomials are not explicitly known.

4. Decay of the inverse for two-dimensional problems. In this section, the decay of the elements for two-dimensional pde problems is examined. We recall some known results and establish new ones. First, the Poisson equation is considered that is easy to handle, as the inverse is explicitly known. Then we will turn to the problem of finding bounds for general tridiagonal problems using results from convergence of iterative methods. Finally, the possibility to numerically compute approximate decays for certain block tridiagonal matrices is considered.

4.1. The Poisson equation. Because the Poisson equation is an isotropic problem, it is enough to look at the decay of the elements in one direction of the underlying mesh, i.e., we can only look at the diagonal blocks of the inverse. This is because a

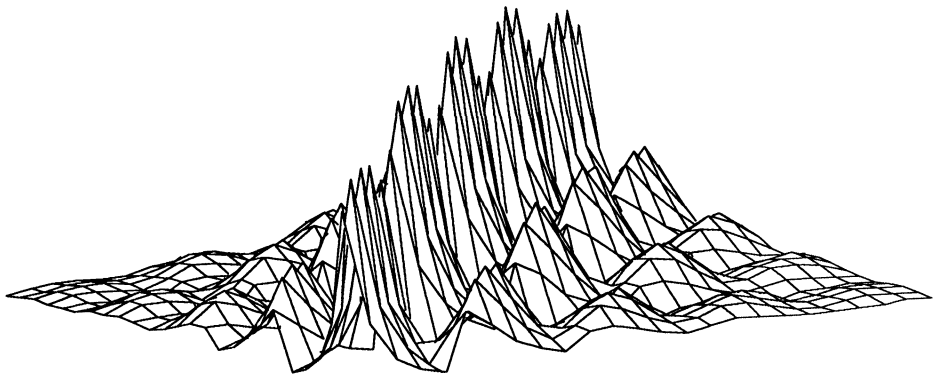


FIG. 1. *Exact inverse of the Poisson problem matrix.*

column (or a row) of the inverse is obtained by putting a Dirac delta function (a "function" being 1 in one point of the mesh and 0 elsewhere) as the right-hand side. Because of the isotropic property of the diffusion equation (as the limit of a time-dependent problem), the right-hand side diffuses the same in both directions. A picture of the inverse of the Poisson problem matrix for a 5×5 mesh is given in Fig. 1.

If we look more closely at what happens for a row of the matrix, starting from the diagonal, we obtain what is shown in Fig. 2. This picture has been obtained for row number 61 in a 121×121 matrix corresponding to a 11×11 mesh.

From the previous section, it is known that

$$(A_T^{-1})_{i,i} = \sum_{l=1}^{n} \alpha_{ii}^{l} \, (T - \mu_l I)^{-1},$$

where

$$\alpha_{ii}^{l} = \frac{S_{i-1}(\mu_l) \; S_{n-i}(\mu_l)}{S_n'(\mu_l)}, \qquad \mu_l = 2\cos\left(\frac{l\pi}{n+1}\right) < 2,$$

and

$$T = \begin{pmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{pmatrix}.$$

Therefore, $T - \mu_l I$ is a Toeplitz matrix with a diagonal element greater than 2. From §2, we know that the elements of the inverses of all these matrices strictly decay away from the diagonal along a row.

THEOREM 4.1. *Let B be the ith diagonal block of the inverse of A_T and $r_+[l]$ the positive root of $r^2 - (4 - \mu_l)r - 1 = 0$; then*

$$\frac{B_{pp}}{B_{pq}} \geq \min_l \{(r_+[l])^{q-p}\}, \quad q > p.$$

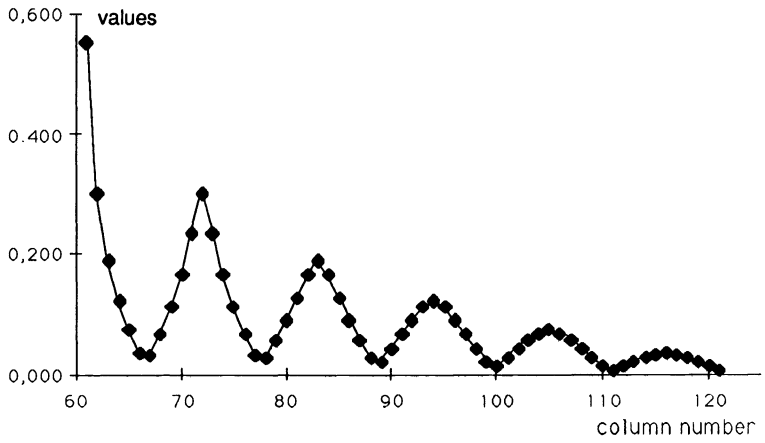


FIG. 2. *A row of the inverse of the Poisson problem matrix.*

Proof. Let $T_l = (T - \mu_l)^{-1}$,

$$B_{pp} = \sum_{l=1}^{n} \alpha_{ii}^{l} (T_l)_{pp} > \sum_{l=1}^{n} \alpha_{ii}^{l} (r_+[l])^{|q-p|} (T_l)_{pq},$$

hence

$$\frac{B_{pp}}{B_{pq}} \geq \frac{\sum_{l=1}^{n} \alpha_{ii}^{l} (r_+[l])^{|q-p|} (T_l)_{pq}}{\sum_{l=1}^{n} \alpha_{ii}^{l} (T_l)_{pq}} \geq \min_l \{(r_+[l])^{|q-p|}\}. \qquad \square$$

This gives a uniform (related to i) estimate of the decay.
Asymptotically, we obtain

$$\frac{B_{pq}}{B_{pp}} \leq C(h),$$

where

$$C(h) = 1 - (q-p)\pi h + O(h^2).$$

However, the bound of Theorem 4.1 is a little pessimistic, as shown by the following numerical example. Consider the linear system from the Poisson equation for $n = 11$. Figure 3 shows the relative decrease of the elements for a row of a diagonal block and the bound given by the previous formulas. We see that the slope is correct, but the values are pessimistic.

4.2. The general block tridiagonal case. Here we consider finding bounds for the decay of the elements of the inverse of a general symmetric positive definite tridiagonal matrix corresponding to the discretization of an elliptic or parabolic problem with a five-point finite difference scheme. The matrix of the problem is

$$A = \begin{pmatrix} D_1 & -A_2^T & & & \\ -A_2 & D_2 & -A_3^T & & \\ & \ddots & \ddots & \ddots & \\ & & -A_{n-1} & D_{n-1} & -A_n^T \\ & & & -A_n & D_n \end{pmatrix}.$$

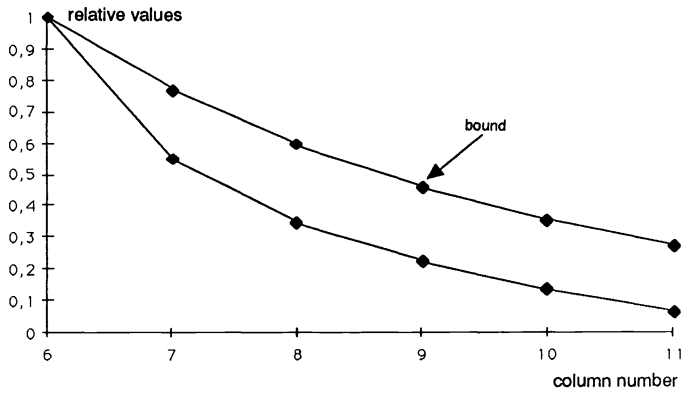


FIG. 3. *Bound for the relative decay for the Poisson problem matrix.*

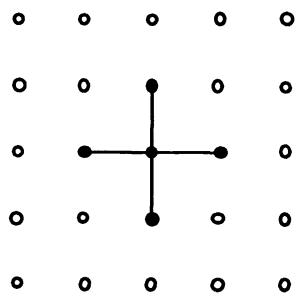


FIG. 4. *Five-point finite difference scheme.*

In this example, matrices D_i are tridiagonal and matrices A_i are diagonal corresponding to the scheme displayed in Fig. 4.

To obtain bounds on the decay of the elements of the inverse, we will follow the same lines as [15]; see also [16]. Consider solving the linear system

$$Ax = b$$

with a Chebyshev first-order iterative method: let x^0 be given and

$$x^{k+1} = x^k + \alpha_k(b - Ax^k).$$

This method converges when A is symmetric positive definite and the coefficients α_k are chosen as the reciprocals of the roots of Chebyshev polynomials.

If $e^k = x - x^k$ is the error we have the following bounds (cf., for instance, [21]).

PROPOSITION 4.2. *Let $\kappa = \lambda_{\max}/\lambda_{\min}$ be the condition number of A; then*

$$\|e^k\|_\infty = \max_i |e_i^k| \leq \|e^k\|_2 \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k \|e^0\|_2.$$

In practice this method is not used because it is unstable; to be stabilized, special orderings of the coefficients must be used. However, we will only use it to obtain bounds on the solution. For this purpose, the previous results can be used to get bounds on the infinity norm. For the Chebyshev method,

$$e^k = P_k(A)e^0,$$

where P_k is a kth-order polynomial. From this, it follows that

$$\|e^k\|_\infty \leq \|P_k(A)\|_\infty \|e^0\|_\infty \leq \sqrt{N}\|P_k(A)\|_2 \|e^0\|_\infty,$$

where $N = n^2$.

Therefore, we have also the following proposition.

PROPOSITION 4.3.

$$\|e^k\|_\infty \leq 2n\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k \|e^0\|_\infty.$$

To compute the jth column x (or the jth row as the matrix is symmetric) of the inverse, a system with $b = e_j$, where $e_j = (0, \cdots, 0, 1, 0, \cdots, 0)^T$, must be solved, the nonzero element being in position j. Now, consider the Chebyshev iterative method with $x^0 = 0$, so $e^0 = x$.

As $x^0 = 0$, the vector $x^1 = \alpha_1 e_j$ has the same sparsity pattern as e_j. The idea is to consider the sparsity patterns of the successive iterates x^k. To do this, it is easier to think in terms of the underlying two-dimensional mesh. Let the mesh points be indexed by two integers (p, q), and $N(p, q)$ denote the set of neighbouring mesh points in the five-point stencil centered on (p, q). Then, the following proposition holds.

PROPOSITION 4.4. *Let $\mathcal{S}(x^k)$ be the set of mesh points corresponding to the sparsity pattern of x^k. If $\mathcal{S}(e_j) = \mathcal{S}(x^1) = (p_1, q_1)$, then*

$$\mathcal{S}(x^k) = \mathcal{S}(x^{k-1}) \bigcup_{(p_l,q_l)\in\mathcal{S}(x^{k-1})} N(p_l, q_l).$$

Proof. It is clear that the sparsity pattern of x^k is deduced from the sparsity pattern of x^{k-1} by a multiplication with A. The vector x^{k-1} can be written as $x^{k-1} = \sum_l \beta_l e_l$, where the index runs across the sparsity pattern of x_{k-1}. So, the sparsity pattern we are looking for is the union of the sparsity patterns of Ae_l for all l in the sparsity pattern of x^{k-1}. But the vector Ae_l is the lth column of A; therefore, there are at most only five nonzero terms corresponding to the mesh point related to the lth component and its four neighbours in the five-point stencil. □

Remark. This result is not restricted to the five-point stencil and can be easily extended, for instance, to sparse matrices arising from finite element methods.

The result from Proposition 4.4 is illustrated in Fig. 5, the black points corresponding to the nonzero components in x^k.

Let S_j^k be the sets of indices $S(x^k)$ generated from $b = e_j$. Regarding the decay of the elements of the inverse, the following general result holds.

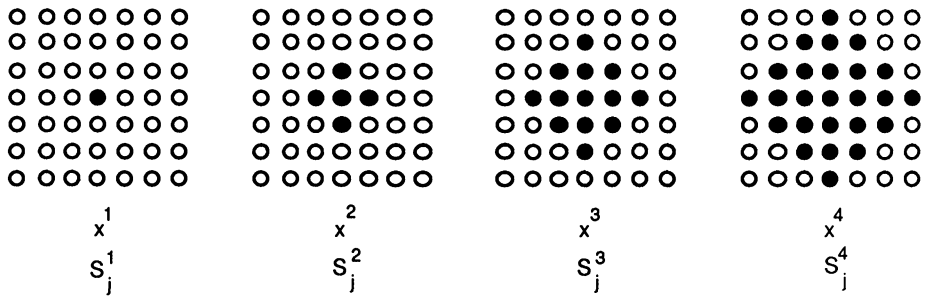


FIG. 5. *Sparsity patterns of* x^1, x^2, x^3, *and* x^4.

THEOREM 4.5.

$$|A_{ij}^{-1}| \leq 2n\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k \max_j |A_{ij}^{-1}| \quad \forall i \notin S_j^k.$$

Proof. We have $e^k = x - x^k$; when $i \notin S_j^k$, the corresponding components of x^k are zero. Hence $|A_{ij}^{-1}| \leq \|e^k\|_\infty$. □

The condition $i \notin S_j^k$ is verified, for instance, for mesh points (p_i, q_i) satisfying

$$|(p_i, q_i) - (p_j, q_j)| \geq kh,$$

where h is the mesh size. The last theorem shows that the elements of the inverse decay as shown in Fig. 1.

Now we specialize to problems arising from finite difference approximations. We suppose that A is a diagonally dominant M-matrix. Then we have the following result.

PROPOSITION 4.6. *When A is a diagonally dominant M-matrix,*

$$(A^{-1})_{ii} \geq \frac{1}{A_{ii}} \quad \forall i,$$
$$\max_j (A^{-1})_{ij} = (A^{-1})_{ii} \quad \forall i.$$

Proof. Denote $C = A^{-1}$. Looking at the AC product, we obtain

$$1 = \sum_{k=1}^{n} a_{ik}c_{ki} = a_{ii}c_{ii} - \sum_{k \neq i} |a_{ik}|c_{ki},$$

because the a_{ik}, $k \neq i$ are nonpositive and the c_{ki} are positive. Therefore,

$$a_{ii}c_{ii} \geq 1,$$

which gives the first result. Now, suppose there exists a $j \neq i$ such that $\max_k c_{ik} = c_{ij} > c_{ii}$; then

$$0 = \sum_{k=1}^{n} a_{jk}c_{ki} = a_{jj}c_{ij} - \sum_{k \neq j} |a_{jk}|c_{ki}.$$

By hypothesis, $0 \leq c_{ki} = c_{ik} < c_{ij}$, so

$$0 > c_{ij} \left(a_{jj} - \sum_{k \neq j} |a_{jk}| \right).$$

But as $c_{ij} \geq 0$ and A is diagonally dominant, this is a contradiction. Therefore the maximum of the elements of the inverse occurs on the diagonal. □

For the Poisson problem we considered in the last section, κ is explicitly known,

$$\kappa \simeq \frac{1}{\sin(\frac{\pi h}{2})^2}.$$

From Theorem 4.5, we know that

$$\frac{c_{ij}}{c_{ii}} \leq C(h), \quad \text{where } C(h) = \frac{2}{h}(1 - k\pi h).$$

This is a factor of $2/h$ off from the formula we obtained in §4.1, which can be quite large. However, this general formula must account for the possible worst case for the decay.

For a general diagonally dominant M-matrix, we have the following theorem.

THEOREM 4.7. *When A is a diagonally dominant M-matrix,*

$$\frac{(A^{-1})_{ij}}{(A^{-1})_{jj}} \leq 2n \left(\frac{\sqrt{\kappa} - 1}{\sqrt{\kappa} + 1} \right)^k \qquad \forall i \notin S_j^k.$$

Still, this bound is not very satisfactory when it is compared, for instance, with the bound obtained for the Poisson problem. In order to obtain more insights into particular problems, we will specialize to generalized strictly diagonally dominant matrices.

DEFINITION 4.8. A is generalized strictly diagonally dominant (GSDD) if there exist a vector $s = (s_i) > 0$ such that

$$|a_{ii}| s_i > \sum_{j \neq i}^{n} |a_{ij}| s_j.$$

This also means that there exists a diagonal matrix S such that $S^{-1}AS$ is strictly diagonally dominant; this is also true for AS.

In particular, strictly diagonally dominant and irreducibly diagonally dominant (cf. [33]) M-matrices and H-matrices are GSDD.

Now, to obtain bounds, simply consider the Jacobi iteration. Let $A = D + L + L^T$, D being the diagonal part and L the strictly lower triangular part of A,

$$x^k = -D^{-1}(L + L^T)x^{k-1} + D^{-1}e_j.$$

Let $J(A) = -D^{-1}(L + L^T)$ be the iteration matrix. Then, we have

$$e^k = J(A)^k e^0.$$

Using these definitions, the following result is obtained.

PROPOSITION 4.9. *If A is* GSDD, *we have*

$$s_i^{-1}|e_i^k| \leq \|J(S^{-1}AS)^k\|_\infty \max_l [s_l^{-1}(A^{-1})_{lj}] \quad \forall j.$$

We also have the following result.

THEOREM 4.10. *If A is* GSDD,

$$(A^{-1})_{ij} \leq s_i \|J(S^{-1}AS)^k\|_\infty \max_l [s_l^{-1}(A^{-1})_{lj}] \quad \forall i \notin S_j^k.$$

What follows is the problem of estimating $\|J(S^{-1}AS)^k\|_\infty$, which is strictly less than 1 as the Jacobi iteration is convergent for a GSDD matrix. Let $B = S^{-1}AS$ and ε, a given (small) positive real number; then we have the following proposition.

PROPOSITION 4.11. *For any matrix B and any $\varepsilon > 0$,*

$$\|B^k\|_\infty^{1/k} \leq \rho(B) + \varepsilon,$$

where $\rho(B)$ is the spectral radius of B.

Proof. Let $B(\varepsilon) = \frac{1}{\rho(B)+\varepsilon} B$; then

$$\rho(B(\varepsilon)) < 1,$$

and $\lim_{k\to\infty} \|B^k(\varepsilon)\|_\infty = 0$. So, for k large enough, $\|B^k(\varepsilon)\|_\infty < 1$. As

$$\|B^k(\varepsilon)\|_\infty = \frac{\|B^k\|_\infty}{(\rho(B)+\varepsilon)^k},$$

the result is proved. □

Now note that $\rho(J(S^{-1}AS)) = \rho(J(A))$. Therefore, the following theorem holds.

THEOREM 4.12. *If A is* GSDD, *we have*

$$(A^{-1})_{ij} \leq s_i [\rho(J(A)) + \varepsilon]^k \max_l [s_l^{-1}(A^{-1})_{lj}] \quad \forall i \notin S_j^k.$$

The matrix S or the vector s can be chosen in many different ways. In fact, if A is an M-matrix and $y > 0$ is any given positive vector, we can get s by solving

$$As = y.$$

It is clear that by choosing y appropriately s can be made, for instance, close to a vector e made of 1s: there exists $\epsilon > 0$, such that $y = Ae + \epsilon > 0$; then $s = e + \epsilon A^{-1}e = e + \epsilon_A$.

THEOREM 4.13. *If A is a diagonally dominant M-matrix, there exists ε such that $\rho(J(A)) + \varepsilon < 1$ and*

$$\frac{(A^{-1})_{ij}}{(A^{-1})_{jj}} \leq \frac{s_i}{1+\epsilon_A} [\rho(J(A)) + \varepsilon]^k \quad \forall i \notin S_j^k.$$

Proof. Use the previous result and Theorem 4.12. □

Remark. If A is strictly diagonally dominant, the factor $(s_i/1+\epsilon_A)$ can be replaced by 1.

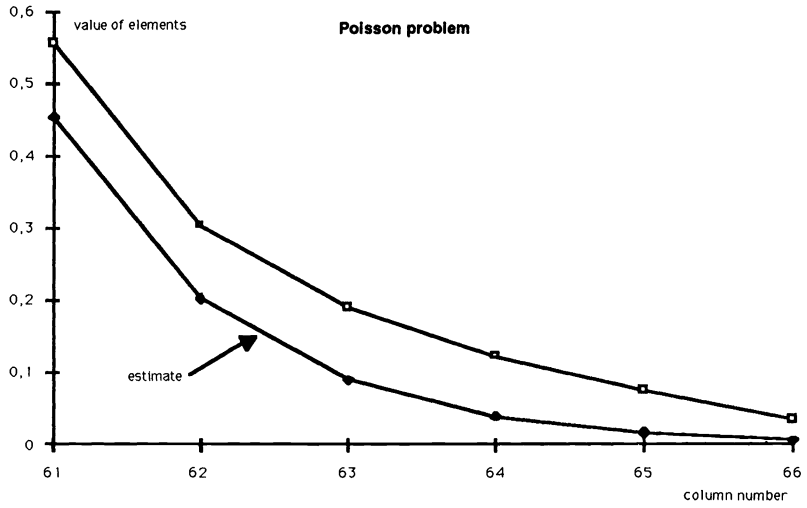


FIG. 6. *Comparison of the actual decay and the approximation.*

If we specialize to the Poisson problem, $\rho(J(A)) = \cos(\pi h) \simeq 1-(\pi^2 h^2/2)$. Hence we get a better estimate than the one given in Theorem 4.5.

4.3. Approximation of the decay for general problems. For general block tridiagonal problems, the precise value of the condition number of the matrix is usually not known. The only information we have for some problems is that $\kappa = O(h^{-2})$. So, it is of interest to be able to compute a numerical approximation of the decay of the elements of the inverse. We can do this in the following way which mimics the INV preconditioner defined in [13]. Instead of computing the LU and UL decompositions as in §3.1, we are going to compute block incomplete factorizations.

Let trid(B) be a tridiagonal matrix with the nonzero elements that are the same as the corresponding ones in B. Then we define two incomplete block factorizations:

$$(\Delta + L)\ \Delta^{-1}\ (\Delta + L^T) \quad \text{and} \quad (\Sigma + L^T)\ \Sigma^{-1}\ (\Sigma + L),$$

where Δ and Σ are block diagonal matrices whose diagonal blocks are tridiagonal and denoted by Δ_i and Σ_i. They are given by the following formulas:

$$\begin{cases} \Delta_1 = D_1, \\ \Delta_i = D_i - A_i\ \text{trid}[\Delta_{i-1}^{-1}]\ (A_i)^T, \end{cases} \quad \begin{cases} \Sigma_n = D_n, \\ \Sigma_i = D_i - (A_{i+1})^T\ \text{trid}[\Sigma_{i+1}^{-1}]\ A_{i+1}. \end{cases}$$

We then approximate the diagonal blocks of the inverse by the inverse of the tridiagonal matrix

$$D_j - A_j\ \text{trid}[\Delta_{j-1}^{-1}]\ A_j^T - A_{j+1}^T\ \text{trid}[\Sigma_{j+1}^{-1}]\ A_{j+1}.$$

When this tridiagonal matrix is computed and factored, we can obtain numerical information on the decay of the elements using the method of §2 and 3. This gives information on the decay along one direction of the two-dimensional mesh. To have

information in the other direction (if the problem at hand is not isotropic), we can compute the other block elements with the formulas developed in §3.

Let us now compare the bounds obtained in this way with the actual decay of the elements on the Poisson problem. Figure 6 shows the actual decay of the elements in row 61 for a matrix on an 11×11 mesh and the decay of the estimate obtained in the previous way. It is seen that although the values are not very good, the behaviour of the curve is quite the same. However, this is a very simple example and this conclusion has to be checked on more general ones.

5. Conclusions. In this paper, we have exhibited useful relationships between the elements of inverses of tridiagonal and block tridiagonal matrices and elements of the Cholesky decompositions of these matrices. In particular, we got very simple expressions for the elements of the inverse of a block tridiagonal matrix. This allows us to develop stable algorithms for computing elements of the inverse when the matrix has more properties, like being diagonally dominant. The characterization of the inverse allows us also to obtain bounds for the decay of the elements of the inverse.

REFERENCES

[1] E. ASPLUND, *Inverse of matrices* $\{a_{ij}\}$ *which satisfy* $a_{ij} = 0$ *for* $j > i + p$, Math. Scand., 7 (1959), pp. 57–60.

[2] S. O. ASPLUND, *Finite boundary value problems solved by Gren's matrix*, Math. Scand., 7 (1959), pp. 49–56.

[3] R. E. BANK AND D. J. ROSE, *Marching algorithms for elliptic boundary value problems.* I: *The constant coefficient case*, SIAM J. Numer. Anal., 14 (1977), pp. 792–829.

[4] R. E. BANK, *Marching algorithms for elliptic boundary value problems.* II: *The variable coefficient case*, SIAM J. Numer. Anal., 14 (1977), pp. 950–970.

[5] J. BARANGER AND M. DUC-JACQUET, *Matrices tridiagonales symétriques et matrices factorisables*, RIRO, R-3 (1971), pp. 61–66.

[6] W. W. BARRETT, *A theorem on inverses of tridiagonal matrices*, Linear Algebra Appl., 27 (1979), pp. 211–217.

[7] W. W. BARRETT AND P. J. FEINSILVER, *Inverses of banded matrices*, Linear Algebra Appl., 41 (1981), pp. 111–130.

[8] W. W. BARRETT AND C. R. JOHNSON, *Determinantal formulas for matrices with sparse inverses*, Linear Algebra Appl., 56 (1984), pp. 73–88.

[9] R. BEVILACQUA, B. CODENOTTI, AND F. ROMANI, *Parallel solution of block tridiagonal linear systems*, Linear Algebra Appl., 104 (1988), pp. 39–57.

[10] W. G. BICKLEY AND J. MCNAMEE, *Matrix and other direct methods for the solution of systems of linear difference equations*, Philos. Trans. Roy. Soc. London Ser. A, 252 (1960), pp. 69–131.

[11] B. BUKHBERGER AND G. A. EMEL'YANENKO, *Methods of inverting tridiagonal matrices*, USSR Comput. Math. and Math. Phys., 13 (1973), pp. 10–20.

[12] J. BUNCH, *Stability of methods for solving Toeplitz systems of equations*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 349–364.

[13] P. CONCUS, G. H. GOLUB, AND G. MEURANT, *Block preconditioning for the conjugate gradient method*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 220–252.

[14] P. CONCUS AND G. MEURANT, *On computing* INV *block preconditionings for the conjugate gradient method*, BIT, 26 (1986), pp. 493–504.

[15] S. DEMKO, *Inverses of band matrices and local convergence of spline projections*, SIAM J. Numer. Anal., 14 (1977), pp. 616–619.

[16] S. DEMKO, W. F. MOSS, AND P. W. SMITH, *Decay rates for inverses of band matrices*, Math. Comp., 43 (1984), pp. 491–499.

[17] V. EIJKHOUT AND B. POLMAN, *Decay rates of banded M-matrices that are near Toeplitz matrices*, Linear Algebra Appl., 109 (1988), pp. 247–277.

[18] D. K. FADEEV, *Properties of the inverse of a Hessenberg matrix*, in Numerical Methods and Computational Issues, Volume 5, V. P. Ilin and V. N. Kublanovskaya, ed., 1981. (In Russian.)

[19] C. F. FISCHER AND R. A. USMANI, *Properties of some tridiagonal matrices and their application to boundary value problems*, SIAM J. Numer. Anal., 6 (1969), pp. 127–141.

[20] E. GALLOPOULOS AND Y. SAAD, *Some fast elliptic solvers on parallel architectures and their complexities*, Internat. J. High Speed Comput., 1 (1989), pp. 113–142.

[21] G. H. GOLUB AND G. MEURANT, *Résolution numérique des grands systèmes linéaires*, Eyrolles, Paris, 1983.

[22] L. GREENGARD, *The rapid evaluation of potential fields in particle systems*, MIT Press, Cambridge, MA, 1987.

[23] P. HENRICI, *Elements of Numerical Analysis*, John Wiley, New York, 1964.

[24] Y. IKEBE, *On inverses of Hessenberg matrices*, Linear Algebra Appl., 24 (1979), pp. 93–97.

[25] Y. KUZNETSOV, *New algorithms for approximate realization of implicit difference schemes*, Soviet. J. Numer. Anal. Math. Modelling, 3 (1988), pp. 99–114.

[26] R. MATTHEIJ AND M. SMOOKE, *Estimates for the inverse of tridiagonal matrices arising in boundary-value problems*, Linear Algebra Appl., 73 (1986), pp. 33–57.

[27] G. MEURANT, *The Fourier/tridiagonal method for the Poisson equation from the point of view of block Cholesky factorization*, Report LBID–764, Lawrence Berkeley Laboratory, Berkeley, CA, 1983.

[28] ———, *A domain decomposition method for parabolic problems*, Appl. Numer. Math., 8 (1991), pp. 427–441.

[29] D. MOSKOVITZ, *The numerical solution of Laplace's and Poisson's equations*, Quart. Appl. Math., 2 (1944), pp. 148–163.

[30] F. ROMANI, *On the additive structure of the inverses of banded matrices*, Linear Algebra Appl., 80 (1986), pp. 131–140.

[31] P. RÓZSA, *On the inverse of band matrices*, in Integral Equations and Operator Theory, Volume 10, Birkhäuser, Boston, 1987, pp. 82–95.

[32] S. A. H. RIZVI, *Inverses of quasi-tridiagonal matrices*, Linear Algebra Appl., 56 (1984), pp. 177–184.

[33] R. S. VARGA, *Matrix Iterative Analysis*, Prentice–Hall, Englewood Cliffs, NJ, 1962.

[34] T. YAMOMOTO AND Y. IKEBE, *Inversion of band matrices*, Linear Algebra Appl., 24 (1979), pp. 105–111.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 729–745, July 1992

004

A UNITARILY CONSTRAINED TOTAL LEAST SQUARES PROBLEM IN SIGNAL PROCESSING*

K. S. ARUN†

Abstract. The problem addressed here is the determination of the total least squares solution, subject to a unitary constraint, of an overdetermined, inconsistent, linear system of equations. The problem arises in many signal processing applications, three of which are briefly presented and studied here. The solution to the constrained total least squares problem is seen to be the same as the solution to the orthogonal Procrustes problem.

Key words. total least squares, orthogonal Procrustes problem, sinusoid retrieval, Toeplitz approximation, motion parameter estimation, rotation matrices, narrow band direction finding, unitary constraint

AMS(MOS) subject classifications. 15A18, 65F20

1. Introduction.

In many signal processing applications, one encounters the problem of estimating a unitary matrix $\mathbf{X}$ in the model $\mathbf{AX} = \mathbf{B}$, from noisy measurements of matrices $\mathbf{A}$ and $\mathbf{B}$. Entries in matrices $\mathbf{A}$, $\mathbf{B}$, and $\mathbf{X}$ may be complex-valued. All three matrices have the same number of columns, $\mathbf{X}$ is square, but $\mathbf{A}$ and $\mathbf{B}$ have more rows than columns, so that the system of equations is overdetermined. To account for noise in the measurements, least squares estimates with and without the unitary constraint and total least squares estimates have been suggested. Here we address the problem of finding the total least squares estimate of $\mathbf{X}$ subject to the constraint that $\mathbf{X}$ be unitary.

In the rest of this section, three signal processing applications are presented where the aforementioned problem is encountered. Section 2 describes the various solutions that have been suggested and presents the constrained total least squares solution to the problem. Surprisingly, the constrained total least squares solution is the same as the constrained least squares solution. In § 3, the results of some numerical experiments on computer-synthesized data with and without the unitary constraint are reported and discussed.

1.1. Motion parameter estimation.

The problem of estimating the translation and rotation of a rigid body between two time instants is a problem faced in many computer vision applications [1], [2]. Noisy measurements $\underline{p}_i$ and $\underline{p}'_i$, $i = 1, 2, \cdots, N$, of the three-dimensional locations of some feature points on the rigid body are made at the two time instants, and it is desired to fit the motion model

$$\underline{p}'_i = \mathbf{R}\underline{p}_i + \underline{T},$$

where $\mathbf{R}$ is a 3×3 rotation matrix, and $\underline{T}$ is a 3×1 translation vector. The rotation hypothesised is around an axis passing through the origin.

A rotation about the origin is a norm-preserving unitary operation, and so the usual approach is to find a 3×1 vector $\underline{T}$ and a unitary 3×3 matrix $\mathbf{R}$ that minimize the sum of squares

$$\sum_{i=1}^{N} \| \underline{p}'_i - \underline{T} - \mathbf{R}\underline{p}_i \|^2.$$

* Received by the editors August 1, 1988; accepted for publication (in revised form) November 28, 1990. This work was supported by Strategic Defense Initiative Office/Innovative Science and Technology grant DAAL03-91-G-0118, managed by the US Army Research Office.

† Coordinated Science Laboratory and Department of Electrical and Computer Engineering, University of Illinois, Urbana, Illinois 61801 (arun@uicsl.csl.uiuc.edu).

The vector-norm used throughout this paper is the usual Euclidean norm. It was shown [3] that if

$$\underline{m} \triangleq \frac{1}{N} \sum_{i=1}^{N} \underline{p}_i$$

and

$$\underline{m}' \triangleq \frac{1}{N} \sum_{i=1}^{N} \underline{p}'_i$$

are the respective centroids of the two sets of three-dimensional points, then the constrained least squares estimate $\hat{\mathbf{R}}_{\text{cls}}$ of the rotation matrix is the solution to the decoupled problem of minimizing

$$\sum_{i=1}^{N} \| (\underline{p}'_i - \underline{m}') - \mathbf{R}(\underline{p}_i - \underline{m}) \|^2 \tag{1}$$

subject to the constraint that $\mathbf{R}$ be unitary, and that the constrained least squares estimate of the translation vector $\underline{T}$ is $\underline{m}' - \hat{\mathbf{R}}_{\text{cls}}\underline{m}$. Thus the problem reduces to the orthogonal Procrustes problem [4] whose solution is listed in § 2.

There is no reason to believe that measurement errors are confined to measurements $\underline{p}'$ from the second frame and that the first frame measurements are noise-free, which is the tacit assumption in the least squares problem formulation. A more natural formulation of the problem that allows for errors in both frames is the total least squares formulation [5] with a unitary constraint on $\mathbf{R}$ as follows. Find a 3×1 vector $\underline{T}$ and a unitary 3×3 matrix $\mathbf{R}$ that minimize

$$\sum_{i=1}^{N} \| \underline{\delta}_i \|^2 + \| \underline{\delta}'_i \|^2$$

subject to the constraint that

$$\underline{p}'_i + \underline{\delta}'_i = \mathbf{R}(\underline{p}_i + \underline{\delta}_i) + \underline{T} \quad \text{for all } i.$$

To decouple the problem of finding $\underline{T}$ from the problem of finding $\mathbf{R}$, note that for a given $\mathbf{R}$ and $\underline{T}$, the errors $\underline{\delta}_i$ and $\underline{\delta}'_i$ are solutions of the underdetermined linear system

$$(-\mathbf{I} \mid \mathbf{R}) \begin{pmatrix} \underline{\delta}_i \\ \underline{\delta}'_i \end{pmatrix} = \underline{p}'_i - \mathbf{R}\underline{p}_i - \underline{T}.$$

The minimum-norm solution to the above system is

$$\underline{\delta}_i = -\tfrac{1}{2}\{(\underline{p}'_i - \underline{T}) - \mathbf{R}\underline{p}_i\}; \qquad \underline{\delta}'_i = \tfrac{1}{2}\{\mathbf{R}^H(\underline{p}'_i - \underline{T}) - \underline{p}_i\},$$

where superscript H denotes Hermitian transposition throughout the paper. Therefore, the cost as a function of $\mathbf{R}$ and $\underline{T}$ is

$$\sum_{i=1}^{N} \tfrac{1}{4} \| (\underline{p}'_i - \underline{T}) - \mathbf{R}\underline{p}_i \|^2 + \tfrac{1}{4} \| \mathbf{R}^H(\underline{p}'_i - \underline{T}) - \underline{p}_i \|^2.$$

Define $\underline{q}'_i \triangleq \underline{p}'_i - \underline{m}'$ and $\underline{q}_i \triangleq \underline{p}_i - \underline{m}$ for every i, and the minimization problem once again gets decoupled. The estimate $\hat{\mathbf{R}}_{\text{ctls}}$ is the solution to the following constrained total

least squares problem: Find a *unitary* 3×3 matrix $\mathbf{R}$ that minimizes

$$\sum_{i=1}^{N} \|\underline{\delta}_i\|^2 + \|\underline{\delta}'_i\|^2 \tag{2}$$

subject to the constraint that

$$\underline{q}'_i + \underline{\delta}'_i = \mathbf{R}(\underline{q}_i + \underline{\delta}_i) \quad \text{for all } i.$$

The estimate $\hat{\underline{T}}_{\text{ctls}}$ is once again given by $\underline{m}' - \hat{\mathbf{R}}_{\text{ctls}}\underline{m}$.

1.2. Sinusoid retrieval. The problem of retrieving multiple sinusoids (with closely spaced frequencies) from estimated covariances is of special interest in a vast range of signal-processing applications [6]–[8]. Very often the covariance sequence may have to be estimated from time-series data, however, it is not uncommon to encounter applications in which the covariance information is directly available. Such situations arise in astronomical star bearing estimation, interference spectroscopy, and some sensor array applications. In interference spectroscopy, the problem is the detection of spectral lines in the intensity distribution $P(\sigma)$ of an electromagnetic source [9], [10]. In the classical two-beam interferometer, the radiation is broken into two beams, and a variable path difference δ is introduced in one of the two paths. The two beams are then recombined and the resultant total intensity (integrated over all wavenumbers σ) is measured. This total intensity varies with the path-difference δ as

$$I(\delta) = \int_{\sigma=0}^{\infty} P(\sigma)\,d\sigma + \int_{\sigma=0}^{\infty} P(\sigma)\cos(2\pi\delta\sigma)\,d\sigma.$$

The variable part of the above function is the so-called interference function, which is obviously also the Fourier transform of the unknown intensity distribution $P(\sigma)$. While $P(\sigma)$ plays the role of the power spectrum, the observed samples of the interference function play the role of the covariance lags, and the problem is one of retrieving sinusoids from the covariances.

All sinusoidal signals: $y(k) = \sum_i c_i \exp(j\omega_i k)$, $|c_i| > 0$, ω_i distinct, can be generated by the following zero-input, state-space model:

$$\underline{x}(k+1) = \mathbf{F}\underline{x}(k), \qquad y(k) = \underline{h}\underline{x}(k),$$

where the model order (the size of the state $\underline{x}$) is equal to the number of complex sinusoids. The eigenvalues of $\mathbf{F}$ are of unit magnitude and equal $\exp(j\omega_i)$, where ω_i are the sinusoid frequencies. Information about the complex amplitudes c_i is contained in $\underline{x}(0)$ and $\underline{h}$. The absolute value of c_i is the amplitude and the angle of c_i is the initial phase of the ith complex sinusoid.

When the sinusoids have random initial phases that are mutually uncorrelated, then the state $\underline{x}$ and the output y are wide-sense stationary processes. Using these facts, it can be shown that the state covariance matrix $\mathbf{P} \triangleq \mathbf{E}[\underline{x}(k)\underline{x}(k)^H]$ is nonsingular and satisfies $\mathbf{P} = \mathbf{F}\mathbf{P}\mathbf{F}^H$, and that the output covariance $r(m) \triangleq \mathbf{E}[y(k+m)y(k)^H]$ satisfies

$$r(m) = \underline{h}\mathbf{F}^m\mathbf{P}\underline{h}^H \quad \text{for all } m.$$

Here, $\mathbf{E}$ denotes the expectation operator; in engineering terms, it is the averaging operator over an ensemble of realizations. For the above properties to hold, it is necessary that the initial phases of the sinusoids in y be mutually uncorrelated; i.e., that the ensemble averages of their pairwise products be all zero. When the initial phases are correlated (the expectation of some of the pairwise products is nonzero), then the state and output processes will not be wide-sense stationary, hence $\mathbf{E}[\underline{x}(k)\underline{x}(k)^H]$ and

$\mathbf{E}[y(k+m)y(k)^H]$ will vary with index k. In such cases, for each realization (sample sequence in the ensemble) of the harmonic process, we can define covariances by temporal averages and use

$$\lim_{T\to\infty}\frac{1}{2T+1}\sum_{k=-T}^{T}$$

in place of expectation. Then $\mathbf{P}$ and $r(m)$ so obtained will be a function of the realization studied, but for each realization, they will satisfy the properties listed below:

$$\mathbf{P} \text{ will be invertible,} \quad \mathbf{P}=\mathbf{F}\mathbf{P}\mathbf{F}^H, \quad \text{and} \quad r(m)=\underline{h}\mathbf{F}^m\mathbf{P}\underline{h}^H \quad \text{for all } m.$$

Therefore, with $\mathbf{E}$ defined as expectation in the wide-sense stationary case and as temporal average otherwise, we have the following results.

The Toeplitz covariance matrix constructed from the output covariances,

$$\mathbf{R}=\begin{bmatrix} r(0) & r(-1) & r(-2) & \cdot\cdot & r(-n) \\ r(1) & r(0) & r(-1) & \cdot\cdot & r(-n+1) \\ r(2) & r(1) & r(0) & \cdot\cdot & r(-n+2) \\ \cdot & \cdot & \cdot & \cdot\cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot\cdot & \cdot \\ r(n) & r(n-1) & r(n-2) & \cdot\cdot & r(0) \end{bmatrix}$$

admits the following factorization:

$$\mathbf{R}=\Theta\Gamma=\begin{bmatrix} \underline{h} \\ \underline{h}\mathbf{F} \\ \underline{h}\mathbf{F}^2 \\ \cdot \\ \cdot \\ \underline{h}\mathbf{F}^n \end{bmatrix}\begin{bmatrix}\mathbf{P}\underline{h}^H & \mathbf{F}^{-1}\mathbf{P}\underline{h}^H & \mathbf{F}^{-2}\mathbf{P}\underline{h}^H & \cdots & \mathbf{F}^{-n}\mathbf{P}\underline{h}^H\end{bmatrix}.$$

The number of columns in Θ and the number of rows in Γ are equal to the model order, which is also equal to the number of complex sinusoids superimposed in the signal. Note that $\Gamma=\mathbf{P}\Theta^H$ because $\mathbf{P}=\mathbf{F}\mathbf{P}\mathbf{F}^H$. When the frequencies ω_i are distinct and every amplitude is nonzero, linear system theory can be used to establish that the pair $(\mathbf{F}, \underline{h})$ is observable, so that Θ has full rank [11]. Because $\mathbf{P}$ is invertible, Γ also has full rank and consequently the rank of $\mathbf{R}$ is equal to the model order [12]. Observe that the ith row of Θ is $\underline{h}\mathbf{F}^{i-1}$, and the ith column of Γ is $\mathbf{F}^{-i+1}\mathbf{P}\underline{h}^H$, so that $\mathbf{F}$ may be obtained by solving the overdetermined system of equations

$$\Theta_1\mathbf{F}=\Theta_2,$$

where $\Theta_1(\Theta_2)$ is obtained from Θ by deleting the last (first) row. If Γ_1 and Γ_2 are defined in a similar manner by deleting the last and first columns, it can be seen that $\mathbf{F}$ also satisfies

$$\mathbf{F}\Gamma_2=\Gamma_1.$$

To estimate the sinusoid frequencies from covariance data, let the singular value decomposition (SVD) of the Toeplitz matrix $\mathbf{R}$ be $\mathbf{R}=\mathbf{U}\Sigma\mathbf{V}^H$, where Σ contains only the nonzero singular values. Since $\mathbf{R}$ is a Hermitian-symmetric positive semidefinite matrix, $\mathbf{V}$ is equal to $\mathbf{U}$. The dimensions of diagonal matrix Σ will equal the rank of $\mathbf{R}$, which is also the model order, say, p. Different factorizations

$$\Theta=\mathbf{U}\Sigma^{1/2}\mathbf{Q}, \qquad \Gamma=\mathbf{Q}^{-1}\Sigma^{1/2}\mathbf{V}^H$$

of the Toeplitz matrix are possible, one for each choice of nonsingular $\mathbf{Q}$, and each will lead to a different state-space realization (they are related to each other by similarity transformations) of the order-p sinusoidal model as

$$\mathbf{F} = \Theta_1^L \Theta_2.$$

Here, the superscript L denotes *any* left inverse. The state-feedback matrix in these coordinates may also be obtained from Γ as

$$\mathbf{F} = \Gamma_1 \Gamma_2^R,$$

where the superscript R denotes any right inverse. The frequencies of the sinusoids may then be found as the angles of the eigenvalues of the $\mathbf{F}$ matrix.

It was recently observed that the choice of factor Θ as $\mathbf{U}\Sigma^{1/2}$ will make the $\mathbf{F}$-matrix unitary [13]. This fact is easily demonstrated by noting that because $\mathbf{R}$ is Hermitian-symmetric and $\mathbf{V} = \mathbf{U}$, this choice makes $\Gamma = \Theta^H$, so that

$$\mathbf{F}\mathbf{F}^H = \Theta_1^L \Theta_2 (\Gamma_1 \Gamma_2^R)^H = \Theta_1^L \Theta_2 \Theta_2^R \Theta_1 = \mathbf{I}.$$

When there is noise present and the covariances are consequently inexact, the Toeplitz approximation method (TAM) of [14] and [12] exploits the rank property of $\mathbf{R}$ and the structure of Θ and Γ to robustly estimate the sinusoid frequencies from inexact covariances. At first, TAM performs an SVD of $\mathbf{R}$, and retains the p principal components (i.e., the p largest singular values and the corresponding singular vectors). Let the singular vectors and singular values *after* the low-rank approximation be U_1, Σ_1, and $\mathbf{V}_1$. Next, TAM picks $\Theta = \mathbf{U}_1 \Sigma_1^{1/2}$ and looks for an approximate solution to $\Theta_1 \mathbf{F} = \Theta_2$ that minimizes the sum of squares

$$\|\Theta_1 \mathbf{F} - \Theta_2\|^2. \tag{3}$$

The norm used for matrices throughout this paper is the Frobenius norm. The TAM estimate is given by

$$\hat{\mathbf{F}}_{\mathrm{TAM}} = \Theta_1^{\dagger} \Theta_2,$$

where the superscript $\dagger$ denotes the pseudoinverse. The sinusoid frequency estimates are, then, the angles of the eigenvalues of $\hat{\mathbf{F}}_{\mathrm{TAM}}$.

Based on the observation that TAM's choice of factor Θ as $\mathbf{U}_1 \Sigma_1^{1/2}$ makes the $\mathbf{F}$-matrix unitary, it has been suggested that the unitary constraint on $\mathbf{F}$ be introduced in the least squares problem of (3) [13]. To be able to use this unitary constraint to advantage, it is important that the factor Θ be chosen correctly. Recall that $\mathbf{F}$ is unitary (in the noise-free case), when Θ is chosen as $\mathbf{U}\Sigma^{1/2}$. For any other choice, $\mathbf{F}$ has eigenvalues on the unit circle, but is not unitary. When the signal has additive white noise riding on it, $\mathbf{U}$ is unaffected, but all singular values get raised by an additive amount equal to the noise variance. Hence, in the presence of noise, an estimate of the noise-free $\mathbf{U}\Sigma^{1/2}$ matrix is

$$\hat{\Theta} = \mathbf{U}_1 (\Sigma_1 - \rho \mathbf{I})^{1/2},$$

where ρ is the arithmetic average of the small singular values (σ_{p+1} and smaller) and $\mathbf{I}$ is the identity matrix of the appropriate dimension. Estimates of Θ_1 and Θ_2 may be obtained by partitioning $\hat{\Theta}$.

Since, in the presence of noise, Θ_1 and Θ_2 are both perturbed, the following constrained *total* least squares formulation is natural. Find a $p \times p$ *unitary* matrix $\mathbf{F}$ to minimize

$$\|\Delta\|^2 + \|\Delta'\|^2 \tag{4}$$

subject to the constraint that

$$(\Theta_1 + \Delta)\mathbf{F} = (\Theta_2 + \Delta').$$

1.3. Direction finding. The problem of finding the directions of arrival of multiple radiating sources using a sensor array has attracted considerable attention over the last decade. The usual assumption is that either the sources or the sensors are narrow-band, making the problem highly structured. High-resolution algorithms based on eigendecomposition [15], [16] require storage of array calibration data and are sensitive to miscalibration resulting, for example, from perturbations in the sensor element locations or sensor gain and phase. Perturbations of the array from the nominal configuration can be significant when the array is long, especially in a sonar towed-array application. The ESPRIT algorithm introduced recently [17] mitigates these difficulties by assuming an array composed of arbitrarily located doublets of sensors with identical (and known) orientation. The algorithm requires that the two sensors in each doublet be identical to each other in response, and that the displacement between the two sensors (separation and angular orientation) be identical for every doublet. It does not require prior knowledge of the actual locations or responses of the individual doublets, and is therefore more robust to perturbations in the array parameters.

Assume that the sources are narrow-band and in the far field, so that the wavefronts impinging on the array are planar. If the signal received at the ith doublet is denoted (x_i, y_i), then the received data in the noiseless case is

$$\underline{z}(t) = \begin{pmatrix} \underline{x}(t) \\ \underline{y}(t) \end{pmatrix} = \begin{pmatrix} \mathbf{A} \\ \mathbf{A}\Phi \end{pmatrix} \underline{s}(t),$$

where $\underline{s}(t)$ is the vector of p source signals as seen from a reference position, $\mathbf{A}$ is the matrix of array-steering vectors (it depends on array calibration including sensor gain and phase and doublet locations, but it need not be known), and Φ is a diagonal matrix of p phase delays between the doublet sensors, one for each of the p sources.

$$\Phi = \operatorname{diag}\{e^{j(\omega/v)d\cos\theta_i}, i = 1, 2, \cdots, p\} = \operatorname{diag}\{e^{j(2\pi/\lambda)d\cos\theta_i}, i = 1, 2, \cdots, p\},$$

where ω is the center frequency of the temporal band, λ is the wavelength at the center frequency, v is the wave velocity, d is the displacement between sensors in a doublet, and θ_i is the direction of arrival of the ith source signal relative to the doublet axis. Since ω, v, and d are known, the problem of direction finding is really a problem of estimating the Φ matrix.

The ESPRIT algorithm exploits the signal structure, especially the fact that the covariance matrix $\mathbf{R}_z$ of $\underline{z}$ ideally has rank equal to the number of sources p (assuming that the sources are not coherent), and that its principal eigenvectors contain information about the Φ matrix. Let the eigendecomposition of $\mathbf{R}_z$ be $\mathbf{R}_z = \mathbf{E}\Lambda E^H$, where Λ has only the p nonzero eigenvalues. Then partition the $\mathbf{E}\Lambda^{1/2}$ matrix as

$$\mathbf{E}\Lambda^{1/2} = \begin{pmatrix} \Theta_x \\ \Theta_y \end{pmatrix}.$$

It can be shown that $\Theta_y = \Theta_x \Psi$, where Ψ is similar to Φ and its eigenvalues contain the direction information [18]. These facts are easily seen by observing that $\mathbf{R}_z$ has the structure

$$\mathbf{R}_z = \begin{pmatrix} \mathbf{A} \\ \mathbf{A}\Phi \end{pmatrix} (\mathbf{P}_s \mathbf{A}^H | \mathbf{P}_s \Phi^H \mathbf{A}^H),$$

where $\mathbf{P}_s$ is the signal covariance matrix. Provided that each of $\mathbf{P}_s$ and $(\mathbf{A}^H|\Phi\mathbf{A}^H)$ has rank p, then $\mathbf{R}_z$ also has rank p and any factorization of $\mathbf{R}_z$ where the inner dimension is p must necessarily be of the form

$$\mathbf{R}_z = \Theta\Gamma = \begin{pmatrix} \mathbf{AQ} \\ \mathbf{A}\Phi\mathbf{Q} \end{pmatrix}(\mathbf{Q}^{-1}\mathbf{P}_s\mathbf{A}^H|\mathbf{Q}^{-1}\mathbf{P}_s\Phi^H\mathbf{A}^H)$$

for some invertible matrix $\mathbf{Q}$. If Θ is partitioned as

$$\Theta = \begin{pmatrix} \Theta_x \\ \Theta_y \end{pmatrix},$$

then we have $\Theta_x\Psi = \Theta_y$ where $\Psi = \mathbf{Q}^{-1}\Phi\mathbf{Q}$.

In the presence of sensor noise, and when the covariances are estimated from finite data, $\mathbf{R}_z$ will not have rank p, and the number p of sources has to be estimated as the number of significantly large eigenvalues. The corresponding principal eigenvectors will not have structure $\Theta_y = \Theta_x\Psi$, and so a total least squares solution is used [18]. Find Ψ that minimizes

$$\|\Delta\|^2 + \|\Delta'\|^2 \tag{5}$$

subject to

$$(\Theta_x + \Delta)\Psi = (\Theta_y + \Delta').$$

It can be shown that if the sources are uncorrelated with each other (i.e., $\mathbf{P}_s$ is diagonal), then the matrix Ψ must be unitary, provided that there is no noise and the covariance matrix is exact. That is the case, because $\mathbf{P}_s$ and Φ will commute with each other, and $\Gamma = [\Gamma_x|\Gamma_y]$ will satisfy

$$\Psi\Gamma_y = \Gamma_x.$$

The choice of $\Theta = \mathbf{E}\Lambda^{1/2}$ ensures that $\Theta^H = \Gamma$, which as in § 1.2 ensures that Ψ is unitary. In the presence of noise, an estimate of the noise-free $\mathbf{E}\Lambda^{1/2}$ is

$$\hat{\Theta} = \mathbf{E}_1(\Lambda_1 - \rho\mathbf{I})^{1/2},$$

where Λ_1 and $\mathbf{E}_1$ are composed of the p largest eigenvalues and corresponding eigenvectors of the perturbed $\mathbf{R}_z$ and ρ is the arithmetic mean of the remaining eigenvalues.

When the sources are uncorrelated with each other and Θ is estimated as above, the direction-finding problem becomes a constrained total least squares problem. Find Ψ that minimizes

$$\|\Delta\|^2 + \|\Delta'\|^2 \tag{6}$$

subject to

$$(\Theta_x + \Delta)\Psi = (\Theta_y + \Delta')$$

and the additional constraint that Ψ be unitary.

2. Constrained total least squares. Before we derive the solution to the unitarily constrained total least squares problem, let us recall the solutions to the constrained least squares problem and to the total least squares problem. These are methods of constructing approximate solutions to an overdetermined inconsistent linear system $\mathbf{AX} = \mathbf{B}$. In the following, it is assumed that $\mathbf{A}$ and $\mathbf{B}$ each have p columns and more rows than columns.

2.1. Procrustes problem. The orthogonal Procrustes problem is to find a *unitary* matrix $\mathbf{X}$ that minimizes the least squares error

$$\|\mathbf{AX}-\mathbf{B}\|^2.$$

The solution to the above constrained least squares problem is as follows. Let the SVD of $\mathbf{H} \triangleq \mathbf{B}^H\mathbf{A}$ be $\mathbf{U}\Sigma\mathbf{V}^H$, where Σ includes all singular values even if they are zero. If $\mathbf{A}$ and $\mathbf{B}$ each have p columns, then so will $\mathbf{U}$, Σ, and $\mathbf{V}$. Then the constrained least squares solution is [19]

$$\hat{\mathbf{X}}_{\text{cls}} = \mathbf{V}\mathbf{U}^H. \tag{7}$$

2.2. Total least squares. Here the problem is to find a matrix $\mathbf{X}$ that minimizes

$$\|\Delta\|^2 + \|\Delta'\|^2$$

subject to $(\mathbf{A}+\Delta)\mathbf{X} = (\mathbf{B}+\Delta')$ [5]. The solution to this problem is obtained from the SVD of the concatenated matrix

$$(\mathbf{A}\,|\,\mathbf{B}) = \mathbf{U}\Sigma\mathbf{V}^H.$$

Partition $\mathbf{V}$ as

$$\mathbf{V} = \begin{pmatrix} \mathbf{V}_{a1} & \mathbf{V}_{a2} \\ \mathbf{V}_{b1} & \mathbf{V}_{b2} \end{pmatrix}$$

so that $\mathbf{V}_{a1}$ is $p \times p$. Then, the total least squares solution is given by [19], [20]

$$\hat{\mathbf{X}}_{\text{tls}} = (\mathbf{V}_{a1}^H)^{-1}\mathbf{V}_{b1}^H \quad \text{or} \quad \hat{\mathbf{X}}_{\text{tls}} = -\mathbf{V}_{a2}\mathbf{V}_{b2}^{-1}, \tag{8}$$

provided that $\mathbf{V}_{a1}$ (likewise $\mathbf{V}_{b2}$) is invertible. For methods to find the total least squares solution when $\mathbf{V}_{a1}$ and hence $\mathbf{V}_{b2}$ are singular or close-to-singular, the reader is referred to [21].

2.3. The CTLS problem. The problem here is the determination of a *unitary* matrix $\mathbf{X}$ that minimizes

$$\|\Delta\|^2 + \|\Delta'\|^2$$

subject to $(\mathbf{A}+\Delta)\mathbf{X} = (\mathbf{B}+\Delta')$. To derive the solution to this constrained total least squares problem, let us obtain an expression for the cost as a function of $\mathbf{X}$. For any $\mathbf{X}$, the corresponding error matrices Δ and Δ' satisfy

$$(\mathbf{X}^H\,|\,-\mathbf{I})\begin{pmatrix} \Delta^H \\ \Delta'^H \end{pmatrix} = -(\mathbf{X}^H\,|\,-\mathbf{I})\begin{pmatrix} \mathbf{A}^H \\ \mathbf{B}^H \end{pmatrix}$$

and the minimum-norm solution $(\Delta\,|\,\Delta')$ to the above system is

$$\begin{aligned}(\Delta\,|\,\Delta') &= -\tfrac{1}{2}(\mathbf{A}\,|\,\mathbf{B})\begin{pmatrix} \mathbf{I} & -\mathbf{X} \\ -\mathbf{X}^H & \mathbf{I} \end{pmatrix} \\ &= -\tfrac{1}{2}(\mathbf{A}-\mathbf{B}\mathbf{X}^H\,|\,\mathbf{B}-\mathbf{A}\mathbf{X}),\end{aligned}$$

using the unitary property of $\mathbf{X}$. Therefore, the cost as a function of $\mathbf{X}$ is

$$\text{Trace}\left[(\Delta\,|\,\Delta')\begin{pmatrix} \Delta^H \\ \Delta'^H \end{pmatrix}\right] = \text{Trace}\,[\tfrac{1}{2}(\mathbf{A}\mathbf{A}^H - \mathbf{B}\mathbf{X}^H\mathbf{A}^H - \mathbf{A}\mathbf{X}\mathbf{B}^H + \mathbf{B}\mathbf{B}^H)].$$

Therefore, the problem of constrained total least squares optimization is really a problem of finding a unitary matrix $\mathbf{X}$ to maximize Trace $(\mathbf{B}^H\mathbf{AX})$. Coincidentally, the solution to the orthogonal Procrustes problem of § 2.2 also maximizes the same quantity, because

$$\|\mathbf{AX}-\mathbf{B}\|^2 = \text{Trace}\,\{(\mathbf{AX}-\mathbf{B})(\mathbf{AX}-\mathbf{B})^H\} = \text{Trace}\,(\mathbf{AA}^H-\mathbf{AXB}^H-\mathbf{BX}^H\mathbf{A}^H+\mathbf{BB}^H)$$

using the unitary constraint on $\mathbf{X}$.

Thus the solution to the constrained total least squares problem is the same as the solution to the orthogonal Procrustes problem.

2.4. Discussion. In the classical least squares model, where the cost function is $\|\mathbf{AX}-\mathbf{B}\|^2$, matrix $\mathbf{A}$ is assumed to be error free, and only $\mathbf{B}$ is considered erroneous. When there is reason to believe that $\mathbf{A}$ is prone to errors and $\mathbf{B}$ is error free, a better cost function to work with is $\|\mathbf{BX}^{-1}-\mathbf{A}\|^2$. In general, the solution so obtained is distinct from the least squares solution. However, with the unitary constraint imposed on $\mathbf{X}$, the new cost is

$$\text{Trace}\,\{(\mathbf{BX}^H-\mathbf{A})(\mathbf{BX}^H-\mathbf{A})^H\} = \text{Trace}\,\{\mathbf{BB}^H+\mathbf{AA}^H-\mathbf{BX}^H\mathbf{A}^H-\mathbf{AX}^H\mathbf{B}^H\},$$

which is exactly the constrained least squares cost in the orthogonal Procrustes problem. Thus with the unitary constraint imposed on the solution matrix, it makes no difference where the error is modelled as coming from, as long as the minimization criterion is sum of squares. Thus it seems reasonable to expect that the constrained total least squares estimate, which assumes that both $\mathbf{A}$ and $\mathbf{B}$ are equally error prone, will not be different.

It can be shown that the cost function minimized by the (unconstrained) total least squares approach is

$$\text{Trace}\,\{(\mathbf{AX}-\mathbf{B})(\mathbf{X}^H\mathbf{X}+\mathbf{I})^{-1}(\mathbf{AX}-\mathbf{B})^H\},$$

which is distinct from both $\|\mathbf{AX}-\mathbf{B}\|^2$ and $\|\mathbf{A}-\mathbf{BX}^{-1}\|^2$. However, with the unitary constraint imposed on $\mathbf{X}$, the total least squares cost also reduces to the cost function of the constrained least squares problem.

The fact that the solutions to the constrained least squares and constrained total least squares problems are the same is both surprising and significant. Some researchers have suggested that a unitary solution to $\mathbf{AX} = \mathbf{B}$ be obtained from noisy measurements of $\mathbf{A}$ and $\mathbf{B}$ by a two-step approach (see the last paragraph of [22], for instance). In the first step, $(\mathbf{A}\,|\,\mathbf{B})$ is replaced by its total least squares approximation $(\mathbf{A}+\Delta\,|\,\mathbf{B}+\Delta')$ and in the second step, the orthogonal Procrustes problem is solved with $\mathbf{A}+\Delta$ and $\mathbf{B}+\Delta'$. It is now obvious that two SVDs are not needed to construct the constrained total least squares solution to $\mathbf{AX} = \mathbf{B}$, and that the first step is redundant.

2.5. Interrelationships. As in § 2.2, let the SVD of $\mathbf{H} = \mathbf{B}^H\mathbf{A}$ be $\mathbf{U}\Sigma\mathbf{V}^H$. Recall that the unitarily constrained total least squares solution to $\mathbf{AX} = \mathbf{B}$ is given by (7) as $\hat{\mathbf{X}}_{\text{cls}} = \mathbf{VU}^H$. Observe that it is also the unitary factor in the so-called *polar decomposition* of square matrix $\mathbf{H}^H$.

The polar decomposition of a $p \times p$ matrix $\mathbf{M}$ is

$$\mathbf{M} = \mathbf{XP}$$

where $\mathbf{X}$ is a $p \times p$ unitary matrix and $\mathbf{P}$ is positive semidefinite [23]. It is easily seen that if $\mathbf{M} = \mathbf{XP}$ is the polar decomposition of $\mathbf{M}$, then $\mathbf{M}$ may also be decomposed as

$$\mathbf{M} = \mathbf{SX}$$

where $\mathbf{S}$ is also positive semidefinite. In fact, we have the relationship $\mathbf{S} = \mathbf{XPX}^H$ between the two positive semidefinite factors.

The polar decomposition of $\mathbf{H}^H = \mathbf{A}^H\mathbf{B} = \mathbf{V}\Sigma\mathbf{U}^H$ is

$$\mathbf{A}^H\mathbf{B} = (\mathbf{V}\mathbf{U}^H)\mathbf{U}\Sigma\mathbf{U}^H = \hat{\mathbf{X}}_{\text{cls}}\mathbf{U}\Sigma\mathbf{U}^H. \tag{9}$$

Alternatively,

$$\mathbf{A}^H\mathbf{B} = \mathbf{V}\Sigma\mathbf{V}^H\hat{\mathbf{X}}_{\text{cls}}. \tag{10}$$

A similar decomposition of $\mathbf{A}^H\mathbf{B}$ can be made in terms of the standard least squares solution $\hat{\mathbf{X}}_{\text{ls}}$ of $\mathbf{AX} = \mathbf{B}$ using normal equations for the least squares solution, which are

$$\mathbf{A}^H\mathbf{B} = (\mathbf{A}^H\mathbf{A})\hat{\mathbf{X}}_{\text{ls}}. \tag{11}$$

The first factor $\mathbf{A}^H\mathbf{A}$ is positive semidefinite, as in (10), but the second factor is not necessarily unitary. There is a similar decomposition of $\mathbf{A}^H\mathbf{B}$ in terms of the *inverse least squares* solution $\hat{\mathbf{X}}_{\text{ils}}$ that minimizes $\|\mathbf{B}\mathbf{X}^{-1} - \mathbf{A}\|^2$:

$$\mathbf{A}^H\mathbf{B} = (\hat{\mathbf{X}}_{\text{ils}}^H)^{-1}\mathbf{B}^H\mathbf{B}. \tag{12}$$

In terms of the total least squares solution $\hat{\mathbf{X}}_{\text{tls}}$, a decomposition of $\mathbf{A}^H\mathbf{B}$ is

$$\mathbf{A}^H\mathbf{B} = (\mathbf{V}_{a1}\Sigma_1^2\mathbf{V}_{a1}^H)\hat{\mathbf{X}}_{\text{tls}} - \hat{\mathbf{X}}_{\text{tls}}(\mathbf{V}_{b2}\Sigma_2^2\mathbf{V}_{b2}^H), \tag{13}$$

where $\mathbf{U}\Sigma\mathbf{V}^H$ is the SVD of $(\mathbf{A}|\mathbf{B})$, and the subscripted matrices in (13) refer to the partitions of $\mathbf{V}$ and Σ as

$$\mathbf{V} = \begin{pmatrix} \mathbf{V}_{a1} & \mathbf{V}_{a2} \\ \mathbf{V}_{b1} & \mathbf{V}_{b2} \end{pmatrix}, \qquad \Sigma = \begin{pmatrix} \Sigma_1 & 0 \\ 0 & \Sigma_2 \end{pmatrix}.$$

Here, as in § 2.2, $\mathbf{V}_{a1}$ and Σ_1 are $p \times p$ matrices. Just as in (9)–(12), in the decomposition of (13) as well, the cofactors of $\hat{\mathbf{X}}$ are positive semidefinite.

Examining (9)–(13) for the scalar-unknown case ($p = 1$), it becomes obvious that the three solutions are related to each other by real numbers (say, r_1, r_2, and r_3):

$$\hat{x}_{ls} = r_1\hat{x}_{\text{cls}}, \quad \hat{x}_{\text{ils}} = r_2\hat{x}_{\text{cls}}, \quad \hat{x}_{\text{tls}} = r_3\hat{x}_{\text{cls}}.$$

In fact, the scalars r_1 and r_2 are nonnegative. Moreover, because $\Sigma_1 \geqq \Sigma_2$ in (13), so is r_3. This means that if one is only interested in the angle of the solution (as in the sinusoid retrieval and direction-finding applications), all four solutions give the same estimate of angle. Thus, although only the constrained total least squares solution lies on the unit circle, if the other three solutions are moved radially to the unit circle, they also yield the same result. Hence, in the sinusoid retrieval and direction-finding applications, if there is only one unknown, all four methods give identical estimates.

The same cannot be claimed in the multivariable case ($p > 1$). There is no obvious relationship between either the eigenvalues or the singular values of the four solutions. In fact, numerical examples in the next section show that the eigenvalues of these solutions can be very different.

2.6. Weighted problems. It is straightforward to show that the solution to the following unitarily constrained weighted total least squares problem

$$\text{Find unitary } \mathbf{X} \text{ to minimize } \|\mathbf{W}(\Delta|\Delta')\| \quad \text{subject to } (\mathbf{A}+\Delta)\mathbf{X} = (\mathbf{B}+\Delta')$$

is also a solution to the unitarily constrained weighted least squares problems

$$\text{Find unitary } \mathbf{X} \text{ to minimize } \|\mathbf{W}(\mathbf{AX}-\mathbf{B})\|,$$

$$\text{Find unitary } \mathbf{X} \text{ to minimize } \|\mathbf{W}(\mathbf{A}-\mathbf{B}\mathbf{X}^H)\|,$$

as long as matrix $\mathbf{W}$ has rank greater than p. The common solution is found from an SVD of $\mathbf{B}^H\mathbf{W}^H\mathbf{W}\mathbf{A}$. If the SVD of $\mathbf{B}^H\mathbf{W}^H\mathbf{W}\mathbf{A}$ is $\mathbf{U}\Sigma\mathbf{V}^H$, where the $p \times p$ matrix Σ contains all p singular values even if they are zero, then $\mathbf{X} = \mathbf{V}\mathbf{U}^H$.

2.7. Computation. The solution to the constrained total least squares problem may be computed using the Golub–Reinsch SVD algorithm or by the following Newton-iterative technique. Recall that the desired matrix is also the unitary factor in the polar decomposition of $\mathbf{A}^H\mathbf{B}$ or of $\mathbf{A}^H\mathbf{W}^H\mathbf{W}\mathbf{B}$ in the weighted case. The following Newton iterations proposed by Higham [24],

$$\mathbf{X}_0 = \mathbf{M},$$

$$\mathbf{X}_{k+1} = \tfrac{1}{2}(\mathbf{X}_k + (\mathbf{X}_k^H)^{-1}), \qquad k = 0, 1, 2, 3, \cdots,$$

converge quadratically to the unitary polar factor of the square matrix $\mathbf{M}$. Convergence can be accelerated by appropriate scaling of the iterate at each step [24]. Because the iterations require only matrix multiplications, inversions, and additions, this approach may be computationally less expensive than the SVD approach, especially on parallel multiprocessor architectures.

3. Numerical examples. This section presents a few numerical examples to illustrate the applicability of the unitarily constrained total least squares problem.

Example 1. In the first example, a 30×2 matrix $\mathbf{A}$ was computed from normal $(0, 1)$-distributed, real-valued, independent random numbers. Both $\mathbf{A}$ and $\mathbf{B} = \mathbf{A}\mathbf{X}$, for

$$\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

were corrupted by additive, zero-mean, real-valued, white noise distributed as normal $(0, 0.05)$, and $\mathbf{X}$ was estimated from the noisy $\mathbf{A}$, $\mathbf{B}$ pair by three methods: least squares (LS), total least squares (TLS), and constrained total least squares (CTLS or CLS). The experiment was repeated for 200 independent sets of additive noise, and the bias and mean square error in the 200 estimates of the first row of $\mathbf{X}$, namely $(0, 1)$, are listed in Table 1.

Example 2. In this simulation, the given data are noisy measurements of the covariance lags

$$r(0)r(1)r(2)\cdots r(12)$$

of a real-valued, sinusoidal process: $r(m) = \cos(2\pi fm)$, $f = 0.25$. The lags were corrupted by additive real-valued perturbations independent of each other. The three variants of TAM using LS, TLS, and CTLS, were employed to estimate f from the perturbed lags. For all methods, a rank-2 approximation was used to estimate Θ, which was picked as $\mathbf{U}_1(\Sigma_1 - \rho\mathbf{I})^{1/2}$ according to the discussion in § 1.2. The experiment was repeated for 200 independent sets of perturbations on the covariances, and the bias and mean square error in the 200 estimates of f from each method are listed in Table 2, for two different

TABLE 1

	Bias		Mean square error	
LS	-1.77×10^{-4}	-40.32×10^{-4}	1.66×10^{-4}	2.12×10^{-4}
TLS	-1.72×10^{-4}	-13.58×10^{-4}	1.66×10^{-4}	1.99×10^{-4}
CTLS	-0.95×10^{-4}	-0.44×10^{-4}	8.76×10^{-5}	6.02×10^{-9}

TABLE 2

Perturbation distribution: normal (0, 0.05)

	Bias	Mean square error
LS	4.85×10^{-5}	3.12×10^{-7}
TLS	4.85×10^{-5}	3.12×10^{-7}
CTLS	4.83×10^{-5}	3.09×10^{-7}

Perturbation distribution: normal (0, 0.5)

	Bias	Mean square error
LS	7.33×10^{-4}	2.34×10^{-4}
TLS	7.00×10^{-4}	2.29×10^{-4}
CTLS	6.97×10^{-4}	2.26×10^{-4}

levels of perturbation. When the standard deviation of the perturbation was 0.5, there were four misses by each method; i.e., for four trials, the estimates of f were very poor. The statistics listed above are for the remaining 196 trials.

Example 3. The three TAM-based algorithms were next tested on

$$r(m) = 4.88 \cos(2\pi f_1 m) + 0.16 \cos(2\pi f_2 m), \quad f_1 = 0.145, \quad f_2 = 0.125.$$

Again, $r(0), r(1), \cdots, r(12)$ were assumed given, but perturbed by real-valued independent errors distributed normally with zero-mean. Mean square errors in the estimates of f over 200 trials are tabulated for various levels of perturbation in Table. 3. When the standard deviation of perturbations was raised to 0.025, there were 52 misses in the 200 trials by all three methods.

Example 4. Data received by an ESPRIT array composed of eight co-oriented, half-wavelength doublets were synthesized for the following scenario. The doublets are located randomly and have differing gain and phase. Two plane waves are incident at angles

$$\theta_1 = 0.41\pi, \qquad \theta_2 = 0.42\pi.$$

160 synchronized snapshots (corresponding to different time instants t) are collected of the sensor data. Source amplitudes $s(t)$ at the 160 time instants are uncorrelated with each other, and are white, zero-mean, and Gaussian. The standard deviations of the source amplitudes are, respectively,

$$A_1 = 10, \qquad A_2 = 1;$$

TABLE 3

Standard deviation of perturbation	Mean square error					
	LS		TLS		CTLS	
	f_1	f_2	f_1	f_2	f_1	f_2
0.001	7.91×10^{-10}	58.6×10^{-8}	7.91×10^{-10}	58.6×10^{-8}	7.58×10^{-10}	57.7×10^{-8}
0.002	3.24×10^{-9}	23.3×10^{-7}	3.24×10^{-9}	23.3×10^{-7}	3.11×10^{-9}	22.9×10^{-7}
0.0025	5.16×10^{-9}	36.3×10^{-7}	5.16×10^{-9}	36.3×10^{-7}	4.97×10^{-9}	35.8×10^{-7}
0.005	2.44×10^{-7}	14.3×10^{-6}	2.45×10^{-7}	14.3×10^{-6}	2.38×10^{-7}	14.1×10^{-6}

TABLE 4

	Bias		Mean square error	
	θ_1/π	θ_2/π	θ_1/π	θ_2/π
LS	1.48×10^{-5}	3.64×10^{-4}	8.61×10^{-8}	6.65×10^{-6}
TLS	1.41×10^{-5}	3.64×10^{-4}	8.53×10^{-8}	6.63×10^{-6}
CTLS	4.56×10^{-5}	3.07×10^{-4}	0.49×10^{-8}	6.33×10^{-6}

i.e.,

$$\mathbf{P}_s = \begin{pmatrix} 100 & 0 \\ 0 & 1 \end{pmatrix}.$$

The 16×160 data matrix $[\underline{z}(1)\underline{z}(2) \cdots \underline{z}(160)]$ was corrupted by 60 independent realizations of normal $(0, 0.07)$ distributed, complex-valued, white noise. The three variants of the ESPRIT algorithm using LS, TLS, and CTLS, were employed on the 60 data sets to estimate θ_1 and θ_2. $\mathbf{R}_z$ was estimated as the temporal average over the 160 snapshots of the noisy $\underline{z}(k)\underline{z}(k)^H$, rank-2 reduction was used and $\mathbf{E}_1(\Lambda_1 - \rho\mathbf{I})^{1/2}$ was used as the estimate of Θ in all methods. The bias and mean square error of the three methods over the 60 estimates is listed in Table 4.

Note that the imposition of the unitary constraint makes a slight improvement in the direction estimate (θ_2) of the weaker signal and considerably reduces the mean square error in the direction estimate (θ_1) of the stronger signal. This reduction occurs despite a substantial increase in bias in the estimate of θ_1, which means that the standard deviation from the mean is much lower for the constrained approach than for the others. This improvement in mean square error is even more pronounced when the signal is made stronger with respect to the additive noise floor.

The experiment was repeated with everything unchanged except for source strengths:

$$A_1 = 25, \qquad A_2 = 1.$$

The results are presented in Table 5 (for the same noise level as before). When the noise standard deviation was raised from 0.07 to 0.70, then all methods missed the weaker direction (θ_2) completely (their estimates were off by large margins), and the mean square error in the θ_1 estimates were as listed in Table 6.

Next, the experiment was repeated with noise standard deviation at 0.07 and source amplitudes

$$A_1 = 1, \qquad A_2 = 10,$$

the roles of strong and weak directions having been reversed. The results over 60 trials are presented in Table 7. Note the improvement in mean square error despite increased bias in the direction estimate (θ_2) of the stronger signal.

TABLE 5

	Bias		Mean square error	
	θ_1/π	θ_2/π	θ_1/π	θ_2/π
LS	-0.54×10^{-5}	-8.41×10^{-4}	5.04×10^{-8}	1.16×10^{-5}
TLS	-0.51×10^{-5}	-8.41×10^{-4}	5.04×10^{-8}	1.16×10^{-5}
CTLS	-2.79×10^{-5}	-8.18×10^{-4}	0.12×10^{-8}	1.18×10^{-5}

TABLE 6

	LS	TLS	CTLS
MSE in θ_1/π	9.75×10^{-7}	9.73×10^{-7}	8.48×10^{-7}

TABLE 7

	Bias		Mean square error	
	θ_1/π	θ_2/π	θ_1/π	θ_2/π
LS	1.38×10^{-4}	-0.41×10^{-4}	2.30×10^{-5}	3.53×10^{-7}
TLS	1.33×10^{-4}	-0.37×10^{-4}	2.29×10^{-5}	3.11×10^{-7}
CTLS	-0.32×10^{-4}	1.30×10^{-4}	2.27×10^{-5}	1.33×10^{-7}

Example 5. Here, the same scenario as in Example 4 was simulated, with the only changes being in the directions of the plane waves and the additive noise level. The directions used to synthesize the data were

$$\theta_1 = 0.10\pi, \qquad \theta_2 = 0.14\pi.$$

Corresponding source strengths were

$$A_1 = 10, \qquad A_2 = 1.$$

The data were corrupted by additive white noise distributed as normal (0, 0.07). The bias and mean square error in the estimates of the two directions over the 60 trials are presented in Table 8.

If, instead of white noise, the additive noise is colored (moving average):

$$r(0) = 0.05, \qquad r(1) = 0.035,$$

then the results are as presented in Table 9.

Note that enforcing the unitary constraint significantly reduces mean square error despite increasing the bias. The bias in this approach is caused by the poor quality of the covariance estimate $\mathbf{R}_z$ from only 160 snapshots. Recall that the unitary property of Ψ holds only when $\mathbf{R}_z$ is known exactly. While the SVD-based rank reduction provides robustness to noise in the data, it does not seem to provide sufficient robustness to covariance estimation errors. However, the bias in these direction estimates reduces as the number of snapshots is increased.

Example 6. The purpose of this example is to demonstrate the reduction in bias in the direction estimates of the constrained approach, as the number of snapshots available is increased. The array simulated here is composed of three co-oriented, randomly located,

TABLE 8

	Bias		Mean square error	
	θ_1/π	θ_2/π	θ_1/π	θ_2/π
LS	0.76×10^{-4}	-6.08×10^{-4}	7.83×10^{-6}	3.83×10^{-5}
TLS	0.79×10^{-4}	-6.12×10^{-4}	7.91×10^{-6}	3.83×10^{-5}
CLS	-2.62×10^{-4}	-3.56×10^{-4}	0.86×10^{-6}	3.71×10^{-5}

TABLE 9

	Bias		Mean square error	
	θ_1/π	θ_2/π	θ_1/π	θ_2/π
LS	0.78×10^{-4}	0.93×10^{-4}	3.23×10^{-7}	1.65×10^{-7}
TLS	0.79×10^{-4}	0.92×10^{-4}	3.21×10^{-7}	1.65×10^{-7}
CLS	-2.50×10^{-4}	3.33×10^{-4}	0.66×10^{-7}	1.61×10^{-7}

half-wavelength, perfectly matched doublets. The doublets have arbitrarily different gain and phase. Two plane waves are incident at angles

$$\theta_1 = 0.10\pi, \qquad \theta_2 = 0.14\pi.$$

As before, source signals are uncorrelated with each other, and are white, zero-mean, Gaussian processes with equal standard deviation $A_1 = A_2 = 1$. The data received by the array is corrupted by twenty independent realizations of normal (0, 0.014) distributed, complex-valued, white noise. The bias and mean square error in the direction estimates over the 20 trials are presented in Table 10 for different numbers of snapshots.

When covariance estimation errors are large (possibly because the number of snapshots is small), then enforcing the unitary constraint on Ψ seems to increase the bias in the resulting direction estimates. When SNR is low, however, the reduction in variance dominates over the increase in bias, so that the mean square error is in fact smaller. At the high SNR end, on the other hand, enforcing the unitary constraint on Ψ may do more harm than good. In many practical direction-finding applications (including sonar), SNR is generally low, and a large number of snapshots (as many as 10,000 per second) is available. In such situations, if the sources are, in fact, uncorrelated, use of the unitary constraint can prove beneficial.

4. Concluding remarks. It was discovered here that the unitarily constrained total least squares problem that arises in a few signal processing systems has a simple analytical

TABLE 10

	Bias		Mean square error	
	θ_1/π	θ_2/π	θ_1/π	θ_2/π
		160 snapshots		
LS	-0.45×10^{-3}	-0.14×10^{-3}	1.32×10^{-5}	6.18×10^{-6}
TLS	-0.45×10^{-3}	-0.13×10^{-3}	1.32×10^{-5}	6.20×10^{-6}
CLS	1.36×10^{-3}	-1.43×10^{-3}	0.98×10^{-5}	5.83×10^{-6}
		1000 snapshots		
LS	-1.21×10^{-3}	1.36×10^{-4}	4.55×10^{-4}	2.36×10^{-4}
TLS	-1.22×10^{-3}	1.42×10^{-4}	4.59×10^{-4}	2.33×10^{-4}
CLS	-0.59×10^{-3}	-1.27×10^{-4}	0.33×10^{-4}	0.15×10^{-4}
		1500 snapshots		
LS	0.25×10^{-4}	-1.71×10^{-4}	2.81×10^{-7}	2.47×10^{-7}
TLS	0.24×10^{-4}	-1.71×10^{-4}	2.81×10^{-7}	2.47×10^{-7}
CLS	-3.77×10^{-4}	1.21×10^{-4}	3.24×10^{-7}	0.77×10^{-7}

solution, and that the solution is the same as the solution to the orthogonal Procrustes problem. The applicability of the unitarily constrained total least squares solution to three problems in motion estimation, sinusoid retrieval, and direction finding was studied.

In the motion estimation problem, the rotation matrix to be estimated from data is known a priori to be unitary, and the constrained total least squares problem applies directly. In the sinusoid retrieval problem and the direction-finding problem, the eigenvalues of intermediate matrices $\mathbf{F}$ and Ψ, respectively, are expected to be on the unit circle. Current methods for estimating these matrices do not exploit this fact. Instead, they estimate the matrix without imposing such a constraint and project the eigenvalues of the estimated matrix radially to the unit circle. Here, a scheme was found for indirectly incorporating the unit-modulus information about the eigenvalues into the matrix estimation procedure. A proper choice of covariance matrix factor Θ was found to make the $\mathbf{F}$ or Ψ matrix unitary in the noise-free situation. In the noisy situation, this factor Θ is estimated via an SVD of the covariance matrix, and the unitary constraint is enforced in the process of estimating the matrix from the factor.

While the use of constraints based on a priori information about the signal or its parameters can improve the quality of the parameter estimates from noisy measurements of the signal, it is well known that their use introduces a different kind of estimation error—errors resulting from inappropriateness of the constraint. It makes the estimates sensitive to any deviation of the signal or its parameters from the constrained model. For instance, if the matrix that needs to be estimated is only near-unitary, and not exactly unitary, then enforcing the unitary constraint on the estimate can make the solution incorrect even when there is no noise. In the sinusoid retrieval and direction-finding applications studied here, deviation from unitariness can occur because of errors in covariance estimation. It was seen that this causes the estimates to be biased. Since unitariness in the direction-finding application relies on uncorrelatedness between sources, the direction estimates obtained by the constrained approach are also sensitive to any correlation between sources.

Regardless of correlation between sources (or initial phases of the sinusoids) and covariance estimation errors, and irrespective of how the factor Θ is chosen, the eigenvalues of the matrix ($\mathbf{F}$ or Ψ) are expected to be on the unit circle. This property translates to unitariness of the matrix only when there is no correlation between sources and the factor Θ is chosen correctly. If a simple method can be found that directly enforces the unit-modulus property of the eigenvalues, then the errors caused by source correlation and covariance estimation errors may be avoided.

For the special case of one unknown ($p = 1$), it can be seen that the unit-modulus property is the same as unitariness, that this does not depend on which factor Θ is chosen, and that it is insensitive to covariance estimation errors. Here the unitary constraint may be enforced without any qualms. However, it was discovered that the solution to the unitarily constrained total least squares problem in the scalar-unknown case may also be obtained by taking either the unconstrained least squares solution or the unconstrained total least squares solution and projecting it radially onto the unit circle.

The unitarily constrained, weighted total least squares problem was also investigated, and a simple solution was found.

Acknowledgments. Discussions with Professors Bhaskar D. Rao and Shankar Chatterjee of the University of California at San Diego, Professor Gene H. Golub of Stanford University, and Professor Alan J. Laub of the University of California at Santa Barbara are gratefully acknowledged.

REFERENCES

[1] S. D. BLOSTEIN AND T. S. HUANG, *Estimating 3-D motion from range data*, in Proc. First Conference on Artificial Intelligence and Applications, Denver, CO, Dec. 1984, pp. 246–250.

[2] D. CYGANSKI AND J. A. ORR, *Applications of tensor theory to object recognition and orientation determination*, IEEE Trans. Pattern Anal. Mach. Intelligence, PAMI-7 (1985), pp. 663–673.

[3] T. S. HUANG, S. D. BLOSTEIN, AND E. A. MARGERUM, *Least squares estimation of motion parameters from 3-D point correspondences*, in Proc. Conference on Computer Vision and Pattern Recognition, Miami Beach, FL, June 1986.

[4] K. S. ARUN, T. S. HUANG, AND S. D. BLOSTEIN, *Least-squares fitting of two 3-D point sets*, IEEE Trans. Pattern Anal. Mach. Intelligence, PAMI-9 (1987), pp. 698–700.

[5] G. H. GOLUB AND C. F. VAN LOAN, *An analysis of the total least squares problem*, SIAM J. Numer. Anal., 17 (1980), pp. 883–893.

[6] S. M. KAY AND S. L. MARPLE, JR., *Spectrum analysis—A modern perspective*, Proc. IEEE, 69 (1981), pp. 1380–1419.

[7] V. F. PISARENKO, *The retrieval of harmonics from a covariance function*, Geophys. J. Royal Astron. Soc., 33 (1973), pp. 247–266.

[8] D. W. TUFTS AND R. KUMARESAN, *Estimation of frequencies of multiple sinusoids: Making linear prediction work like maximum likelihood*, Proc. IEEE, 70 (1982), pp. 975–989.

[9] J. E. CHAMBERLAIN, *The Principles of Interferometric Spectroscopy*, John Wiley, New York, 1979.

[10] M. H. COHEN, D. L. JAUNCEY, K. I. KELLERMANN, AND B. G. CLARK, *Radio interferometry at one-thousandth second of arc*, Science, 162 (1968), pp. 88–94.

[11] T. KAILATH, *Linear Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[12] S. Y. KUNG, K. S. ARUN, AND D. V. BHASKARRAO, *State-space and singular value decomposition based methods for the harmonic retrieval problem*, J. Opt. Soc. Amer., 73 (1983), pp. 1799–1811.

[13] K. S. ARUN AND B. D. RAO, *An improved Toeplitz approximation method*, in Proc. 1988 IEEE Internat. Conference on ASSP, New York, 1988, pp. 2352–2355.

[14] S. Y. KUNG, *A Toeplitz approximation method and some applications*, in Proc. Internat. Symposium on the Mathematical Theory of Networks and Systems, Santa Monica, CA, 1981, pp. 262–266.

[15] R. O. SCHMIDT, *Multiple emitter location and signal parameter estimation*, in Proc. RADC Spectral Estimation Workshop, Griffiss Air Force Base, Rome, NY, 1979, pp. 243–258.

[16] G. BIENVENU AND L. KOPP, *Optimality of high resolution array processing using eigenvalues*, IEEE Trans. Acoust., Speech Signal Process., ASSP-31 (1983), pp. 1235–1248.

[17] A. PAULRAJ, R. ROY, AND T. KAILATH, *Estimation of signal parameters via rotational invariance techniques—ESPRIT*, in Proc. 19th Annual Asilomar Conference on Circuits, Systems, and Computers, Pacific Grove, CA, 1985, pp. 83–89.

[18] R. ROY AND T. KAILATH, *Total least squares ESPRIT*, in Proc. 21st Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, 1987, pp. 297–301.

[19] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

[20] M. D. ZOLTOWSKI, *Signal processing applications of the method of total least squares*, in Proc. 21st Annual Asilomar Conference on Signals, Systems, and Computers, Pacific Grove, CA, 1987, pp. 290–296.

[21] S. VAN HUFFEL AND J. VANDEWALLE, *Analysis and solution of the nongeneric total least squares problem*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 360–372.

[22] M. D. ZOLTOWSKI, *Novel techniques for estimation of array signal parameters based on matrix pencils, subspace rotations, and total least squares*, in Proc. 1988 IEEE Internat. Conference on Acoustics, Speech, and Signal Processing, New York, 1988, pp. 2861–2864.

[23] F. R. GANTMACHER, *The Theory of Matrices*, Chelsea, New York, 1959.

[24] N. J. HIGHAM, *Computing the polar decompositon—with applications*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1160–1174.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 746–763, July 1992

005

THE ANALYSIS FOR THE TOTAL LEAST SQUARES PROBLEM WITH MORE THAN ONE SOLUTION*

MUSHENG WEI†

Abstract. This paper presents an analysis of the solutions of the total least squares problem (TLS) $AX \approx B$ in cases where the matrix (A, B) may have multiple smallest singular values. General formulas for the minimum norm TLS solutions are given; the difference between the TLS and the LS solutions is obtained; the error bounds for the perturbed TLS solutions with or without minimal length are deduced. The analysis is useful especially for rank deficient problems and generalizes previous results of Golub and Van Loan, Van Huffel and Vandewalle, and Zoltowski. Numerical results for a practical application are also given to verify the error bounds.

Key words. total least squares, rank deficient, perturbation

AMS(MOS) subject classifications. 65F20, 15A18

1. Introduction In this paper we use the following notations. Let $C^{m\times n}(R^{m\times n})$ be the set of $m \times n$ matrices with complex (real) entries and $C^m = C^{m\times 1}$ be the set of m-dimensional vectors. For a matrix $A \in C^{m\times n}$, let $A^H \in C^{n\times m}$ be the (complex) conjugate transpose of A ($A^H = A^T$ when $A \in R^{m\times n}$), $A^+ \in C^{n\times m}$ the pseudoinverse of A, Rank (A) the rank of A, $R(A)$ the range of A, I_n the identity matrix of order n, $\varnothing_{m\times n}$ the $m \times n$ matrix of all zero entries (if no confusion occurs, we will drop the subindex), $\|\cdot\| = \|\cdot\|_2$ the spectral norm, and $\|\cdot\|_F$ the Frobenius norm. Span $\{v_1, \cdots, v_n\}$ denotes the vector space spanned by $v_1, \cdots, v_n$.

The problem of linear parameter estimation arises in a broad class of scientific problems. It can be described by a linear equation

$$Ax \approx b, \tag{1.1a}$$

where A is the $m \times n$ data matrix and b is the m-dimensional observation vector. A and b are the perturbed version of the exact but unobservable data A_0 and b_0, respectively, i.e., the exact relation is

$$A_0 x = b_0. \tag{1.1b}$$

In the ordinary least squares (LS) approach to (1.1a), the measurements in A are assumed to be free of error and all errors are confined to b. This results in the following equivalent problem:

$$\text{minimize } \|r\| \quad \text{subject to } b - r \in R(A). \tag{1.2}$$

The total least squares (TLS) approach to fitting, on the other hand, is appropriate when there are errors in both A and b. If we rewrite (1.1a) as

$$(A, b)\begin{pmatrix} x \\ -1 \end{pmatrix} = 0, \tag{1.3a}$$

then the TLS approach amounts to considering the following problem:

$$\underset{\Delta A, \Delta b}{\text{minimize}} \ \|(\Delta A, \Delta b)\|_F \quad \text{subject to } b - \Delta b \in R(A - \Delta A). \tag{1.4a}$$

* Received by the editors October 16, 1989; accepted for publication (in revised form) November 28, 1990. This work was supported by the Education Committee, People's Republic of China.

† Department of Mathematics, East China Normal University, Shanghai 200062, China.

Golub and Van Loan first gave a mathematical analysis of the TLS problem [2]. They also extended the problem to cover the case in which B is an $m \times d$ ($d \geqq 1$) observation matrix [3]. Then (1.3a) and (1.4a) become, respectively,

$$(A,B)\begin{pmatrix} X \\ -I \end{pmatrix} = \varnothing \tag{1.3b}$$

and

$$\underset{\Delta A,\Delta B}{\text{minimize}} \ \|(\Delta A, \Delta B)\|_F \quad \text{subject to } B-\Delta B \subseteq R(A-\Delta A). \tag{1.4b}$$

Van Huffel and Vandewalle [9], [10], [12] investigated problem (1.4b), presented a detailed analysis, and extended the analysis to cover the nongeneric TLS case. They also proposed an efficient algorithm to compute the TLS solutions [11]. Zoltowski [17] also developed an analysis for the problem.

The main tool for solving TLS problems is the singular value decomposition (SVD) for A and (A, B) (or (A, b) when $d = 1$). In this paper we assume that $m \geqq n + d$. Let the SVD for A and (A, B) be

$$\bar{U}^H A \bar{V} = \bar{\Sigma}, \qquad U^H (A,B) V = \Sigma, \tag{1.5}$$

respectively, where

$$\begin{aligned} &\bar{\Sigma} = \operatorname{diag}(\bar{\sigma}_1, \cdots, \bar{\sigma}_n) \quad \text{with } \bar{\sigma}_1 \geqq \cdots \geqq \bar{\sigma}_n \geqq 0, \\ &\Sigma = \operatorname{diag}(\sigma_1, \cdots, \sigma_{n+d}) \quad \text{with } \sigma_1 \geqq \cdots \geqq \sigma_n \geqq \sigma_{n+1} \geqq \cdots \geqq \sigma_{n+d} \geqq 0, \end{aligned} \tag{1.6}$$

and four matrices U, $\bar{U} \in C^{m \times m}$, $\bar{V} \in C^{n \times n}$, and $V \in C^{(n+d)\times(n+d)}$ are all unitary.

Golub and Van Loan [2], [3]; Van Huffel and Vandewalle [9], [10], [12]; and other researchers mainly consider the case in which $\bar{\sigma}_n > \sigma_{n+1}$. This condition guarantees the existence and uniqueness of a solution to the TLS problem (1.4) [3], [9], [10]. However, in many practical applications, one may encounter the case $\bar{\sigma}_p > \sigma_{p+1} = \cdots = \sigma_{n+1}$ for some $p < n$. In this case, the TLS problem may have more than one solution. For example, in electromagnetics data processing problems, one has for some sample rate ΔT a discrete time series

$$f_l = \sum_{j=1}^{r} c_j \exp(l \lambda_j \Delta T), \qquad l = 0, 1, \cdots, m+n-1,$$

obtained from experimental or computational results, where $m \geqq n \geqq r$; λ_j and c_j for $1 \leqq j \leqq r$ are the resonance poles and residues to be determined [8]. When applying Prony's method [8], [15], one may need to solve a TLS problem that is rank deficient (also see the example in § 5). These TLS problems are more complicated but their analysis is also important and useful.

In this paper we extend the results in [2], [3], [9], [10], [12], and [17] in order to analyze the above-mentioned TLS problems with more than one solution. We deduce the formulas for the minimum norm TLS solution X_{TLS}, the relationship between the TLS and the LS solutions, and perturbation bounds of the TLS solutions. In this paper, we study the TLS problem (1.4b) but all the results also hold for the TLS problem (1.4a), in which one only needs to set $d = 1$.

Note also that our attention is focused on the algebraic properties and the perturbation theory for the TLS problem. To simplify the discussion we assume that no scaling or weighting is needed. Of course, weighting and scaling play an important role in solving

TLS problems. They affect the singular values of the matrices and thus affect the accuracy of the solutions. It requires a separate paper to handle this matter [17].

This paper is arranged as follows. In § 2, we discuss various formulas for X_{TLS}; in § 3, we compare the TLS solution with the LS solution; in § 4, we study the perturbation bounds for the TLS solutions, with or without minimal length; in § 5, we present the numerical results of a test problem to verify the inequalities; finally, in § 6, we make some concluding remarks.

To complete this section, we list some known results that are necessary for our discussion (see also, e.g., [3]).

- Let $G, H \in C^{m\times n}$ and the singular values σ_j of G and H be arranged in decreasing order. Then

$$|\sigma_j(G)-\sigma_j(H)| \leqq \|G-H\| = \sigma_1(G-H) \quad \text{for } j=1, \cdots, n. \tag{1.7}$$

- Let the singular values of $A \in C^{m\times n}$ and $(A, B) \in C^{m\times(n+d)}$ be as in (1.6). Then the following interlacing relation holds:

$$\sigma_j \geqq \bar{\sigma}_j \geqq \sigma_{j+d} \quad \text{for } j=1, \cdots, n. \tag{1.8}$$

- Let the SVD of $A \in C^{m\times n}$ be given in (1.5). If $k \leqq r = \mathrm{Rank}\,(A)$ and $A_k = \sum_{i=1}^{k} \bar{\sigma}_i \bar{u}_i \bar{v}_i^H$, then by the Eckart–Young theorem,

$$\begin{aligned} &\min_{\mathrm{Rank}\,(D)=k} \|A-D\| = \|A-A_k\| = \bar{\sigma}_{k+1}, \\ &\min_{\mathrm{Rank}\,(D)=k} \|A-D\|_F = \|A-A_k\|_F = \left(\sum_{i=k+1}^{n} \bar{\sigma}_i^2\right)^{1/2}. \end{aligned} \tag{1.9}$$

- Let $A, B \in C^{m\times n}$. Then one has [6], [14]

$$B^+ - A^+ = -B^+(B-A)A^+ + B^+(I-AA^+) - (I-B^+B)A^+. \tag{1.10}$$

2. The TLS solutions. In this section, we discuss various formulas of the minimum norm TLS solution in the case where

$$\bar{\sigma}_p > \sigma_{p+1} = \cdots = \sigma_{n+1},$$

with σ_j and $\bar{\sigma}_j$ the singular values of (A, B) and A, respectively.

An extended CS decomposition [7] can be used to study the TLS problem. For some integer $p \leqq n$, partition V into the following form:

$$V = \begin{pmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \end{pmatrix} \begin{matrix} n \\ d \end{matrix} \tag{2.1}$$

$$p, n+d-p.$$

Assume that $\mathrm{Rank}\,(V_{11}) = r \leqq p$ and that $d_1 = \cdots = d_t = 1 > d_{t+1} \geqq \cdots \geqq d_r > 0$ are the nonzero singular values of V_{11}. Set

$$C = \mathrm{diag}\,(d_{t+1}, \cdots, d_r) \quad \text{and} \quad S = \mathrm{diag}\,(\sqrt{1-d_{t+1}^2}, \cdots, \sqrt{1-d_r^2}).$$

It was shown [7] that there exist unitary matrices W_1, W_2, Z_1, and Z_2 with appropriate sizes, such that

$$V = \begin{pmatrix} W_1 & \\ & W_2 \end{pmatrix} \begin{pmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{pmatrix} \begin{pmatrix} Z_1^H & \\ & Z_2^H \end{pmatrix}, \tag{2.2}$$

where

$$D_{11} = \operatorname{diag}(I_t, C, 0_{(n-r)\times(p-r)}), \qquad D_{12} = \operatorname{diag}(0_{t\times(d+t-p)}, S, I_{n-r}),$$
$$D_{21} = \operatorname{diag}(0_{(d+t-p)\times t}, S, I_{p-r}), \qquad D_{22} = \operatorname{diag}(I_{d+t-p}, -C, 0_{(p-r)\times(n-r)}). \tag{2.3}$$

First, we deduce the following result.

LEMMA 2.1. *Let* $A \in C^{m\times n}$, $B \in C^{m\times d}$, *and the* SVD *for* A *and* (A, B) *be* (1.5):

$$\bar{U}^H A \bar{V} = \bar{\Sigma}, \qquad U^H(A, B)V = \Sigma.$$

For some integer $p \leqq n$, *partition* V *into the form in* (2.1). *Then*

(1) *Both* V_{22} *and* V_{11} *are of full rank or neither of them is of full rank.*

(2) $\bar{\sigma}_p > \sigma_{p+1}$ *implies that both* V_{11} *and* V_{22} *are of full rank.*

Proof. The proof is in two parts.

(1) The assertion is obvious from the extended *CS* decomposition (2.2), (2.3), because both V_{11} and V_{22} have $p - r$ zero singular values.

(2) If $\bar{\sigma}_p > \sigma_{p+1}$ but Rank $(V_{11}) = p_1 < p$, then from the SVD (1.5) for (A, B) and the partitioning (2.1) for V, we have

$$A = U_1\Sigma_1 V_{11}^H + U_2\Sigma_2 V_{12}^H, \qquad B = U_1\Sigma_1 V_{21}^H + U_2\Sigma_2 V_{22}^H, \tag{2.4}$$

where U_1 and U_2 are the first p and the last $(m - p)$ columns of U, respectively, $\Sigma_1 = \operatorname{diag}(\sigma_1, \cdots, \sigma_p)$, $\Sigma_2 = \operatorname{diag}(\sigma_{p+1}, \cdots, \sigma_{n+d})$. Then from (2.4) and (1.7),

$$\bar{\sigma}_j \leqq \sigma_j(U_1\Sigma_1 V_{11}^H) + \|U_2\Sigma_2 V_{12}^H\| \leqq \sigma_j(U_1\Sigma_1 V_{11}^H) + \sigma_{p+1}, \qquad j = 1, \cdots, n.$$

Now $p_1 < p$ by assumption, so $\sigma_j(U_1\Sigma_1 V_{11}^H) = 0$ for $j = p_1 + 1, \cdots, p, \cdots, n$. In particular,

$$\bar{\sigma}_p \leqq \sigma_{p+1}.$$

One then gets a contradiction. So Rank $(V_{11}) = p$. One also has from Lemma 2.1(1) that Rank $(V_{22}) = d$. This completes the proof of the lemma. □

From Lemma 2.1, we can easily obtain the following results.

COROLLARY 2.1. *Let the* SVD *for* A *and* (A, B) *be as in* (1.5). *For some* $p \leqq n$, *let* V *be partitioned as in* (2.1). *If* V_{11} (*or* V_{22}) *is not of full rank, then*

$$\sigma_{p+d} \leqq \bar{\sigma}_p \leqq \sigma_{p+1}. \tag{2.5}$$

LEMMA 2.2. *Let the* SVD *for* A *and* (A, B) *be as in* (1.5), $p \leqq n$ *be some integer, and* V *be partitioned as in* (2.1). *Then the following two conditions are equivalent*:

$$\sigma_p > \sigma_{p+1} = \cdots = \sigma_{n+d} \quad \text{and} \quad V_{11}, V_{22} \quad \text{are of full rank}; \tag{2.6}$$

$$\bar{\sigma}_p > \sigma_{p+1} = \cdots = \sigma_{n+d}. \tag{2.7}$$

Proof. (2) ⇒ (1). See Lemma 2.1(2) and note that $\sigma_p \geqq \bar{\sigma}_p$.

(1) ⇒ (2). Consider matrix $A^HA - \sigma_{n+1}^2 I$. On the one hand, one has $A^HA - \sigma_{n+1}^2 I = V_{11}DV_{11}^H$ with $D = \operatorname{diag}(\sigma_1^2 - \sigma_{n+1}^2, \cdots, \sigma_p^2 - \sigma_{n+1}^2)$ being nonsingular. Then

$$p = \text{Rank}\,(D) \geqq \text{Rank}\,(V_{11}DV_{11}^H) \geqq \text{Rank}\,(V_{11}^+ V_{11}DV_{11}^H(V_{11}^H)^+) = \text{Rank}\,(D),$$

that is,

$$p = \text{Rank}\,(A^HA - \sigma_{n+1}^2 I). \tag{2.8}$$

On the other hand, by the interlacing (1.8) and the conditions in (2.6), one gets $\bar{\sigma}_{p+1} = \cdots = \bar{\sigma}_n = \sigma_{n+1}$. Then from

$$A^H A - \sigma_{n+1}^2 I = \bar{V}\,\mathrm{diag}\,(\bar{\sigma}_1^2, \cdots, \bar{\sigma}_n^2)\bar{V}^H - \sigma_{n+1}^2 \bar{V}\bar{V}^H$$

$$= \bar{V}\,\mathrm{diag}\,(\bar{\sigma}_1^2 - \sigma_{n+1}^2, \cdots, \bar{\sigma}_p^2 - \sigma_{n+1}^2, 0, \cdots, 0)\bar{V}^H,$$

with $\bar{V}$ being an $n \times n$ unitary matrix, one obtains

(2.9) $\mathrm{Rank}\,(A^H A - \sigma_{n+1}^2 I) = \mathrm{Rank}\,(\mathrm{diag}\,(\bar{\sigma}_1^2 - \sigma_{n+1}^2, \cdots, \bar{\sigma}_p^2 - \sigma_{n+1}^2, 0, \cdots, 0)) = p.$

This implies $\bar{\sigma}_p > \sigma_{n+1}$. □

Remarks. Let us note the following.

(1) Van Huffel obtained results similar to those in Lemma 2.1 and Corollary 2.1 for the case $p = n$ [9, pp. 69–70, p. 27, Thm. 1-5].

(2) In Lemma 2.1, the condition $\bar{\sigma}_p > \sigma_{p+1}$ cannot be relaxed to $\sigma_p > \sigma_{p+1}$. One can easily construct a matrix (A, B) such that $\sigma_p > \sigma_{p+1}$, but neither V_{11} nor V_{22} is of full rank.

Next we discuss various formulas of the minimum norm TLS solution X_{TLS}. Van Huffel [9, p. 30] obtained a formula

$$X_{\mathrm{TLS}} = -(V_{12}Q_2)(V_{22}Q_2)^{-1} (= -V_{12}V_{22}^{+}),$$

where Q_1 and Q_2 are the first $(n - p)$ and the last d columns of an $(n + d - p)$ by $(n + d - p)$ unitary matrix Q such that $V_{22}Q_1 = \varnothing$. Zoltowski [17] also obtained a similar formula:

$$X_{\mathrm{TLS}} = -V_{12}V_{22}^{+} = V_{11}V_{21}^{H}(I_d - V_{21}V_{21}^{H})^{-1}.$$

By applying the extended *CS* decomposition, one can obtain more equivalent formulas. We first discuss the following case.

THEOREM 2.1. *Consider the* TLS *problem* (1.4b). *Let the* SVD *for A and C be given by* (1.5). *Assume that for some* $p \leqq n$, *conditions in* (2.6) (*and so in* (2.7)) *hold. Then the* TLS *problem* (1.4b) *is solvable. Partition V as in* (2.1) *and let* $Q = (Q_1, Q_2) \in C^{(n+d-p)\times(n+d-p)}$ *be any unitary matrix with* $V_{22}Q_2$ *nonsingular, where* Q_1 *and* Q_2 *are the first* $n - p$ *and the last d columns of Q, respectively; then one correction matrix* $(\Delta A, \Delta B)$ *satisfying* (1.4b) *is*

$$(\Delta A, \Delta B) = \sigma_{n+1} U_2 Q_2 ((V_{12}Q_2)^H, (V_{22}Q_2)^H). \tag{2.10}$$

The minimum norm TLS *solution is*

$$\begin{aligned} X_{\mathrm{TLS}} &= (V_{11}^H)^{+} V_{21}^H \\ &= -V_{12}V_{22}^H(I_d - V_{21}V_{21}^H)^{-1} = (A^HA - \sigma_{n+1}^2 I_n)^{+} A^H B \\ &= (A^HA - \sigma_{n+1}^2 V_{12}V_{12}^H)^{+}(A^HB - \sigma_{n+1}^2 V_{12}V_{22}^H), \end{aligned} \tag{2.11}$$

with the correction matrix $(\Delta A, \Delta B)$ *defined in* (2.10) *where* $V_{22}Q_1 = \varnothing$.

Proof. By applying the extended *CS* decomposition (2.2), (2.3), we have

$$(D_{11}^H)^{+} D_{21}^H = -D_{12}D_{22}^{+} = \mathrm{diag}\,(\varnothing_{t\times(d+t-p)}, SC^{-1}, \varnothing_{(n-r)\times(p-r)})$$

with $r = p$. So

$$(V_{11}^H)^{+} V_{21}^H = W_1 (D_{11}^H)^{+} D_{21}^H W_2^H = -W_1 D_{12} D_{22}^{+} W_2^H = -V_{12}V_{22}^{+}.$$

This proves the first formula in (2.11). Because V is a unitary matrix and V_{22} has full row rank d, $V_{22}V_{22}^H = I_d - V_{21}V_{21}^H$ is invertible. So

$$-V_{12}V_{22}^H(I_d - V_{21}V_{21}^H)^{-1} = -V_{12}V_{22}^H(V_{22}V_{22}^H)^+ = -V_{12}V_{22}^+ = X_{\mathrm{TLS}}.$$

This proves the second formula in (2.11). From (2.4) and the condition $\sigma_{p+1} = \cdots = \sigma_{n+d}$, we have

$$\text{(2.12a)} \quad A^HA = V_{11}\Sigma_1^2V_{11}^H + \sigma_{p+1}^2V_{12}V_{12}^H, \qquad A^HB = V_{11}\Sigma_1^2V_{21}^H + \sigma_{p+1}^2V_{12}V_{22}^H.$$

From the proof of (2.8) and the fact that $V_{12}V_{22}^H + V_{11}V_{21}^H = 0$,

$$\text{(2.12b)} \quad A^HA - \sigma_{p+1}^2I = V_{11}DV_{11}^H, \qquad A^HB = V_{11}(\Sigma_1^2 - \sigma_{p+1}^2I)V_{21}^H = V_{11}DV_{21}^H,$$

with $D = \mathrm{diag}\,(\sigma_1^2 - \sigma_{p+1}^2, \cdots, \sigma_p^2 - \sigma_{p+1}^2)$ nonsingular. Also note that

$$\text{(2.12c)} \quad (V_{11}\Sigma_1^2V_{11}^H)^+ = (V_{11}^H)^+\Sigma_1^{-2}V_{11}^+, \qquad (V_{11}DV_{11}^H)^+ = (V_{11}^H)^+D^{-1}V_{11}^+$$

because D, Σ_1^2 are nonsingular and V_{11} has full column rank p. Then the third formula in (2.11) follows from (2.12b) and (2.12c), and the last one follows from (2.12a) and (2.12c). The remaining assertions are the consequence of the results in [2], [3], [9], and [10]. □

In practical applications, when $d > 1$, $\sigma_{n+1}, \cdots, \sigma_{n+d}$ rarely coincide. However, if one considers the TLS problem as an approximation to a true but unknown relation

$$A_0X = B_0,$$

then Rank (A_0, B_0) = Rank $(A_0) \leqq n$, so that $\sigma_{n+1}, \cdots, \sigma_{n+d}$ are just the perturbation of zero. In this case it is realistic to define an error bound ε such that all singular values σ_i, satisfying $|\sigma_i - \sigma_{n+1}| < \varepsilon$, are considered coinciding with σ_{n+1}. For these cases, one may use the formulas of (2.11), but the associated correction matrix may have a Frobenius norm larger than the minimal value $(\sum_{i=1}^d \sigma_{n+i}^2)^{1/2}$, as already pointed out in [9, pp. 31–32]. Specifically, we have Theorem 2.2.

THEOREM 2.2. *Consider* TLS *problem* (1.4b). *Let the* SVD *for* A *and* (A, B) *be given by* (1.5). *Assume that for some integers* p *and* q, *with* $p \leqq n$ *and* $p < q < n + d$,

$$\sigma_p > \sigma_{p+1} = \cdots = \sigma_q > \sigma_{q+1} \geqq \cdots \geqq \sigma_{n+d},$$

and let V *be partitioned as in* (2.1), *in which* V_{22} *is of full rank. If one picks out* $X_{\mathrm{TLS}} = (V_{11}^H)^+V_{21}^H$, *then*

$$\begin{aligned} X_{\mathrm{TLS}} &= (V_{11}^H)^+V_{21}^H = V_{11}V_{21}^H(I_d - V_{21}^HV_{21})^{-1} \\ &= -V_{12}V_{22}^+ = -V_{12}V_{22}^H(I_d - V_{21}V_{21}^H)^{-1} \\ &= (A^HA - V_{12}\Sigma_2^2V_{12}^H)^+(A^HB - V_{12}\Sigma_2^2V_{22}^H), \end{aligned} \qquad \text{(2.13)}$$

with

(2.14)

$$(\Delta A, \Delta B) = U_2\Sigma_2Q_2Q_2^H(V_{12}^H, V_{22}^H), \qquad \sigma_{n+1}\sqrt{d} \geqq \|(\Delta A, \Delta B)\|_F \geqq \sqrt{\sum_{i=1}^d \sigma_{n+i}^2},$$

where Q_1 *and* Q_2 *are the first* $n - p$ *and the last* d *columns of a unitary matrix* $Q = (Q_1, Q_2) \in C^{(n+d-p)\times(n+d-p)}$ *with* $V_{22}Q_1 = 0$.

Proof. The proof of (2.13) is exactly the same as that in Theorem 2.1. Assertion (2.14) is a direct consequence of the results in [2], [3], [9], and [10]. □

Remark. If we insist on minimizing $\|(\Delta A, \Delta B)\|_F$, and if for some $1 \leqq e \leqq d$, $\sigma_p > \sigma_{p+1} = \cdots = \sigma_{n+e} > \sigma_{n+e+1} \geqq \cdots \geqq \sigma_{n+d}$ and $\sigma_{n+1}, \cdots, \sigma_{n+d}$ differ significantly, then we need to keep the last $d - e$ columns of V and pick e columns from the right singular vectors associated with σ_{n+1} to form an $(n+d) \times d$ matrix

$$\begin{pmatrix} W_1 \\ W_2 \end{pmatrix}$$

such that the lower $d \times d$ matrix W_2 is nonsingular. Then a TLS solution can be expressed as $X = -W_1 W_2^{+}$. This strategy has been suggested in [9, p. 31].

3. Comparison of X_{TLS} and X_{LS}. In this section, we compare the minimum norm TLS solution with the LS solution. Because we have already established the formulas for X_{TLS}, the work is rather straightforward. First, we discuss the TLS problem considered in Theorem 2.2.

THEOREM 3.1. *Consider the* TLS *problem* (1.4b). *Assume that the conditions in Theorem* 2.2 *hold. If one picks out* X_{TLS} *as in* (2.13), *then the following hold.*

(1) *If* $p = n$, *then*

$$\|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_n^2} \|X_{\mathrm{TLS}}\|,$$

$$\|B - AX_{\mathrm{TLS}}\| \leqq \|B - AX_{\mathrm{LS}}\| + \frac{\sigma_{n+1}^2}{\bar{\sigma}_n} \|X_{\mathrm{TLS}}\|. \tag{3.1a}$$

(2) *If* $p < n$ *and* Rank $(A) = r > p$, *then*

$$\|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_r^2} \|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\| + \|X_{\mathrm{TLS}}\|,$$

$$\|B - AX_{\mathrm{TLS}}\| \leqq \|B - AX_{\mathrm{LS}}\| + \frac{\sigma_{n+1}^2}{\bar{\sigma}_r} \|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\|. \tag{3.2a}$$

Let $A_p = \sum_{j=1}^{p} \bar{\sigma}_j \bar{u}_j \bar{v}_j^H$ *and* $X_p = A_p^{+} B$; *then*

$$\|X_{\mathrm{TLS}} - X_p\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_p^2} \|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\| + \frac{2\sigma_{n+1}^2}{\bar{\sigma}_p^2 - \sigma_{n+1}^2} \|X_{TLS}\|,$$

$$\|B - AX_{\mathrm{TLS}}\| \leqq \|B - AX_p\| + \frac{\sigma_{n+1}^2}{\bar{\sigma}_p} \|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\| + \sigma_{n+1} \|X_{\mathrm{TLS}}\|. \tag{3.3}$$

(3) *If* $p < n$ *and* Rank $(A) = p$, *then*

$$\|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_p^2} \|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\| + \frac{\sigma_{n+1}^2}{\bar{\sigma}_p^2 - \sigma_{n+1}^2} \|X_{\mathrm{TLS}}\|,$$

$$\|B - AX_{\mathrm{TLS}}\| \leqq \|B - AX_{\mathrm{LS}}\| + \frac{\sigma_{n+1}^2}{\bar{\sigma}_p} \|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\|. \tag{3.4}$$

In (3.2a)–(3.4),

$$\|V_{12}^H X_{\mathrm{TLS}} - V_{22}^H\| = \|V_{22}^{+}\| = \frac{\|X_{\mathrm{TLS}}\|}{\|V_{21}\|}, \tag{3.5}$$

the last formula in (3.5) *holds for* $X_{\mathrm{TLS}} \neq 0$.

Proof. Because $X_{\mathrm{TLS}} = (A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+ (A^H B - V_{12}\Sigma_2^2 V_{22}^H)$, $X_{\mathrm{LS}} = A^+ B = (A^H A)^+ A^H B$, and $X_p = A_p^+ B = (A_p^H A_p)^+ A^H B$, we have from the matrix decomposition (1.10) that

$$\begin{aligned} X_{\mathrm{TLS}} - X_{\mathrm{LS}} &= -((A^H A)^+ - (A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+)(A^H B - V_{12}\Sigma_2^2 V_{22}^H) \\ &\quad -(A^H A)^+ V_{12}\Sigma_2^2 V_{22}^H \\ &= (A^H A)^+ V_{12}\Sigma_2^2 (V_{12}^H X_{\mathrm{TLS}} - V_{22}^H) + (I - A^+ A) X_{\mathrm{TLS}}, \\ A(X_{\mathrm{TLS}} - X_{\mathrm{LS}}) &= (A^H)^+ V_{12}\Sigma_2^2 (V_{12}^H X_{\mathrm{TLS}} - V_{22}^H), \end{aligned} \tag{3.6}$$

and also because $(A_p^H A_p)^+ (A_p^H A_p - A^H A) = \varnothing$,

$$\begin{aligned} X_{\mathrm{TLS}} - X_p &= -((A_p^H A_p)^+ - (A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+)(A^H B - V_{12}\Sigma_2^2 V_{22}^H) \\ &\quad -(A_p^H A_p)^+ V_{12}\Sigma_2^2 V_{22}^H \\ &= (A_p^H A_p)^+ V_{12}\Sigma_2^2 (V_{12}^H X_{\mathrm{TLS}} - V_{22}^H) + (I - A_p^+ A_p) X_{\mathrm{TLS}}, \\ A(X_{\mathrm{TLS}} - X_p) &= (A_p^H)^+ V_{12}\Sigma_2^2 (V_{12}^H X_{\mathrm{TLS}} - V_{22}^H) + (A - A_p) X_{\mathrm{TLS}}. \end{aligned} \tag{3.7}$$

Furthermore, by applying the extended *CS* decomposition (2.2) and (2.3),

$$\begin{aligned} & V_{12}^H X_{\mathrm{TLS}} - V_{22}^H = -V_{22}^+, \\ & \|V_{12}\| = \sqrt{1 - d_p^2} \quad \text{for } p = n \quad \text{and} \quad \|V_{12}\| = 1 \quad \text{for } p < n, \\ & \|V_{21}\| = \sqrt{1 - d_p^2}, \qquad \|V_{11}^+\| = \|V_{22}^+\| = d_p^{-1}, \\ & \|X_{\mathrm{TLS}}\| = \|V_{11}^+\| \|V_{21}\|. \end{aligned} \tag{3.8}$$

Then we have the following results:

(1) If $p = n$, then $I - A^+ A = \varnothing$ and $\|A^+\| = 1/\bar{\sigma}_n$, so (3.1a) immediately follows from (3.6) and (3.8).

(2) If $p < n$ and $r = \mathrm{Rank}\,(A) > p$, then

$$\bar{\sigma}_r \leqq \sigma_{n+1}, \quad \|A^+\| = \frac{1}{\bar{\sigma}_r}, \quad \text{and} \quad \|(I - A^+ A) X_{\mathrm{TLS}}\| \leqq \|X_{\mathrm{TLS}}\|.$$

Then from (3.6) we get (3.2a).

Note that from (1.7),

$$\sigma_j(A^H A - V_{12}\Sigma_2^2 V_{12}^H) \geqq \sigma_j(A^H A) - \|V_{12}\Sigma_2^2 V_{12}^H\| \geqq \bar{\sigma}_j^2 - \sigma_{n+1}^2$$

for $j = 1, \cdots, n$ and $\mathrm{Rank}\,(A^H A - V_{12}\Sigma_2^2 V_{12}^H) = p$. Then

$$\|(A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+\| = \frac{1}{\sigma_p(A^H A - V_{12}\Sigma_2^2 V_{12}^H)} \leqq \frac{1}{\bar{\sigma}_p^2 - \sigma_{n+1}^2}. \tag{3.9}$$

So we have

$$\begin{aligned} & \|(I - A_p^+ A_p) X_{\mathrm{TLS}}\| \\ & \quad = \|(I - A_p^+ A_p)(A^H A - V_{12}\Sigma_2^2 V_{12}^H)(A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+ X_{\mathrm{TLS}}\| \\ & \quad = \|(I - A_p^+ A_p)((A^H A - A_p^H A_p) - V_{12}\Sigma_2^2 V_{12}^H)(A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+ X_{\mathrm{TLS}}\| \\ & \quad \leqq 2\sigma_{n+1}^2 \|(A^H A - V_{12}\Sigma_2^2 V_{12}^H)^+ X_{\mathrm{TLS}}\| \leqq \frac{2\sigma_{n+1}^2}{\bar{\sigma}_p^2 - \sigma_{n+1}^2} \|X_{\mathrm{TLS}}\|. \end{aligned} \tag{3.10}$$

From (3.7), (3.10), and the facts that $\|A_p^+\| = 1/\bar{\sigma}_p$ and $\|A - A_p\| = \bar{\sigma}_{p+1} \leqq \sigma_{n+1}$, the inequalities in (3.3) follow.

(3) When $p < n$ and Rank $(A) = p$, then we have that (similar to the proof of (3.10))

$$\|(I - A^+A)X_{\mathrm{TLS}}\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_p^2 - \sigma_{n+1}^2}\|X_{\mathrm{TLS}}\|. \tag{3.11}$$

So the inequalities in (3.4) also hold. The equalities in (3.5) directly follow from (3.8). This completes the proof of the theorem. □

For the TLS problem considered in Theorem 2.1, we obtain Theorem 3.2.

THEOREM 3.2. *Consider the* TLS *problem* (1.4b) *and assume that the conditions in Theorem* 2.1 *hold. Then*

(1) *If* $\sigma_{n+1} = 0$, *then*

$$X_{\mathrm{TLS}} = X_{\mathrm{LS}}. \tag{3.1b}$$

(2) *If* $\sigma_{n+1} > 0$ *and* $p \leqq n$, *then*

$$\|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_p^2}\|X_{\mathrm{TLS}}\|, \qquad \|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\| \leqq \frac{\sigma_{n+1}^2}{\bar{\sigma}_p^2 - \sigma_{n+1}^2}\|X_{\mathrm{LS}}\|,$$
$$\|B - AX_{\mathrm{TLS}}\| \leqq \|B - AX_{\mathrm{LS}}\| + \frac{\sigma_{n+1}^2}{\bar{\sigma}_p}\|X_{\mathrm{TLS}}\|. \tag{3.2b}$$

Proof. The proof consists of the following:

(1) If $\sigma_{n+1} = 0$, then from (2.11),

$$X_{\mathrm{TLS}} = (A^HA - \sigma_{n+1}^2 I)^+A^HB = (A^HA)^+A^HB = A^+B = X_{\mathrm{LS}}.$$

(2) Choose a unitary matrix $Q = (q_1, \cdots, q_{n-p+d}) \in C^{(n-p+d)\times(n-p+d)}$ such that $V_{22}Q = (\varnothing, \Gamma)$ with $d \times d$ matrix Γ nonsingular. Then from

$$\begin{pmatrix} A^H \\ B^H \end{pmatrix}(A, B)\begin{pmatrix} V_{12} \\ V_{22} \end{pmatrix}q_j = \sigma_{n+1}^2\begin{pmatrix} V_{12} \\ V_{22} \end{pmatrix}q_j, \qquad j = 1, \cdots, n-p,$$

we obtain

$$\begin{aligned} &A^HA(V_{12}q_j) = \sigma_{n+1}^2(V_{12}q_j), \\ &B^HA(V_{12}q_j) = \varnothing, \end{aligned} \qquad j = 1, \cdots, n-p. \tag{3.12}$$

So $V_{12}q_j((1/\sigma_{n+1})AV_{12}q_j)$ are the right (left) singular vectors of A associated with σ_{n+1} for $j = 1, \cdots, n-p$ and

$$B \perp \mathrm{Span}\{\bar{u}_{p+1}, \cdots, \bar{u}_n\}.$$

From this,

$$X_{\mathrm{LS}} = A^+B = \sum_{j=1}^{n} \bar{\sigma}_j^{-1}\bar{v}_j\bar{u}_j^HB = \sum_{j=1}^{p} \bar{\sigma}_j^{-1}\bar{v}_j\bar{u}_j^HB = A_p^+B = X_p, \tag{3.13}$$

where

$$A_p = \sum_{j=1}^{p} \bar{\sigma}_j\bar{u}_j\bar{v}_j^H \quad \text{and} \quad X_p = A_p^+B.$$

Also note that $X_{\mathrm{TLS}} = (A^H A - \sigma_{p+1}^2 I_n)^+ A^H B$ and

$$(A^H A - \sigma_{p+1}^2 I)^+ = \bar{V}\ \mathrm{diag}\,((\bar{\sigma}_1^2 - \sigma_{p+1}^2)^{-1}, \cdots, (\bar{\sigma}_p^2 - \sigma_{p+1}^2)^{-1}, 0, \cdots, 0)\bar{V}^H,$$

$$(A_p^H A_p)^+ = \bar{V}\ \mathrm{diag}\,(\bar{\sigma}_1^{-2}, \cdots, \bar{\sigma}_p^{-2}, 0, \cdots, 0)\bar{V}^H, \tag{3.14}$$

$$I - A_p^+ A_p = \bar{V}\ \mathrm{diag}\,(\varnothing_p, I_{n-p})\bar{V}^H.$$

So $(I - A_p^+ A_p)X_{\mathrm{TLS}} = 0$. Then from (3.7), (3.8), and (3.14),

$$\begin{aligned} X_{\mathrm{TLS}} - X_{\mathrm{LS}} &= X_{\mathrm{TLS}} - X_p = (A_p^H A_p)^+ V_{12}\sigma_{p+1}^2(-V_{22}^+) \\ &= \sigma_{p+1}^2 (A_p^H A_p)^+ X_{\mathrm{TLS}} = \sigma_{p+1}^2 (A^H A - \sigma_{p+1}^2 I)^+ X_{\mathrm{LS}}, \\ A(X_{\mathrm{TLS}} - X_{\mathrm{LS}}) &= \sigma_{p+1}^2 (A_p^H)^+ X_{\mathrm{TLS}}. \end{aligned} \tag{3.15}$$

Then the estimates in (3.2b) follow from (3.15). This completes the proof of the theorem. □

Remarks. We note the following

(1) In practical applications, especially in rank deficient problems where Rank $(A^H A - V_{12}\Sigma_2^2 V_{12}^H) = p$ and $X_{\mathrm{LS}} = X_p$, the difference $\|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\|$ is small. However, if $X_{\mathrm{LS}} \neq X_p$, then we would expect that X_{TLS} is not close to X_{LS}. In this case, $\|X_{\mathrm{TLS}} - X_p\|$ is much smaller than $\|X_{\mathrm{TLS}} - X_{\mathrm{LS}}\|$ when $\sigma_{n+1} \ll \bar{\sigma}_p$. So we should pick out X_P as the minimum norm LS solution rather than X_{LS}. For a detailed discussion of the rank deficient LS problem, see [6], [14], and [15].

(2) Golub and Van Loan [2] obtained similar results for $p = n$ and $d = 1$, while Van Huffel and Vandewalle [8], [12] obtained similar results for $p = n$ and $d \geqq 1$.

4. Perturbation of the TLS solutions. In this section, we derive the perturbation bounds for the TLS solutions, with or without minimal length. We begin with Theorem 4.1.

THEOREM 4.1. *Consider the* TLS *problem* (1.4b). *Assume that the conditions in Theorems* 2.1 *or* 2.2 *hold. Partition* V *as in* (2.1) *and let* $A' \in C^{m\times n}$, $B' \in C^{m\times d}$, *and* $(A', B') = (A, B) + E$ *with* $\|E\| = \eta \leqq \frac{1}{6}(\bar{\sigma}_p - \sigma_{n+1})$, *and the* SVD *for* A', C' *be*

$$(\bar{U}')^H A' \bar{V}' = \bar{\Sigma}', \qquad (U')^H (A', B') V' = \Sigma', \tag{4.1}$$

where $\bar{\Sigma}' = \mathrm{diag}\,(\bar{\sigma}'_1, \cdots, \bar{\sigma}'_n)$ *and* $\Sigma' = \mathrm{diag}\,(\sigma'_1, \cdots, \sigma'_{n+d})$. *Partition* V' *conformally with* V *as in* (2.1), *and replace* V_{ij} *by* V'_{ij} *for* $i, j = 1, 2$. *Define* $X'_{\mathrm{TLS}} = ((V'_{11})^H)^+ (V'_{21})^H$. *Then one has the following estimates*:

$$\begin{aligned} \|X_{\mathrm{TLS}} - X'_{\mathrm{TLS}}\| &\leqq \frac{\eta + \sigma_{n+1}}{\bar{\sigma}_p - \sigma_{n+1}}(3 + 5\|X_{\mathrm{TLS}}\|) \\ &\leqq \frac{6(\eta + \sigma_{n+1})}{\bar{\sigma}_p - \sigma_{n+1}}\sqrt{1 + \|X_{\mathrm{TLS}}\|^2} \leqq \frac{12(\eta + \sigma_{n+1})}{\bar{\sigma}_p - \sigma_{n+1}} \frac{\sigma_1}{\|B\| - \sigma_{n+1}}\|X_{\mathrm{TLS}}\|, \end{aligned} \tag{4.2}$$

where the third inequality holds for $X_{\mathrm{TLS}} \neq \varnothing$.

Proof. By the perturbation theory of the singular values,

$$\bar{\sigma}'_p - \sigma'_{p+1} \geqq \bar{\sigma}_p - \sigma_{p+1} - |\bar{\sigma}_p - \bar{\sigma}'_p| - |\sigma'_{p+1} - \sigma_{p+1}| \geqq \tfrac{2}{3}(\bar{\sigma}_p - \sigma_{p+1}) > 0. \tag{4.3}$$

So V'_{11} and V'_{22} are of full rank from Lemma 2.1, and the formulas in (2.13) can be

used for X'_{TLS}. Let

$$\hat{A} = A - U_2\Sigma_2 V_{12}^H, \qquad \hat{B} = B - U_2\Sigma_2 V_{22}^H,$$
$$\hat{A}' = A' - U_2'\Sigma_2'(V_{12}')^H, \qquad \hat{B}' = B' - U_2'\Sigma_2'(V_{22}')^H. \tag{4.4}$$

Then it is easy to check that X_{TLS} and X'_{TLS} are the minimum norm LS solutions to

$$\hat{A}X = \hat{B}, \qquad \hat{A}'X' = \hat{B}', \tag{4.5}$$

respectively. So $X_{\mathrm{TLS}} = \hat{A}^+\hat{B}$ and $X'_{\mathrm{TLS}} = (\hat{A}')^+\hat{B}'$. From (1.10),

$$\begin{aligned} X_{\mathrm{TLS}} - X'_{\mathrm{TLS}} &= -((\hat{A}')^+ - \hat{A}^+)\hat{B} + (\hat{A}')^+(\hat{B} - \hat{B}') \\ &= (\hat{A}')^+(\hat{A}' - \hat{A})X_{\mathrm{TLS}} + (I - (\hat{A}')^+\hat{A}')X_{\mathrm{TLS}} + (\hat{A}')^+(\hat{B} - \hat{B}'). \end{aligned} \tag{4.6}$$

Note that Rank $(\hat{A})$ = Rank $(\hat{A}') = p$, $\sigma_p(\hat{A}) \geqq \bar{\sigma}_p - \|U_2\Sigma_2 V_{12}^H\| \geqq \bar{\sigma}_p - \sigma_{p+1}$. Similarly, $\sigma_p(\hat{A}') \geqq \bar{\sigma}'_p - \sigma'_{p+1}$. So

$$\|\hat{A}^+\| \leqq \frac{1}{\bar{\sigma}_p - \sigma_{p+1}}, \qquad \|(\hat{A}')^+\| \leqq \frac{1}{\bar{\sigma}'_p - \sigma'_{p+1}} \leqq \frac{3}{2(\bar{\sigma}_p - \sigma_{p+1})}. \tag{4.7}$$

Furthermore, we have from $X_{\mathrm{TLS}} = \hat{A}^H(\hat{A}^H)^+X_{\mathrm{TLS}}$ that

$$(I - (\hat{A}')^+\hat{A}')X_{\mathrm{TLS}} = (I - (\hat{A}')^+\hat{A})(\hat{A} - \hat{A}')^H(\hat{A}^H)^+X_{\mathrm{TLS}},$$
$$\|\hat{A}' - \hat{A}\| \leqq \|A' - A\| + \|U_2\Sigma_2 V_{12}^H\| + \|U_2'\Sigma_2'(V_{12}')^H\| \leqq 2(\eta + \sigma_{p+1}), \tag{4.8}$$
$$\|\hat{B}' - \hat{B}\| \leqq \|B' - B\| + \|U_2\Sigma_2 V_{22}^H\| + \|U_2'\Sigma_2'(V_{22}')^H\| \leqq 2(\eta + \sigma_{p+1}).$$

Substituting these into (4.6), one obtains

$$\begin{aligned} \|X_{\mathrm{TLS}} - X'_{\mathrm{TLS}}\| &\leqq \|(\hat{A}')^+\|\,\|\hat{A}' - \hat{A}\|\,\|X_{\mathrm{TLS}}\| \\ &\quad + \|\hat{A}^+\|\,\|\hat{A}' - \hat{A}\|\,\|X_{\mathrm{TLS}}\| + \|(\hat{A}')^+\|\,\|\hat{B}' - \hat{B}\| \\ &\leqq \frac{\eta + \sigma_{p+1}}{\bar{\sigma}_p - \sigma_{p+1}}(3 + 5\|X_{\mathrm{TLS}}\|). \end{aligned}$$

This is the first inequality of (4.2). Applying the Schwartz inequality we get

$$3 + 5\|X_{\mathrm{TLS}}\| \leqq \sqrt{34(1 + \|X_{\mathrm{TLS}}\|^2)} \leqq 6\sqrt{1 + \|X_{\mathrm{TLS}}\|^2}. \tag{4.9}$$

From this, the second inequality of (4.2) follows. For any $y \in C^m$,

$$\begin{aligned} y^H BB^H y &= y^H B(V_{21}V_{21}^H + V_{22}V_{22}^H)B^H y \\ &= y^H B(V_{21}V_{21}^H + V_{22}Q_2Q_2^H V_{22}^H)B^H y, \end{aligned}$$

where Q_2 is the same as in Theorem 2.1 with $V_{22}Q_1 = \varnothing$. So we get $\|B\|^2 \leqq \|BV_{21}\|^2 + \|BV_{22}Q_2\|^2$, and then

$$\|B\| \leqq \|BV_{21}\| + \|BV_{22}Q_2\|. \tag{4.10}$$

On the other hand, from the decomposition (1.5) for (A, B),

$$-BV_{22}Q_2 = (A, B)\begin{pmatrix} V_{12}Q_2 \\ V_{22}Q_2 \end{pmatrix} - AV_{12}Q_2 = U_2\Sigma_2 Q_2 - AV_{12}Q_2,$$

and so

$$\|BV_{22}Q_2\| \leqq \sigma_{n+1} + \|AV_{12}Q_2\|. \tag{4.11}$$

Note that V_{22} and $V_{22}Q = (\varnothing, V_{22}Q_2)$ have the same set of singular values. From (2.2) and (2.3), the smallest singular value of $V_{22}Q_2$ is $d_p > 0$. So the biggest singular value of $V_{12}Q_2$ is $(1 - d_p^2)^{1/2}$. That is, $\|V_{12}Q_2\| = \|V_{21}\|$ from (3.8). Then from (4.9), (4.10), and (4.11), we have

$$\begin{aligned} \|B\| - \sigma_{n+1} &\leqq \|B\|\|V_{21}\| + \|A\|\|V_{12}Q_2\| \\ &\leqq 2\|(A,B)\|\|V_{21}\| = 2\sigma_1\|V_{21}\|. \end{aligned} \tag{4.12}$$

Now $X_{\mathrm{TLS}} \neq \varnothing$ implies $\|B\| > \sigma_{n+1}$. So

$$\frac{1}{\|V_{21}\|} \leqq \frac{2\sigma_1}{\|B\| - \sigma_{n+1}}$$

and from (3.8), the third inequality of (4.2) holds. □

When $p = n$, Rank $(\hat{A}') = n$. So in (4.8), $I - \hat{A}'^{+}\hat{A}' = \varnothing$. We then obtain a corollary.

COROLLARY 4.1. *If in Theorem* 4.1, $p = n$, *then the following inequalities hold*:

$$\begin{aligned} \|X_{\mathrm{TLS}} - X'_{\mathrm{TLS}}\| &\leqq \frac{3(\eta + \sigma_{n+1})}{\bar{\sigma}_n - \sigma_{n+1}}(1 + \|X_{\mathrm{TLS}}\|) \\ &\leqq \frac{9(\eta + \sigma_{n+1})}{2(\bar{\sigma}_n - \sigma_{n+1})}\sqrt{1 + \|X_{\mathrm{TLS}}\|^2} \\ &\leqq \frac{9(\eta + \sigma_{n+1})}{\bar{\sigma}_n - \sigma_{n+1}} \frac{\sigma_1}{\|B\| - \sigma_{n+1}}\|X_{\mathrm{TLS}}\|. \end{aligned} \tag{4.13}$$

The last inequality holds for $X_{\mathrm{TLS}} \neq \varnothing$.

In rank deficient problems, the TLS solutions are not unique. To obtain X_{TLS}, it is important to determine p. It is possible that in perturbed TLS problems, perturbed singular values satisfy $\bar{\sigma}'_q > \sigma'_{q+1}$ for some integer q, $n \geqq q > p$. In this case, the perturbed minimum norm TLS solution X'_{TLS} is not close to X_{TLS}. However, it is indeed close to a TLS solution to (1.4b). Specifically, we have Theorem 4.2.

THEOREM 4.2. *Consider the* TLS *problem* (1.4b). *Assume that the conditions in Theorems* 2.1 *or* 2.2 *hold. Let* $A' \in C^{m\times n}$, $B' \in C^{m\times d}$, *and* $(A', B') = (A, B) + E$ *with* $\|E\| \leqq \frac{1}{6}(\bar{\sigma}_p - \sigma_{n+1})$, *and the* SVD *for* A' *and* (A', B') *be as in* (4.1). *If for some integer* q, $n \geqq q > p$, $\bar{\sigma}'_q > \sigma'_{q+1}$, *partition* V' *as*

$$V' = \begin{pmatrix} V''_{11} & V''_{12} \\ V''_{21} & V''_{22} \end{pmatrix} \begin{matrix} n \\ d \end{matrix} \tag{4.14}$$

$$q, n + d - q.$$

Let $X' = -V''_{12}(V''_{22})^{+}$. *Then there exists a* TLS *solution* X *to* (1.4b), *such that*

$$\begin{aligned} \|X' - X\| &\leqq \frac{\eta + \sigma_{n+1}}{\bar{\sigma}_p - \sigma_{n+1}}(3 + 5\|X_{\mathrm{TLS}}\| + 2\|X'\|) \\ &\leqq \frac{\eta + \sigma_{n+1}}{\bar{\sigma}_p - \sigma_{n+1}}(6\sqrt{1 + \|X_{\mathrm{TLS}}\|^2} + 2\|X'\|) \\ &\leqq \frac{\eta + \sigma_{n+1}}{\bar{\sigma}_p - \sigma_{n+1}}\left(\frac{12\sigma_1}{\|B\| - \sigma_{n+1}} + 2\right)\|X\|. \end{aligned} \tag{4.15}$$

Proof. Note that both V''_{11} and V''_{22} are of full rank by Lemma 2.1. Also note that

$$((V'_{11})^H,(V'_{21})^H)\begin{pmatrix} X' \\ -I \end{pmatrix} = \varnothing.$$

So

$$X' = X'_{\mathrm{TLS}} + (I-((V'_{11})^H)^+(V'_{11})^H)Y = X'_{\mathrm{TLS}} + (I-(\hat{A}')^+\hat{A}')Y \tag{4.16}$$

for some $Y \in C^{n\times d}$, while $\hat{A}'$ is defined in (4.4). Let

$$X = X_{\mathrm{TLS}} + (I-\hat{A}^+\hat{A})(I-(\hat{A}')^+\hat{A}')Y; \tag{4.17}$$

then X is a TLS solution to (1.4b). So we obtain

$$\begin{aligned} \|X'-X\| &\leqq \|X'_{\mathrm{TLS}}-X_{\mathrm{TLS}}\| + \|\hat{A}^+\hat{A}(I-(\hat{A}')^+\hat{A}')Y\| \\ &= \|X'_{\mathrm{TLS}}-X_{\mathrm{TLS}}\| + \|\hat{A}^+(\hat{A}-\hat{A}')(I-(\hat{A}')^+\hat{A}')Y\|, \end{aligned} \tag{4.18}$$

and the estimates in (4.15) follow from Theorem 4.1 and equations (4.7) and (4.8). □

Remark. We see from Theorems 4.1 and 4.2 that when $\|E\| \leqq \frac{1}{6}(\bar{\sigma}_p - \sigma_{n+1})$, then for any TLS solution X' of the perturbed TLS problem, we can find a TLS solution of the original TLS problem, such that the relative difference

$$\frac{\|X'-X\|}{\|X\|}$$

has the same order as

$$\frac{\|X'_{\mathrm{TLS}}-X_{\mathrm{TLS}}\|}{\|X_{\mathrm{TLS}}\|}.$$

If we just look for a TLS solution, then it is not necessary to find the correct number p, but a number $q \geqq p$, such that V''_{22} is of full rank.

5. Numerical experiments. In this section, we provide numerical experiments to verify the perturbation and comparison results proved in the previous sections. The problem taken is the same as mentioned in [15, Thm. 5.1], which is the basic result of Prony's method for extracting the poles from time-transient data [8]. The time-transient data that we take are

$$f_l = \sum_{j=1}^{12} C_j z_j^l, \tag{5.1}$$

with $l = 0, 1, \cdots, m+n-1$, $z_j = \exp(\lambda_j \Delta T)$ for $j = 1, \cdots, 12$. The λ_j's and C_j's are to be determined. We assume that C_j and z_j are nonzero and z_j are distinct for $j = 1, \cdots, 12$.

Let

$$a_j = \begin{pmatrix} f_{j-1} \\ \vdots \\ f_{j+m-2} \end{pmatrix}, \qquad A_n = (a_1, \cdots, a_n), \tag{5.2}$$

and consider the linear system

$$A_n x = -a_{n+1} = b_n. \tag{5.3}$$

Assume that $m \geqq n$, $m \geqq 12$. Then it is well known [4], [15] that Rank $(A_n) = \min(n, 12)$. So if $n \geqq 12$, then (5.3) is compatible. In this case, for any solution $x = (\alpha_0, \cdots, \alpha_{n-1})^T$, construct a polynomial

$$P_n(z) = z^n + \alpha_{n-1} z^{n-1} + \cdots + \alpha_1 z + \alpha_0, \tag{5.4}$$

then $P_n(z)$ has zeros $z_1, \cdots, z_{12}$.

In the following tests, we did the computations on a FACOM-M340S with double machine precision, and computed the SVD for A_n and (A_n, b_n) and the zeros of the polynomial $P_n(z)$ with SSL2 subroutines DASVD1 and DRJETR, which are based on the algorithms of Golub and Reinsch [1] and Jenkins and Traub [5], respectively.

In all tables, R.E. denotes the relative error, $(E\text{-}k)$ means that the value is in the interval $[.1E-k, 1.E-k)$, x (for $n = 13$) denotes a TLS solution obtained by

$$x_j = -v_{j,k}/v_{n+1,k} \quad \text{for } j = 1, \cdots, n, \qquad |v_{n+1,k}| = \max_{p+1 \leqq l \leqq n+1} |v_{n+1,l}|. \tag{5.5}$$

We took λ_j, C_j as given in Table 1(a) (which are the same values as in [8]), and the sampling rate $\Delta T = 0.1$. We chose $m + n = 80$ for $n = 12, 13$, then computed the

TABLE 1

(a) *Six pairs of poles and residues.*[1]

λ_j	C_j
$-0.082 \pm 0.926i$	1
$-0.147 \pm 2.874i$	1
$-0.188 \pm 4.835i$	1
$-0.220 \pm 6.800i$	1
$-0.247 \pm 8.767i$	1
$-0.270 \pm 10.733i$	1

(b) *True* x_{TLS}.

α_j	$n = 12$	$n = 13$
0	0.7938982298043*E*00	0.7402257380948*E*00
1	−0.7653369105027*E* + 1	−0.6342055198480*E* + 1
2	0.3537685520596*E* + 2	0.2533178915986*E* + 2
3	−0.1035787060425*E* + 3	−0.6119928170028*E* + 2
4	0.2137774135039*E* + 3	0.9574601563874*E* + 2
5	−0.3274778523845*E* + 3	−0.9156088539969*E* + 2
6	0.3816416365930*E* + 3	0.2836241974257*E* + 2
7	−0.3408584282216*E* + 3	0.6382737256781*E* + 2
8	0.2315475588881*E* + 3	−0.1249649342604*E* + 3
9	−0.1166855106588*E* + 3	0.1227507194322*E* + 3
10	0.4141797164398*E* + 2	−0.7806765325649*E* + 2
11	−0.9301418820089*E* + 1	0.3274538698142*E* + 2
12		−0.8369025082069*E* + 1

(c) *The* SVs *of* A_n *and* (A_n, b_n).

	$n = 12$	$n = 13$
σ_1	0.30*E*02	0.30*E*02
σ_{12}	0.39*E* − 4	0.15*E* − 3
$\bar{\sigma}_{12}$	0.76*E* − 5	0.39*E* − 4

[1] Note: The data are taken from Van Blaricum and Mittra [8].

λ_j and C_j. Table 1(b) lists the true x_{TLS} for $n = 12, 13$ and Table 1(c) lists the singular values of A_n and (A_n, b_n) for $n = 12, 13$, respectively.

Since linear system (5.3) is compatible for $n \geqq 12$, we may use the estimates in (4.15) with $\sigma_{n+1} = 0$ (note that from Table 1(b), $\|x\| > 1$):

$$\|x' - x\| \leqq \frac{\eta}{\bar{\sigma}_p}(3 + 5\|x_{\mathrm{TLS}}\| + 2\|x'\|) \leqq \frac{10\eta}{\bar{\sigma}_p}\|x\|. \tag{5.6}$$

Table 2 lists the numerical results in which the errors are introduced by roundoff. Because

TABLE 2

(a) x_{TLS} *with roundoff errors.*

α_j	$n = 12$	R.E.	$n = 13$	R.E.
0	0.79389823049*E*00	(*E* − 8)	0.74022573813*E*00	(*E* − 9)
1	−0.76533691109*E* + 1	(*E* − 8)	−0.63420551987*E* + 1	(*E* − 10)
2	0.35376855230*E* + 2	(*E* − 9)	0.25331789161*E* + 2	(*E* − 9)
3	−0.10357870611*E* + 3	(*E* − 9)	−0.61199281702*E* + 2	(*E* − 10)
4	0.21377741362*E* + 3	(*E* − 9)	0.95746015640*E* + 2	(*E* − 9)
5	−0.32747785254*E* + 3	(*E* − 9)	−0.91560885401*E* + 2	(*E* − 10)
6	0.38164163675*E* + 3	(*E* − 9)	0.28362419742*E* + 2	(*E* − 10)
7	−0.34085842834*E* + 3	(*E* − 9)	0.63827372570*E* + 2	(*E* − 9)
8	0.23154755896*E* + 3	(*E* − 9)	−0.12496493426*E* + 3	(*E* − 11)
9	−0.11668551069*E* + 3	(*E* − 10)	0.12275071943*E* + 3	(*E* − 11)
10	0.41417971651*E* + 2	(*E* − 9)	−0.78067653257*E* + 2	(*E* − 10)
11	−0.93014188209*E* + 1	(*E* − 10)	0.32745386982*E* + 2	(*E* − 10)
12			−0.83690250821*E* + 1	(*E* − 11)

(b) $n = 13$, *True and computed* TLS solutions.

α_j	True x	x with roundoff	R.E.
0	0.8514624213313*E*00	0.85146242138*E*00	(*E* − 10)
1	−0.7414403470421*E* + 1	−0.74144034708*E* + 1	(*E* − 10)
2	0.3028860086281*E* + 2	0.30288600864*E* + 2	(*E* − 10)
3	−0.7571216418515*E* + 2	−0.75712164184*E* + 2	(*E* − 10)
4	0.1256993390451*E* + 3	0.12569933905*E* + 3	(*E* − 11)
5	−0.1374452932055*E* + 3	−0.137445293208*E* + 3	(*E* − 10)
6	0.8183596111829*E* + 2	0.81835961118*E* + 2	(*E* − 11)
7	0.1606815156801*E* + 2	0.16068151569*E* + 2	(*E* − 10)
8	−0.9252175515349*E* + 2	−0.92521755156*E* + 2	(*E* − 10)
9	0.1064013830813*E* + 3	0.10640138308*E* + 3	(*E* − 11)
10	−0.7226439331825*E* + 2	−0.72264393319*E* + 2	(*E* − 10)
11	0.3144212298599*E* + 2	0.31442122986*E* + 2	(*E* − 11)
12	−0.8228910544638*E* + 1	−0.82289105446*E* + 1	(*E* − 11)

(c) R.E. *in computed poles.*

	$n = 12$	$n = 13$ from x_{TLS}	$n = 13$ from x
−0.082 ± 0.926*i*	(*E* − 8)	(*E* − 9)	(*E* − 9)
−0.147 ± 2.874*i*	(*E* − 8)	(*E* − 10)	(*E* − 9)
−0.188 ± 4.835*i*	(*E* − 8)	(*E* − 9)	(*E* − 10)
−0.220 ± 6.800*i*	(*E* − 9)	(*E* − 10)	(*E* − 10)
−0.247 ± 8.767*i*	(*E* − 10)	(*E* − 10)	(*E* − 10)
−0.270 ± 10.733*i*	(*E* − 9)	(*E* − 12)	(*E* − 12)

the (double) machine precision is about 10^{-16}, so $\eta \sim 10^{-16}C(n, m)\sigma_1 \sim 10^{-14}\sigma_1$, where $C(n, m)$ is a constant depending on n, m [3]. Table 2(a) lists the computed x_{TLS} for $n = 12$, 13, and Table 2(b) lists the true and the computed TLS solutions obtained with (5.5). For $n = 12$, $\|\Delta x_{\mathrm{TLS}}\|/\|x_{\mathrm{TLS}}\| = .42E - 9$; for $n = 13$, $\|\Delta x_{\mathrm{TLS}}\|/\|x_{\mathrm{TLS}}\| = .16E - 10$ and $\|\Delta x\|/\|x\| = .19E - 10$. Table 2(c) lists the errors of computed poles that are obtained from $P_n(z)$ with the coefficients listed in Table 2(a,b).

Table 3 lists the numerical results with f_j's containing normally distributed perturbation with mean zero and standard deviation 10^{-10}. We computed $\eta = \|(\Delta A_n, \Delta b_n)\|$

TABLE 3

(a) x_{TLS} *with error* $.12E - 8$.

α_j	$n = 12$	R.E.	$n = 13$	R.E.
0	0.79389811208*E*00	$(E - 6)$	0.74022572934*E*00	$(E - 7)$
1	−0.76533681778*E* + 1	$(E - 6)$	−0.63420551525*E* + 1	$(E - 8)$
2	0.35376851622*E* + 2	$(E - 7)$	0.25331789091*E* + 2	$(E - 8)$
3	−0.10357869715*E* + 3	$(E - 7)$	−0.61199281851*E* + 2	$(E - 8)$
4	0.21377739789*E* + 3	$(E - 7)$	0.95746016595*E* + 2	$(E - 7)$
5	−0.32747783211*E* + 3	$(E - 7)$	−0.91560887770*E* + 2	$(E - 7)$
6	0.38164161678*E* + 3	$(E - 7)$	0.28362423499*E* + 2	$(E - 7)$
7	−0.34085841367*E* + 3	$(E - 7)$	0.63827368354*E* + 2	$(E - 7)$
8	0.23154755103*E* + 3	$(E - 8)$	−0.12496493082*E* + 3	$(E - 8)$
9	−0.11668550769*E* + 3	$(E - 7)$	0.12275071740*E* + 3	$(E - 8)$
10	0.41417970939*E* + 2	$(E - 7)$	−0.7806752428*E* + 2	$(E - 7)$
11	−0.93014187404*E* + 1	$(E - 7)$	0.32745386769*E* + 2	$(E - 8)$
12			−0.83690250563*E* + 1	$(E - 8)$

(b) $n = 13$, *True and computed* TLS *solutions.*

α_j	True x	x with error $.12E - 8$	R.E.
0	0.8464494801999*E*00	0.84644941241*E*00	$(E - 7)$
1	−0.7366077517118*E* + 1	−0.73660769990*E* + 1	$(E - 7)$
2	0.3006521946970*E* + 2	0.30065217621*E* + 2	$(E - 7)$
3	−0.7505813330768*E* + 2	−0.75058129310*E* + 2	$(E - 7)$
4	0.1243494763689*E* + 3	0.12434947079*E* + 3	$(E - 8)$
5	−0.1353774876176*E* + 3	−0.13537748292*E* + 3	$(E - 8)$
6	0.7942614713242*E* + 2	0.79426145976*E* + 2	$(E - 7)$
7	0.1822044662383*E* + 2	0.18220443849*E* + 2	$(E - 7)$
8	−0.9398382451989*E* + 2	−0.93983820023*E* + 2	$(E - 7)$
9	0.1071381747428*E* + 3	0.10713817112*E* + 3	$(E - 8)$
10	−0.7252592035766*E* + 2	−0.72525918559*E* + 2	$(E - 7)$
11	0.3150085528093*E* + 2	0.31500854752*E* + 2	$(E - 7)$
12	−0.8235224881845*E* + 1	−0.82352248099*E* + 1	$(E - 8)$

(c) R.E. *in computed poles.*

	$n = 12$	$n = 13$ from x_{TLS}	$n = 13$ from x
−0.082 ± 0.926*i*	$(E - 5)$	$(E - 6)$	$(E - 5)$
−0.147 ± 2.874*i*	$(E - 5)$	$(E - 6)$	$(E - 5)$
−0.188 ± 4.835*i*	$(E - 5)$	$(E - 7)$	$(E - 6)$
−0.220 ± 6.800*i*	$(E - 5)$	$(E - 6)$	$(E - 6)$
−0.247 ± 8.767*i*	$(E - 6)$	$(E - 7)$	$(E - 6)$
−0.270 ± 10.733*i*	$(E - 7)$	$(E - 8)$	$(E - 8)$

and found that $\eta = .12E - 8$ in all three cases. Table 3(a) lists the computed x_{TLS} for $n = 12, 13$, and Table 3(b) lists the true and the computed TLS solutions obtained with (5.5). For $n = 12$, $\|\Delta x_{\mathrm{TLS}}\| / \|x_{\mathrm{TLS}}\| = .54E - 7$; for $n = 13$, $\|\Delta x_{\mathrm{TLS}}\| / \|x_{\mathrm{TLS}}\| = .32E - 7$ and $\|\Delta x\| / \|x\| = .40E - 7$. Table 3(c) lists the errors of computed poles that are obtained from $P_n(z)$ with the coefficients listed in Table 3(a,b).

Note that the relative errors of the computed TLS solutions match the bound in (5.6). On the other hand, comparing the TLS solutions with the LS solutions presented in [15], we observe that the two sets of solutions have the same order of accuracy. Also note that when $n = 12$, A_n is of full rank, while when $n = 13$, A_n is rank deficient.

6. Concluding remarks. In this paper we discuss TLS problem $AX \approx B$ in which (A, B) may have multiple smallest singular values. In particular, rank deficient problems belong to this class. The results of this paper generalize those of full rank TLS problems [2], [3], [9], [10], [12], [17] to the rank deficient case. Various formulas for the minimum norm TLS solution are given. The relationship and the difference between the minimum norm TLS and the minimum norm LS solutions are obtained. If the original TLS problem is slightly perturbed, the perturbation bounds for the TLS solutions with or without minimal length are deduced. It is proved that the upper bound for the relative difference

$$\frac{\|X' - X\|}{\|X\|}$$

has the same order as

$$\frac{\|X'_{\mathrm{TLS}} - X_{\mathrm{TLS}}\|}{\|X_{\mathrm{TLS}}\|},$$

where X_{TLS} and X'_{TLS} are the minimum norm TLS solutions of the original and perturbed problems, respectively, while X and X' are also TLS solutions of the same problems but different from X_{TLS} and X'_{TLS}. In a practical example the perturbation bounds proven in the paper are verified.

For a detailed discussion of the algebraic relations between the TLS and LS problems with more than one solution, see [16].

Acknowledgment. We thank Dr. P. Van Dooren and the referees, who suggested using the extended CS decomposition to simplify the discussion and who made other useful suggestions. One referee also provided related material.

REFERENCES

[1] G. H. GOLUB AND C. REINSCH, *Singular value decomposition and least squares solutions*, Numer. Math., 14 (1970), pp. 403–420.

[2] G. H. GOLUB AND C. F. VAN LOAN, *An analysis of the total least squares problem*, SIAM J. Numer. Anal., 17 (1980), pp. 883–893.

[3] ———, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[4] G. HEINIG AND K. ROST, *Algebraic Methods for Toeplitz-Like Matrices and Operators*, Birkhäuser, Boston, 1984.

[5] M. A. JENKINS AND J. F. TRAUB, *A three-stage algorithm for real polynomials using quadratic iteration*, SIAM J. Numer. Anal., 7 (1970), pp. 543–566.

[6] C. L. LAWSON AND R. J. HANSON, *Solving Least Squares Problems*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

[7] C. C. PAIGE AND M. A. SAUNDERS, *Towards a generalized singular value decomposition*, SIAM J. Numer. Anal., 18 (1981), pp. 398–405.

[8] M. L. VAN BLARICUM AND R. MITTRA, *Problems and solutions associated with Prony's method for processing transient data*, IEEE Trans. Antennas and Propagation, AP-26 (1978), pp. 174–182.

[9] S. VAN HUFFEL, *Analysis of the total least squares problem and its use in parameter estimation*, Ph.D. thesis, ESAT Laboratory, Department of Electrical Engineering, Katholieke Universität Leuven, Leuven, Belgium, June 1987.

[10] S. VAN HUFFEL AND J. VANDEWALLE, *Analysis and solution of the nongeneric total least squares problem*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 360–372.

[11] ———, *The partial total least squares algorithm*, J. Comput. Appl. Math., 21 (1988), pp. 333–341.

[12] ———, *Algebraic connections between the least squares and total least squares problems*, Numer. Math., 55 (1989), pp. 431–449.

[13] ———, *Analysis and properties of the generalized total least squares problem $AX \approx B$ when some or all columns in A are subject to error*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 294–315.

[14] P. Å. WEDIN, *Perturbation theory for pseudo-inverses*, BIT, 13 (1973), pp. 217–232.

[15] M. WEI, *The perturbation of consistent least squares problems*, Linear Algebra Appl., 112 (1989), pp. 231–245.

[16] ———, *Algebraic relations between the total least squares and least squares problems with more than one solution*, Numer. Math., to appear.

[17] M. D. ZOLTOWSKI, *Generalized minimum norm and constrained total least squares with applications to array processing*, in Proc. SPIE Signal Processing III, San Diego, CA, 975 (1988), pp. 78–85.

SPECIAL SECTION ON

Iterative Methods in Numerical Linear Algebra

JULY 1992 Volume 13, Number 3

INTRODUCTION

More than 180 mathematicians from all corners of the world attended the First Copper Mountain Conference on Iterative Methods. The meeting, held April 1–5, 1990, took place at the Copper Mountain Resort, which is located 70 miles west of Denver. During the four days of the meeting, over 100 talks on current research were presented. Topics included nonsymmetric systems, preconditioning strategies, parallel implementations, applications, software, multigrid methods, domain decomposition, eigenvalue problems, integral equations, nonlinear systems, indefinite problems, discretization techniques, complex matrix problems, and common software standards.

There are two special issues devoted to chronicling the presentations made at the Copper Mountain Conference, one in *SIMAX* and the other in the *SIAM Journal on Scientific and Statistical Computing* (*SISSC*). The review process followed the normal SIAM policies for selecting referees and making recommendations. This issue represents a rich mix of papers on a wide variety of topics related to iterative methods. There are two aspects of this collection that we would like to underscore. First, much of the research represented in these articles was motivated or influenced by the need to develop new algorithms for the growing variety of parallel processing computers. Second, the increasing interaction between the multigrid community and the iterative method community is

reflected in the many articles that incorporate multigrid and multilevel ideas into the construction of preconditioners and domain decomposition strategies. We also remark that the articles in this issue reflect the new and exciting work on iterative methods for nonsymmetric linear systems that was presented at the meeting.

A special effort was made to bring students to the meeting. The vehicle for this effort was a Student Paper Competition, in which students were asked to submit an original research paper consisting primarily of their own work. Out of ten submissions, three winners were selected. First place went to Barry Smith of the Courant Institute at New York University. Second place was awarded to Doug James of North Carolina State University. Third place honors were shared by Sverker Holmgren and Kurt Otto from Uppsala University in Stockholm, Sweden. Barry Smith's paper appeared in the special issue of *SISSC*; the other two winning papers appear in this issue.

We would like to thank the members of the program committee for their help in organizing the meeting. They are: Seymour Parter (chair), Loyce Adams, Steve Ashby, Howard Elman, Roland Freund, Anne Greenbaum, David Kincaid, Steve McCormick, Ahmed Sameh, Paul Saylor, Olof Widlund, and David Young. In particular, we would like to give special thanks to Howard Elman and Anne Greenbaum, who, in addition to helping to organize the meeting, acted as special *SIMAX* editors for this issue. Through their efforts, the articles contained in this special issue were carefully refereed and brought into print on schedule. We would also like to thank the following persons for their generous support of this meeting: Fred Howes of the Applied Mathematics Program of the National Science Foundation, Don Austin from the Applied Mathematical Sciences Program of the Department of Energy, Andy White from the Advanced Computing Laboratory at Los Alamos National Laboratories, and Bob Huddleston of the Computing Division of Lawrence Livermore National Laboratories. Without their help, this meeting could not have taken place.

As this issue goes to press, planning for the next Copper Mountain Conference on Iterative Methods is in its final stages. It will be held April 9–16, 1992, in Copper Mountain, Colorado. Plans again include special journal issues in *SISSC* and *SIMAX*. It is our hope that the lively interaction and the fine quality of presentations and papers that marked the first meeting can be duplicated at the upcoming meeting.

Thomas A. Manteuffel
Gene H. Golub

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 767–777, July 1992

006

CIRCULANT AND SKEWCIRCULANT MATRICES FOR SOLVING TOEPLITZ MATRIX PROBLEMS*

THOMAS HUCKLE†

Abstract. In recent papers, numerous authors studied the solutions of symmetric positive definite Toeplitz systems $Tx = b$ by the conjugate gradient method for different families of circulant preconditioners C. In this paper new circulant/skewcirculant approximations are introduced to T and their properties are studied. The main interest is directed to the skewcirculant case. Furthermore, algorithms for computing the eigenvalues of T are formulated, based on the Lanczos algorithm and Rayleigh quotient iteration. For some numerical examples the spectra of $C^{-1}T$ are compared and the behaviour of the introduced eigenvalue algorithms is displayed.

Key words. Toeplitz matrix, circulant matrix, preconditioned conjugate gradient method, Lanczos algorithm, Rayleigh quotient iteration

AMS(MOS) subject classifications. 65F10, 65F15

0. Introduction. In recent papers, Strang and R. Chan [17], [5], [3], [4] and T. Chan [6] studied the use of circulant matrices C for solving systems of linear equations $T_n x = b$ with

$$T_n := T(t_0, t_1, \cdots, t_{n-1}) := \begin{pmatrix} t_0 & t_1 & \cdots & \cdots & t_{n-1} \\ t_1 & t_0 & t_1 & & \vdots \\ \vdots & t_1 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & \ddots & t_1 \\ t_{n-1} & \cdots & \cdots & t_1 & t_0 \end{pmatrix}$$

a symmetric positive definite Toeplitz matrix. Thereby a symmetric circulant matrix C is defined by $C = T(c_0, c_1, \cdots, c_{k-1}, c_k, c_{k-1}, \cdots, c_1)$ for even n and $C = T(c_0, c_1, \cdots, c_k, c_k, \cdots, c_1)$ for odd n with $k = \lfloor n/2 \rfloor$.

Applying the preconditioned conjugate gradient method [12], [10] to solving the system $T_n x = b$, we must find a matrix C such that the eigenvalues of $C^{-1}T_n$ are clustered and $Cy = d$ can be solved very fast. For circulant C the second condition is fulfilled by a Fast Fourier Transform (FFT) (see, e.g., [8]). In [17], Strang proposed $C_s := T(t_0, t_1, t_2, \cdots, t_2, t_1)$, and with Chan he proved that if the underlying generating function f, the Fourier coefficients of which give the entries of T_n, is a positive function in the Wiener class, then for n sufficiently large, C_n and C_n^{-1} are uniformly bounded in the l_2 norm and the eigenvalues of $C_s^{-1} T_n$ are clustered around 1 [5].

Chan introduced

$$C_f := T\left(t_0, \frac{(n-1)t_1 + t_{n-1}}{n}, \cdots, \frac{t_{n-1} + (n-1)t_1}{n}\right).$$

C_f minimizes $\|C - T_n\|_F$ over all symmetric circulant matrices, where $\| \quad \|_F$ denotes the Frobenius norm. The spectrum of C_f is again clustered around 1 (see [6]).

In §§ 1 and 2 we consider circulant approximations to T_n proposed in [11], [17], [6], and [4] and skewcirculant approximations, which can be defined in the same way,

* Received by the editors April 5, 1990; accepted for publication (in revised form) January 3, 1991.

† Institut für Angewandte Mathematik und Statistik, Universität Würzburg, Am Hubland, D-8700 Würzburg, Germany (huckle@vax.rz.uni-wuerzburg.dbp.de).

and study some of their properties and relationships. Thereby a symmetric matrix S is said to be skewcirculant if $S = T(s_0, s_1, \cdots, s_k, -s_k, \cdots, -s_1)$ for odd n and $S = T(s_0, s_1, \cdots, s_{k-1}, 0, -s_{k-1}, \cdots, -s_1)$ for even n [8], [9]. If T_n is indefinite any circulant approximation will generally be indefinite, but preconditioning is only possible for positive definite C. However, from C_s and C_f we can always construct positive definite circulant matrices such that the spectrum of $C^{-1}T_n$ is approximately clustered around ± 1.

Every $n \times n$ circulant matrix C has the same eigenvectors. Thus approximating T_n by C can be considered as approximating the eigenvalues of T_n and ordering the eigenvectors of C such that they are "near" to those of T_n. Therefore these eigenvectors may be efficient start vectors in iterative algorithms for determining any, especially the minimal eigenvalue of T_n in $O(n \log(n))$ arithmetic operations. In § 3 we give different eigenvalue algorithms based on Rayleigh quotient iteration, proposed in [17], and the Lanczos algorithm [14], [15].

Finally, for some numerical examples we compare the clustering properties of the circulant/skewcirculant approximations and test the eigenvalue algorithms proposed in § 3.

1. A new circulant/skewcirculant approximation to T. The purpose of this section is to derive estimations for the minimum eigenvalues of a real symmetric Toeplitz matrix T using a representation $T = C + S$ with C a circulant and S a skewcirculant matrix. For symmetric Toeplitz matrices T, T_1, T_2 let us define a scalar product by $(T_1, T_2)_E := e_1^T T_1 T_2 e_1$ with $e_1 = (1, 0, \cdots, 0)^T$, and the corresponding norm for symmetric Toeplitz matrices

$$\|T\|_E^2 := \|Te_1\|_2^2 = \sum_{i=0}^{n-1} t_i^2.$$

Similar to the best circulant Frobenius norm approximation to T, we can search for a circulant matrix C_e with

$$\|C_e - T\|_E = \min_{C \text{ circulant}} \|C - T\|_E.$$

The solution to this problem is

$$C_e := T\left(t_0, \frac{t_1 + t_{n-1}}{2}, \cdots, \frac{t_{n-1} + t_1}{2}\right).$$

In the same way we can define the skewcirculant matrix

$$S_e := T\left(t_0, \frac{t_1 - t_{n-1}}{2}, \cdots, \frac{t_{n-1} - t_1}{2}\right),$$

which solves

$$\min_{S \text{ skewcirculant}} \|S - T\|_E.$$

Then for $C_a := C_e + (a - t_0)I$ and $S_b := S_e + (b - t_0)I$ with $a + b = t_0$ it holds that

$$C_a + S_b = T. \tag{1}$$

Let us define

$$\mathbf{T} := \{T/T \text{ real symmetric } n\times n \text{ Toeplitz matrix}\},$$

$$\mathbf{C} := \{C\in\mathbf{T}/C \text{ circulant}\}, \mathbf{C_0} := \{C\in\mathbf{C}/c_0 = 0\},$$

$$\mathbf{S} := \{S\in\mathbf{T}/S \text{ skewcirculant}\}, \mathbf{S_0} := \{S\in\mathbf{S}/s_0 = 0\}.$$

Then the matrices $T(0, 1, 0, \cdots, 0, 1)$, $T(0, 0, 1, 0, \cdots, 0, 1, 0)$, $\cdots$, form a basis of $\mathbf{C_0}$, and $T(0, 1, 0, \cdots, 0, -1)$, $T(0, 0, 1, 0, \cdots, 0, -1, 0)$, $\cdots$, form a basis of $\mathbf{S_0}$, and it holds that

$$\mathbf{C}\perp\mathbf{S_0}, \quad \mathbf{C_0}\perp\mathbf{S}, \quad \mathbf{C}\oplus\mathbf{S_0} = \mathbf{T} = \mathbf{C_0}\oplus\mathbf{S}$$

relative to $(.,.)_E$, and thus every real symmetric Toeplitz matrix can be uniquely represented in the form $T = C_0 + S_0 + t_0 * I$ with $C_0 \in \mathbf{C_0}$ and $S_0 \in \mathbf{S_0}$.

Now (1) can be used to derive estimations for the minimal eigenvalue $\mu_1(T)$. All eigenvectors of T can be chosen to be reciprocal or antireciprocal; this means $Jx = x$ or $Jx = -x$ with

$$J = \begin{pmatrix} & & 1 \\ & \iddots & \\ 1 & & \end{pmatrix}. \tag{2}$$

For the following, let us assume that the eigenvector x_T, corresponding to $\mu_1(T)$, is reciprocal and unique up to a scalar factor. Following Kato [13, Chap. I., eq. (6.79) and Chap. II, Problem (6.2)] it holds that

$$\mu_1(T) = \mu_1(C_a + S_b) \geqq \mu_1(C_a) + \mu_1(S_b) = \mu_1(C_e) + \mu_1(S_e) - t_0.$$

Furthermore, for

$$T(x) := 2S_b + (x-1)(S_b - C_a) = 2C_a + (x+1)(S_b - C_a) = (1+x)S_b + (1-x)C_a$$

the minimal eigenvalue $\mu_1(T(x)) =: \mu_1(x)$ is a concave, piecewise, holomorphic function of x, and $T(-1) = 2C_a$, $T(0) = T$, and $T(1) = 2S_b$. Denote by x_S, respectively x_C, the normalized reciprocal eigenvector corresponding to $\mu_1(1)$ and $T(1)$, respectively, $\mu_1(-1)$ and $T(-1)$. Then we have the series expansions

$$\mu_1(x) = 2\mu_1(S_b) + (x-1)x_S^T(S_b - C_a)x_S + \cdots,$$

$$\mu_1(x) = 2\mu_1(C_a) + (x+1)x_C^T(S_b - C_a)x_C + \cdots.$$

Hence $\mu_1(1)$ is the maximum of $\mu_1(x)$ if and only if $x_S^T(S_b - C_a)x_S = 0$; this implies that

$$b = (x_S^T C_e x_S + t_0 - \mu_1(S_e))/2, \qquad a = t_0 - b. \tag{3}$$

Since $\mu_1(x)$ is concave this gives an upper bound for $\mu_1(T)$:

$$\mu_1(T) \leqq 2\mu_1(S_b) = \mu_1(S_e) + x_S^T C_e x_S - t_0,$$

with b defined by (3). Similarly, $\mu_1(-1)$ is the maximum if and only if $x_C^T(S_b - C_a)x_C = 0$; this gives

$$\mu_1(T) \leqq \mu_1(C_e) + x_C^T S_e x_C - t_0.$$

Thus we have proved the following theorem.

THEOREM 1. *For the minimum eigenvalue of a symmetric Toeplitz matrix T with circulant and skewcirculant approximations C_e and S_e, it holds that*

$$(4)\quad \mu_1(C_e)+\mu_1(S_e)-t_0 \leqq \mu_1(T) \leqq \min\{\mu_1(S_e)+x_S^T C_e x_S, \mu_1(C_e)+x_C^T S_e x_C\}-t_0.$$

Thereby, x_T, x_C, and x_S are assumed to be unique reciprocal (antireciprocal) eigenvectors of T, C_e, and S_e, corresponding to $\mu_1(T)$.

These bounds can be used as an initial interval in the Cybenko–Van Loan algorithm for computing the minimum eigenvalue of a symmetric positive definite Toeplitz matrix [7] or in the iterative eigenvalue estimations considered in § 3. By FFT they can be determined in $O(n \log(n))$ operations. The lower bound in (4) seems to be an especially useful completion of the eigenvalue inclusions given in [7].

In addition, C_e is connected with the circulant approximation $C_c := T(t_0, t_1 + t_{n-1}, \cdots, t_{n-1} + t_1)$ introduced by Chan in [4] by the equation $C_c = 2*C_{t_0/2} = 2*C_e - t_0*I$.

2. Circulant and skewcirculant preconditioners for Toeplitz systems. The Toeplitz property gives no reason to prefer circulant to skewcirculant matrices. Hence for preconditioning the system $T_n x = b$, the skewcirculant approximations S_s and S_f should also be considered, where $S_s := T(t_0, t_1, t_2, \cdots, -t_2, -t_1)$ and

$$S_f := T\left(t_0, \frac{(n-1)t_1 - t_{n-1}}{n}, \cdots, \frac{t_{n-1}-(n-1)t_1}{n}\right),$$

with

$$\|S_f - T_n\|_F = \min_{S \text{ skewcirculant}} \|S - T_n\|_F.$$

Therefore let us first extend the results for circulant approximations derived in [5], [6], [3], and [4] to the skewcirculant case.

Let $T = T(t_0, t_1, \cdots)$ be a real single infinite positive definite Toeplitz matrix in the Wiener class. This means $0 < \sum_{k=-\infty}^{\infty} t_k e^{ik\Theta}$ and $\sum_{k=0}^{\infty} |t_k| < M < \infty$, and T_n is a finite section of T. Following [5], for $n = 2m$ we can partition T_n, C_s, and S_s in the form

$$T_n = \begin{pmatrix} D & R \\ R^T & D \end{pmatrix}, \quad C_s = \begin{pmatrix} D & R_c \\ R_c^T & D \end{pmatrix}, \quad S_s = \begin{pmatrix} D & R_s \\ R_s^T & D \end{pmatrix},$$

with $D = T(t_0, \cdots, t_{m-1})$, $R_c = T(t_m, \cdots, t_1)$; R has diagonals $t_1, \cdots, t_{n-1}$, and R_s has diagonals $t_1, \cdots, t_{m-1}, 0, -t_{m-1}, \cdots, -t_1$. Symmetric Toeplitz matrices are centrosymmetric [2]. Thus using the matrix

$$Q = \frac{1}{\sqrt{2}} \begin{pmatrix} I & I \\ -J & J \end{pmatrix},$$

with J defined by (2), the eigenvalue problems $T_n x = \lambda C_s x$ and $T_n x = \sigma S_s x$ split into

$$(5)\qquad (D-RJ)y = \lambda_-(D-R_cJ)y \quad \text{and} \quad (D+RJ)z = \lambda_+(D+R_cJ)z$$

and

$$(6)\qquad (D-RJ)y = \sigma_-(D-R_sJ)y \quad \text{and} \quad (D+RJ)z = \sigma_+(D+R_sJ)z.$$

Defining $H_c := (R_c - R)J$ and $H_s := (R_s - R)J$, we get

$$H_c = \begin{pmatrix} h_1 & h_2 & \cdot & 0 \\ h_2 & & & 0 \\ \cdot & & & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \quad \text{and} \quad H_s = \begin{pmatrix} g_1 & g_2 & \cdot & g_m \\ g_2 & & & 0 \\ \cdot & & & 0 \\ g_m & 0 & 0 & 0 \end{pmatrix},$$

with $h_j = t_j - t_{n-j}$, $g_j = -t_j - t_{n-j}$ for $j \leqq m$. Thus (5) and (6) are equivalent to

$$(7) \qquad \frac{(1-\lambda_+)}{\lambda_+}(D+RJ)z = H_c z, \qquad \frac{(1-\lambda_-)}{\lambda_-}(D-RJ)y = -H_c y$$

and

$$(8) \qquad \frac{(1-\sigma_+)}{\sigma_+}(D+RJ)z = H_s z, \qquad \frac{(1-\sigma_-)}{\sigma_-}(D-RJ)y = -H_s y.$$

As the order n increases, D, H_c, and H_s approach singly infinite Toeplitz and Hankel matrices $\tilde{D} = T(t_0, t_1, t_2, \cdots)$,

$$\tilde{H}_c = \begin{pmatrix} t_1 & t_2 & t_3 & \cdot \\ t_2 & t_3 & \cdot & \\ t_3 & \cdot & & \\ \cdot & & & \end{pmatrix},$$

$\tilde{H}_s = -\tilde{H}_c$, and RJ converges against 0. In [5] Strang and Chan proved that the clustering of $C_s^{-1}T_n$ around 1 depends on the clustering around 0 of the eigenvalues of the infinite Hankel–Toeplitz problem

$$\tilde{H}_c\tilde{z} = \tilde{\nu}\tilde{D}\tilde{z} \quad \text{and} \quad \tilde{H}_c\tilde{y} = -\tilde{\mu}\tilde{D}\tilde{y},$$

which is the limit of (7) for $n \to \infty$. In the same way the clustering of $S_s^{-1}T_n$ depends on the eigenvalue distribution of

$$\tilde{H}_s\tilde{z} = \tilde{\nu}\tilde{D}\tilde{z} \quad \text{and} \quad \tilde{H}_s\tilde{y} = -\tilde{\mu}\tilde{D}\tilde{y},$$

the limit of (8). In view of $\tilde{H}_s = -\tilde{H}_c$, for large n the eigenvalues of $S_s^{-1}T_n$ are clustered around 1 in the same way as the eigenvalues of $C_s^{-1}T_n$.

Furthermore, the eigenvalues of the skewcirculant matrix S_s are of the form

$$\lambda_j = t_0 + t_1\sigma w^j + \cdots + t_{m-1}(\sigma w^j)^{m-1} + 0 - t_{m-1}(\sigma w^j)^{m+1} - \cdots - t_1(\sigma w^j)^{n-1}$$

$$= t_0 + t_1(\sigma w^j + (\sigma w^j)^{-1}) + \cdots = \sum_{k=1-m}^{m-1} t_k e^{(2j+1)k\pi i/n},$$

with $\sigma = e^{i\pi/n}$ and $w = \sigma^2$ (see [8]); thus S_s and S_s^{-1} are positive definite and uniformly bounded for positive definite T in the Wiener class and n large enough (see [5, eq. (3)] for the circulant case). Hence we have proved the following.

THEOREM 2. *Suppose T is real and positive and in the Wiener class. Then for large n the skewcirculants S_s and S_s^{-1} are uniformly bounded and positive definite, and the eigenvalues of $S_s^{-1}T_n$ are clustered around* 1.

In [3] and [4] Chan proved further properties of C_s and C_f. It can be seen immediately that all results also hold for the skewcirculant approximations S_s and S_f. Hence S_s and S_f are asymptotically equal, and S_s minimizes $\|S - T_n\|_1 = \|S - T_n\|_\infty$ over all possible symmetric skewcirculant matrices S. Note that C_e of § 1 is not asymptotically equal to C_s.

In order to show that skewcirculant matrices are efficient preconditioners, it remains to prove that the system $Sy = d$ can be solved very fast. Every skewcirculant matrix S can be written in the form $S = \Delta^H C \Delta$ with a circulant matrix C and a diagonal matrix Δ [8, p. 85]. Thus the linear equation $Sy = d$ can be solved via FFT, also.

All in all, the skewcirculant approximations to T_n have the same properties as the circulant approximations and are efficient preconditioners. It depends on the special structure of T_n which of the two alternatives is preferable, e.g., in some examples the skewcirculant matrix S_s is positive definite while C_s is indefinite ([5, example 1]), and in some other examples the eigenvalues of $S_s^{-1} T_n$ are better clustered than that of $C_s^{-1} T_n$.

In [11, Chap. 7.6] Szegö and Grenander have introduced another family of circulant matrices that approximates a Hermitian–Toeplitz matrix T_n and can be used for preconditioning. In the following, we consider only the case of real symmetric T_n. Set $U := (u_1, \cdots, u_n) := (u_{k,j})_{k,j=1}^n$ with $u_{k,j} := e^{2\pi ikj/n}/\sqrt{n}$ and $D := \operatorname{diag}(d_1, \cdots, d_n)$ with

$$d_k := t_0 + 2 \sum_{j=1}^{p} \left(1 - \frac{j}{p}\right) t_j \cos\left(\frac{2\pi jk}{n}\right) \tag{9}$$

for $1 \leqq p \leqq n$. Then $C_g := U^H D U$ is a circulant matrix with double eigenvalues $d_k = d_{n-k}$, $k = 1, \cdots, \lceil n/2 \rceil$, and one or two simple eigenvalues d_n and $d_{n/2}$ (for even n). For positive definite T in the Wiener class, (9) shows that C_g is positive definite and bounded for large p and n. The eigenspace to d_k, $k < \lceil n/2 \rceil$, is equal to $\operatorname{span}(u_k, u_{n-k}) = \operatorname{span}(v_k, v_{n-k})$ with

$$v_k^T := u_k^T + u_{n-k}^T = \left(\frac{2}{\sqrt{n}} \cos\left(\frac{2\pi kj}{n}\right)\right)_{j=1}^{n},$$

$$v_{n-k}^T := e^{-2\pi ik/n} u_k^T + e^{2\pi ik/n} u_{n-k}^T = \left(\frac{2}{\sqrt{n}} \cos\left(\frac{2\pi kj}{n}\right)\right)_{j=0}^{n-1}.$$

Then $v_k \pm v_{n-k}$ are the reciprocal, respectively antireciprocal, eigenvectors corresponding to d_k. In [8, Chap. 7.3] for Hermitian–Toeplitz matrices Szegö and Grenander defined the norm as

$$|T_n|^2 = \frac{1}{n} \sum_{j=1}^{n} \lambda_j(T_n)^2$$

with $\lambda_j(T_n)$ the eigenvalues of T_n, and they proved in [8, Chap. 7.6, p. 113] that

$$\begin{aligned} |T_n - C_g|^2 &\leqq 2\frac{2}{n} \sum_{j=1}^{p} j \left(1 - \frac{j}{p}\right)^2 |t_j|^2 + 2\left(\frac{2}{n} \sum_{j=1}^{p} \frac{j^2(p-j)}{p^2} |t_j|^2 + \frac{2}{n} \sum_{j=p+1}^{n} (n-j)|t_j|^2\right) \\ &= \frac{4}{n} \sum_{j=1}^{p} \frac{j(p-j)}{p} |t_j|^2 + \frac{4}{n} \sum_{j=1}^{n} (n-j)|t_j|^2 \\ &\leqq \frac{4k}{n} \sum_{j=1}^{k} |t_j|^2 + \frac{4p}{n} \sum_{j=k+1}^{p} |t_j|^2 + 4 \sum_{j=p+1}^{n} |t_j|^2, \end{aligned}$$

with $1 \leqq k \leqq p \leqq n$. Hence for C_g to be a good approximation to T_n, we have to choose $k \leqq p \leqq n$ such that $\sum_{j=k+1}^{\infty} |t_j|^2$ gets small and $n \gg k$. Furthermore, a good choice for p seems to be $p = n$. For this case, it follows directly from (9) that C_g and C_f have the

same eigenvalues and corresponding eigenvectors and thus are equal. Thereby, the eigenvalues of a circulant matrix $T(c_0, c_1, \cdots, c_{n-1})$ are given by (see, e.g., [8])

$$\lambda_j = \sum_{k=0}^{n-1} c_k e^{2\pi ijk/n}.$$

Therefore the best circulant Frobenius norm approximation to T_n is the Szegö–Grenander approximant with $p = n$. As another example, for $p = n/2$, we get $C_g = T(t_0, (1 - 2/n)t_1, \cdots, (2/n)t_{m-1}, \cdots, (1 - 2/n)t_1)$, which again is asymptotically equal to C_s and C_f. The results of Szegö and Grenander can be extended to the skewcirculant case by defining $V := (v_{k,j})_{k,j=1}^{n}$ with $v_{k,j} := e^{\pi ij(2k+1)/n}/\sqrt{n}$ and $\Delta := \operatorname{diag}(\delta_1, \cdots, \delta_n)$ with

$$\delta_k := t_0 + 2\sum_{j=1}^{p}\left(1 - \frac{j}{p}\right)t_j \cos\left(\frac{\pi(2k+1)j}{n}\right)$$

for $1 \leqq p \leqq n$. Then $S_g := V^H \Delta V$ is a skewcirculant approximation to T_n.

In the end of this section we will be concerned with indefinite Hermitian–Toeplitz matrices. If $T_n = U\Delta U^T$ is indefinite, any of the above-considered circulant approximations $C = V\Sigma V^T$ with Δ and Σ diagonal, U and V orthogonal matrices, will generally be indefinite also. Therefore the eigenvalues of $C^{-1}T_n$ may generally be complex. An obvious way to generate a positive definite circulant matrix for preconditioning is the choice of $\tilde{C} := V|\Sigma|V^T$. For C a good approximation to T_n, let us assume that the same holds for the eigenvalues and eigenvectors. Thus we have that $V^T U = I + E$ with small E, and the eigenvalues of $C^{-1}T$ are clustered around 1. Then we get

$$\begin{aligned}\tilde{C}^{-1/2} T_n \tilde{C}^{-1/2} &= V|\Sigma|^{-1/2} V^T U \Delta U^T V |\Sigma|^{-1/2} V^T \\ &= V|\Sigma|^{-1/2}(I+E)\Delta(I+E^T)|\Sigma|^{-1/2}V^T = V\Delta|\Sigma|^{-1}V^T + B\end{aligned}$$

with small B, and the eigenvalues of $\tilde{C}^{-1}T_n$ are clustered around ± 1. Hence $\tilde{C}$ may be a good choice for preconditioning the indefinite linear system $T_n x = b$. In § 4, we give an example for the clustering of the circulant approximations in the indefinite case.

3. Computing eigenvalues and eigenvectors of a symmetric Toeplitz matrix. Computing the eigenvalues of a symmetric Toeplitz matrix T_n, especially the minimum eigenvalue, is a problem of considerable interest [16], [7], [18]. By a suitable shift we can assume that T_n is positive definite. For the following, let C and S be any circulant approximation to T_n. All eigenvectors of T_n (and C, respectively S) can be chosen to be reciprocal or antireciprocal. Hence assuming that the eigenvectors of C and S are good approximations to those of T_n leads to the following algorithm for computing the minimum eigenvalue of T_n.

(i) Compute the smallest eigenvalues of C and S and the corresponding reciprocal (antireciprocal) eigenvectors $v_r(C)$, $v_r(S)$, $v_a(C)$, and $v_a(S)$, and set $U_r := (v_r(C), v_r(S))$, $U_a := (v_a(C), v_a(S))$; if the minimum eigenvalue of C or S is a simple eigenvalue and has no reciprocal or antireciprocal eigenvector, then we can complete U_r, respectively U_a, by an eigenvector of the next smallest eigenvalue.

(ii) Compute a lower bound σ for $\lambda_{\min}(T_n)$, e.g., using C_e and S_e with formula (4), or set $\sigma = 0$. Define $A := (T_n - \sigma I)^{-1}$.

(iii) Compute $\lambda_{\max}(U_r^T A U_r)$ and $\lambda_{\max}(U_a^T A U_a)$ with corresponding eigenvectors x_r and x_a. Set $y_r := U_r x_r$, $y_a := U_a x_a$.

(iv) Apply one of the following eigenvalue algorithms to get estimations for the minimum reciprocal, respectively antireciprocal, eigenvalue of T_n: (a) Lanczos algorithm on A and y_r, respectively y_a; (b) Rayleigh quotient iteration with start vector y_r, respectively y_a [14], [15].

Note that each of the algorithms generates approximations to the eigenvectors of T_n that are either reciprocal or antireciprocal. The Rayleigh quotient iteration for computing eigenvalues of T_n was proposed by Strang in [5].

Computing the eigenvectors and eigenvalues of C and S requires $O(n \log(n))$ operations. The cost of each step in (iv) depends on the method used for solving $(T_n - \mu I)x = b$. For example, solving Yule–Walker equations takes $O(n)$ operations in VLSI architecture (see [7]), while the new "fast" algorithms take $O(n \log(n)^2)$ operations (see [1]). Also, the cg method can be applied to $(T_n - \sigma I)x = b$ with preconditioner $C - \sigma I$ or $S - \sigma I$. Furthermore, the matrix A is positive definite, and thus the methods for solving the linear systems, which appear by using the Lanczos algorithm, are stable.

In view of the low number of arithmetic operations required for the above algorithms, it may also be efficient to use the first or second computed estimate of $\lambda_{\min}(T_n)$ together with the lower bound in (4) as an initial interval for the Cybenko–Van Loan method [7].

Obviously the algorithm can be generalized to compute $\lambda_{\max}(T_n)$ or $\lambda_i(T_n)$, $1 \leqq i \leqq n$, by starting with the ith eigenvalues and eigenvectors of C and S.

4. Examples. For comparing the circulant/skewcirculant approximations to T_n and for testing the eigenvalue algorithms described above, we consider the following examples:

1. $t_i = 1/(i+1)^2$ for $i = 0, \cdots, 19$; cf. [17] and [6];
2. $t_0 = 1$ and $t_i =$ random $(-.2, .2)$ for $i = 1, \cdots, 19$;
3. $t_i = \cos(i)/(i+1)$ for $i = 0, \cdots, 14$; cf. [6];
4. $t_0 = 6.2$, $t_i =$ random $(-1, 1)$ for $i = 1, \cdots, 10$ and $t_i = 0$ for $i = 11, \cdots, 19$;
5. $T = \sum_{k=1}^{40} w_k T_{2\pi\Theta_k}$ a 40×40 matrix with $(T_\Theta)_{i,j} = \cos(\Theta(i-j))$ for $i, j = 1, \cdots, 40$ and w_k and Θ_k are uniformly distributed random numbers taken from $[0, 1]$; cf. [7];
6. $t_0 = 0$ and $t_i = 1/(i+1)^2$ for $i = 1, \cdots, 39$.

Table 1 demonstrates the lower bound of formula (4); thereby $r := \sum_{k=1}^{n-1} |t_k|$. Figs. 1–4 show the eigenvalue distribution of $X^{-1}T$ for the various circulant/skewcirculant approximations X. Here the circulant approximation of Strang has the best clustering around 1. For both examples, the Szegö approximation with $p = n$ is superior to the choices $p = n/2$ and $p = \sqrt{n}$. For example 3, the skewcirculant approximation of Strang is better than the circulant one.

All in all, in the Wiener class the Strang approximation seems to have the best clustering property. The decision between the circulant and the skewcirculant preconditioner depends on the structure of the Toeplitz matrix. In general, C_s and S_s have the

TABLE 1

Nr.	$\mu_1(C_e) + \mu_1(S_e) - t_0$	$\mu_1(T)$	$\mu_n(T)$	t_0	r
1	0.6457	0.6464	2.0289	1.0	0.6
2	−0.3967	0.0064	1.8451	1.0	2.073
3	0.4514	0.4536	2.3584	1.0	1.39
4	2.0180	2.3841	11.139	6.2	5.2
5	−8.2445	0.1083	53.115	19.367	69.1

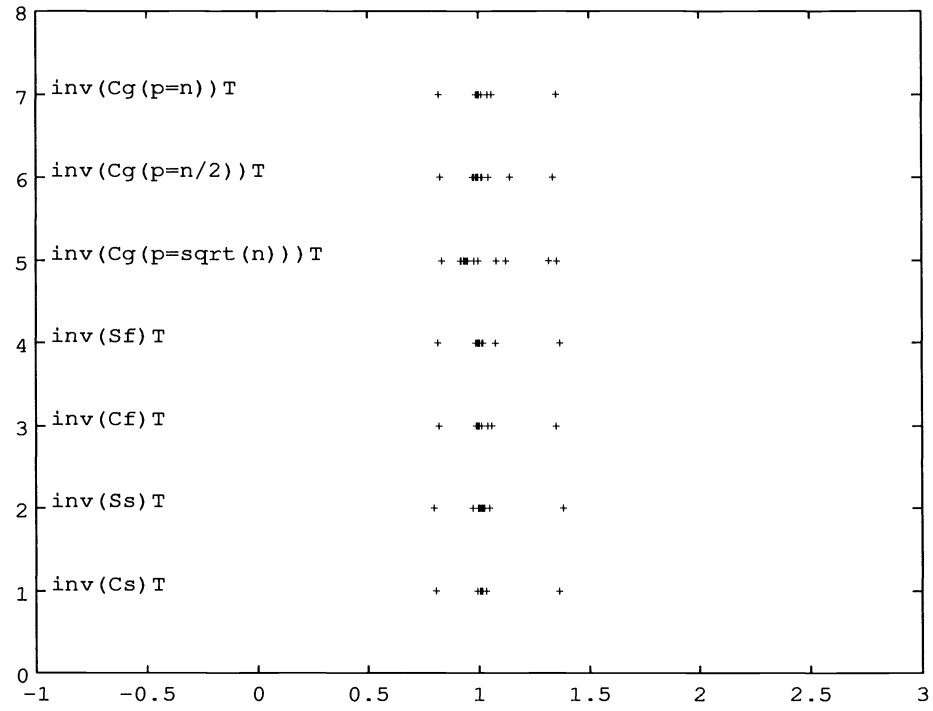


FIG. 1. *Eigenvalue distribution of* $X^{-1}T$ *for example* 1.

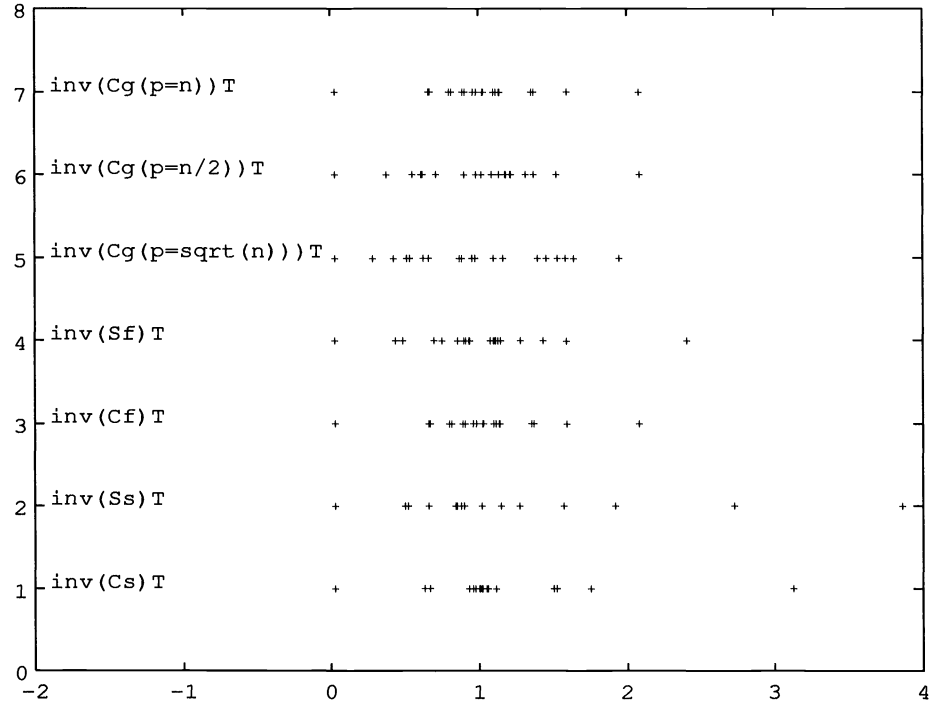


FIG. 2. *Eigenvalue distribution of* $X^{-1}T$ *for example* 2.

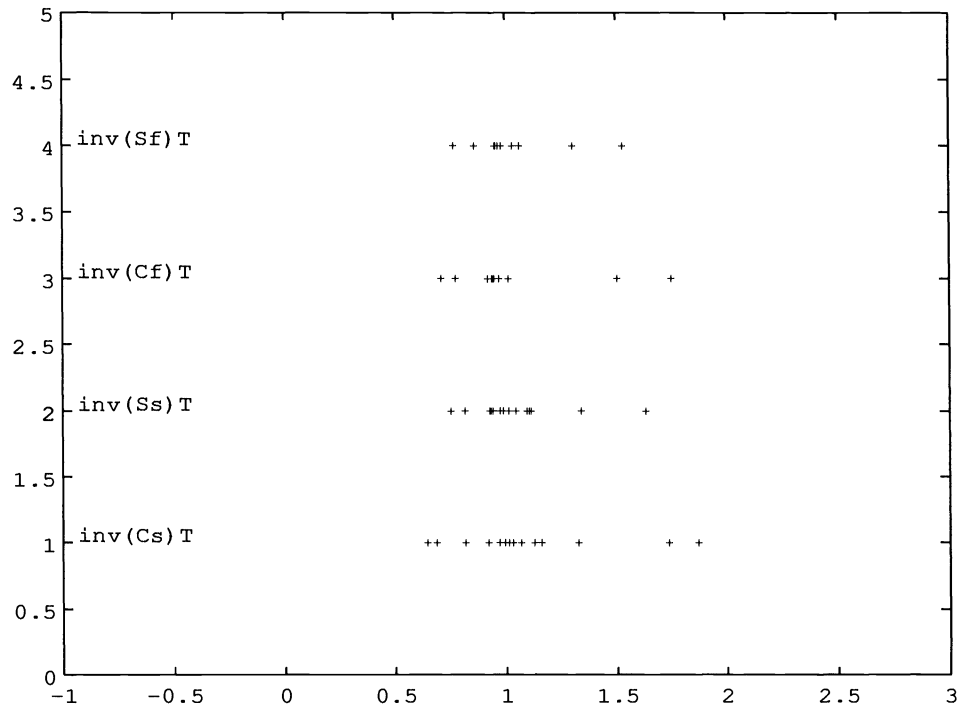


FIG. 3. *Eigenvalue distribution of* $X^{-1}T$ *for example* 3.

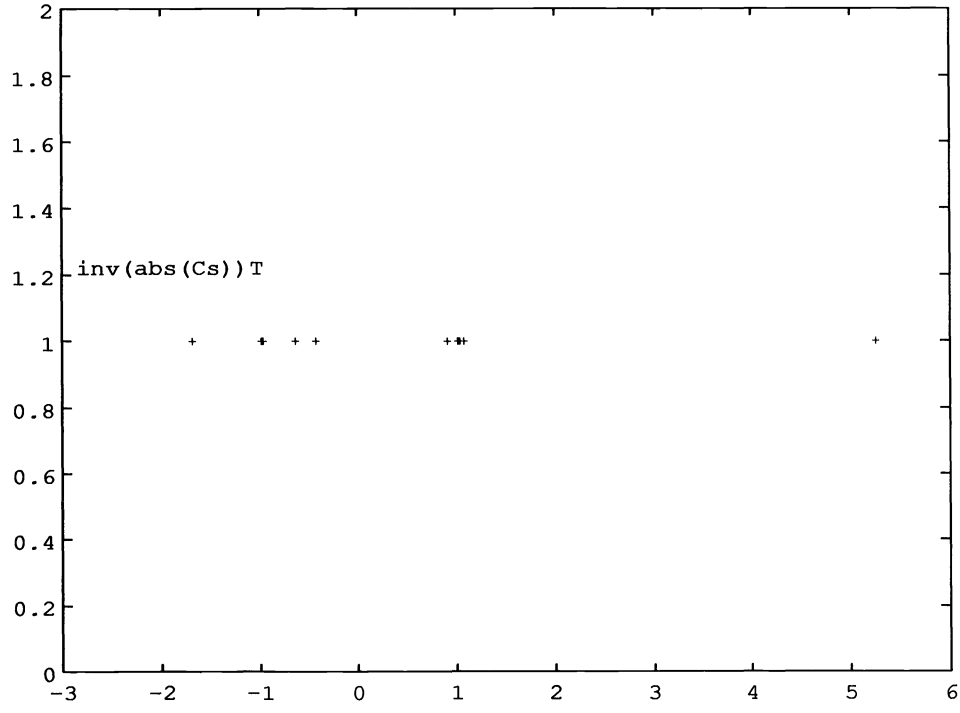


FIG. 4. *Eigenvalue distribution of* $X^{-1}T$ *for example* 6.

TABLE 2
Angle between u_T and its approximations u_C and u_S.

Nr.	$\cos(\angle(u_T, u_C))$	$\cos(\angle(u_T, u_S))$	$\cos(\angle(u_T, \text{span}\{u_C, u_S\}))$
1	0.9125	0.9996	0.999998
2	0.2238	0.9632	0.97145
3	0.9995	0.9140	0.999966
4	0.9185	0.9499	0.994668
5	0.4419	0.7923	0.7845

disadvantage of possibly becoming indefinite, and thus the Frobenius norm approximations are preferable.

Table 2 shows the angle between the true eigenvector u_T of T_n and its projections on the subspaces spanned by the eigenvectors u_C and u_S, corresponding to the minimum eigenvalues of the circulant/skewcirculant approximations C_f and S_f.

In view of Table 2 for examples 1 and 3 the eigenvalue estimation given by steps (i)–(iii) is satisfactory for many cases without use of further computation. For examples 2 and 4 we get an estimation of the same accuracy after three Lanczos steps; example 5 requires eight steps. For the general case, in our opinion the best eigenvalue algorithm (iv) is a combination of the Lanczos and the Rayleigh quotient method.

REFERENCES

[1] J. R. BUNCH, *Stability of methods for solving Toeplitz systems of equations*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 349–364.

[2] A. CANTONI AND P. BUTLER, *Eigenvalues and eigenvectors of symmetric centrosymmetric matrices*, Linear Algebra Appl., 13 (1976), pp. 275–288.

[3] R. H. CHAN, *The spectrum of a family of circulant preconditioned Toeplitz systems*, SIAM J. Numer. Anal., 26 (1989), pp. 503–506.

[4] ———, *Circulant preconditioners for Hermitian-Toeplitz systems*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 542–550.

[5] R. H. CHAN AND G. STRANG, *Toeplitz equations by conjugate gradients with circulant preconditioner*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 104–119.

[6] T. F. CHAN, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 766–771.

[7] G. CYBENKO AND C. VAN LOAN, *Computing the minimum eigenvalue of a symmetric positive definite Toeplitz matrix*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 123–131.

[8] P. J. DAVIS, *Circulant Matrices*, John Wiley, New York, 1979.

[9] P. DELSARTE AND Y. GENIN, *Spectral properties of finite Toeplitz matrices*, in Proc. 1983 Conference on Mathematical Theory of Network and Systems, Beer-Sheva, Israel, 1983; Lecture Notes in Computer Science 58, Springer-Verlag, Berlin, New York, 1984, pp. 194–213.

[10] G. H. GOLUB AND C. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

[11] U. GRENANDER AND G. SZEGÖ, *Toeplitz forms and their applications*, University of California Press, Berkeley, CA, 1958.

[12] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.

[13] T. KATO, *Perturbation Theory for Linear Operators*, Springer-Verlag, Berlin, 1976.

[14] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45 (1950), pp. 255–282.

[15] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[16] V. P. PISARENKO, *The retrieval of harmonics from a covariance function*, Geophys. J. R. Astr. Soc., 33 (1973), pp. 347–366.

[17] G. STRANG, *A proposal for Toeplitz matrix calculations*, Stud. Appl. Math., 74 (1986), pp. 171–176.

[18] W. F. TRENCH, *Numerical solution of the eigenvalue problem for Hermitian-Toeplitz matrices*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 135–146.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 778–795, July 1992

007

HOW FAST ARE NONSYMMETRIC MATRIX ITERATIONS?*

NOËL M. NACHTIGAL†, SATISH C. REDDY‡, AND LLOYD N. TREFETHEN§

Abstract. Three leading iterative methods for the solution of nonsymmetric systems of linear equations are CGN (the conjugate gradient iteration applied to the normal equations), GMRES (residual minimization in a Krylov space), and CGS (a biorthogonalization algorithm adapted from the biconjugate gradient iteration). Do these methods differ fundamentally in capabilities? If so, which is best under which circumstances? The existing literature, in relying mainly on empirical studies, has failed to confront these questions systematically. In this paper it is shown that the convergence of CGN is governed by singular values and that of GMRES and CGS by eigenvalues or pseudo-eigenvalues. The three methods are found to be fundamentally different, and to substantiate this conclusion, examples of matrices are presented for which each iteration outperforms the others by a factor of size $O(\sqrt{N})$ or $O(N)$ where N is the matrix dimension. Finally, it is shown that the performance of iterative methods for a particular matrix cannot be predicted from the properties of its symmetric part.

Key words. iterative method, conjugate gradient iteration, normal equations, Krylov space, pseudospectrum, CGN, GMRES, BCG, CGS

AMS(MOS) subject classification. 65F10

1. Introduction.

More than a dozen parameter-free iterative methods have been proposed for solving nonsymmetric systems of linear equations

$$Ax = b, \qquad A \in \mathbb{C}^{N \times N}. \tag{1.1}$$

A rough list is given in Table 1, and for a more detailed classification we recommend [1], [6], and [12]. In this paper we concentrate on the three methods that we believe are the most important: CGN, GMRES, and CGS. Quickly summarized, CGN is a name for the conjugate gradient iteration applied to the normal equations; this idea can be implemented in various ways, of which the most robust in the presence of rounding errors may be the program LSQR [18]. GMRES is the most robust of the Krylov space orthogonalization and residual minimization methods. CGS is a modification of BCG, the biconjugate gradient iteration, that appears to outperform BCG consistently. To the best of our knowledge none of the other iterations proposed to date significantly outperform CGN, GMRES, and CGS.

This leaves us with the questions: do CGN, GMRES, and CGS themselves differ significantly in capabilities? If so, which of them is best for which matrices? In the literature, these questions have for the most part been approached empirically by case studies of "real world" matrices and preconditioners. However, although such case studies are indispensable as proofs of feasibility, the answers they provide are not very sharp or general. We believe that the experimental approach is an inefficient route to the understanding of fundamental properties of algorithms and a poor basis for predicting the results of future computations.

In this paper we attempt a more systematic assessment of the convergence of nonsymmetric matrix iterations. The first half of the paper deals with generalities, presenting

* Received by the editors April 5, 1990; accepted for publication (in revised form) April 17, 1991.

† Research Institute for Advanced Computer Science, Mail Stop T041-5, NASA Ames Research Center, Moffett Field, California 94035. This research was supported by National Science Foundation grant 8814314-DMS.

‡ Courant Institute of Mathematical Sciences, 251 Mercer St., New York, New York 10012.

§ Department of Computer Science, Cornell University, Ithaca, New York 14853 (lnt@cs.cornell.edu). The research of this author was supported by a National Science Foundation Presidential Young Investigator Award and Air Force grant AFOSR-87-0102.

TABLE 1

Iterative methods for nonsymmetric systems $Ax = b$. (The differences among these algorithms are slight in some cases.) The terminology approximately follows Elman [6] *and Gutknecht* [12]. *References not listed can be found in those two papers and in* [21].

I. Methods based on the normal equations	
CGN = CGNR	Hestenes and Stiefel '52 [14]
CGNE	Craig '55
LSQR	Paige and Saunders '82 [18]
II. Orthogonalization methods	
GCG	Concus and Golub '76, Widlund '78
—	Axelsson '79, '80
ORTHOMIN	Vinsome '76
ORTHORES	Young and Jea '80
ORTHODIR	Young and Jea '80
FOM	Saad '81
GCR	Elman '82 [6], Eisenstat et al. '83 [5]
GMRES	Saad and Schultz '86 [23]
III. Biorthogonalization methods	
BIOMIN = BCG	Lanczos '52 [16], Fletcher '76 [8]
BIORES = BO	Lanczos '50, Jea and Young '83
BIODIR	Jea and Young '83
BIOMIN^2 = CGS	Sonneveld '89 [25]
BIORES^2	Gutknecht '90 [12]
BIODIR^2	Gutknecht '90 [12]
BiCGSTAB	Van der Vorst '90 [30]
QMR	Freund '90 [9], [10]
IV. Other methods	
USYMLQ	Saunders, Simon, and Yip '88 [24]
USYMQR	Saunders, Simon, and Yip '88 [24]

various results concerning the matrix properties that control the convergence of CGN (§ 2), GMRES (§ 3), and CGS (§ 4). In particular, we show that the convergence of CGN depends on the singular values of A, whereas the convergence of GMRES and CGS depends on its eigenvalues (if A is close to normal) or pseudo-eigenvalues (if A is far from normal). Many of the results we present are already known, especially those connected with CGN, but the fundamental distinction between the roles of eigenvalues and singular values seems to be often overlooked.

These general considerations lead to the conclusion that CGN, GMRES, and CGS indeed differ fundamentally in capabilities. In § 5 we substantiate this claim by constructing simple, artificial examples which show that in certain circumstances each of these three iterations outperforms the others by a factor on the order of $\sqrt{N}$ or N or more. We emphasize that these examples are in no way intended to be representative of realistic computations. They are offered entirely for the insight they provide.

Section 6 discusses the relationship between convergence rates and the properties of the symmetric part of a matrix, or as we prefer to think of it, the field of values. Using the examples of § 5 for illustration, we argue that a well-behaved symmetric part is neither

necessary nor sufficient for rapid convergence, and that therefore, considering the symmetric part is not a reliable way to analyze iterative methods.

Our discussion of all of these iterations is intentionally simplified. We largely ignore many important issues such as sparsity and other structure, machine architecture, rounding errors, storage limitations, the effect of truncation or restarts, and the possibility of hybrid Krylov space iterations, which in some cases may be the fastest of all [17]. Most important, we ignore the issue of preconditioning, without which all of these methods are often useless (see Example R, below). For a broader view of matrix iterations the reader should consult references such as [13], [21], and [22]. For an empirical comparison of CGN, GMRES, and CGS, see [19].

Throughout the paper we use the following standard notation:

$\|\ \|$ = 2-norm,
N = dimension of A,
Λ = spectrum of A,
Σ = set of singular values of A,
A^* = conjugate transpose of A,
x_0 = initial guess,
x_n = nth iterate,
$e_n = A^{-1}b - x_n$ = nth error,
$r_n = b - Ax_n = Ae_n$ = nth residual.

For any set $S \subseteq \mathbb{C}$ and function $f(z)$ defined on S we shall also find it convenient to write

$$\|f\|_S = \sup_{z \in S} |f(z)|.$$

CGN, GMRES, and CGS can each be described in a few lines of pseudocode—or programmed in a few lines of Matlab. The formulas are given in Fig. 1.

2. CGN. Perhaps the most obvious nonsymmetric iterative method is the application of the conjugate gradient iteration to the normal equations

$$A^*Ax = A^*b, \tag{2.1}$$

an idea that dates to the original CG paper by Hestenes and Stiefel [14]. (Of course, A^*A is never formed explicitly.) This algorithm, which we shall call CGN,[1] constructs the unique sequence of vectors

$$x_n \in x_0 + \langle A^*r_0, (A^*A)A^*r_0, \cdots, (A^*A)^{n-1}A^*r_0 \rangle \tag{2.2}$$

with minimal residual at each step:

$$\|r_n\| = \text{minimum}. \tag{2.3}$$

A statement equivalent to (2.3) is the orthogonality condition

$$r_n \perp \langle AA^*r_0, (AA^*)^2r_0, \cdots, (AA^*)^nr_0 \rangle. \tag{2.4}$$

The beauty of this algorithm is that thanks to the CG connection, x_n can be found by a three-term recurrence relation. For details, see [6].

[1] Though algorithms of the normal equations type are usually based on (2.1), an alternative (often called Craig's method) is the sequence $AA^*y = b$, $x = A^*y$. There is no universal agreement on names for these algorithms, but the most common choices are CGNR and CGNE, respectively. In this paper we use the neutral term CGN in place of CGNR, since except for a few details, most of what we say applies to CGNE as well.

CGN

$\beta_0 := 0;\ p_0 := 0$
For $n := 1, 2, \cdots$
$p_n := A^* r_{n-1} + \beta_{n-1} p_{n-1}$
$\alpha_n := \|A^* r_{n-1}\|^2 / \|A p_n\|^2$
$x_n := x_{n-1} + \alpha_n p_n$
$r_n := r_{n-1} - \alpha_n A p_n$
$\beta_n := \|A^* r_n\|^2 / \|A^* r_{n-1}\|^2$

GMRES

$v_1 := r_0 / \|r_0\|;\ e_1 = (1, 0, 0, \cdots)^T$
For $n := 1, 2, \cdots$
For $j := 1, \cdots, n$
$h_{jn} := v_j^* A v_n$
$\hat{v}_{n+1} := A v_n - \sum_{j=1}^{n} h_{jn} v_j$
$h_{n+1,n} := \|\hat{v}_{n+1}\|$
$v_{n+1} := \hat{v}_{n+1} / h_{n+1,n}$.
$y_n :=$ least-squares solution to $H_n y_n \approx e_1 \|r_0\|$
$x_n := x_0 + \sum_{j=1}^{n} (y_n)_j v_j$

CGS

$q_0 := p_0 := 0;\ \rho_0 := 1,\ \tilde{r}_0 = r_0$ or some other choice
For $n := 1, 2, \cdots$
$\rho_n := \tilde{r}_0^* r_{n-1}$
$\beta_n := \rho_n / \rho_{n-1}$
$u_n := r_{n-1} + \beta_n q_{n-1}$
$p_n := u_n + \beta_n (q_{n-1} + \beta_n p_{n-1})$
$v_n := A p_n$
$\sigma_n := \tilde{r}_0^* v_n$
$\alpha_n := \rho_n / \sigma_n$
$q_n := u_n - \alpha_n v_n$
$r_n := r_{n-1} - \alpha_n A (u_n + q_n)$
$x_n := x_{n-1} + \alpha_n (u_n + q_n)$

FIG. 1. CGN, GMRES, *and* CGS. *Each iteration begins with an initial guess* x_0 *and initial residual* $r_0 = b - Ax_0$. *See* [23] *for details of more efficient implementations of* GMRES *and* [18] *for the* LSQR *implementation of* CGN.

To investigate convergence rates we note that at each step we have

$$x_n = x_0 + q_{n-1}(A^*A) A^* r_0 \tag{2.5}$$

for some polynomial q_{n-1} of degree $n - 1$. Subtracting this equation from the exact solution $A^{-1}b$ gives

$$e_n = p_n(A^*A) e_0 \tag{2.6}$$

for some polynomial $p_n(z) = 1 - z q_{n-1}(z)$ of degree n with $p_n(0) = 1$. Since $r_n = A e_n$ and $A p_n(A^*A) = p_n(AA^*)A$, multiplying (2.6) by A gives

$$r_n = p_n(AA^*) r_0 \tag{2.7}$$

for the same polynomial $p_n(z)$. We conclude that convergence will be rapid if and only if polynomials p_n exist for which $\|p_n(AA^*) r_0\|$ decreases rapidly, and a sufficient condition for this is that $\|p_n(AA^*)\|$ should decrease rapidly. Exact convergence in exact arithmetic occurs in at most n steps if the degree of the minimal polynomial of AA^* is n. Convergence to a tolerance ε occurs if AA^* has a "pseudominimal polynomial" p_n with $p_n(0) = 1$ and $\|p_n(AA^*)\| \leqq \varepsilon$.

At this point singular values enter into the picture. Since AA^* is a normal matrix with spectrum Σ^2, we have

$$\|p_n(AA^*)\| = \|p_n\|_{\Sigma^2} \tag{2.8}$$

for any polynomial p_n, where we have defined $\|p_n\|_{\Sigma^2} = \sup_{z\in\Sigma^2} |p_n(z)|$ as mentioned in the Introduction. In other words, the rate of convergence of CGN is determined by the real approximation problem of minimizing $\|p_n\|_{\Sigma^2}$ subject to $p_n(0) = 1$. We have proved the following theorem.

THEOREM 1. *For the* CGN *iteration applied to an arbitrary matrix* A,

$$\frac{\|r_n\|}{\|r_0\|} \leqq \inf_{\substack{p_n\in P_n \\ p_n(0)=1}} \|p_n\|_{\Sigma^2}. \tag{2.9}$$

Greenbaum has shown that for each n, there exists an initial residual r_0 such that equality in (2.9) is attained [11]. Thus this theorem describes the upper envelope of the convergence curves corresponding to all possible initial guesses for the CGN iteration applied to a fixed matrix A and right-hand side b. Particular initial guesses make observed convergence curves lie below the envelope, but the improvement is rarely dramatic.

We emphasize that *the convergence of* CGN *is determined solely by the singular values of* A. Any two matrices with the same singular values have identical worst-case convergence rates.[2] If A is normal, the moduli of the eigenvalues are equal to the singular values, but the arguments of the eigenvalues are irrelevant to convergence. If A is not normal, convergence rates cannot be determined from eigenvalues alone.

One choice of a polynomial p_n in (2.9) is the Chebyshev polynomial T_n transplanted to the interval $[\sigma_{\min}^2, \sigma_{\max}^2]$ and normalized by $p_n(0) = 1$, where $\sigma_{\min}$ and $\sigma_{\max}$ denote the extreme singular values of A. Elementary estimates lead from here to the familiar corollary

$$\frac{\|r_n\|}{\|r_0\|} \leqq 2\left(\frac{\kappa-1}{\kappa+1}\right)^n, \tag{2.10}$$

where $\kappa = \sigma_{\max}/\sigma_{\min}$ is the condition number of A. Thus, loosely speaking, CGN converges in at most $O(\kappa)$ iterations. Unlike (2.9), however, this inequality is far from sharp in general, unless the singular values of A are smoothly distributed.

Another choice of p_n is a product of two polynomials p_k and p_{n-k} of lower degree. Together with Greenbaum's sharp form of Theorem 1, this yields another corollary of Theorem 1:

$$\frac{\|r_n\|}{\|r_0\|} \leqq \frac{\|r_k\|}{\|r_0\|}\left(\sup_{\tilde{r}_0\in\mathbb{C}^N} \frac{\|\tilde{r}_{n-k}\|}{\|\tilde{r}_0\|}\right) \tag{2.11}$$

for any $k \leqq n$. To put it in words: the envelope described by (2.9) is concave downwards, so the convergence of CGN tends to accelerate in the course of the iteration.

The convergence of CGN is strictly monotonic:

$$\|r_{n+1}\| < \|r_n\| \quad \text{if } \|r_n\| > 0. \tag{2.12}$$

---

[2] In fact the determining effect of the singular values applies to all initial vectors, not just to the worst case; see the penultimate paragraph of § 3.

One of the many ways to prove this is to note that for sufficiently small ε, $\|I - \varepsilon AA^*\| < 1$ (see § 6). Equation (2.12) follows since $p_{n+1}(z)$ must be at least as good as the product of $p_n(z)$ and the monomial $1 - \varepsilon z$.

The results of this section are essentially all known. In particular, theorems related to (2.11) can be found in [28].

3. GMRES. Residual minimization methods minimize the residual in a simpler Krylov space at the price of more arithmetic. They construct the unique sequence $\{x_n\}$ with

$$x_n \in x_0 + \langle r_0, Ar_0, \cdots, A^{n-1} r_0 \rangle \tag{3.1}$$

satisfying

$$\|r_n\| = \text{minimum}. \tag{3.2}$$

An equivalent statement is the orthogonality condition

$$r_n \perp \langle Ar_0, A^2 r_0, \cdots, A^n r_0 \rangle. \tag{3.3}$$

This condition is implemented by "brute force" in the sense that at the nth step, linear combinations of n vectors are manipulated. The GMRES iteration is a robust implementation of (3.1)–(3.3) by means of an Arnoldi construction of an orthonormal basis for the Krylov space, which leads to an $(n + 1) \times n$ Hessenberg least-squares problem [23]. At each step we have

$$e_n = p_n(A)e_0, \qquad r_n = p_n(A)r_0, \tag{3.4}$$

where $p_n(z)$ is a polynomial of degree n with $p_n(0) = 1$. Convergence will be rapid if and only if polynomials p_n exist for which $\|p_n(A)r_0\|$ decreases rapidly, and a sufficient condition for this is that $\|p_n(A)\|$ should decrease rapidly. Convergence to a tolerance ε occurs in n steps if there exists a polynomial p_n with $p_n(0) = 1$ and $\|p_n(A)\| \leqq \varepsilon$.

These formulas lead us to look at eigenvalues rather than singular values. If A is a normal matrix with spectrum Λ, then for any polynomial p_n,

$$\|p_n(A)\| = \|p_n\|_\Lambda. \tag{3.5}$$

From this we obtain the following analogue of Theorem 1.

THEOREM 2. *For the* GMRES *iteration applied to a normal matrix* A,

$$\frac{\|r_n\|}{\|r_0\|} \leqq \inf_{\substack{p_n \in P_n \\ p_n(0) = 1}} \|p_n\|_\Lambda. \tag{3.6}$$

As in Theorem 1, we expect that this bound will be reasonably sharp in practice, though it is not known that equality need be attained for any r_0. Thus if A is normal, the convergence of GMRES is determined by the eigenvalues of A via the complex approximation problem of minimizing $\|p_n\|_\Lambda$ subject to $p_n(0) = 1$. Complex approximation problems are harder than real ones, and no convergence bound as memorable as (2.10) results. Equation (2.11), on the other hand, carries over to this case without modification.

Unfortunately, nonsymmetric matrices are rarely normal. Two methods of analysis of the convergence of GMRES for general matrices have been proposed. The first, the standard approach in the literature, is based on the assumption that A is not too far from normal. For any matrix A that can be diagonalized as $A = V\Lambda V^{-1}$, the natural generalization of (2.8) is

$$\|p_n\|_\Lambda \leqq \|p_n(A)\| \leqq \kappa(V)\|p_n\|_\Lambda. \tag{3.7}$$

Combining (3.4) and (3.7) gives the following theorem.

THEOREM 3. *For the* GMRES *iteration applied to a diagonalizable matrix* A,

$$\frac{\|r_n\|}{\|r_0\|} \leqq \kappa(V) \inf_{\substack{p_n \in P_n \\ p_n(0)=1}} \|p_n\|_\Lambda, \tag{3.8}$$

where $\kappa(V)$ *is the condition number of any matrix of eigenvectors of* A.

This theorem indicates that if $\kappa(V)$ is not too large, it is still a reasonable approximation to say that the convergence of GMRES is determined by the eigenvalues of A.

The second approach is motivated by matrices for which $\kappa(V)$ is huge or infinite, that is, matrices whose eigenvalues are highly sensitive to small perturbations in the matrix entries. Let $\Lambda_\varepsilon \supseteq \Lambda$ denote the ε-pseudospectrum of A, i.e., its set of ε-pseudo-eigenvalues: those points $z \in \mathbb{C}$ that are eigenvalues of some matrix $A + E$ with $\|E\| \leqq \varepsilon$ or, equivalently, those points $z \in \mathbb{C}$ with $\|(zI - A)^{-1}\| \geqq \varepsilon^{-1}$. Let L be the arc length of the boundary $\partial\Lambda_\varepsilon$. By a contour integral we can readily show that

$$\|p_n\|_\Lambda \leqq \|p_n(A)\| \leqq \frac{L}{2\pi\varepsilon} \|p_n\|_{\Lambda_\varepsilon} \tag{3.9}$$

for any $\varepsilon > 0$ [26]. This inequality leads to the following theorem.

THEOREM 4. *For the* GMRES *iteration applied to an arbitrary matrix* A,

$$\frac{\|r_n\|}{\|r_0\|} \leqq \frac{L}{2\pi\varepsilon} \inf_{\substack{p_n \in P_n \\ p_n(0)=1}} \|p_n\|_{\Lambda_\varepsilon} \tag{3.10}$$

for any $\varepsilon > 0$.

Loosely speaking, if A is far from normal, then the convergence of GMRES depends on polynomial approximation problems defined on the pseudospectra, not just the spectrum. See [17], [26], and [27] for examples and further discussion of this phenomenon.

The convergence of GMRES, unlike CGN, is not always strictly monotonic; we can have $\|r_{n+1}\| = \|r_n\|$. A necessary and sufficient condition for strict monotonicity at every step n (and for all r_0) is that the field of values of A should lie in an open half-plane with respect to the origin. This half-plane condition is discussed further in § 6.

Neither Theorem 3 nor Theorem 4 is sharp, nor necessarily close to sharp even for worst-case initial residuals r_0. To the best of our knowledge the convergence of GMRES, unlike that of CGN, cannot be reduced completely to a problem in approximation theory.

It is readily shown that if A and $\tilde{A}$ are unitarily similar, then their behaviors under GMRES are identical in the sense that there exists a bijection $r_0 \mapsto \tilde{r}_0$ on $\mathbb{C}^N$ such that the convergence curve for A with initial vector r_0 is the same as the convergence curve for $\tilde{A}$ with initial vector $\tilde{r}_0$. The analogous statement for CGN would be that the behaviors of A and $\tilde{A}$ under CGN are identical in the same sense if AA^* and $\tilde{A}\tilde{A}^*$ are unitarily similar, which is equivalent to A and $\tilde{A}$ having the same singular values. See the remarks following Theorem 1 in § 2.

We cannot complete a discussion of GMRES without mentioning the important point that in practice, residual minimization methods are usually not applied in the "pure" form described above. To keep storage requirements under control, GMRES is often restarted after each k steps for some integer k (e.g., 5 or 10 or 20), and ORTHOMIN is generally truncated in a different but analogous way so that the algorithm works always with a k-dimensional Krylov substance. Besides the desire to keep the discussion simple, we have avoided mentioning this issue because we believe that restarting or truncating

these iterations is not an entirely satisfactory idea, since the resulting algorithms tend to spend a great deal of time relearning information obtained in previous cycles. For a discussion of this point, see [17], where we advocate the use of hybrid methods instead.

4. BCG and CGS. The BCG, or biconjugate gradient iteration, constructs non-optimal approximations in the same Krylov subspace as GMRES, but with less work per step [8], [16]. Thus, like GMRES, BCG constructs a sequence of vectors

$$x_n \in x_0 + \langle r_0, Ar_0, \cdots, A^{n-1} r_0 \rangle, \tag{4.1}$$

which implies

$$e_n = p_n(A)e_0, \qquad r_n = p_n(A)r_0 \tag{4.2}$$

for some polynomial p_n of degree n. The difference is that instead of (3.3), p_n is now determined by the orthogonality condition

$$r_n \perp \langle \tilde{r}_0, A^*\tilde{r}_0, \cdots, (A^*)^{n-1}\tilde{r}_0 \rangle, \tag{4.3}$$

where $\tilde{r}_0 \in \mathbb{C}^N$ is a vector often taken equal to r_0. Since GMRES is optimal in the sense of (3.2), BCG can never outperform it if one measures performance by the number of iterations required to reduce $\|r_n\|$ by a certain amount. However, BCG computes its choice of x_n by three-term recurrence relations. Consequently the nth step of BCG requires $O(1)$ vector operations rather than the $O(n)$ vector operations required by GMRES, making it potentially much faster in total work. Equally important, the amount of storage required does not grow with n.

CGS, which stands for "CG squared," is a modification of BCG due to Sonneveld [25]. Sonneveld's observation is that by reorganizing the BCG algorithm in a certain way one can replace (4.2) by

$$e_n = p_n^2(A)e_0, \qquad r_n = p_n^2(A)r_0 \tag{4.4}$$

for the *same* polynomial p_n, with no increase in the amount of work per step. Furthermore, whereas BCG (like CGN) requires vector multiplications by both A and A^*, which may be awkward for certain sparse data structures or parallel machines, or may be impossible when matrix-free algorithms are in use, CGS only requires multiplications by A.

We will not give further details of these algorithms or much information about their convergence properties, which are less well understood than for CGN and GMRES. For discussion of these matters, including remarkable connections with orthogonal polynomials, continued fractions, Padé approximation, and the *qd* algorithm, see [2], [12], [20], and [29]. The following remarks, most of which can be derived from the description above, will suffice.

First, thanks to (4.4), CGS typically converges (or diverges) faster than BCG by a factor of between 1 and 2.

Second, except for that factor of 2, CGS can outperform GMRES in total work but not in number of iterations. In fact, at each step we obviously have that

$$\|r_n^{\text{GMRES}}\| \leqq \|r_n^{\text{BCG}}\|, \qquad \|r_{2n}^{\text{GMRES}}\| \leqq \|r_n^{\text{CGS}}\| \tag{4.5}$$

if all three methods begin with the same r_0, regardless of the choice of $\tilde{r}_0$.

Third, for a symmetric matrix and $\tilde{r}_0 = r_0$, BCG reduces to the CG iteration [8].

Finally, far from converging monotonically, BCG and CGS are susceptible to the possibility of breakdown—division by zero—if $\rho_{n-1} = 0$ or $\sigma_n = 0$ at some step (see Fig. 1). Breakdown will not occur in the generic case, but numerical analysts are well trained to expect that where infinities may arise with probability zero, numbers large enough to

be troublesome in floating-point arithmetic are likely to appear more often than that. Moreover, as our example S below will show, the mere requirement that r_0 and $\tilde{r}_0$ be real is enough to guarantee breakdown in certain cases. In the face of such reasonable grounds for suspicion, it is remarkable how frequently BCG and CGS turn out to be effective.

Various results are known about conditions under which BCG and CGS break down or converge exactly, assuming exact arithmetic [20], [12]. For example, it can be shown that if GMRES obtains the exact solution at a certain step n, then BCG and CGS do the same if they do not break down [20]. Unfortunately, much less is known about what matters in practice: approximate breakdown and approximate convergence.

5. Eight examples. So much for the generalities. Now back to the original questions: how different are CGN, GMRES, and CGS, and when? What convergence curves—$\log \|r_n\|$ as a function of n—are possible?

To show that none of these algorithms is dispensable, three examples would suffice. As our goal has been to learn as much as possible in the process, however, we have actually constructed $2^3 = 8$ examples in an attempt to nail down the space of matrices at every corner. Table 2 summarizes these examples by listing numbers of iterations—not work estimates. For CGN and CGS the two are proportional, but for GMRES the work per step increases linearly with the number of iterations if the matrix is sparse, and so does the storage. Thus if a sparse matrix requires $O(\sqrt{N})$ iterations for both GMRES and CGS, CGS is the winner in both work and storage by a factor $O(\sqrt{N})$.

GMRES and CGS construct iterates in essentially the same Krylov space and are relatively hard to distinguish. Therefore, we begin the discussion with the first four examples in the table, for which these two behave comparably. With each example we present a computed convergence curve corresponding to dimension $N = 40$, except in two cases with $N = 400$, and a random real initial vector x_0 and right-hand side b with independent normally distributed elements of mean 0 and variance 1. Bigger dimensions do not change the curves significantly. For CGS we take $\tilde{r}_0 = r_0$, except in Example $B_{\pm 1}$.

To fully explain these experiments we mention that the curves plotted below represent actual residuals, not residual estimates computed by the iterative algorithm; as it happens, in these examples it makes little difference. Plots of errors rather than residuals also look qualitatively similar for these examples.

Example I: all methods good (Fig. 2). By Theorem 1, CGN converges in one step (for all initial data) if and only if all the singular values of A are equal, that is, if and

TABLE 2

*Numbers of iterations required for convergence to a fixed precision for our eight example matrices for worst-case initial residuals. * denotes divergence.*

Name of matrix	CGN	GMRES	CGS	
I	1	1	1	all methods good
R	N	N	N	all methods bad
C	1	N	N	CGN wins
B_1	N	2	2	CGN loses
D	N	$2\sqrt{N}$	$\sqrt{N}$	CGS wins
S	1	2	*	CGS loses
$B_{\pm 1}$	N	2	*	GMRES wins
B_κ	2	$2\sqrt{N}$	$\sqrt{N}$	GMRES loses

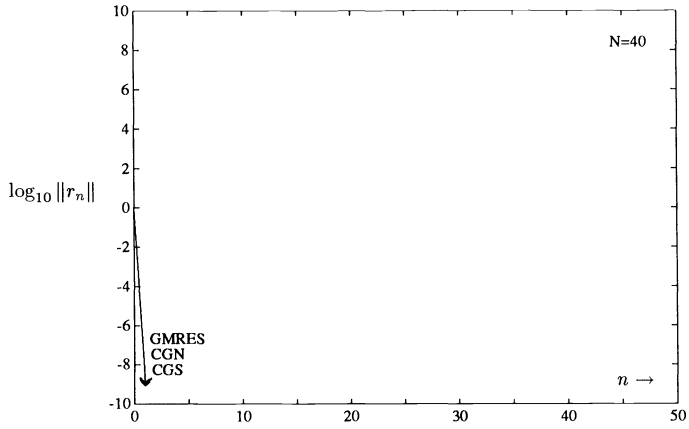


FIG. 2. *Example I (identity). All three iterations converge in one step.*

only if A is a multiple of an orthogonal matrix. By a slight extension of Theorem 3, GMRES converges in one step (and CGS also, by the remark at the end of § 4) if and only if A is diagonalizable and all its eigenvalues are equal, that is, if and only if A is a multiple of the identity. Since the identity is orthogonal, the latter condition implies the former, and these conditions are simultaneously satisfied if and only if A is a scalar multiple of the identity. Thus up to a scale factor there is a unique matrix that is handled perfectly by CGN, GMRES, and CGS: $A = I$.

Example R: all methods bad (Fig. 3). The opposite extreme would be a matrix for which all three iterations made no progress whatever until step N. By (2.12) no such example exists, but we can easily find a matrix for which all three algorithms make negligible progress until step N. By Theorems 1 and 2 any normal matrix with suitably troublesome eigenvalues and singular values will suffice, such as $A = \text{diag}(1, 4, 9, \cdots, N^2)$. For a more interesting example, consider a random matrix R of dimension N. To be precise (although the details are not very important), let the elements of R be independent normally distributed random numbers with mean 0 and variance 1. Such a matrix has condition number $O(N)$ on average and smoothly distributed singular values [4], so by

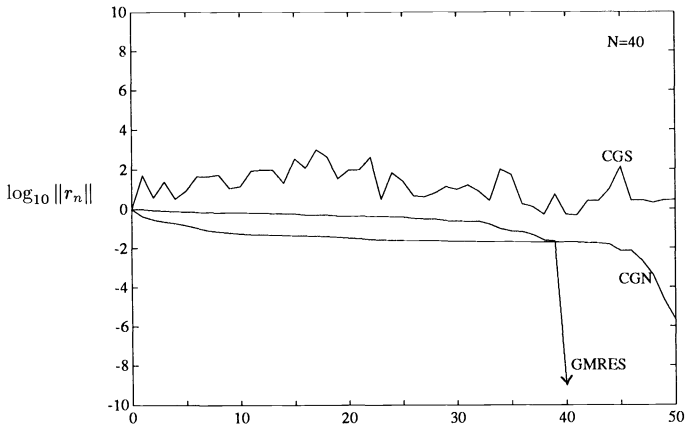


FIG. 3. *Example R (random). All three iterations require N steps.*

Theorem 1, CGN will require N steps for convergence. The eigenvalues are approximately uniformly distributed in a disk of radius $\sqrt{N}$ about the origin, suggesting that GMRES and CGS will also require N steps. In other words, *no known iterative method solves random matrix problems in better than $O(N)$ iterations*. (It would certainly be startling if this were not true, since such an iteration would beat Gaussian elimination on average even in the absence of preconditioning.) These predictions are confirmed by the experiment presented in Fig. 3. Note that the CGS convergence curve is wiggly, while the other two are monotonic, and that only GMRES exhibits the convergence in N steps that would be achieved by all three methods in exact arithmetic.

Example C: CGN *wins* (*Fig.* 4). Suppose we want a matrix for which CGN converges in one step but GMRES and CGS make no progress at all (for worst-case initial data) until step N. As mentioned above, the first requirement will be met if and only if A is a multiple of an orthogonal matrix. For the second, we must have $r_0 = r_1 = \cdots = r_{N-1}$, or by (3.3) and (4.3), $r_0 \perp \langle Ar_0, A^2r_0, \cdots, A^{N-1}r_0 \rangle$ and $r_0 \perp \langle \tilde{r}_0, A^*\tilde{r}_0, \cdots, (A^*)^{N-2}\tilde{r}_0 \rangle$. These conditions are simultaneously satisfied for suitable r_0 if A is a multiple of an orthogonal matrix with minimal polynomial $1 - z^N$, such as the circulant matrix

$$C = \begin{pmatrix} 0 & 1 & & & & \\ & 0 & 1 & & & \\ & & 0 & 1 & & \\ & & & 0 & 1 & \\ & & & & 0 & 1 \\ 1 & & & & & 0 \end{pmatrix} \quad (N \times N). \tag{5.1}$$

It is obvious why this matrix is indigestible by GMRES and CGS: C represents a circulant shift upwards by one position, while C^{-1} is a circulant shift downwards. It takes $N - 1$ shifts in one direction to approximate a single shift in the other direction, and thus Krylov spaces provide very poor approximations. This example has been mentioned before by Brown [3], van der Vorst [29], and undoubtedly others.

Example B_1: CGN *loses* (*Fig.* 5). Now we want to reverse the pattern of the last example. As mentioned above, convergence in one step of GMRES and CGS implies that the matrix has just a single nondefective eigenvalue, hence is a multiple of the identity, entailing convergence in one step of CGN also. Thus a perfect example in this category cannot exist. However, a nearly perfect example can be found if we settle for

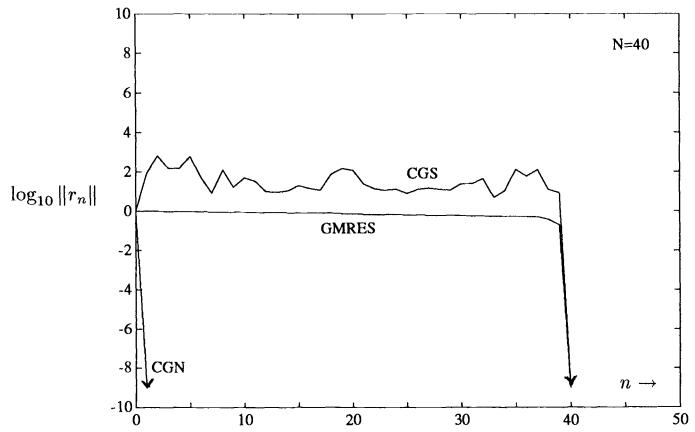


FIG. 4. *Example C (circulant shift)*. CGN *converges in one step, but* GMRES *and* CGS *require N steps.*

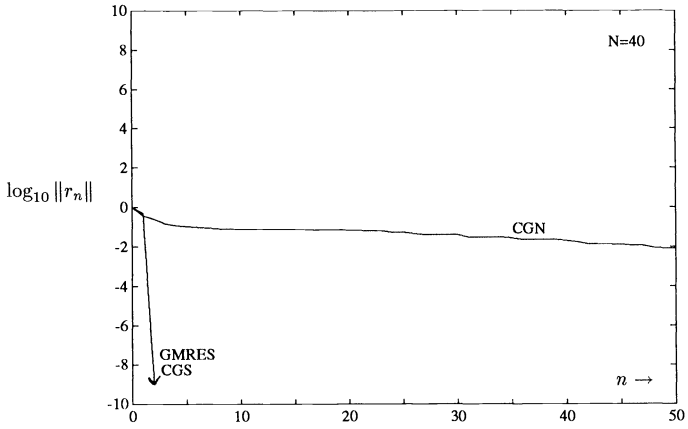


FIG. 5. *Example* B_1 (*block-diagonal matrix with eigenvalue* 1). CGN *requires N steps for convergence, but* GMRES *and* CGS *converge in two steps.*

convergence in *two* steps of GMRES and CGS. Thus we need a matrix whose minimal polynomial has degree 2 but which is ill-conditioned, with singular values spread over a wide range. Such an example is the block-diagonal matrix

$$B_1 = \begin{pmatrix} M_1 & & & & \\ & M_2 & & & \\ & & M_3 & & \\ & & & \ddots & \\ & & & & M_{N/2} \end{pmatrix} \quad (N \times N), \tag{5.2}$$

with

$$M_j = \begin{pmatrix} 1 & j-1 \\ 0 & 1 \end{pmatrix} \qquad 1 \le j \le N/2. \tag{5.3}$$

Obviously the minimal polynomial has degree 2, while the varying values of j ensure a troublesome distribution of singular values in the range approximately $[2/N, N/2]$. Incidentally, the diagonal elements of M_j might just as well have been taken to be any two numbers α and β of the same sign, so long as they remain the same in every block.

The four examples above show that CGN is sometimes better than GMRES and CGS by a factor $O(N)$ and sometimes worse by the same factor. This leaves us with the problem of distinguishing GMRES and CGS, which calls for examples of a different style. To make CGS look worse than GMRES, we construct examples in which CGS breaks down, at least for worst-case initial data. To make CGS look better than GMRES, we construct sparse examples in which both iterations take $O(\sqrt{N})$ steps, implying that the work and storage estimates for GMRES are $O(\sqrt{N})$ times larger. Alternatively, $O(\sqrt{N})$ may be replaced by a constant and these examples may be interpreted as showing that CGS may outperform GMRES by an arbitrary factor.

Example D: CGS *wins* (*Fig.* 6). For an example in this category it suffices to pick any diagonal matrix with condition number $\kappa = O(N)$ and smoothly distributed positive entries. BCG then behaves exactly like CG, requiring $O(\sqrt{N})$ iterations, since the condition number is $O(N)$, and GMRES behaves almost the same but not identically since it is

minimizing a different norm. CGS does better by at most a factor of 2. CGN, however, squares the condition number and requires $O(N)$ steps.

For a particularly clean version of this idea, define

$$D = \mathrm{diag}\,(x_1, x_2, \cdots, x_N), \tag{5.4}$$

where $\{x_j\}$ denotes the set of Chebyshev extreme points scaled to the interval $[1, \kappa]$ for some $\kappa > 1$,

$$y_j = \cos\frac{(j-1)\pi}{N-1}, \quad x_j = 1 + \frac{1}{2}(y_j + 1)(\kappa - 1), \quad 1 \leqq j \leqq N. \tag{5.5}$$

Then we expect steady convergence of GMRES at the rate indicated by (2.10) with κ replaced by $\sqrt{\kappa}$, and convergence of CGS at about twice this rate. If we set

$$\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^{2\sqrt{N}} = \varepsilon, \quad \text{i.e., } \kappa = \left(\frac{1+\varepsilon^{1/2\sqrt{N}}}{1-\varepsilon^{1/2\sqrt{N}}}\right)^2, \tag{5.6}$$

then GMRES and CGS will converge to accuracy ε in about $2\sqrt{N}$ and $\sqrt{N}$ steps, respectively. Confirming this prediction, Fig. 6 shows the results of an experiment with $\varepsilon = 10^{-10}$ and dimension $N = 400$ rather than the usual $N = 40$.

Example S: CGS *loses* (*Fig.* 7). Let S be the skew-symmetric matrix

$$S = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \otimes I_{N/2}, \tag{5.7}$$

that is, an $N \times N$ block-diagonal matrix with 2×2 blocks. This matrix is normal and has eigenvalues $\pm i$ and singular value 1. Therefore, by Theorems 1 and 2, CGN converges in one step and GMRES in two steps, as shown in Fig. 7. On the other hand, CGS encounters a division by zero at the first step for any real initial vector r_0, assuming $\tilde{r}_0 = r_0$. If $\tilde{r}_0$ is chosen at random, the zero denominator is avoided generically and convergence is achieved in practice, but the expected result of that division remains infinite.

An analogous example, though essentially of dimension 4 rather than 2, has been discussed by Joubert [15].

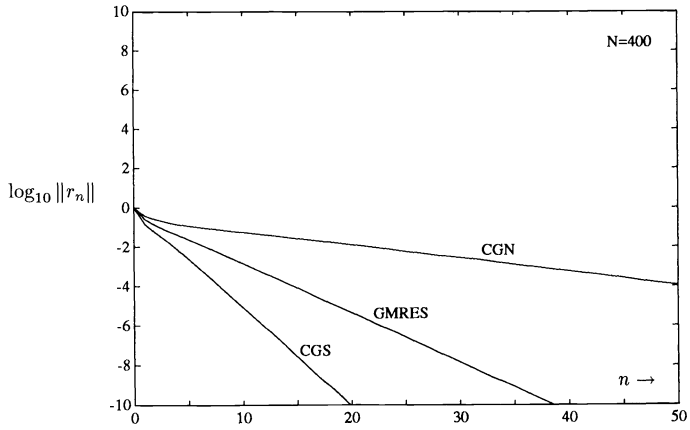


FIG. 6. *Example D* (*diagonal matrix with condition number N*). *The dimension is* $N = 400$. CGS *requires* $\sqrt{N}$ *steps for convergence, while* CGN *and* GMRES *require* $O(N)$ and $2\sqrt{N}$ *steps, respectively—hence a total work estimate in both cases comparable to* $O(N)$ *steps of* CGS.

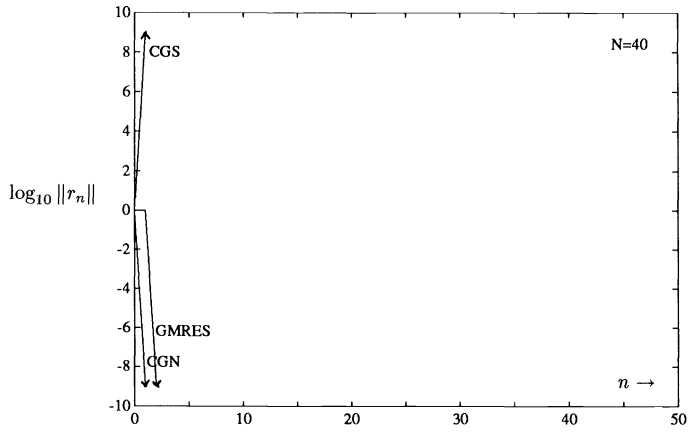


FIG. 7. *Example S (skew-symmetric).* CGS *breaks down at the first step, while* CGN *and* GMRES *converge in one and two steps, respectively.*

Example $B_{\pm 1}$: GMRES *wins* (*Fig.* 8). For this example we want a matrix like that of Fig. 5, except for which CGS breaks down. This is easily accomplished by defining a matrix $B_{\pm 1}$ by (5.2) but with (5.3) replaced by

$$M_j = \begin{pmatrix} 1 & j-1 \\ 0 & -1 \end{pmatrix}, \qquad 1 \leqq j \leqq N/2. \tag{5.8}$$

As with the matrix S above, CGS will encounter a division by zero at the first step if r_0 and $\tilde{r}_0$ are chosen appropriately, and this is what we have done in Fig. 8. Generically, however, this example does not break down.

Example B_κ: GMRES *loses* (*Fig.* 9). For this final example it is natural to modify the idea of matrices B_1 and $B_{\pm 1}$ again so that instead of fixed eigenvalues and varying singular values, we have fixed singular values and varying eigenvalues. In particular, let B_κ be defined as in (5.2) but with (5.3) replaced by

$$M_j = \begin{pmatrix} x_j & \gamma_j \\ 0 & \kappa/x_j \end{pmatrix}, \quad \gamma_j = (\kappa^2 + 1 - x_j^2 - \kappa^2/x_j^2)^{1/2}, \quad 1 \leqq j \leqq N/2, \tag{5.9}$$

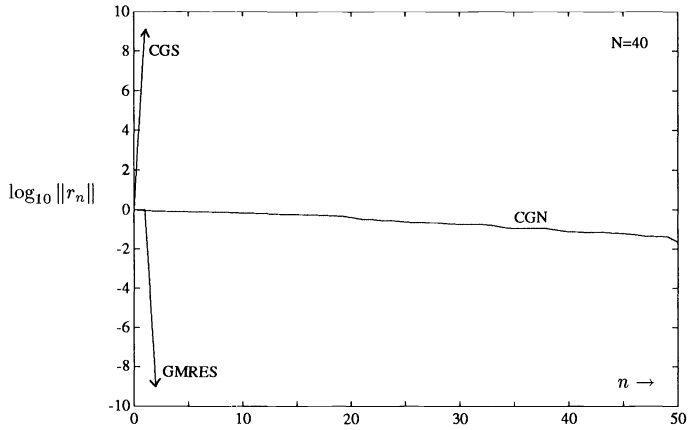


FIG. 8. *Example* $B_{\pm 1}$ *(block diagonal matrix with eigenvalues* ± 1*). To make* CGS *break down,* r_0 *and* $\tilde{r}_0$ *have been chosen diabolically.*

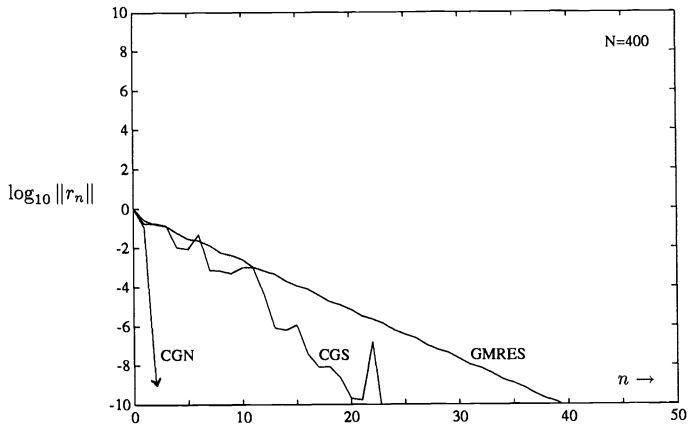


FIG. 9. *Example B_κ (block-diagonal matrix with singular values 1, κ). As in Fig. 6, the dimension is N = 400.* CGN *takes two steps for convergence,* CGS *takes $\sqrt{N}$ steps, and* GMRES *takes $2\sqrt{N}$ steps, for a total* GMRES *work estimate comparable to $O(N)$ steps of* CGS.

where $\{x_j\}$ are again Chebyshev points scaled to the interval $[1, \kappa]$ as in (5.5), but with N replaced by $N/2$. It is readily verified that each block M_j has the same singular values 1 and κ, whereas the eigenvalues lie throughout the interval $[1, \kappa]$. Taking again N = 400, $\varepsilon = 10^{-10}$, and κ defined by (5.6) gives the results shown in Fig. 9.

6. Symmetric parts and half-plane conditions. In the literature on nonsymmetric matrix iterations, much attention has been given to the behavior of the *symmetric* or more properly *Hermitian part* of a matrix, defined by $M = \frac{1}{2}(A + A^*)$. In particular, Eisenstat, Elman, and Schultz [5] and Elman [6] show that if M is positive definite, then various truncated and restarted Krylov space iterations are guaranteed to converge with a convergence rate bounded according to

$$\frac{\|r_n\|}{\|r_0\|} \leqq \left[1 - \frac{\lambda_{\min}(M)^2}{\sigma_{\max}(A)^2}\right]^{n/2}, \tag{6.1}$$

where $\sigma_{\max}(A) = \lambda_{\max}(A^*A)^{1/2}$ is the largest singular value of A. Among other algorithms, these results apply to GMRES (k) for any $k \geqq 1$, that is, GMRES restarted every k steps.

Theorems of this kind can be made rotationally invariant by restating them in terms of the *field of values* of a matrix, defined by $W = \{x^*Ax/x^*x,\ x \in \mathbb{C}^N\}$. The real part of W is equal to the interval $[\lambda_{\min}(M), \lambda_{\max}(M)]$, and therefore the statement that M is positive definite is equivalent to the statement that W lies in the open right half-plane. More generally, it is enough to assume that W lies in any open half-plane $\{z : \mathrm{Re}\,(e^{-i\theta}z) > 0\}$. We call this assumption the *half-plane condition*; it is also sometimes said that A is *definite*. The basis of these convergence theorems is the observation that the half-plane condition implies $\|1 - \varepsilon e^{-i\theta}A\| < 1$ for all sufficiently small $\varepsilon > 0$. The mathematics involved is the same as in standard results in numerical analysis on logarithmic norms, or in functional analysis, the Hille–Yosida theorem [27].

These theorems are important, but we believe they are of limited utility for choosing between iterative methods. The reason is that they are based on the relatively trivial case in which $k = 1$, analogous to a steepest-descent iteration; for $k > 2$ the half-plane condition is sufficient but not necessary for convergence. This fact is well known in principle, but nevertheless the opinion seems to have become widespread that the half-plane condition is what matters in practice. See, for example, [7] and [24].

To show that a well-behaved symmetric part is not necessary for rapid convergence of GMRES, it is enough to look at the matrices S, B_1, or $B_{\pm 1}$. For example, consider B_1. The field of values is the disk about $z = 1$ of radius $N/4$, which implies $\lambda_{\min}(M) = 1 - N/4$, $\lambda_{\max}(M) = 1 + N/4$. We could hardly be further from satisfying the half-plane condition, but GMRES converges in two steps.

Conversely, (6.1) shows that a sufficiently well-behaved symmetric part guarantees rapid convergence of GMRES. To show that mere positive definiteness of M is not enough, however, consider a normal matrix along the lines of the matrix C of (5.1), but with eigenvalues only at the roots of unity in the right half-plane. Since the condition number is 1, CGN converges in one step, whereas GMRES still requires many steps.

7. Conclusions and exhortation. Of the many parameter-free nonsymmetric matrix iterations proposed to date, we believe that CGN, GMRES, and CGS are the best. So far as we know, for calculations in exact arithmetic with performance measured by the residual norm $\|r_n\|$, no other iteration ever outperforms these three by more than a constant factor, except in certain examples involving special initial residuals r_0.

The convergence of CGN is determined by the singular values of A; the eigenvalues have nothing to do with it except insofar as they determine the singular values. If A is normal or close to normal, the convergence of GMRES is determined by the eigenvalues of A; the singular values, and in particular the condition number, have nothing to do with it. More precisely, by Theorems 1 and 2, the convergence of GMRES and CGN for a normal matrix depends on how well 0 can be approximated on the spectrum Λ by polynomials $p_n(z)$ and $p_n(r^2)$, respectively, with $p_n(0) = 1$ and $r = |z|$. It follows that we can expect CGN to be the winner if the singular values are clustered but the eigenvalues tend to surround the origin, whereas GMRES will be the winner if the eigenvalues are as tightly clustered as the singular values.

If A is far from normal, on the other hand, the convergence of GMRES becomes slower by a potentially unbounded factor than eigenvalues alone would suggest. In some such cases, the convergence is approximately determined by the pseudospectra of A instead.

The above statements about GMRES apply also, approximately, to CGS, but the convergence of CGS is affected additionally by instabilities that are not yet fully understood. When matrix-vector multiplications are much more expensive than vector operations and storage, CGS can outperform GMRES by at most a factor of 2. When the cost of vector operations and storage is significant, however, as is typical in sparse matrix calculations, Examples D and B_κ have established that CGS may outperform GMRES by a factor of order $\sqrt{N}$. Taken together, our examples show that CGN, GMRES, and CGS each outperform the others in some cases by factors of order $\sqrt{N}$ or N.

In summary, these three algorithms are genuinely distinct in their behavior. Until something better comes along, there is a place for all of them in scientific computing.

Having confined the discussion to generalities and contrived examples throughout the paper, we close with two editorial remarks of the more usual kind. First, we believe CGN is underrated. Despite the squaring of the condition number, this algorithm sometimes outperforms the competition; too many authors dismiss it with a flurry of rhetoric.[3] Second, CGS is a remarkable algorithm that deserves attention. It outperforms the more

[3] The "squaring of the condition number" we refer to is the fact that Σ^2 rather than Σ or Λ is what governs the convergence of CGN in exact arithmetic (Theorem 1). Whether rounding errors are amplified by a factor on the order of the square of the condition number is quite a different matter and is not discussed here. With the LSQR implementation, they need not be [18].

familiar BCG frequently by a factor of 1 to 2, and it converges in a number of iterations as low as GMRES far more often than the available theory might suggest. Yet, despite these impressive results, the convergence curves generated by CGS are frequently so erratic that it is hard to imagine that this algorithm can be completely right. We suspect an even better algorithm may be waiting to be discovered.[4]

CGN, GMRES, and CGS are so easy to program that there is little excuse for not taking the trouble to do so. We propose that until a fundamentally superior matrix iteration is invented, researchers in this field adopt the policy that no plot of convergence rates is complete unless it includes curves for CGN, GMRES, and CGS.

Acknowledgments. We have benefited from the generous advice of many people, including Peter Brown, Howard Elman, Roland Freund, Joe Grcar, Anne Greenbaum, Martin Gutknecht, David Keyes, Tom Manteuffel, Kapil Mathur, Olavi Nevanlinna, Chris Paige, Lothar Reichel, Youcef Saad, Michael Saunders, Robert Schreiber, Henk van der Vorst, and Jacob White. We especially thank Anne Greenbaum for pointing out an error in an earlier version of Theorems 1 and 2 and for several other valuable suggestions.

And none of this would have been any fun without Matlab.

REFERENCES

[1] S. F. ASHBY, T. A. MANTEUFFEL, AND P. E. SAYLOR, *A taxonomy for conjugate gradient methods*, SIAM J. Numer. Anal., 27 (1990), pp. 1542–1568.

[2] C. BREZINSKI, *Padé-Type Approximation and General Orthogonal Polynomials*, Birkhäuser-Verlag, Boston, 1980.

[3] P. BROWN, *A theoretical comparison of the Arnoldi and* GMRES *algorithms*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 58–78.

[4] A. EDELMAN, *Eigenvalues and condition numbers of random matrices*, Ph.D. thesis and Numer. Anal. Report 89-7, Department of Mathematics, Massachusetts Institute of Technology, Cambridge, MA, 1989.

[5] S. C. EISENSTAT, H. C. ELMAN, AND M. H. SCHULTZ, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.

[6] H. C. ELMAN, *Iterative methods for large, sparse, nonsymmetric systems of linear equations*, Ph.D. thesis and Res. Report #229, Department of Computer Science, Yale University, New Haven, CT, 1982.

[7] H. ELMAN AND R. STREIT, *Polynomial iteration for nonsymmetric indefinite linear systems*, in Numerical Analysis, J. D. Hennert, ed., Lecture Notes in Mathematics 1230, Springer-Verlag, Berlin, New York, 1986.

[8] R. FLETCHER, *Conjugate gradient methods for indefinite systems*, in Proc. of the Dundee Biennial Conference on Numerical Analysis, G. A. Watson, ed., Springer-Verlag, New York, 1975.

[9] R. W. FREUND, *Conjugate gradient-type methods for linear systems with complex symmetric coefficient matrices*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 425–448.

[10] R. W. FREUND AND N. M. NACHTIGAL, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices*, Part II, Tech. Report 90.46, Research Institute for Advanced Computer Science, NASA Ames Research Center, Moffett Field, CA, 1990.

[11] A. GREENBAUM, *Comparison of splittings used with the conjugate gradient algorithm*, Numer. Math., 33 (1979), pp. 181–194.

[12] M. H. GUTKNECHT, *The unsymmetric Lanczos algorithms and their relations to Padé approximation, continued fractions, and the qd algorithm*, Preliminary Proceedings, Copper Mountain Conference on Iterative Methods, 1990.

[4] Footnote added in revision. Since this paper was first submitted for publication, two notable additions to the field of CGS-type iterations have been introduced: the Bi-CGSTAB algorithm of van der Vorst [30] and the QMR algorithm of Freund [9], [10], both of which we inserted into Fig. 1 at the revision stage. It appears that these algorithms represent progress towards improving the erratic convergence of CGS. For a survey of these and related developments, see [31].

[13] L. A. HAGEMAN AND D. M. YOUNG, *Applied Iterative Methods*, Academic Press, New York, 1981.
[14] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.
[15] W. D. JOUBERT, *Generalized conjugate gradient and Lanczos methods for the solution of nonsymmetric systems of linear equations*, Ph.D. thesis and Report CNA-238, Center for Numerical Analysis, University of Texas, Austin, TX, 1990.
[16] C. LANCZOS, *Solution of systems of linear equations by minimized iteration*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 33–53.
[17] N. M. NACHTIGAL, L. REICHEL, AND L. N. TREFETHEN, *A hybrid* GMRES *algorithm for nonsymmetric linear systems*, SIAM J. Matrix Anal. Appl., this issue, pp. 796–825.
[18] C. C. PAIGE AND M. A. SAUNDERS, LSQR: *An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[19] G. RADICATI, Y. ROBERT, AND S. SUCCI, *Iterative algorithms for the solution of nonsymmetric systems in the modelling of weak plasma turbulence*, J. Comp. Phys., 80 (1989), pp. 489–497.
[20] Y. SAAD, *The Lanczos biorthogonalization algorithm and other oblique projection methods for solving large unsymmetric systems*, SIAM J. Numer. Anal., 19 (1982), pp. 485–506.
[21] Y. SAAD, *Krylov subspace methods on supercomputers*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 1200–1232.
[22] Y. SAAD AND M. H. SCHULTZ, *Conjugate gradient-like algorithms for solving nonsymmetric linear systems*, Math. Comp., 44 (1985), pp. 417–424.
[23] Y. SAAD AND M. H. SCHULTZ, GMRES: *A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comp., 7 (1986), pp. 856–869.
[24] M. A. SAUNDERS, H. D. SIMON, AND E. L. YIP, *Two conjugate-gradient-type methods for unsymmetric linear equations*, SIAM J. Numer. Anal., 25 (1988), pp. 927–940.
[25] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comp., 10 (1989), pp. 36–52.
[26] L. N. TREFETHEN, *Approximation theory and numerical linear algebra*, in Algorithms for Approximation II, J. C. Mason and M. G. Cox, eds., Chapman and Hall, London, 1990.
[27] L. N. TREFETHEN, *Pseudospectra of matrices*, in Proc. 14th Dundee Biennial Conf. on Numer. Anal., D. F. Griffiths and G. A. Watson, eds., to appear.
[28] A. VAN DER SLUIS AND H. A. VAN DER VORST, *The rate of convergence of conjugate gradients*, Numer. Math., 48 (1986), pp. 543–560.
[29] H. A. VAN DER VORST, *The convergence behavior of some iterative solution methods*, in Proc. Fifth Internat. Symposium Numer. Meth. Engrg., Vol. 1, R. Gruber, J. Periaux, and R. P. Shaw, eds., Springer-Verlag, 1989.
[30] ———, *Bi*-CGSTAB: *A fast and smoothly converging variant of Bi*-CG *for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.
[31] R. W. FREUND, G. H. GOLUB, AND N. M. NACHTIGAL, *Iterative solution of linear systems*, Acta Numerica, to appear.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 796–825, July 1992

008

A HYBRID GMRES ALGORITHM FOR NONSYMMETRIC LINEAR SYSTEMS*

NOËL M. NACHTIGAL†, LOTHAR REICHEL‡, AND LLOYD N. TREFETHEN§

Abstract. A new hybrid iterative algorithm is proposed for solving large nonsymmetric systems of linear equations. Unlike other hybrid algorithms, which first estimate eigenvalues and then apply this knowledge in further iterations, this algorithm avoids eigenvalue estimates. Instead, it runs GMRES until the residual norm drops by a certain factor, then re-applies the polynomial implicitly constructed by GMRES via a Richardson iteration with Leja ordering. Preliminary experiments suggest that the new algorithm frequently outperforms the restarted GMRES algorithm.

Key words. iterative method, Krylov subspace, CGNR, GMRES, CGS, hybrid, pseudospectrum

AMS(MOS) subject classification. 65F10

1. Introduction. In this paper we present a new point of view regarding nonsymmetric matrices, and as a natural outgrowth, a new hybrid iterative algorithm. The point of view is that if a matrix is nonsymmetric (more precisely, nonnormal), any attempt to make use of its eigenvalues should be viewed with caution. The new algorithm is a hybrid scheme in which a few steps of GMRES [29] are followed by a Richardson iteration based on the polynomial implicitly constructed by GMRES, with the factors ordered in a Leja sequence for stability [25]. Unlike other hybrid algorithms, this one never estimates any eigenvalues. It is simpler than other hybrid iterations, but more robust, and appears to outperform other methods in many cases.

We begin with a brief explanation and survey of hybrid methods, assuming that the reader is already familiar with GMRES, the Arnoldi process, and polynomial iterations. Suppose we are given a large nonsymmetric system of linear equations

$$Ax = b, \quad A \in \mathbb{C}^{N \times N}, \quad x, b \in \mathbb{C}^N, \tag{1.1}$$

where A may be the matrix that results after preconditioning. The many nonhybrid iterative methods that have been proposed for solving such systems can be divided into two categories: (i) those that require no a priori information about A, of which three of the most important are CGN, CGS, and GMRES,[1] and (ii) those that do require a priori information about A, such as the Richardson and Chebyshev iterations. The idea of a hybrid iteration is to combine these approaches in a two-phase algorithm:

Phase I: acquire information about A via an iteration of type (i);

Phase II: apply that information in further iterative steps of type (ii).

In practice, of course, things need not be quite so simple; a robust code may loop back to Phase I one or more times to ensure an adequate convergence rate.

* Received by the editors April 5, 1990; accepted for publication (in revised form) June 5, 1991.

† Research Institute for Advanced Computer Science, Mail Stop T041-5, NASA Ames Research Center, Moffett Field, California 94035 (santa@riacs.edu) This research was supported by Air Force Office of Scientific Research grant 87-0102 and by National Science Foundation grant 8814314-DMS.

‡ Department of Mathematics and Computer Science, Kent State University, Kent, Ohio 44242 (reichel@mcs.kent.edu). This research was supported in part by IBM Bergen Scientific Centre and National Science Foundation grant DMS-8704196.

§ Department of Computer Science, Cornell University, Ithaca, New York 14853 (lnt@cs.cornell.edu).

[1] CGN is a name for the conjugate gradient iteration applied to the normal equations [14], CGS is a biorthogonalization algorithm adapted from the biconjugate gradient iteration [36], and GMRES is an algorithm based on residual minimization in a Krylov subspace [29].

The assumption underlying the hybrid idea is that algorithms of type (i) cost more per step than those of type (ii), so that a switch from the one to the other is potentially advantageous. This assumption frequently holds for GMRES and for the many other Krylov subspace iterations such as ORTHORES, ORTHOMIN, and ORTHODIR, because these algorithms have the unfortunate property that the work and storage required to carry out the nth step grow in proportion to n. The goal of a hybrid algorithm is to recover some of this factor $O(n)$. On the other hand the assumption does not hold for CGS nor for CGN in problems where A^* is as easy to apply to a vector as A. Thus the natural realm of applicability of hybrid methods is to problems where Krylov subspace methods take fewer steps than the alternatives. For a discussion of when this is likely to be the case, see [20].

The recent literature on hybrid methods begins with a paper of Manteuffel [18]. In Manteuffel's algorithm, a number of extreme eigenvalues of A are first estimated by a modified power iteration (Phase I). These eigenvalue estimates are then surrounded by an ellipse, and a Chebyshev iteration is carried out with parameters corresponding to that ellipse (Phase II). Schematically,

Manteuffel '78:
modified power iteration → eigenvalue estimates → ellipse → Chebyshev iteration.

Ashby has implemented this algorithm in a Fortran code package called ChebyCode [2], which incorporates many safeguards and extra features omitted in this outline.

Let x_0 denote the approximation to the solution $A^{-1}b$ at the beginning of an iterative process, which may correspond to Phase I or Phase II depending on context. We use the following (standard) notation:

$$n\text{th iterate: } x_n,$$

$$n\text{th error: } e_n = A^{-1}b - x_n,$$

$$n\text{th residual: } r_n = Ae_n = b - Ax_n.$$

Manteuffel's algorithm delivers Phase II iterates x_n satisfying

$$x_n = x_0 + q_{n-1}(A)r_0, \qquad q_{n-1} \in P_{n-1}$$

and

$$e_n = p_n(A)e_0, \quad r_n = p_n(A)r_0, \quad p_n \in P_n, \quad p_n(0) = 1, \tag{1.2}$$

where $p_n(z) = 1 - zq_{n-1}(z)$ is a Chebyshev polynomial shifted to the ellipse of eigenvalue estimates. (P_n denotes the set of polynomials of degree less than or equal to n.) The same equations (1.2) hold for other hybrid Krylov subspace iterations, including our own. The various algorithms differ only in the choice of the sequence of polynomials $p_n(z)$, known as *residual polynomials*, and in the mechanics of how they are applied. Our goal is to make $\|p_n(A)r_0\|$ small, and the obvious way to achieve this is to try to make $\|p_n(A)\|$ small.

One modification of Manteuffel's algorithm is to replace the Chebyshev iteration of Phase II by a more general iteration, an idea first proposed by Smolarski and Saylor [34], [35]. Phase I of their algorithm constructs a polygon containing eigenvalue estimates, then solves a discrete least-squares approximation problem on that polygon to obtain an effective residual polynomial $p_\nu(z)$ for some integer ν. Phase II applies this polynomial one or more times by means of a cyclic Richardson iteration

$$p_{k\nu}(z) = [p_\nu(z)]^k, \qquad k = 1, 2, \cdots. \tag{1.3}$$

In outline:

Smolarski and Saylor '81:
modified power iteration → eigenvalue estimates → polygon →

L^2-optimal $p_\nu(z)$ → Richardson iteration.

The advantage of such an algorithm is that since the approximation problem is posed on an arbitrary domain of estimated eigenvalues, there is no restriction to matrices whose spectra are well approximated by an ellipse. Throughout this paper, $p_\nu(z)$ denotes a residual polynomial of fixed degree which forms the basis of a cyclical Phase II iteration defined by (1.3).

Another modification of Manteuffel's algorithm is to replace the power iteration of Phase I by an Arnoldi iteration, which is now the standard method for estimating eigenvalues of nonsymmetric matrices iteratively. In fact this difference is not as great as the names suggest, for the modified power iteration is essentially the same as the Arnoldi iteration. Together with the Arnoldi point of view, however, comes the important additional advantage that an approximate solution in Phase I can be conveniently constructed by the closely related GMRES algorithm. This kind of hybrid was first proposed by Elman, Saad, and Saylor [7]:

Elman, Saad, and Saylor '86:
Arnoldi/GMRES → eigenvalue estimates → ellipse → Chebyshev iteration.

To be more precise about what we mean by an "Arnoldi/GMRES" calculation, the Arnoldi and GMRES iterations both make use of a Hessenberg matrix obtained by the orthogonalization of a sequence of Krylov vectors. The Arnoldi iteration estimates eigenvalues of A by computing the eigenvalues of the square part of this matrix, while GMRES finds approximate solutions to $Ax = b$ by solving a least-squares problem involving the same matrix made rectangular by the addition of an extra row. In a Phase I calculation of Arnoldi/GMRES type, both of these computations are carried out simultaneously, so that Phase II begins with both eigenvalue information and a good initial guess.

The most general hybrid algorithms combine both of these modifications, and thus differ from Manteuffel's algorithm in both Phases I and II. One of the first of these to be proposed was the PSUP algorithm of Elman and Streit [8], in which an Arnoldi/GMRES iteration to obtain eigenvalue estimates is followed by a solution of an L^∞ approximation problem on a polygonal domain, with the resulting polynomial iteration implemented by a matrix version of Horner's rule:

Elman and Streit '86:
Arnoldi/GMRES → eigenvalue estimates → polygon →

L^∞-optimal $p_\nu(z)$ → Horner iteration.

Another algorithm of this type, developed at about the same time by Saad [28], solves an L^2 approximation problem on a polygon after first constructing a well-conditioned basis of shifted Chebyshev polynomials, then applies the resulting polynomial in a second-order Richardson iteration:

Saad '87:
Arnoldi/GMRES → eigenvalue estimates → polygon → Chebyshev basis →

L^2-optimal $p_\nu(z)$ → second-order Richardson iteration.

More recently, Saylor and Smolarski have also modified their method to take advantage of GMRES [32], [33]:

Saylor and Smolarski '91:
Arnoldi/GMRES → eigenvalue estimates → polygon →

L^2-optimal $p_\nu(z)$ → Richardson iteration.

Finally, two recent algorithms developed since this article was first drafted make use of numerically computed Schwarz–Christoffel conformal maps [17], [43]:

Li '91:
Arnoldi/GMRES → eigenvalue estimates → polygon → conformal map →

rational approximation → (k,l)-step iteration.

Starke and Varga '91:
Arnoldi/GMRES → eigenvalue estimates → polygon →

conformal map → Faber polynomials → Richardson iteration.

The above algorithms are summarized in Table 1.1. We hasten to add that these algorithms differ in many important ways that we have not mentioned and indeed, all of the one- or two-line summaries of this section represent only the barest of skeletons.

This completes our survey of existing hybrid algorithms of the fully specified sort, where procedures for both Phases I and II are given, so that the algorithm is applicable in principle to an arbitrary matrix. In addition, however, there is a large literature of "polynomial iterations" or "semi-iterative methods" devoted to Phase II by itself, on the assumption that eigenvalue estimates are already available, and each of these becomes a full-fledged hybrid algorithm as soon as it is coupled with an Arnoldi/GMRES iteration for Phase I. For example, Opfer and Schober construct a first-order Richardson iteration from an L^∞-optimal polynomial [22]; Eiermann [5] and Gutknecht [13] investigate Faber and Faber-CF approximations, respectively; Fischer and Reichel [9], [24] and Tal-Ezer [37] derive $p_\nu(z)$ by polynomial interpolation in Féjer points (conformal images of roots of unity); and Gragg and Reichel recommend the use of polynomials orthogonal on the boundary of the eigenvalue domain [11]. There are also a number of important papers by Eiermann, Niethammer, Varga, and others on further aspects of these iterations and their connections with approximation theory and complex analysis; an example is [6]. We will not attempt a survey, but merely note in conclusion that whereas the idea of estimating eigenvalues by the Arnoldi process clearly predominates for Phase I of hybrid algorithms in the current literature, the possibilities for Phase II are numerous.

TABLE 1.1
Hybrid algorithms.

		Phase II: Chebyshev	Phase II: Other
Phase I	Power Method	Wrigley [42] Manteuffel [18] Saylor [30]	Smolarski and Saylor [34], [35]
	Arnoldi/GMRES	Elman, Saad, and Saylor [7]	Elman and Streit [8] Saad [28] Saylor and Smolarski [32], [33] Li [17] Starke and Varga [43]

Hybrid iterative algorithms are closely related to the idea of *polynomial preconditioners* [15], [27]. The polynomial $p_\nu(z)$ constructed by the hybrid algorithm proposed in this paper might be applied as a preconditioner, and a recursive version of this idea has recently been considered by Joubert [16, § 3.5].[2] In the symmetric case, there is a long history of hybrid algorithms and polynomial preconditioners motivated mainly by the search for parameters for Chebyshev iterations. In particular, a recursive conjugate-gradient preconditioner analogous to Joubert's has recently been proposed for the symmetric case by O'Leary [21]. Various further combinations of hybrid and preconditioning ideas will undoubtedly be investigated in the years to come.

2. The trouble with eigenvalues. What the existing hybrid methods have in common is that they all estimate eigenvalues, construct a domain enclosing them in the complex plane, and then calculate a polynomial $p_\nu(z)$ that is small in some sense on that domain.

(2.1) *Existing algorithms*:
Arnoldi/GMRES → eigenvalue estimates → enclosing domain → $p_\nu(z)$ → iteration.

In this section we explain why we consider the introduction and subsequent manipulation of eigenvalue estimates inappropriate. The principal problem is that eigenvalues do not generally contain enough information to capture the behavior of a matrix efficiently in the nonnormal case and, in particular, even though the scalar $p_\nu(\lambda)$ may be small whenever λ is an eigenvalue of A, it does not follow that the matrix $p_\nu(A)$ is small in norm. A secondary problem, relevant even for normal matrices, is that the smallness of $p_\nu(z)$ on a set of estimated eigenvalues does not imply that it is small at the exact eigenvalues.

To begin the discussion with the first of these problems, let us suppose first that exact eigenvalues rather than mere estimates happen to be available at the end of Phase I. On the face of it this should be the ideal situation for the standard iterations in Phase II. Following [38], however, we can show by an example that eigenvalue information may be utterly misleading as to the actual behavior of A. Let A be a large upper-triangular Toeplitz matrix of the form

$$A = \begin{pmatrix} 1 & 1 & \frac{1}{2} & & & \\ & 1 & 1 & \frac{1}{2} & & \\ & & 1 & 1 & \frac{1}{2} & \\ & & & 1 & 1 & \frac{1}{2} \\ & & & & 1 & 1 \\ & & & & & 1 \end{pmatrix} \quad (N \times N). \tag{2.2}$$

This matrix has just the single eigenvalue $\{1\}$.[3] Thanks to this simple eigenvalue distribution, one might naturally expect a Phase II iteration to achieve rapid convergence with the sequence of residual polynomials $p_n(z) = (1 - z)^n$. In actuality, however, this choice will lead to geometric divergence at a rate approximating $(\frac{3}{2})^n$ for large N and $n \ll N$. The reason is that for practical purposes A behaves much more nearly as if its spectrum were

$$\Lambda_{\text{practical}} = f(\bar{D}), \tag{2.3}$$

[2] Joubert's work, though formulated in terms of preconditioning rather than hybrid iterations, is the closest we have seen in the literature to the idea proposed here. His implementation does not take advantage of the $O(\nu)$ speedup in Phase II, however, so little improvement over existing methods is achieved.

[3] In fact it is defective, but the reader should not make too much of that, for the defectiveness is an inessential property that could be removed without changing the matrix behavior significantly by adding small perturbations to the diagonal elements. In any case, similar examples are readily constructed that are nondefective to begin with, such as the matrix (3.6) in § 3.

where $\bar{D}$ is the closed unit disk and $f(z)$ is the *symbol* of this Toeplitz matrix [26],

$$f(z) = 1 + z + \tfrac{1}{2}z^2. \tag{2.4}$$

This domain is illustrated in Fig. 2.1. Although it contains the exact spectrum $\Lambda = \{1\}$, it is much larger than that and, in particular, the fact that it extends to the right as far as the point $z = 2.5$ is what causes divergence at the rate $(\frac{3}{2})^n$ of a Richardson iteration based on $p_n(z) = (1 - z)^n$.

On the other hand, if we take $p_n(z)$ to be a sequence of polynomials that are small on $\Lambda_{\text{practical}}$ instead of just Λ, the convergence of the Richardson iteration for the same example becomes rapid.

This example is contrived, but similar phenomena occur frequently in scientific computing. Convection-diffusion equations, for example (the favorite test problems in papers on iterative methods), sometimes lead to matrices with misleading spectra closely related to (2.2). The matrices that arise in spectral methods also have misleading eigenvalues [23]. We are convinced that this pattern is a common one throughout applications involving nonnormal matrices [39].

If eigenvalues are not the right information on which to base a Phase II iteration, might some different information perform better? It will not do to look at Jordan structure, for aside from the impracticability of estimating Jordan blocks in Phase I, we have already noted that small perturbations would make the Jordan canonical form of this and any other matrix diagonal without changing the eigenvalues very much, in which case we are back where we started. Another unsuitable idea is to consider the spectrum of A in the operator limit $N \to \infty$. That would be satisfactory for the example above, but not for many other problems in which the limit process is less sharp, as occur, for example, in spectral methods. A third idea is to replace the spectrum of A by the the *field of values* $W(A)$, i.e., the set of Rayleigh quotients x^*Ax/x^*x, $x \in \mathbb{C}^N$. However, fields of values are too big to be appropriate for eigenvalue-style applications, besides being always convex. For example, although the matrices

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad \begin{pmatrix} 1 & 3 \\ 0 & 1 \end{pmatrix} \tag{2.5}$$

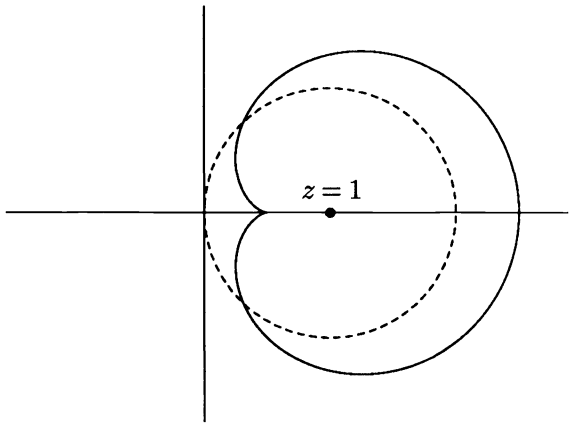


FIG. 2.1. *Spectrum (dot) and pseudospectrum (bounded by solid curve) of the matrix A of* (2.2) *for large dimension N. Since the pseudospectrum extends outside the dashed circle, a Richardson iteration based on the residual polynomials* $p_n(z) = (1 - z)^n$ *will diverge as n increases with* $n \ll N$.

are far from singular, their fields of values both contain the origin, and therefore no residual polynomial satisfying the normalization condition $p_n(0) = 1$ can ever be smaller than 1 on $W(A)$. For a further discussion of this point, see [20, § 6].

We believe that if the idea of working with a domain in the complex plane is to be retained, a better approach might be to replace Λ by a fourth candidate, the *ε-pseudospectrum* [23], [26], [38], [39], defined by

$$\Lambda_\varepsilon = \{\lambda \in \mathbb{C}: \|(\lambda I - A)^{-1}\| \geqq \varepsilon^{-1}\}, \tag{2.6}$$

where $\varepsilon > 0$ is a small parameter which for hybrid methods should be taken to be on the order of the residual reduction that has been achieved by Phase I (i.e., $\varepsilon \approx \tau$, with τ defined by (3.3), below). Equivalently, Λ_ε can be defined in terms of perturbations of eigenvalues

$$\Lambda_\varepsilon = \{\lambda \in \mathbb{C}: \lambda \text{ is an eigenvalue of } A + E \text{ for some } \|E\| \leqq \varepsilon\}. \tag{2.7}$$

We have discussed pseudospectra elsewhere, however, and will not pursue the idea further here. Instead, our more fundamental proposal (in § 3) is that domains in the complex plane need not be manipulated at all.

Before leaving the subject of eigenvalue-related quantities, however, we must mention a remarkable phenomenon that may partially explain why existing hybrid algorithms work as well as they do. Eigenvalues *estimates* are sometimes more reliable than exact eigenvalues! We have noticed this effect in our experiments and Manteuffel informs us that he has noticed it too [19]. One way to explain it is to note that eigenvalue estimates tend to come closer to a pseudospectrum than to the exact spectrum [39], and it is usually the pseudospectrum that provides the better iteration parameters. A related phenomenon in another context has been mentioned in [40, § 7].

However, this eigenvalue-estimate effect is not robust enough to provide a foundation for an algorithm to be applied to arbitrary matrices, and to illustrate this we will now turn to the second problem with eigenvalue estimates mentioned in the opening paragraph of this section. For a trivial example, take

$$A = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}. \tag{2.8}$$

After one step of the Arnoldi iteration, the estimated eigenvalue is

$$\lambda_{\text{Arnoldi}} = \frac{r_0^T A r_0}{r_0^T r_0} = 0, \tag{2.9}$$

assuming r_0 is real. Considering the symmetry about the origin of the actual eigenvalues $\pm i$, this may seem a natural enough choice, but it is fatal for any polynomial iteration that attempts to construct polynomials $p_n(z)$ that are small on the spectrum but normalized by $p_n(0) = 1$. For example, if $p_n(z)$ is constructed as a product of terms $(1 - z/\lambda_i)$ corresponding to various eigenvalue estimates λ_i (see (5.3)), an eigenvalue estimate $\lambda = 0$ will lead to a factor $(1 - z/0)$ and a consequent division by zero. We shall return to this example at the end of § 3.

For a richer example along the same lines, consider the matrix

$$A = \begin{pmatrix} I + R_1 & 0 \\ 0 & -I + R_2 \end{pmatrix} \qquad (N \times N), \tag{2.10}$$

where R_1 and R_2 are dense matrices of dimension $N/2$ with independent normally distributed random elements of standard deviation $\frac{1}{4}\sqrt{N/2}$. For large N, the eigenvalues of

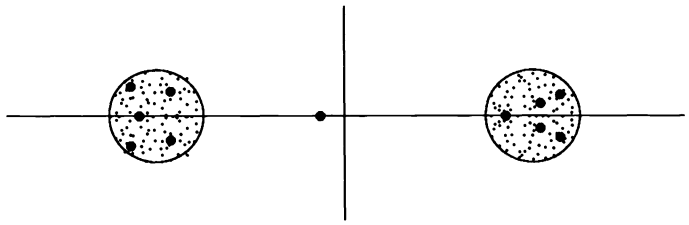


FIG. 2.2. *Arnoldi eigenvalue estimates at step* $n = 11$ *for the matrix* A *of* (2.10), $N = 200$. *The small dots are the exact eigenvalues of* A. *One of the estimates appears near the origin, far from the exact spectrum, which will cause a residual polynomial based on these estimates to perform poorly.*

A are approximately uniformly distributed in the disks $|z \pm 1| \leqq \frac{1}{4}$, well away from the origin, and the condition number is $\kappa(A) \approx 2.06$ [4]. Fig. 2.2 shows that ten of the Arnoldi estimates match this spectrum reasonably well at step $n = 11$, but the eleventh, thanks to the symmetry, appears near the origin. Since this spurious eigenvalue is not exactly at the origin, it does not cause a division by zero, but it certainly leads a polynomial iteration astray.

To summarize, it is not entirely safe to base a matrix iteration on exact eigenvalues, if they happen to be available, nor, so far as we are aware, on eigenvalues estimated by any existing methods. Of course, a new eigenvalue estimator might be found with better properties—and in fact the algorithm we are about to propose might be described in those terms. Since the successful operation of such an algorithm depends on its estimating eigenvalues *incorrectly*, however, we see little to be gained by interpreting it as an eigenvalue estimator.

3. The hybrid GMRES algorithm. We propose that in (2.1), the middle steps should be eliminated:

(3.1) *New algorithm*:
GMRES $\rightarrow p_\nu(z) \rightarrow$ iteration.

The GMRES iteration of Phase I constructs a sequence of residual polynomials that minimize the norm of the residual

$$\text{(3.2)} \qquad \text{GMRES: } \|r_n\| = \|p_n(A)r_0\| = \min_{\substack{p \in P_n \\ p(0)=1}} \|p(A)r_0\|, \qquad n = 1, 2, \cdots.$$

Correspondingly, what Phase II requires is another sequence of residual polynomials. Why translate from polynomials to eigenvalue estimates and back again? We propose to take precisely the GMRES polynomial $p_\nu(z)$ obtained at some step ν of Phase I and continue applying it over and over again in Phase II cyclically as in (1.3):

HYBRID GMRES ALGORITHM
Start with a *random* initial guess x_0.
Phase I: Run GMRES until $\|r_n\|$ drops by a suitable amount. Set $\nu := n$.
Phase II: Re-apply the GMRES polynomial $p_\nu(z)$ cyclically until convergence.

This algorithm is purely a GMRES hybrid, for no Arnoldi eigenvalue estimates are calculated. No domain of estimated eigenvalues is constructed and no approximation problem is solved in the complex plane. The omission of these steps makes our hybrid algorithm simpler than most of those previously proposed.

Of course, many issues have been left out of this description, such as:

1. What is a "suitable" reduction in $\|r_n\|$ for terminating Phase I?
2. How shall $p_\nu(z)$ be constructed from the GMRES iteration?

3. How shall $p_\nu(z)$ be applied in Phase II?
4. What safeguards should be added to ensure convergence?
5. What are the properties of this algorithm in theory?
6. How does it perform in practice?

Most of these questions will be addressed in the next few sections, but not definitively. We have little doubt that improvements can be effected in many of the details of our implementation.

Before we turn to these issues, however, a few remarks will clarify the idea of our algorithm; further details are given in § 6. Suppose that at the νth GMRES step we have

$$\frac{\|r_\nu\|}{\|r_0\|} = \frac{\|p_\nu(A)r_0\|}{\|r_0\|} = \tau \tag{3.3}$$

for some $\tau < 1$. Our hope is that we then have

$$\|p_\nu(A)\| \approx \tau, \tag{3.4}$$

so that further iterations with the same polynomial $p_\nu(z)$ will continue to reduce the residual. Of course, such a conclusion can never be guaranteed, for just as adaptive integrators can always be fooled by integrands with spikes in places that fail to get sampled, adaptive matrix iterators can always be fooled by initial residuals r_0 with small components in key directions. Nevertheless it is a reasonable hope that (3.4) may hold, provided that x_0 (or more precisely r_0) is chosen at random, and provided also that τ lies well enough below 1 so that $p_\nu(z)$ is forced to contain some genuine information about A. Probabilistic theorems to this effect could be proved.

Thus what GMRES "knows" about the matrix A at the end of Phase I, with a little luck, is no more and not much less than (3.4). It does not know anything very precise about the eigenvalues of A, and in particular, there is no reason to expect that the roots of $p_\nu(z)$ must always be good approximations to eigenvalues (though in some cases they will be). More generally, consider the family of lemniscates defined by

$$L_\alpha = \{ z \in \mathbb{C} : |p_\nu(z)| = \alpha \}, \qquad \alpha \geqq 0. \tag{3.5}$$

The set of roots of $p_\nu(z)$ is the same as the lemniscate L_0, which we have just claimed to be of little significance. But there is a choice of α of greater interest:

$$L_\tau = \text{"the GMRES lemniscate."}$$

Roughly speaking, the domain enclosed by L_τ is GMRES's best concept at step ν of the effective spectrum of A. (We hope to make this statement more precise in later work.) In running our hybrid algorithm, we have found it informative to plot L_τ at the end of Phase I (by sampling $\log |p_\nu(z)|$ on a grid and calling a contour plotter). On the same plot we generally display the zeros of $p_\nu(z)$ and also the lemniscate L_1 that passes through the origin. These plots of lemniscates give a graphic indication of the manner in which A may be causing difficulty, and in the practical world this translates into guidance in the design of preconditioners.

For example, consider the following banded Toeplitz matrix investigated by Grcar [12]:

$$A = \begin{pmatrix} 1 & 1 & 1 & 1 & & & \\ -1 & 1 & 1 & 1 & 1 & & \\ & -1 & 1 & 1 & 1 & 1 & \\ & & -1 & 1 & 1 & 1 & 1 \\ & & & \ddots & \ddots & & \ddots \end{pmatrix} \qquad (N \times N). \tag{3.6}$$

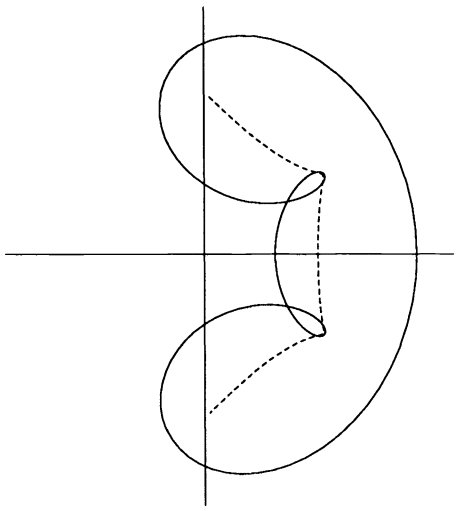


FIG. 3.1. *Spectrum (along the dashed curve) and ε-pseudospectrum (enclosed by the solid curve) of the matrix A of* (3.6), *for large N and small ε. (The figure looks approximately the same for any N and ε satisfying, say,* $e^{-N/50} < \varepsilon < 1/50$. *See* [26].)

Like (2.2), this is a matrix whose effective spectrum is quite different from its spectrum. In this case $\Lambda_{\text{practical}}$ is the region of the complex plane enclosed by $f(S)$, where S is the unit circle and

$$f(z) = -z^{-1} + 1 + z + z^2 + z^3 \tag{3.7}$$

for large N. Figure 3.1 shows Λ and $\Lambda_{\text{practical}}$ for this matrix, assuming N is close to ∞, and Fig. 3.2 shows the lemniscates L_τ computed by GMRES at steps $n = 2$, 3, 6, and 20

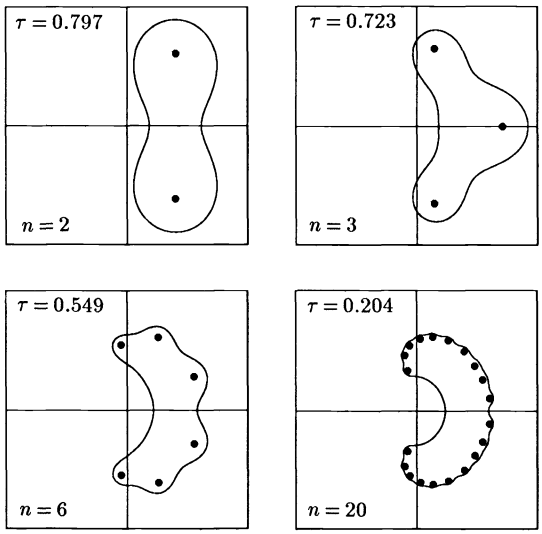


FIG. 3.2. GMRES *zeros and lemniscate for the same matrix at steps* $n = 2, 3, 6, 20$, *with* $N = 200$. *Each square represents the domain* $[-5, 5] \times [-5, 5]$. *The lemniscates approximate the pseudospectra, not the spectrum.*

for the case $N = 200$. Clearly, GMRES has done quite a good job of locating $\Lambda_{\text{practical}}$. It neither can nor should locate the exact spectrum Λ. Note that for this example the convex hull of $\Lambda_{\text{practical}}$ encloses the origin, which means that ChebyCode and some of the other existing hybrid algorithms would fail.

To close this section, let us return to the 2×2 example (2.8) that breaks an Arnoldi hybrid algorithm. In the first step GMRES makes no progress whatever with this matrix, and the corresponding residual polynomial is $p_1(z) = 1$. Thus after one step we have

$$\text{Arnoldi: } p_1(z) = 1 - z/0,$$

$$\text{GMRES: } p_1(z) = 1 - z/\infty.$$

To speak conventionally in terms of eigenvalues, we might say that GMRES has chosen the only other possible eigenvalue estimate besides 0 that is symmetric with respect to the spectrum $\pm i$, namely, ∞; the equation analogous to (2.9) is

$$\lambda_{\text{GMRES}} = \frac{r_0^T A^T A r_0}{r_0^T A r_0} = \infty. \tag{3.8}$$

This choice has made all the difference, however, since it has led to a polynomial $p_1(z)$ that is finite rather than infinite. Brown has pointed out that this phenomenon is general: when the Arnoldi iteration divides by 0, GMRES stagnates harmlessly, and vice versa [3]. Thus, although the performance of our hybrid algorithm can certainly be disappointing, if GMRES converges slowly or if (3.4) fails to hold and some kind of restart is necessary, the finiteness of $\|p_\nu(A)\|$ implies that at least it can never break down.

We have now presented a number of arguments in support of the view that the residual polynomial $p_\nu(z)$ in a hybrid algorithm should be derived from the GMRES method rather than from Arnoldi eigenvalue estimates. This idea also appears to be supported by numerical experiments. Throughout our computations for this project we have subjected each example matrix to two versions of our program, one based on GMRES and the other on Arnoldi. Each of the two methods sometimes outperforms the other, but the Arnoldi variant usually converges more slowly, and it fails considerably more often. (Of course, a failure is not absolute; with a robust implementation it will mean a return to Phase I as described in § 7.) A few comparisons of this sort are reported in Fig. 8.8, below.

4. Construction of $p_\nu(z)$. Our implementation of the hybrid GMRES algorithm calculates the coefficients of $p_\nu(z)$ explicitly. We have not investigated the stability of this procedure, and it may be that there are better ways to find the roots of $p_\nu(z)$, for example, by solving an eigenvalue or generalized eigenvalue problem (see the additional remarks on stability in the next section). However, computational experience indicates that the explicit approach works well in practice.

Here is how the coefficients are determined. Let K_n denote the $N \times n$ matrix of Krylov vectors

$$K_n = (r_0 \quad Ar_0 \quad \cdots \quad A^{n-1} r_0). \tag{4.1}$$

The Arnoldi/GMRES process constructs an $N \times n$ matrix of orthonormal vectors spanning the same space

$$V_n = (v_1 \quad v_2 \quad \cdots \quad v_n), \tag{4.2}$$

by applying the iterative formula

$$v_{n+1} = h_{n+1,n}^{-1}(Av_n - V_n h_n), \qquad h_n = (h_{1n}, \cdots, h_{nn})^T, \tag{4.3}$$

where the numbers h_{ij} are the elements of a Hessenberg matrix of inner products. (We use the same notation as in [29].) Since the columns of V_n and K_n span the same space for each n, we have that

$$V_n = K_n C_n$$

for some upper-triangular matrix

$$C_n = \begin{pmatrix} c_{11} & \cdots & c_{1n} \\ & \ddots & \vdots \\ & & c_{nn} \end{pmatrix}. \tag{4.4}$$

This matrix is not formed during the GMRES iteration as presented in [29], but to find $p_\nu(z)$ explicitly we will need it. The appropriate formula comes from (4.3):

$$\begin{pmatrix} c_{1,n+1} \\ \vdots \\ c_{n+1,n+1} \end{pmatrix} = h_{n+1,n}^{-1} \begin{pmatrix} 0 \\ c_{1,n} \\ \vdots \\ c_{n,n} \end{pmatrix} - h_{n+1,n}^{-1} \begin{pmatrix} C_n h_n \\ 0 \end{pmatrix}. \tag{4.5}$$

By inserting the calculation (4.5) in the GMRES iteration, we generate the elements of C_n column by column as the iteration proceeds.

Having solved a Hessenberg least-squares problem at step $n = \nu$, GMRES produces an iterate x_ν of the form

$$x_\nu = x_0 + V_\nu y \tag{4.6}$$

for some vector y of dimension ν. Since $V_\nu y = K_\nu C_\nu y$, it follows that the vector $C_\nu y$ contains the coefficients of the polynomial $q_{\nu-1}(z)$ of (1.2):

$$C_\nu y = (\alpha_0, \cdots, \alpha_{\nu-1})^T, \qquad q_{\nu-1}(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_{\nu-1} z^{\nu-1}. \tag{4.7}$$

Since $p_\nu(z) = 1 - z q_{\nu-1}(z)$, this gives us the coefficients of $p_\nu(z)$ as well.

5. Richardson iteration for Phase II. Phase I is complete and we have determined the polynomials $q_{\nu-1}(z)$ and $p_\nu(z)$ implicit in the GMRES iteration. We now face the question of how best to re-apply these polynomials for the further iterations of Phase II.

As mentioned in the Introduction, many ideas have been advanced for this phase of a hybrid algorithm, of which one of the simplest is the Horner iteration of Elman and Streit [8]. From (1.2) and (4.7) we have that

$$x_n - x_0 = q_{n-1}(A) r_0, \qquad q_{n-1}(z) = \alpha_0 + \alpha_1 z + \cdots + \alpha_{n-1} z^{n-1}, \tag{5.1}$$

and therefore $x_n - x_0$ is the final result w of the loop

$$\begin{aligned} &w := \alpha_{n-1} r_0 \\ &\textbf{For } i := 1 \textbf{ to } n-1 \\ &\qquad w := A w + \alpha_{n-i-1} r_0. \end{aligned} \tag{5.2}$$

In our experiments this method has worked quite well. So has a related and even simpler method in which one forms $q_{\nu-1}(A) r_0$ as a student would do who had never heard of Horner's rule—for the familiar factor-of-2 advantage of the Horner formula vanishes when we are dealing with matrices rather than scalars. The disadvantage of such approaches is that the intermediate steps may correspond to residuals so large that information may be lost due to rounding errors, though this has not troubled us in practice.

The alternative we have preferred is to factor $p_\nu(z)$ numerically,

$$p_\nu(z) = \prod_{i=1}^{\nu} (1 - z/\zeta_i), \tag{5.3}$$

and then carry out a first-order Richardson iteration[4] along the lines of Smolarski and Saylor [34], [35]:

$$\begin{aligned} &\textbf{For } j := 1 \textbf{ to } \nu \\ &\qquad x_j := x_{j-1} + r_{j-1}/\zeta_j. \end{aligned} \tag{5.4}$$

(One could also formulate the calculation in terms of $q_{\nu-1}(z)$ rather than $p_\nu(z)$ by means of the "grand leap" iteration described in [31], a method that can be slightly more efficient than (5.4).) The reader may object that finding the roots of a polynomial is an ill-conditioned problem, so that incorporating a root-finding step in a hybrid algorithm is likely to make the algorithm unstable. Though we have not yet analyzed the matter fully, we believe that this concern is about half justified. On the one hand, ill-conditioning in the rootfinding problem per se is probably not important, for the success of our algorithm ultimately depends on the size of $p_\nu(z)$ in the complex plane, not the locations of its roots. On the other hand, the size of $p_\nu(z)$ is itself an ill-conditioned function of its coefficients in general. Thus there is a stability issue, but it lies not in the rootfinding but in the representation of $p_\nu(z)$ by its coefficients in the basis of monomials, as alluded to at the beginning of § 4. The ideal hybrid algorithm might begin by constructing a more stable basis in which to represent $p_\nu(z)$. We do not know how worthwhile this extra complication would be in practice.

There is still another question of stability to be addressed. The factorization (5.3) offers a choice of the order in which to label the roots ζ_j, and as discussed first by Young and more recently by Anderssen and Golub [1], Fischer and Reichel [9], [25], and Tal-Ezer [37], this ordering is important for stability. The reason is that although the final result $p_\nu(A)$ will be small in exact arithmetic, floating-point errors may destroy this property unless (approximately speaking) the intermediate products $p_j(A)$ also are reasonably small. This issue is not academic; the factors at stake are potentially enormous.

How can an ordering of the roots $\{\zeta_j\}$ be efficiently selected to ensure that the intermediate products $p_j(A)$ are small? Our choice has been the *weighted Leja ordering* described in [25], which is defined by the condition

$$|\zeta_j| \prod_{i=1}^{j-1} |\zeta_j - \zeta_i| = \max_{j \le l \le \nu} |\zeta_l| \prod_{i=1}^{j-1} |\zeta_l - \zeta_i|, \qquad j = 1, 2, \cdots, \nu - 1, \tag{5.5}$$

assuming the points ζ_j are distinct. (At the first step (5.5) reduces to the condition $|\zeta_1| = \max_{1 \le l \le \nu} |\zeta_l|$.) The idea behind this ordering is that it tends to approximately *equidistribute* the points ζ_j in the sense of potential theory. The Leja ordering is easy to calculate, and in the examples we have looked at, it performs dramatically better than more elementary alternatives.

The Richardson iteration with Leja ordering also has the appealing property that since the polynomials $p_n(A)$ tend to decrease steadily in norm, the Phase II iteration can be meaningfully stopped at any point rather than just at the end of a cycle of ν steps.

---

[4] When A is real, the introduction of complex arithmetic by a complete factorization of $p_\nu(z)$ is unnecessary. One can factor it instead into linear and quadratic terms with real coefficients and obtain a Richardson iteration with steps of both first and second order. See [22], [31], or [34] for details.

It is not hard to argue that (5.5) cannot be exactly the right ordering condition to impose in all cases. For example, this algorithm has a sensitivity to multiple points ζ_j that is unnatural and that in contrived examples may cause instability. A more perfect, if more complicated, ordering algorithm might involve the minimization of an appropriate Leja product not just over the points $\zeta_j, \cdots, \zeta_\nu$ but over some approximation to the lemniscate L_τ. Nevertheless, our experience indicates that the Leja ordering is a reliable solution to the instability problem in the great majority of cases.

6. Switching criterion; behavior of the idealized hybrid GMRES algorithm. The principal feature of our algorithm that we have not yet specified is when Phase I should be terminated—in other words, the choice of ν. Optimizing this decision is a complicated matter, for it depends on both the problem and the computing environment. For example, if matrix-vector products are far more time consuming than other operations and plenty of storage is available, then one might as well stay in Phase I forever with a "GMRES(∞)" iteration. On the other hand, if storage is so limited that only a few vectors can be retained, then GMRES(∞) is out of the question and one must switch quickly to Phase II. Considerations such as these suggest that to a certain extent users of a hybrid GMRES algorithm will inevitably have to make some of the decisions themselves if the aim is optimal performance.

More can be said, however, if we are willing to make some simplifying assumptions. In particular, let us assume that storage is unlimited, so that the only goal is to minimize the computing time. Assume further that only operations on N-vectors are significant, and define a *vector operation*, our fundamental work unit, to be the cost of an "axpy" operation $ax + y$ involving a scalar a and N-vectors x and y. Finally, assume also that

(6.1) one matrix-vector multiplication costs δ vector operations

for some $\delta \geqq 0$. For a sparse matrix on a serial computer, δ is approximately the average number of nonzero elements per row.

These assumptions are mechanical ones, whose degree of validity depends on straightforward factors readily checked. To motivate our choice of ν, we are now going to make two further highly idealized assumptions that are in another category entirely—approximately true in some cases, perhaps, but sometimes far from true, almost never true exactly, and in any case unverifiable. First, we assume that the GMRES iteration of Phase I accelerates as it proceeds, or at worst, converges steadily:

$$\text{(6.2)} \qquad \text{Phase I:} \quad \frac{\|r_{n+m}\|}{\|r_0\|} \leqq \frac{\|r_n\|}{\|r_0\|}\frac{\|r_m\|}{\|r_0\|} \quad \text{for all } m, n \geqq 0.$$

(If $\|r_n\|/\|r_0\| = \|p_n(A)\|$ at each step n, as in (3.4), then (6.2) follows as a corollary, but in general we only have $\|r_n\|/\|r_0\| \leqq \|p_n(A)\|$.) Second, we assume that the Richardson iteration of Phase II converges steadily at exactly the same rate as in Phase I:

$$\text{(6.3)} \qquad \text{Phase II:} \quad \frac{\|r_{k\nu}\|}{\|r_0\|} = \left(\frac{\|r_\nu\|}{\|r_0\|}\right)^k \quad \text{for all } k \geqq 0.$$

Our strategy for choosing ν is motivated by the following idea:

(6.4) Goal: equal amounts of work in Phase I and Phase II.

Figure 6.1 explains the thinking behind (6.4) by illustrating the kind of convergence behavior we are hoping for under idealized circumstances. The hybrid algorithm cannot take fewer iterations than GMRES(∞), but with luck it will take nearly as few. If ν is large this will correspond to a large reduction in the total computing time.

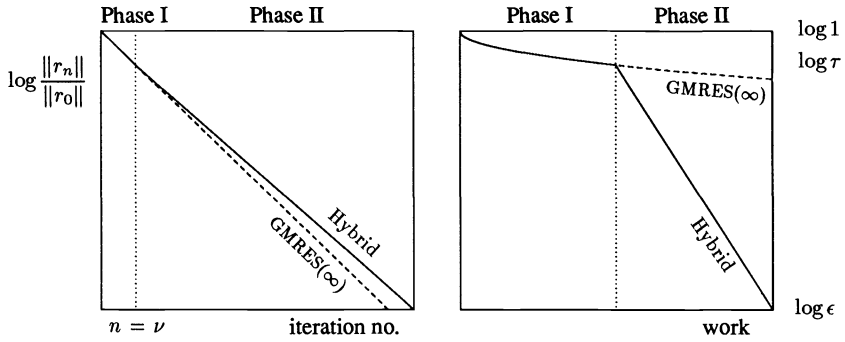


FIG. 6.1. *Convergence of the hybrid* GMRES *algorithm under idealized circumstances as a function of iteration number and computing time. The switchover step* ν *is determined by the condition that equal amounts of time are spent in Phase* I *and Phase* II. *The resulting factor of improvement over* GMRES(∞) *is* $O(\nu^2)$.

Now we work out the algebra required to implement (6.4). Suppose we have just completed step ν of Phase I, the residual has been reduced by the factor of

$$\frac{\|r_\nu\|}{\|r_0\|} = \tau, \tag{6.5}$$

and our desired accuracy is:

$$\text{convergence tolerance:} \quad \frac{\|r_{\text{final}}\|}{\|r_0\|} = \varepsilon \qquad (\varepsilon < \tau). \tag{6.6}$$

According to estimates in [29], the work performed so far is

$$\text{Phase I work:} \quad \nu(\nu + 3 + \delta) \text{ vector operations.} \tag{6.7}$$

On the other hand in the Richardson iteration of Phase II the work per step will be $1 + \delta$ vector operations, and by (6.3), the total number of steps to convergence will be $\nu \log \varepsilon / \log \tau$—hence in Phase II, $\nu(\log \varepsilon / \log \tau - 1)$. This implies that

$$\text{Phase II work:} \quad \nu(1+\delta)\left(\frac{\log \varepsilon}{\log \tau} - 1\right) \text{ vector operations.} \tag{6.8}$$

The condition (6.4) can be realized by equating the right-hand sides of (6.7) and (6.8):

$$\text{switching condition:} \quad \nu + 3 + \delta = (1+\delta)\left(\frac{\log \varepsilon}{\log \tau} - 1\right). \tag{6.9}$$

To summarize, here is how we decide when to terminate Phase I. During Phase I the left-hand side of (6.9) increases monotonically and the right-hand side decreases monotonically (because τ is decreasing). We switch to Phase II as soon as the left-hand side exceeds the right-hand side.

Besides its aesthetic appeal, the condition (6.4) has some more solid justification. In particular, we have the following theorem.

THEOREM 1. *Suppose the assumptions above hold, including* (6.2) *and* (6.3), *and let the transition from Phase* I *to Phase* II *be determined by* (6.9). *Then the hybrid algorithm converges, and no other choice of* ν *could have reduced the computing time by more than a factor of two.*

Proof. Increasing ν can shorten the computation only by reducing the length of Phase II, and since Phase II consumes only half of the computing time, the improvement

can be at most a factor of two. On the other hand decreasing ν would mean entering Phase II with an inferior polynomial $p_\nu(z)$ and further to travel with it, by (6.2) and (6.3), so that the work in Phase II would have to increase. Since that work is already half of the total, the maximum possible improvement is again a factor of two. □

So long as the assumptions (6.2) and (6.3) hold, the reasoning above shows that our strategy for choosing ν is actually optimal in the sense that any other choice might lead to a penalty of a factor greater than two.

7. Practical safeguards. In practice, of course, our idealized assumptions do not always hold. They may fail in several ways, and one of these is particularly important: equation (3.4) may fail, leaving us with

$$\tau = \frac{\|r_\nu\|}{\|r_0\|} \ll \|p_\nu(A)\|. \tag{7.1}$$

In such circumstances (6.3) will be far from satisfied, and the Richardson iteration of Phase II may converge much more slowly than expected or may not converge at all. The reason why (7.1) may occur is that the GMRES algorithm depends upon the particular initial residual r_0 and, consequently, the coefficients of $p_\nu(z)$ are affected by which components happen to be well represented in that vector.

There are several ways in which one might modify the hybrid algorithm to try to minimize the risk of occurrence of (7.1). For example, one might impose a threshold value of τ—insist that switchover to Phase II not take place until $\|r_n\|$ has been reduced by a factor of at least, say, 2. Or one could monitor the details of convergence in Phase I more carefully than we have proposed, forbidding switchover until some evidence has accumulated that the rate of convergence is steady. Another, more expensive, idea would be to apply Phase I to two or more independent vectors r_0 in parallel—"block GMRES." This would lead to a more reliable polynomial $p_\nu(z)$, though the extra work would be partly wasted since the residual r_n of actual interest would not be reduced. For problems with multiple right-hand sides, however, such an idea would be natural.

But there is a more fundamental implication of (7.1), and that is that any robust hybrid iterative code must include safeguards for coping with failure. If the convergence of the Phase II iteration proves unsatisfactory, there are various actions that may be taken. The simplest might be to restart the hybrid algorithm entirely from scratch from the current best available solution x_n. This would mean throwing away the information obtained in the GMRES steps already carried out, but if $p_\nu(z)$ has performed disappointingly, one might argue that that information is unreliable anyway.

The approach we have used instead is to return to the original GMRES iteration of Phase I and resume that iteration where it was interrupted. Returning to Phase I in this way is an easy matter if one has retained the necessary vectors in storage. Once a new polynomial $p_{\nu'}(z)$ is obtained that is deemed to be substantially better than the old one, we cycle back again to Phase II. To be precise, here is our current scheme, whose effects in one example can be seen in Fig. 8.6 below:

1. If any cycle of ν steps of Phase II reduces $\|r_n\|$ by a factor less than $\sqrt{\tau}$—that is, if the convergence is more than twice as slow as expected—return to Phase I.
2. Carry out additional GMRES steps $\nu + 1, \nu + 2, \cdots, \nu'$ of Phase I until the total computing time in Phase I has doubled, and calculate a new polynomial $p_{\nu'}(z)$.
3. Begin a new Phase II iteration with the new polynomial $p_{\nu'}(z)$, starting from the previous best value x_n, which will come either from the previous Phase II if the convergence there was slow but positive, or from the new Phase I if there was actual divergence in the previous Phase II.

Since this algorithm reverts to Phase I whenever the convergence of Phase II is going badly, it can never do much worse than GMRES(∞), as stated in the following theorem.

THEOREM 2. *The hybrid algorithm with the safeguards described above always converges and never requires more than three times as much computer time as* GMRES(∞).

Sketch of proof. The factor of 3 is attained if the first Phase II computation proceeds twice as slowly as expected in a case where GMRES(∞) would have converged to the desired precision at step $\nu + 1$. Careful consideration of the details of the algorithm, which we shall omit, shows that further cycling between Phase II and Phase I never leads to a factor greater than 3. □

Of course we generally expect convergence much faster than for GMRES. We remind the reader that Theorem 2 depends on our assumption that storage is not limited, so that the hybrid algorithm can be implemented as described. It also ignores rounding errors.

The details of the safeguarding procedure proposed above are arbitrary. There are many other ways to make a hybrid scheme robust, and we hope to have more to say on the subject in the future.

8. Numerical experiments. Three sorts of problems are chosen most often for testing numerical algorithms: realistic, artificial, and random. Realistic test problems have the advantage that they are tied directly to applications and thus, in a sense, are most reliable. Artificial problems have the advantage that they can be made cleaner and more extreme in their behavior, so that they provide more insight into fundamentals. As for random problems, they also have advantages in some contexts, but not here, for none of the known nonsymmetric matrix iterations beat the $O(N^3)$ (serial) performance of direct methods for random matrices [20]. In other words, iterative methods are useful only for matrices with special properties, which they typically acquire through preconditioning.

In this section we apply our hybrid algorithm to some test problems of the artificial kind and illustrate some of its good and bad properties in the process. We hope to investigate more realistic problems in the future.

Each of our experiments compares four algorithms:

1. Hybrid GMRES (solid curves),
2. Restarted GMRES(ν) (solid curves),
3. CGN (dashed curves),
4. CGS (dots).[5]

So far as we know, these are the best matrix iterations available[6] [20]. To keep the comparison simple, our restarted algorithm is GMRES(ν), where ν is the same switchover step number determined adaptively by the hybrid algorithm. Thus our restarted and hybrid GMRES iterations are identical for the first ν steps, and from that point on the hybrid algorithm re-applies the same residual polynomial $p_\nu(z)$ cyclically, while the restarted algorithm finds a succession of new optimal polynomials of degree ν. Except in Fig. 8.6, all of the hybrid results shown come from the idealized algorithm described in § 6, with none of the safeguards mentioned in § 7.

In each experiment the dimension is $N = 1000$ (except as noted), the convergence tolerance is $\varepsilon = 10^{-5}$, and the right-hand side b and the initial guess x_0 are random real vectors with independent normally distributed elements.

[5] CGS convergence curves are often so erratic that they obscure the rest of the plot. To avoid this clutter without suppressing convergence rates that are often very impressive, we have settled for a single dot representing the residual at the end of each CGS iteration.

[6] See the further remarks about CGS in the Conclusions.

For each example we present three plots. The first plot shows $\log_{10} \|r_n\|$ as a function of the iteration number n. By this measure the hybrid algorithm can do no better than GMRES(ν), whereas CGS and CGN may do better or worse depending on the matrix. The second plot shows $\log_{10} \|r_n\|$ as a function of work measured by vector operations, defined in § 6. By this measure the hybrid algorithm may outperform restarted GMRES by a factor as high as $O(\nu)$. The third plot shows the roots of $p_\nu(z)$ in the complex plane together with the associated GMRES lemniscate L_τ and the lemniscate L_1 passing through the origin. As mentioned in § 3, L_τ gives an indication of the spectrum or τ-pseudospectrum of A.

Example 1. Our first and simplest example, shown in Fig. 8.1, is the triangular Toeplitz matrix (2.2). This is an example where the hybrid algorithm outperforms GMRES(ν) very cleanly. Plotted against the step number, the GMRES(ν) residual converges smoothly and linearly and the hybrid algorithm lags a little behind. Plotted against work, however, that linear convergence curve becomes scalloped, a common phenomenon for restarted GMRES which reflects the fact that later cycles tend to waste time redetermining information that was already obtained in earlier cycles. The hybrid algorithm now does much better, achieving rapid and steady convergence after the point of switchover. In fact, the figure matches the idealized curves of Fig. 6.1 remarkably well.

A comparison of Figs. 2.1 and 8.1 reveals that GMRES has done a good job of locating the τ-pseudospectrum of A.

In this example the hybrid algorithm is the fastest of the four algorithms asymptotically and is roughly tied with CGN for the specified tolerance $\varepsilon = 10^{-5}$. CGS converges erratically and somewhat more slowly. GMRES(ν) converges much more slowly.

Example 2. A similar but somewhat more complicated Toeplitz example is the Grcar matrix (3.6) (Fig. 8.2). As mentioned above, this is a case where ChebyCode and some of the other hybrid algorithms would fail since the pseudospectrum does not lie in a half-plane. Again the hybrid GMRES algorithm substantially outperforms GMRES(ν). CGS does about equally well. CGN does much better, however, because this is a matrix whose singular values (smoothly distributed in the interval [0.89, 3.24]) are much better behaved than its eigenvalues and pseudo-eigenvalues (encircling the origin).

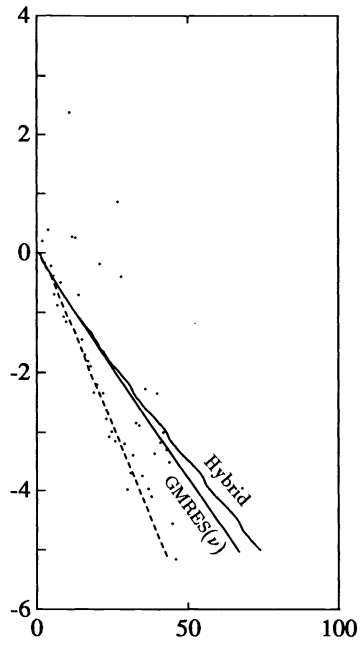
Example 3. For an example in which CGN does poorly, consider the tridiagonal Toeplitz matrix (Fig. 8.3)

$$A = \begin{pmatrix} 5.1 & 3 & & & & \\ 2 & 5.1 & 3 & & & \\ & 2 & 5.1 & 3 & & \\ & & 2 & 5.1 & 3 & \\ & & & 2 & 5.1 & \end{pmatrix} \quad (1000 \times 1000). \tag{8.1}$$

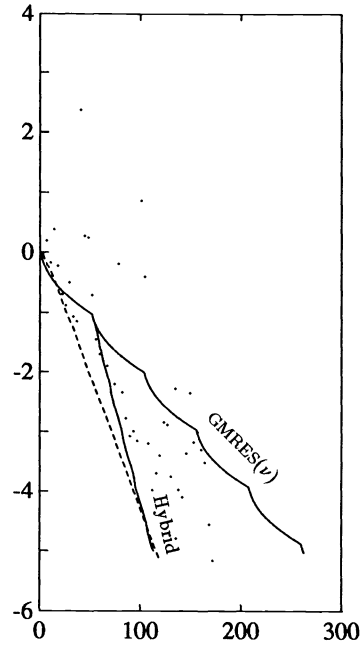
The symbol of this matrix is $f(z) = 2z^{-1} + 5.1 + 3z$, which maps the unit circle into an ellipse whose intersection with the real axis is [0.1, 10.1]. Consequently the condition number is $\kappa \approx 101$ for large N, and since the spectrum and pseudospectra do not wrap around the origin, the Krylov subspace iterations do relatively well.

In this example the convergence of the Richardson iteration of Phase II is disappointing; (6.3) does not hold very closely. Nevertheless, the plot of $\|r_n\|$ against work reveals that the hybrid iteration is the fastest. The GMRES lemniscate closely matches the elliptical pseudospectrum.

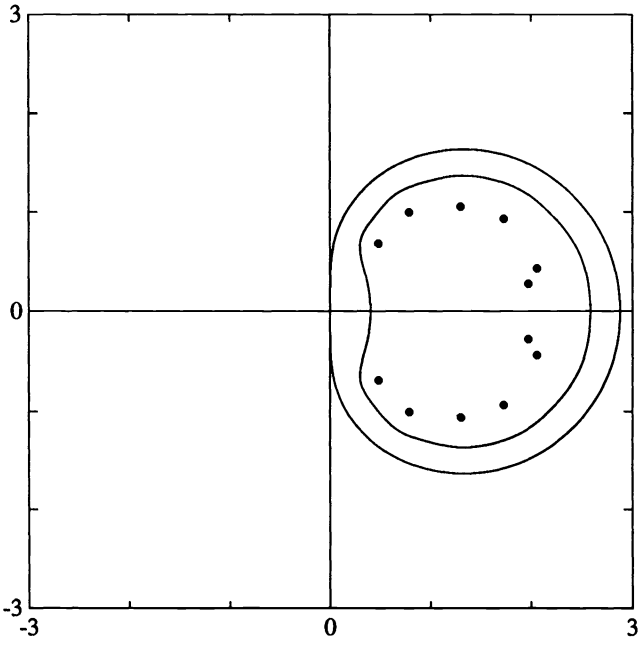
Example 4. Our fourth example is bidiagonal but not Toeplitz (Fig. 8.4). On the diagonal this matrix has the elements $.5 - \omega^1, \cdots, .5 - \omega^{1000}$, where $\omega = -.5 + i\sqrt{3}/2$ is a cube root of unity. The superdiagonal contains uniformly distributed random



(a) $\log_{10} \|r_n\|$ *vs.* n

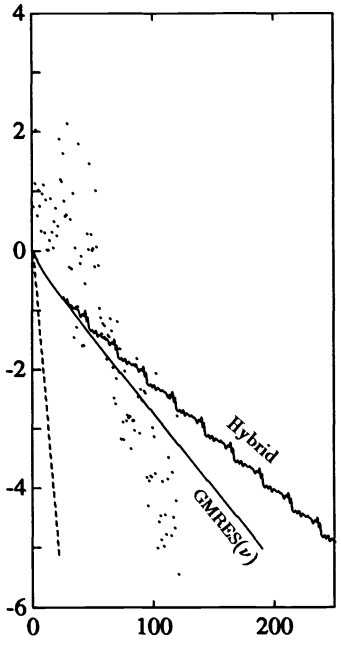


(b) $\log_{10} \|r_n\|$ *vs. work*

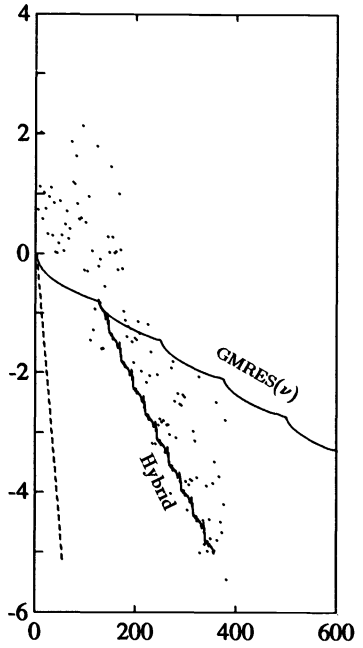


(c) GMRES *lemniscate at step* $\nu = 12$

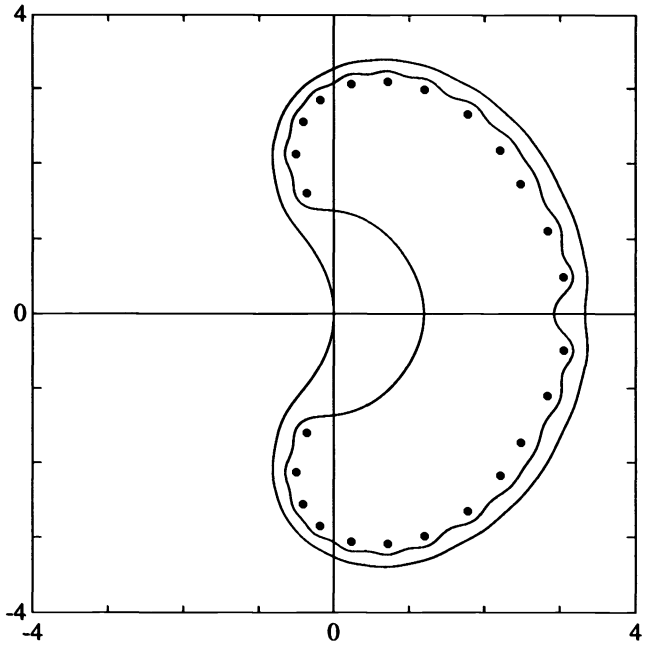
FIG. 8.1. *Example* 1: *the Toeplitz matrix* (2.2). *The* CGN *and hybrid* GMRES *algorithms are the winners. For this and the subsequent examples, the dimension is* $N = 1000$, *except as noted.*



(a) $\log_{10} \|r_n\|$ *vs.* n

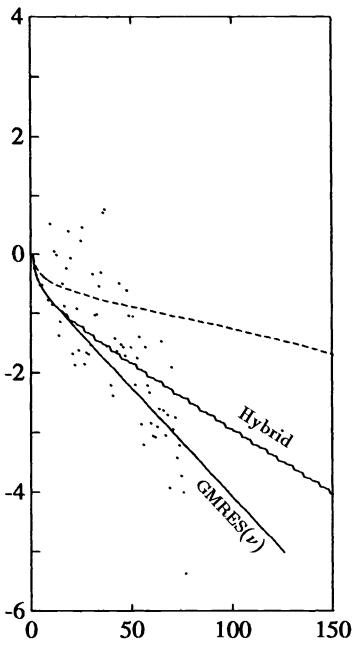


(b) $\log_{10} \|r_n\|$ *vs. work*

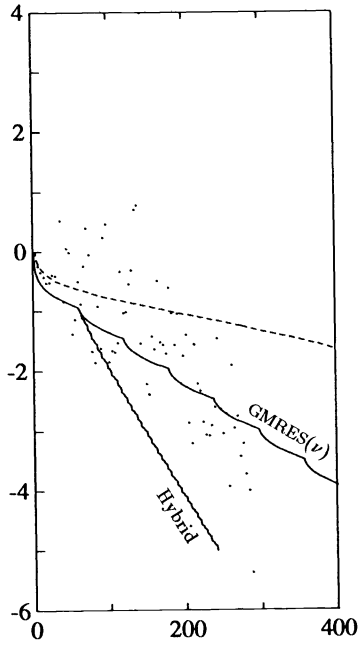


(c) GMRES *lemniscate at step* $\nu = 24$

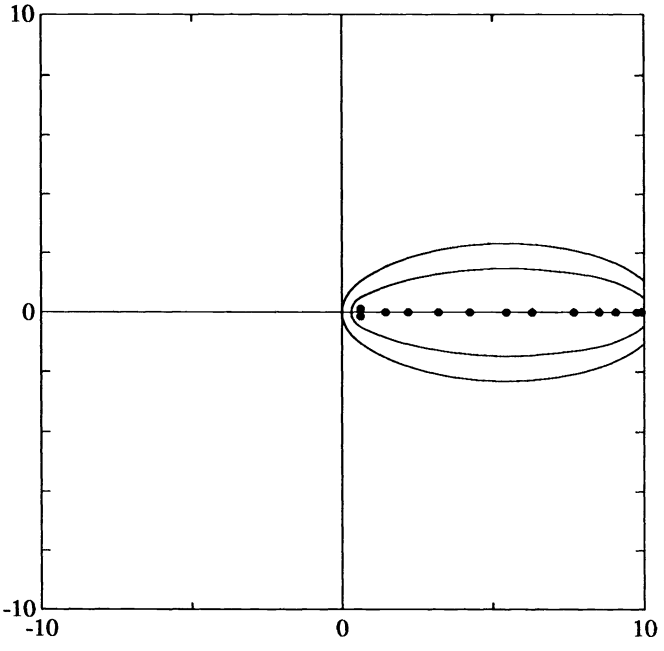
FIG. 8.2. *Example* 2: *Grcar's Toeplitz matrix* (3.6). *The hybrid* GMRES *algorithm again beats* GMRES(ν), *but* CGN *does much better.*



(a) $\log_{10} \|r_n\|$ *vs. n*

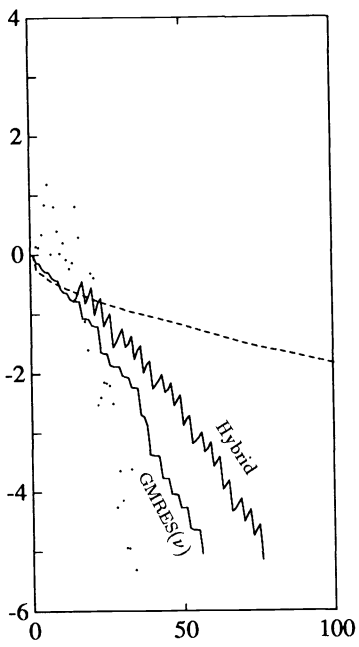


(b) $\log_{10} \|r_n\|$ *vs. work*

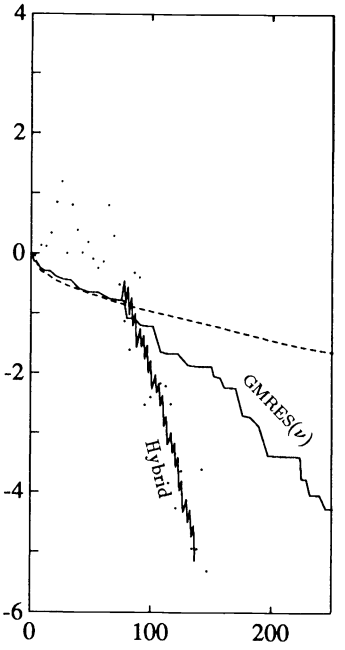


(c) GMRES *lemniscate at step* $\nu = 13$

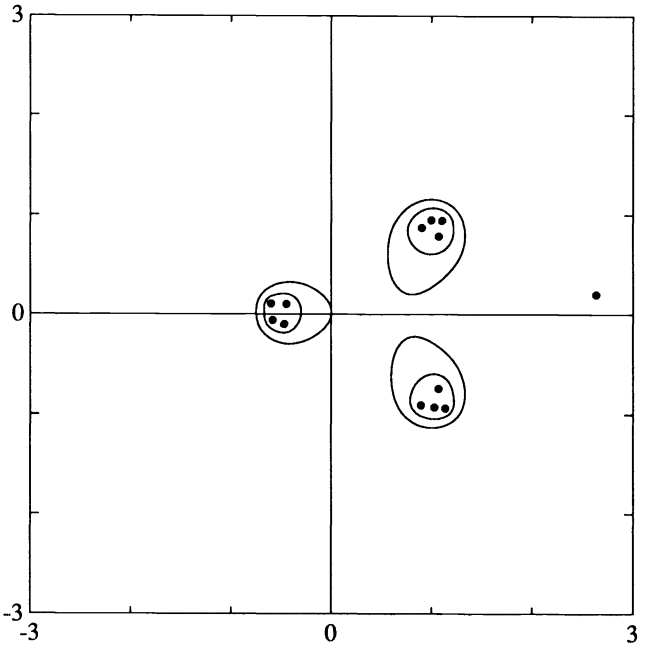
FIG. 8.3. *Example* 3: *the tridiagonal Toeplitz matrix* (8.1). *This matrix has condition number* $\kappa \approx 101$, *and* CGN *converges much more slowly than the other iterations. The hybrid* GMRES *algorithm is the winner.*



(a) $\log_{10} \|r_n\|$ *vs.* n



(b) $\log_{10} \|r_n\|$ *vs. work*



(c) GMRES *lemniscate at step* $\nu = 13$

FIG. 8.4. *Example* 4: *a bidiagonal matrix with three distinct eigenvalues. The* CGS *and hybrid* GMRES *algorithms are the most efficient.*

numbers in the interval [0, 1.5]. This is a matrix whose spectrum consists of just three points but whose pseudospectra are larger domains surrounding those points. The singular values are not very tightly clustered, and we find that CGN converges more slowly than any of the other algorithms. Among the three Krylov subspace iterations the hybrid GMRES iteration does best.

Figure 8.4(c) reveals an outlying root of $p_\nu(z)$ that is worth a comment. Since the pseudospectra have approximate three-fold symmetry but ν (=13) is not divisible by three, it is not surprising that one of the linear factors of the residual polynomial should be nearly useless (compare Fig. 2.2). The GMRES lemniscate L_τ contains a very small lobe near the outlying root (too small to be apparent in the picture), and thus this example illustrates that the connection of L_τ with a pseudospectrum of A is not perfect. The outlying root does no harm to the hybrid iteration, however.

Example 5. Finally, we give an example in which the hybrid GMRES algorithm performs poorly, at least in its idealized form described in § 6. Let A be a diagonal matrix of dimension $N = 1001$ whose diagonal entries are complex numbers lying on the unit semicircle in the right half-plane. Rather than a uniform distribution of points along the semicircle with respect to arc length, we take a uniform distribution with respect to the imaginary coordinate,

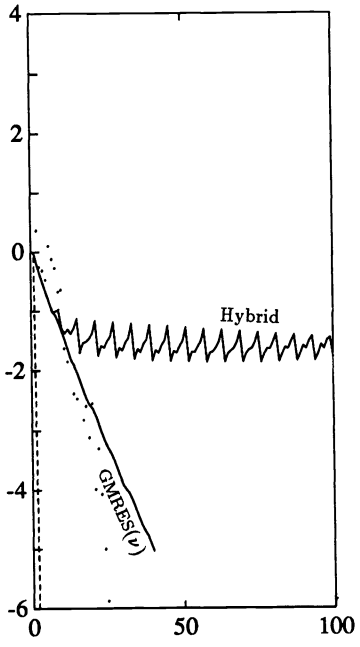
$$a_{jj} = e^{i\theta_j}, \quad \theta_j = \sin^{-1}((j-501)/500), \quad 1 \leqq j \leqq 1001.$$

These points are sparsely located near $\pm i$, and as a result, for most initial residuals r_0, GMRES can reduce the residual significantly without going to the considerable trouble of making $|p_\nu(z)|$ substantially smaller than 1 near $z = \pm i$. This is exactly what is revealed in Fig. 8.5. Assumption (6.3) does not hold closely, and we end up with a Phase II iteration that makes little progress. GMRES(ν) beats the hybrid algorithm by a large factor, and CGS does even better. Since the singular values are all equal to 1, CGN converges in one step.

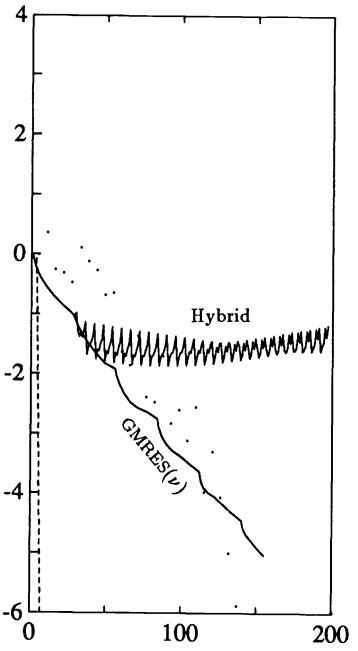
These observations, and Fig. 8.5, pertain to the idealized hybrid algorithm with none of the safeguards mentioned in § 7. In practice, of course, one would never permit so many iterations to be wasted in Phase II before returning to the GMRES iteration to get better information about A. In Fig. 8.6, we do this. The same example is run with the safeguarded hybrid algorithm described in § 7 and the convergence becomes acceptable. Note the plateau in Fig. 8.6(b), revealing a return to Phase I that generates an improved residual polynomial $p_{\nu'}(z)$ without reducing the best available residual.

This example is worth dwelling on because it reveals how important the quality of the information in r_0 is to achieving rapid convergence in Phase II. To put it succinctly, for hybrid iterative algorithms, *multiplicities matter*—even if the matrix is normal. Eigenvalues of higher multiplicities correspond to larger eigenspaces, so they tend to be better represented in a random initial vector, which increases their influence on $p_\nu(z)$. To demonstrate this, Fig. 8.7 repeats the computation of Fig. 8.5 for a new matrix, which is exactly the same as before, except that the multiplicities of the end eigenvalues $\pm i$ have been increased from 1 to 101. Thus the dimension of A is now 1201. The convergence of the hybrid algorithm without safeguards becomes quite rapid, and the explanation can be seen in the difference of the lemniscates L_1 in Figs. 8.5 and 8.7.

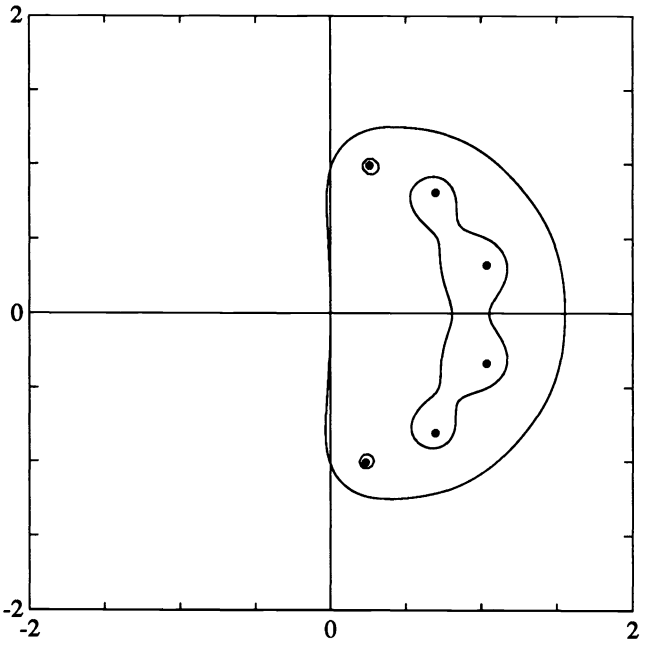
We close this section with four final examples to illustrate the difference between our hybrid GMRES algorithm, based on the residual polynomial $p_\nu(z)$ derived from GMRES, and a "hybrid Arnoldi" algorithm in which $p_\nu(z)$ is taken to be the normalized polynomial whose roots are the Arnoldi eigenvalue estimates at step ν. Figure 8.8 compares the convergence of these two algorithms for Examples 2, 3, 5, and 7 above. For Example 3 the Arnoldi variant is faster in Phase II by about 50 percent, but in our experience this



(a) $\log_{10} \|r_n\|$ *vs. n*

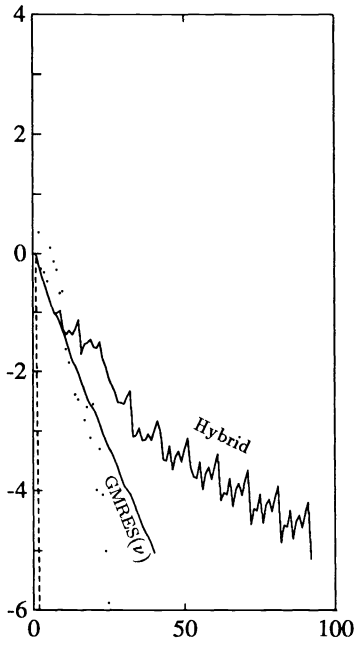


(b) $\log_{10} \|r_n\|$ *vs. work*

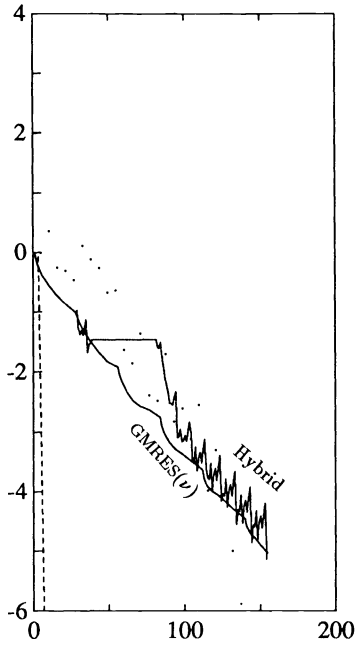


(c) GMRES *lemniscate at step* $\nu = 6$

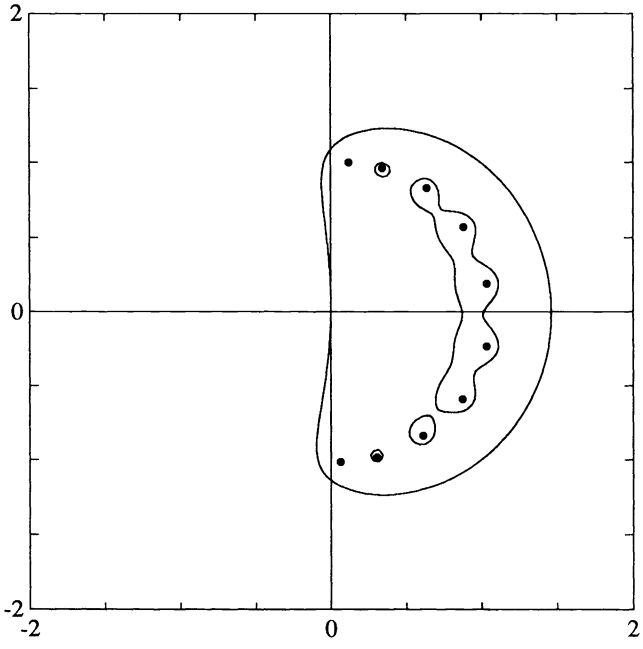
FIG. 8.5. *Example* 5: *a diagonal matrix of dimension* $N = 1001$ *with eigenvalues on the unit semicircle in the right half-plane. The hybrid algorithm without safeguards constructs a residual polynomial* $p_\nu(z)$ *that is not much smaller than* 1 *near* $z = \pm i$, *and the convergence is very slow.*



(a) $\log_{10} \|r_n\|$ *vs. n*

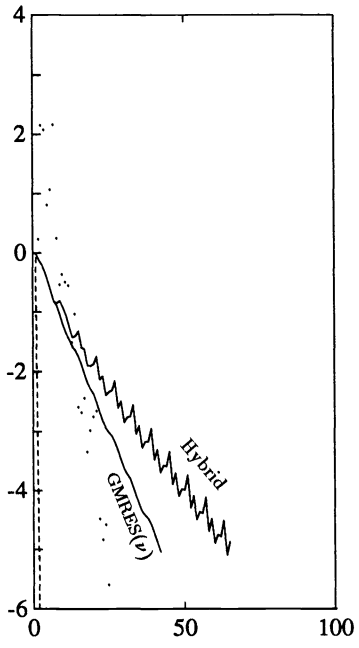


(b) $\log_{10} \|r_n\|$ *vs. work*

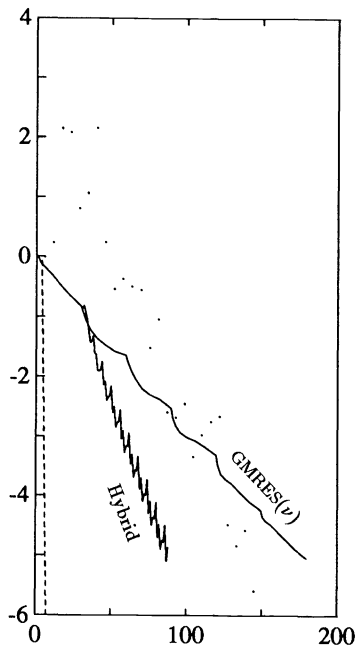


(c) GMRES *lemniscate at step* $\nu' = 10$

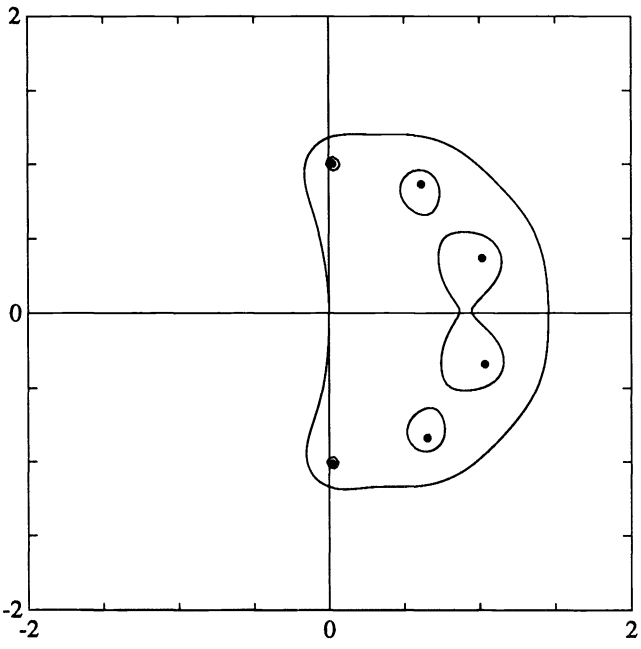
FIG. 8.6. *Example* 5 *again, but solved now with the safeguarded hybrid* GMRES *algorithm described in* § 7. *The algorithm returns to Phase* I *to get better information, and ends up solving the problem with reasonable efficiency.*



(a) $\log_{10} \|r_n\|$ *vs.* n



(b) $\log_{10} \|r_n\|$ *vs. work*



(c) GMRES *lemniscate at step* $\nu = 6$

FIG. 8.7. *Example* 5 *a third time, except that now, the matrix has been changed by increasing the multiplicities of the eigenvalues* $\pm i$ *from* 1 *to* 101, *so that the dimension becomes* 1201. *Now* $p_\nu(z)$ *contains better information, and the convergence of the hybrid algorithm, even without safeguards, is rapid.*

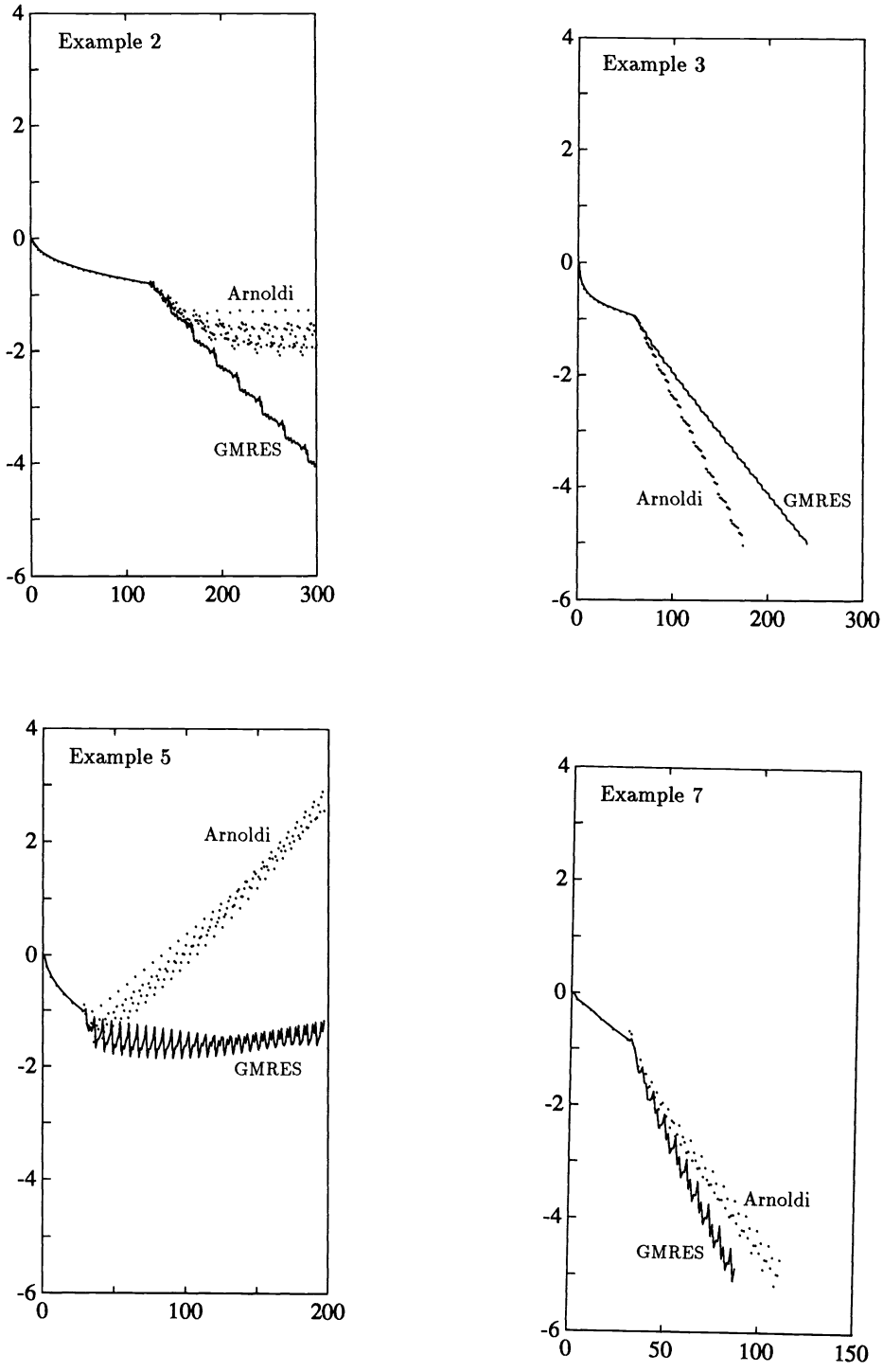


FIG. 8.8. *Comparison of the hybrid* GMRES *algorithm with a "hybrid Arnoldi" variant for Examples* 2, 3, 5, *and* 7 *above. Usually, though not always, the Arnoldi variant performs less well, for the reasons discussed in* § 2.

is not typical. More often it is slower, as in Example 7. In addition, as in Examples 2 and 5, it is not uncommon for the Arnoldi variant to stall, necessitating a return to Phase I that may not be required by the hybrid GMRES algorithm. In principle the hybrid Arnoldi algorithm can break down completely with a division by zero, as mentioned in § 2, but of course the probability of such an event is zero.

9. Conclusions. In conclusion, we would like to summarize the relationships as we see them between our hybrid GMRES algorithm and the four principal classes of competing algorithms for the iterative solution of nonsymmetric linear systems: the restarted and truncated Krylov space iterations such as GMRES(k) and ORTHOMIN(k); previous hybrid algorithms; the normal equations–conjugate gradients combination known as CGN; and the Lanczos-type algorithms such as CGS. We assume as usual that the cost of vector operations is significant enough that a "pure" Krylov space iteration such as GMRES(∞) is not competitive.

The comparison with the first two groups of alternatives turns on the question: how good is the information contained in the initial steps of an Arnoldi/GMRES iteration? The first group, the restarted and truncated algorithms such as GMRES(k), are motivated by the assumption that this information is not reliable and should be replaced regularly as the iteration proceeds even if this increases the work per step substantially. The second group, the existing hybrid algorithms summarized in our Introduction, are motivated by an opposite assumption: that initial Arnoldi/GMRES steps may produce information solid enough that it makes sense to perform further manipulations and "data compression" upon it, in particular, the solution of an approximation problem in the complex plane that typically leads to an iteration polynomial of lower degree.

Our hybrid GMRES algorithm entails an assumption intermediate between these two. It assumes that the information coming from Arnoldi/GMRES steps is too valuable to be discarded, but not so solid that further data compression is appropriate. Of course the validity of this assumption depends upon various factors, notably, the initial vector for the GMRES iteration and the choice of the switchover step ν. We believe that it is a reasonable assumption in many cases, however, and this view of the matter, combined with our numerical experiments, leads us to believe that for most problems our algorithm is faster than GMRES(k) and more robust than other hybrids.

The third comparison, with CGN, is relatively straightforward, at least in principle. The hybrid GMRES algorithm should be the winner when A is ill-conditioned, loosely speaking, or more precisely, when its squared singular values are less favorably distributed than its (pseudo-) eigenvalues in the sense described in [18].

In our opinion, the most serious competitors are the Lanczos-type algorithms such as CGS [4], whose work and storage requirements, unlike those of GMRES and ORTHOMIN, do not grow with the iteration number. These algorithms do not minimize anything, and their convergence is often quite erratic, but it is usually very fast. Most recently (since the time when this manuscript was first submitted for publication), algorithms in this class with less erratic convergence curves have been developed by Freund [10] and van der Vorst [41]. Examples can be devised for which either CGS or hybrid GMRES is superior. We hope that a fuller understanding of the comparison between these two classes of iterative methods will come with further analysis, experiments, and algorithmic development.

Acknowledgments. For advice on many topics we are grateful to Howard Elman, Martin Gutknecht, Wayne Joubert, Tom Manteuffel, Youcef Saad, Paul Saylor, and Richard Varga. For making the computations easy we are grateful to Matlab.

REFERENCES

[1] R. S. ANDERSSEN AND G. H. GOLUB, *Richardson's non-stationary matrix iterative procedure*, Report STAN-CS-72-304, Department of Computer Science, Stanford University, Stanford, CA 1972.

[2] S. F. ASHBY, *CHEBYCODE: A FORTRAN implementation of Manteuffel's adaptive Chebyshev algorithm*, Tech. Report 1203, Department of Computer Science, University of Illinois, Urbana, IL, 1985.

[3] P. BROWN, *A theoretical comparison of the Arnoldi and* GMRES *algorithms*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 58–78.

[4] A. EDELMAN, Personal communication, 1990.

[5] M. EIERMANN, *On semiiterative methods generated by Faber polynomials*, Numer. Math., 56 (1989), pp. 139–156.

[6] M. EIERMANN, W. NIETHAMMER, AND R. S. VARGA, *A study of semiiterative methods for nonsymmetric systems of linear equations*, Numer. Math., 47 (1985), pp. 505–533.

[7] H. C. ELMAN, Y. SAAD, AND P. E. SAYLOR, *A hybrid Chebyshev Krylov subspace algorithm for solving nonsymmetric systems of linear equations*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 840–855.

[8] H. ELMAN AND R. STREIT, *Polynomial iteration for nonsymmetric indefinite linear systems*, in Numerical Analysis, Lecture Notes in Mathematics No. 1230, J. P. Hennert, ed., Springer-Verlag, Berlin, New York, 1986.

[9] B. FISCHER AND L. REICHEL, *A stable Richardson iteration method for complex linear systems*, Numer. Math., 54 (1988), pp. 225–242.

[10] R. FREUND, *Conjugate gradient type methods for linear systems with complex symmetric coefficient matrices*, SIAM J. Sci. Statist. Comp., 13 (1992), pp. 425–448.

[11] W. B. GRAGG AND L. REICHEL, *On the application of orthogonal polynomials to the iterative solution of linear systems of equations with indefinite or non-hermitian matrices*, Linear Algebra Appl., 88 (1987), pp. 349–371.

[12] J. F. GRCAR, *Operator coefficient methods for linear equations*, Report SAND89-8691, Sandia National Laboratories, 1989.

[13] M. H. GUTKNECHT, *An iterative method for solving linear equations based on minimum norm Pick-Nevanlinna interpolation*, in Approximation Theory V, C. K. Chui, L. L. Schumaker, and J. D. Ward, eds., Academic Press, New York, 1986.

[14] M. R. HESTENES AND E. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.

[15] O. G. JOHNSON, C. A. MICCHELLI, AND G. PAUL, *Polynomial preconditioners for conjugate gradient calculations*, SIAM J. Numer. Anal., 20 (1983), pp. 362–376.

[16] W. JOUBERT, *Iterative methods for the solution of nonsymmetric systems of linear equations*, Report CNA-242, Center for Numerical Analysis, University of Texas, Austin, Texas, 1990.

[17] X. LI, *An adaptive method for solving nonsymmetric linear systems involving application of* SCPACK, J. Comp. Appl. Math., to appear.

[18] T. A. MANTEUFFEL, *Adaptive procedure for estimating parameters for the nonsymmetric Tchebychev iteration*, Numer. Math., 31 (1978), pp. 183–208.

[19] ———, Personal communication, 1989.

[20] N. M. NACHTIGAL, S. C. REDDY, AND L. N. TREFETHEN, *How fast are nonsymmetric matrix iterations?*, SIAM J. Matrix Anal. Appl., this issue, pp. 778–795.

[21] D. P. O'LEARY, *Yet another polynomial preconditioner for the conjugate gradient algorithm*, Linear Algebra Appl., 154–158 (1991), pp. 377–388.

[22] G. OPFER AND G. SCHOBER, *Richardson's iteration for nonsymmetric matrices*, Linear Algebra Appl., 58 (1984), pp. 343–367.

[23] S. C. REDDY AND L. N. TREFETHEN, *Lax-stability of fully discrete spectral methods via stability regions and pseudo-eigenvalues*, Comp. Meth. Appl. Mech. Engrg., 80 (1990), pp. 147–164.

[24] L. REICHEL, *Polynomials by conformal mapping for the Richardson iteration method for complex linear systems*, SIAM J. Numer. Anal., 25 (1988), pp. 1359–1368.

[25] L. REICHEL, *The application of Leja points to Richardson iteration and polynomial preconditioning*, Linear Algebra Appl., 154–156 (1991), pp. 389–414.

[26] L. REICHEL AND L. N. TREFETHEN, *Eigenvalues and pseudo-eigenvalues of Toeplitz matrices*, Linear Algebra Appl., 162–164 (1992), pp. 153–185.

[27] Y. SAAD, *Krylov subspace methods on supercomputers*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 1200–1232.

[28] Y. SAAD, *Least-squares polynomials in the complex plane and their use for solving nonsymmetric linear systems*, SIAM J. Numer. Anal., 24 (1987), pp. 155–169.

[29] Y. SAAD AND M. H. SCHULTZ, GMRES: *A generalized minimum residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[30] P. E. SAYLOR, *Use of the singular value decomposition with the Manteuffel algorithm for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 1 (1980), pp. 210–222.

[31] P. E. SAYLOR, *Leapfrog variants of iterative methods for linear algebraic equations*, J. Comp. Appl. Math., 24 (1988), pp. 169–193.

[32] P. E. SAYLOR, *An adaptive algorithm for Richardson's iteration*, in Iterative Methods for Large Linear Systems, D. R. Kincaid and L. J. Hayes, eds., Academic Press, New York, 1990.

[33] P. E. SAYLOR AND D. C. SMOLARSKI, *Implementation of an adaptive algorithm for Richardson's method*, Linear Algebra Appl., 154–156 (1991), pp. 615–646.

[34] D. C. SMOLARSKI, *Optimum semi-iterative methods for the solution of any linear algebraic system with a square matrix*, Tech. Report 1077, Department of Computer Science, University of Illinois, Urbana, IL, 1981.

[35] D. C. SMOLARSKI AND P. E. SAYLOR, *An optimum iterative method for solving any linear system with a square matrix*, BIT, 28 (1988), pp. 163–178.

[36] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[37] H. TAL-EZER, *Polynomial approximation of functions of matrices and applications*, J. Sci. Comput., 4 (1989), pp. 25–60.

[38] L. N. TREFETHEN, *Approximation theory and numerical linear algebra*, in Algorithms for Approximation II, J. C. Mason and M. G. Cox, eds., Chapman and Hall, London, 1990.

[39] L. N. TREFETHEN, *Pseudospectra of matrices*, in Proc. 14th Dundee Biennial Conf. on Numer. Anal., D. F. Griffiths and G. A. Watson, eds., to appear.

[40] L. N. TREFETHEN AND M. R. TRUMMER, *An instability phenomenon in spectral methods*, SIAM J. Numer. Anal., 24 (1987), pp. 1008–1023.

[41] H. A. VAN DER VORST, *Bi*-CGSTAB: *A fast and smoothly converging variant of Bi*-CG *for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.

[42] H. E. WRIGLEY, *Accelerating the Jacobi method for solving simultaneous equations by Chebyshev extrapolation when the eigenvalues of the iteration matrix are complex*, Comput. J., 6 (1963), pp. 169–176.

[43] G. STARKE AND R. S. VARGA, *A hybrid Arnoldi-Faber iterative method for nonsymmetric systems of linear equations*, Numer. Math., to appear.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 826–846, July 1992

009

DIAGONAL SCALINGS OF THE LAPLACIAN AS PRECONDITIONERS FOR OTHER ELLIPTIC DIFFERENTIAL OPERATORS*

A. GREENBAUM†

Abstract. The use of diagonal scalings of the Laplacian matrix as preconditioners for matrices arising from other second-order self-adjoint elliptic differential operators is considered. It is proved that if a diffusion operator with a piecewise constant but discontinuous diffusion coefficient is preconditioned by a diagonal scaling of the Laplacian, then, in the limit as the mesh size goes to zero, the optimal diagonal scaling is just the identity. If, on the other hand, the Laplacian is scaled on each side by the square root of the diagonal of the matrix corresponding to the diffusion operator, then the condition number of the preconditioned matrix grows like $O(h^{-2})$, instead of $O(1)$. This is in contrast to the case in which the diffusion coefficient is smoothly varying, in which case numerical evidence suggests that the optimal diagonal scaling is approximately equal to the square root of the diagonal of the matrix.

Key words. Laplacian, preconditioners, elliptic operators

AMS(MOS) subject classification. 65F

1. Introduction. In [2], experiments were reported using a numerical optimization code to determine the preconditioner of a specified form that, for a given coefficient matrix, minimized the condition number of the preconditioned system. One of the more interesting experiments involved finding the optimal diagonal scaling of the Laplacian to use as a preconditioner for other second-order self-adjoint elliptic differential operators. Similar experiments had previously been carried out in [1], and the use of preconditioners of this form has also been discussed in [6].

Let A_h be the matrix arising from a finite element or finite difference approximation for the problem

$$-\nabla \cdot a\nabla u = f \quad \text{in } \Omega; \qquad u = 0 \quad \text{on } \partial\Omega, \tag{1.1}$$

where the positive coefficient a varies throughout the domain Ω and is bounded away from zero. Let Δ_h be the Laplacian matrix arising from the same finite element or finite difference approximation for the problem

$$-\Delta u = f \quad \text{in } \Omega; \qquad u = 0 \quad \text{on } \partial\Omega. \tag{1.2}$$

Let D be any positive definite diagonal matrix. One might consider using the matrix

$$M = D\Delta_h D \tag{1.3}$$

as a preconditioner for the matrix A_h in an iterative algorithm such as the Chebyshev or conjugate gradient method to solve problem (1.1). At each iteration it is then necessary to solve a linear system with coefficient matrix M, but such linear systems are generally much easier to solve than the original problem with matrix A_h. It is trivial to invert the diagonal matrix D, and, on a uniform rectangular grid, Δ_h can be solved with a fast Poisson solver. On an irregular region, Δ_h can be solved by embedding the region in a rectangle and using an integral equation formulation of the problem [4]. The number

* Received by the editors April 5, 1990; accepted for publication (in revised form) January 2, 1991. This work was supported by the Applied Mathematical Sciences Program of the U.S. Department of Energy under contract DE-AC02-76ER03077 and by the Advanced Research Projects Agency of the Department of Defense under contract F49620-87-C-0065.

† Courant Institute of Mathematical Sciences, 251 Mercer St., New York, New York 10012 (na.greenbaum@na-net.ornl.gov).

of iterations required by the Chebyshev or conjugate gradient algorithms can be bounded in terms of the condition number of the preconditioned system, and so one might then ask the question: What is the best diagonal matrix D to use in (1.3) in order to minimize this condition number? That is, find a positive definite diagonal matrix D_h such that

$$\min_{D \in \{\text{positive definite diagonal matrices}\}} \kappa((D\Delta_h D)^{-1}A_h) = \kappa((D_h\Delta_h D_h)^{-1}A_h), \tag{1.4}$$

where $\kappa(M^{-1}A_h)$ is the ratio of the largest to smallest eigenvalue of $M^{-1}A_h$ or the condition number of the symmetrically preconditioned matrix $M^{-1/2}A_hM^{-1/2}$. This is equivalent to finding a matrix D_h, which minimizes

$$\kappa(\Delta_h^{-1}(D^{-1}A_hD^{-1}))$$

over all positive definite diagonal matrices D, since the eigenvalues of $\Delta_h^{-1}(D^{-1}A_hD^{-1})$ are the same as those of $(D\Delta_hD)^{-1}A_h$. The problem was stated in this second form in [1].

In this paper we prove a somewhat counterintuitive result about the optimal diagonal scaling D_h when the diffusion coefficient a is piecewise constant but discontinuous. Both the result and the method of proof became apparent from studying the numerical results of the optimization code, thus indicating the usefulness of such a code as a tool in the study of preconditioners. The result is that in the limit as the mesh size h goes to zero, the optimal diagonal scaling D_h approaches the identity (or a scalar multiple of the identity, since scalar factors do not affect the condition number). This is in contrast to the case of a smoothly varying diffusion coefficient a, in which case numerical evidence suggests that the optimal diagonal scaling D_h is approximately equal to the square root of the diagonal of the matrix A_h.

2. A piecewise constant diffusion coefficient: Theoretical results. The first theorem that we prove is very general in nature, applying to arbitrary matrices and preconditioners with a certain algebraic property. It characterizes a space in which the extreme values of the Rayleigh quotient must be attained. The next two theorems use this result and apply to matrices arising from specific forms of (1.1), with preconditioners of the form (1.3).

THEOREM 2.1. *Let A and C be two n by n symmetric positive definite* (SPD) *matrices and assume that certain rows of C are just scalar multiples of the corresponding rows of A; that is, there is a nonempty set S such that for each $i \in S$ there is a scalar c_i such that*

$$C_{ij} = c_i A_{ij} \quad \forall j = 1, \cdots, n. \tag{2.1}$$

Then the extreme values of the Rayleigh quotient v^TAv/v^TCv are obtained for certain vectors v satisfying either

$$(Av)_i = 0 \quad \forall i \in S \tag{2.2}$$

or

$$v_j = 0 \quad \forall j \notin S. \tag{2.3}$$

Proof. Let w be an arbitrary vector and let v be a vector that satisfies (2.2) and that matches w in all components outside of S. Such a vector exists since A is SPD and hence every principal submatrix is nonsingular. The vector w can be written in the form

$$w = v + \hat{v},$$

where $\hat{v}$ satisfies (2.3). Hence $v^TA\hat{v} = \hat{v}^TAv = 0$ and we have

$$w^TAw = v^TAv + \hat{v}^TA\hat{v}.$$

Since $(Cv)_i$ is also zero for all $i \in S$, it also follows that

$$w^T C w = v^T C v + \hat{v}^T C \hat{v}.$$

Thus the Rayleigh quotient is given by

$$\frac{w^T A w}{w^T C w} = \frac{v^T A v + \hat{v}^T A \hat{v}}{v^T C v + \hat{v}^T C \hat{v}}$$

and since each term in the numerator and denominator is nonnegative, it satisfies

$$\min\left\{\frac{v^T A v}{v^T C v}, \frac{\hat{v}^T A \hat{v}}{\hat{v}^T C \hat{v}}\right\} \leqq \frac{w^T A w}{w^T C w} \leqq \max\left\{\frac{v^T A v}{v^T C v}, \frac{\hat{v}^T A \hat{v}}{\hat{v}^T C \hat{v}}\right\},$$

from which the desired result follows. □

The spaces defined by (2.2) and (2.3) are both A-orthogonal and C-orthogonal to one another, and together they span all of R^n. The technique of expressing a given vector as a sum of vectors from each space will be used in the proofs throughout this paper.

We first use Theorem 2.1 to prove a result about the one-dimensional problem

$$\begin{aligned} -\frac{d}{dx}\left(a\frac{du}{dx}\right) &= f, \qquad x \in (0,1), \\ u(0) &= u(1) = 0, \end{aligned} \tag{2.4}$$

where the coefficient $a(x)$ has the form

$$a(x) = \begin{cases} a_1, & \text{if } x < .5, \\ a_2, & \text{if } x > .5, \end{cases} \qquad a_1, a_2 > 0, \quad a_1 \neq a_2. \tag{2.5}$$

Let A_h be the matrix arising from a continuous piecewise linear finite element approximation for this problem on a uniform grid of size h. Assume that the grid contains a node at the point of discontinuity of $a(x)$. The matrix A_h is then given by

$$A_h = \begin{bmatrix} 2a_1 & -a_1 & & & & & & \\ -a_1 & & & & & & & \\ & & -a_1 & & & & & \\ & -a_1 & 2a_1 & -a_1 & & & & \\ & & -a_1 & a_1+a_2 & -a_2 & & & \\ & & & -a_2 & 2a_2 & & & \\ & & & & & & -a_2 & \\ & & & & & -a_2 & 2a_2 & \end{bmatrix}. \tag{2.6}$$

Let Δ_h be the one-dimensional Laplacian matrix arising from a continuous piecewise linear finite element approximation for the problem

$$\begin{aligned} -\frac{d^2u}{dx^2} &= f, \qquad x \in (0,1), \\ u(0) &= u(1) = 0, \end{aligned} \tag{2.7}$$

on the same uniform grid. The matrix Δ_h is just tridi $(-1, 2, -1)$. We prove the following theorem.

THEOREM 2.2. *Let A_h and Δ_h be as defined above and let D_h be a positive definite diagonal matrix that satisfies* (1.4). *Then*

(i) *D_h has the form*

$$D_h = \begin{pmatrix} d_{1,h} I_h & & \\ & \bar{d}_h & \\ & & d_{2,h} I_h \end{pmatrix},$$

where $d_{1,h}$, $d_{2,h}$, and $\bar{d}_h$ are positive scalars and I_h is the identity of order $(n-1)/2$, where $h = 1/n + 1$.

(ii) *In the limit as $h \to 0$, these scalars approach each other; that is,*

$$\lim_{h \to 0} d_{1,h} = \lim_{h \to 0} d_{2,h} = \lim_{h \to 0} \bar{d}_h \equiv d,$$

and the condition number of the optimally preconditioned matrix approaches that of the matrix preconditioned by the simple Laplacian:

$$\lim_{h \to 0} \kappa((D_h \Delta_h D_h)^{-1} A_h) = \lim_{h \to 0} \kappa(\Delta_h^{-1} A_h) = \max\left\{\frac{a_1}{a_2}, \frac{a_2}{a_1}\right\}.$$

(iii) *If $\hat{D}_h$ is any matrix of the form*

$$\hat{D}_h = \begin{pmatrix} \hat{d}_{1,h} I_h & & \\ & \hat{d}_h & \\ & & \hat{d}_{2,h} I_h \end{pmatrix} \tag{2.8}$$

and the positive scalars $\hat{d}_{1,h}$, $\hat{d}_{2,h}$, and $\hat{d}_h$ approach different *limits as $h \to 0$ (more generally, if there exist positive constants ε and δ such that for all h less than δ either $|d_{1,h} - d_h| > \varepsilon$ or $|d_{2,h} - d_h| > \varepsilon$), then*

$$\kappa((\hat{D}_h \Delta_h \hat{D}_h)^{-1} A_h) \geqq O(h^{-2}) \quad \text{as } h \to 0.$$

We prove this theorem through a series of lemmas (Lemmas 2.1–2.4). For simplicity we drop the subscript h when it is clear which variables depend on h. The point of discontinuity of a, $x = .5$, is grid point number $(n+1)/2$, which we denote by m.

LEMMA 2.1. *Let D be any matrix of the form*

$$D = \begin{pmatrix} d_1 I & & \\ & \bar{d} & \\ & & d_2 I \end{pmatrix}, \tag{2.9}$$

where d_1, d_2, and $\bar{d}$ are positive scalars and I is the identity matrix of order $m-1$. Define M to be the matrix $D\Delta D$, where Δ is the Laplacian matrix. The vectors v for which the Rayleigh quotient $v^T Av / v^T Mv$ attains its extreme values satisfy

$$(Av)_i = 0, \qquad i = 1, \cdots, m-2, m+2, \cdots, n, \tag{2.10}$$

or, equivalently,

$$\begin{aligned} v_{m-1-j} &= \frac{n-1-2j}{n-1} v_{m-1}, \qquad j = 1, \cdots, m-2 \\ v_{m+1+j} &= \frac{n-1-2j}{n-1} v_{m+1}, \qquad j = 1, \cdots, m-2. \end{aligned} \tag{2.11}$$

Proof. Result (2.10) follows from Theorem 2.1 since all rows of M, except rows $m-1$, m, and $m+1$, are just scalar multiples of the corresponding rows of A and since either of the extreme values a_1/d_1^2 or a_2/d_2^2, which can be taken on by the Rayleigh quotient for vectors that are zero outside $\{1, \cdots, m-2, m+2, \cdots, n\}$, can also be taken on for vectors satisfying (2.10).

By definition of the matrix A, equations (2.10) are equivalent to

$$\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & & & \\ & & & -1 & \\ & & -1 & 2 & \end{pmatrix}\begin{pmatrix} v_{m-2} \\ v_{m-3} \\ \vdots \\ v_1 \end{pmatrix} = \begin{pmatrix} v_{m-1} \\ 0 \\ \vdots \\ 0 \end{pmatrix},$$

$$\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & & & \\ & & & -1 & \\ & & -1 & 2 & \end{pmatrix}\begin{pmatrix} v_{m+2} \\ v_{m+3} \\ \vdots \\ v_n \end{pmatrix} = \begin{pmatrix} v_{m+1} \\ 0 \\ \vdots \\ 0 \end{pmatrix}.$$

It is easy to check that vectors of the form (2.11) are the solutions to these equations and, thus, the equivalence of (2.10) and (2.11). □

Vectors satisfying (2.10) or (2.11) are called *discrete harmonic*.

LEMMA 2.2 (Theorem 2.2(iii)). *Let D be any positive definite matrix of the form* (2.9). *If d_1, d_2, and $\bar{d}$ approach different limits as $h \to 0$, then*

$$\kappa(M^{-1}A) \geqq O(h^{-2}) \quad \text{as } h \to 0,$$

where $M = D\Delta D$.

Proof. For any vector v satisfying (2.10) and (2.11), we can write

$$\begin{aligned} v^TAv &= v_{m-1}(Av)_{m-1} + v_m(Av)_m + v_{m+1}(Av)_{m+1} \\ &= v_{m-1}a_1\left(2v_{m-1} - \frac{n-3}{n-1}v_{m-1} - v_m\right) + v_m((a_1+a_2)v_m - a_1v_{m-1} - a_2v_{m+1}) \\ &\quad + v_{m+1}a_2\left(2v_{m+1} - v_m - \frac{n-3}{n-1}v_{m+1}\right). \end{aligned}$$

After simplification this becomes

$$v^TAv = a_1\left[(v_m - v_{m-1})^2 + \frac{2}{n-1}v_{m-1}^2\right] + a_2\left[(v_m - v_{m+1})^2 + \frac{2}{n-1}v_{m+1}^2\right]. \tag{2.12}$$

Similarly, v^TMv can be written as

$$v^TMv = (\bar{d}v_m - d_1v_{m-1})^2 + \frac{2}{n-1}d_1^2v_{m-1}^2 + (\bar{d}v_m - d_2v_{m+1})^2 + \frac{2}{n-1}d_2^2v_{m+1}^2. \tag{2.13}$$

Taking $v_{m-1} = v_m = v_{m+1} = 1$ and dividing (2.13) by (2.12) gives

$$\begin{aligned} \frac{v^TMv}{v^TAv} &= \frac{(\bar{d}-d_1)^2 + \frac{2}{n-1}d_1^2 + (\bar{d}-d_2)^2 + \frac{2}{n-1}d_2^2}{\frac{2}{n-1}(a_1+a_2)} \\ &\geqq \frac{n-1}{2}\cdot\frac{\max\{(d_1-\bar{d})^2, (d_2-\bar{d})^2\}}{a_1+a_2} \geqq O(h^{-1}). \end{aligned}$$

Taking

$$v_{m-1}=\frac{\bar{d}}{d_1},\quad v_{m+1}=\frac{\bar{d}}{d_2},\quad v_m=1,$$

and dividing (2.12) by (2.13) gives

$$\frac{v^TAv}{v^TMv}=\frac{a_1\left[(1-\bar{d}/d_1)^2+\frac{2}{n-1}(\bar{d}/d_1)^2\right]+a_2\left[(1-\bar{d}/d_2)^2+\frac{2}{n-1}(\bar{d}/d_2)^2\right]}{\frac{2}{n-1}(2\bar{d}^2)}$$

$$\geqq\frac{n-1}{2}\cdot\frac{\max\{a_1(1/\bar{d}-1/d_1)^2,a_2(1/\bar{d}-1/d_2)^2\}}{2}\geqq O(h^{-1}).$$

Hence, by definition of κ, we have

$$\kappa(M^{-1}A)=\left(\max_{v\neq 0}\frac{v^TAv}{v^TMv}\right)\Big/\left(\min_{v\neq 0}\frac{v^TAv}{v^TMv}\right)=\left(\max_{v\neq 0}\frac{v^TAv}{v^TMv}\right)\cdot\left(\max_{v\neq 0}\frac{v^TMv}{v^TAv}\right)$$

$$\geqq\left(\frac{n-1}{2}\right)^2\cdot\frac{\max\{a_1(1/\bar{d}-1/d_1)^2,a_2(1/\bar{d}-1/d_2)^2\}}{2}$$

$$\cdot\frac{\max\{(d_1-\bar{d})^2,(d_2-\bar{d})^2\}}{a_1+a_2}\geqq O(h^{-2}).$$ □

LEMMA 2.3 (Theorem 2.2(ii)). *If D is a matrix of the form* (2.9) *that satisfies*

$$\min_{\hat{D}\text{ of the form (2.9)}}\kappa((\hat{D}\Delta\hat{D})^{-1}A)=\kappa((D\Delta D)^{-1}A),$$

then d_1, d_2, and $\bar{d}$ approach the same limit as $h\to 0$, and

$$\kappa(M^{-1}A)\to\kappa(\Delta^{-1}A)=\max\left\{\frac{a_1}{a_2},\frac{a_2}{a_1}\right\}\quad\text{as } h\to 0,$$

where $M=D\Delta D$.

Proof. From Lemma 2.2, it is clear that d_1, d_2, and $\bar{d}$ must approach the same limit, since otherwise the condition number of the preconditioned matrix would be greater than or equal to $O(h^{-2})$. Yet the condition number of the matrix preconditioned by the simple Laplacian is

$$\max\left\{\frac{a_1}{a_2},\frac{a_2}{a_1}\right\}=O(1),$$

which is well known, and can also be seen from (2.12) and (2.13). We can assume without loss of generality that this limit is 1, since scalar factors do not affect the condition number. Taking $v_{m-1}=v_m=0$, then, and dividing (2.12) by (2.13), we have

$$\frac{v^TAv}{v^TMv}=\frac{a_2}{d_2^2}\to a_2.$$

Taking $v_m=v_{m+1}=0$ and dividing (2.12) by (2.13) gives

$$\frac{v^TAv}{v^TMv}=\frac{a_1}{d_1^2}\to a_1.$$

Hence, in the limit as $h \to 0$, the condition number of the optimally preconditioned matrix is no better than

$$\max\left\{\frac{a_1}{a_2}, \frac{a_2}{a_1}\right\},$$

which is the condition number of the matrix preconditioned by the Laplacian. □

LEMMA 2.4 (Theorem 2.2(i)). *If D is a matrix of the form* (2.9) *that satisfies*

$$\min_{\hat{D} \text{ of the form (2.9)}} \kappa((\hat{D}\Delta\hat{D})^{-1}A) = \kappa((D\Delta D)^{-1}A),$$

then D also satisfies

$$\min_{\tilde{D} \in \{\text{positive definite diagonal matrices}\}} \kappa((\tilde{D}\Delta\tilde{D})^{-1}A) = \kappa((D\Delta D)^{-1}A).$$

Moreover, any positive definite diagonal matrix that satisfies this equation is of the form (2.9).

Proof. Let $\tilde{D} = \operatorname{diag}(\tilde{d}_i)$, $i = 1, \cdots, n$, be any positive definite diagonal matrix and let $\hat{D}$ be the matrix of the form (2.9) whose $(m-1)$st, mth, and $(m+1)$st diagonal elements are equal to those of $\tilde{D}$. Define $\hat{M} = \hat{D}\Delta\hat{D}$ and $\tilde{M} = \tilde{D}\Delta\tilde{D}$. Let v be a vector satisfying (2.10). Then $v^T\tilde{M}v$ satisfies

$$v^T\tilde{M}v = v^T\tilde{D}\hat{D}^{-1}\hat{M}\hat{D}^{-1}\tilde{D}v = w^T\hat{M}w,$$

where $w = \hat{D}^{-1}\tilde{D}v$ matches v in components $m-1$, m, and $m+1$. As in Theorem 2.1, then, w can be written in the form $w = v + \hat{v}$, where $\hat{v}_{m-1} = \hat{v}_m = \hat{v}_{m+1} = 0$, and hence $\hat{v}^T\hat{M}v = v^T\hat{M}\hat{v} = 0$. It follows that

$$v^T\tilde{M}v = w^T\hat{M}w = v^T\hat{M}v + \hat{v}^T\hat{M}\hat{v} \geqq v^T\hat{M}v.$$

Since by Theorem 2.1 the largest value of the Rayleigh quotient $v^T\hat{M}v/v^TAv$ is obtained for a vector v satisfying (2.10), it follows that

$$\max_{v \neq 0} \frac{v^T\tilde{M}v}{v^TAv} \geqq \max_{v \neq 0} \frac{v^T\hat{M}v}{v^TAv}. \tag{2.14}$$

Now let a vector w be given by

$$w = \tilde{D}^{-1}\hat{D}v,$$

where v again satisfies (2.10). Then $w^T\tilde{M}w$ is equal to $v^T\hat{M}v$ and, since the $(m-1)$st, mth, and $(m+1)$st elements of w match those of v, we can again write w in the form $w = v + \hat{v}$, where $\hat{v}_{m-1} = \hat{v}_m = \hat{v}_{m+1} = 0$. Hence $\hat{v}^TAv = v^TA\hat{v} = 0$ and we have

$$w^TAw = v^TAv + \hat{v}^TA\hat{v} \geqq v^TAv.$$

Since by Theorem 2.1 the Rayleigh quotient $v^TAv/v^T\hat{M}v$ obtains its largest value for some v satisfying (2.10), it follows that

$$\max_{w \neq 0} \frac{w^TAw}{w^T\tilde{M}w} \geqq \max_{v \neq 0} \frac{v^TAv}{v^T\hat{M}v}. \tag{2.15}$$

From (2.14), (2.15), and the definition of κ, then, the desired result follows:

$$\kappa(M^{-1}A) \leqq \kappa(\hat{M}^{-1}A) = \left(\max_{v \neq 0} \frac{v^TAv}{v^T\hat{M}v}\right) \cdot \left(\max_{v \neq 0} \frac{v^T\hat{M}v}{v^TAv}\right) \leqq \kappa(\tilde{M}^{-1}A).$$

Since the inequalities in (2.14) and (2.15) are strict unless $\hat{v}$ is zero, i.e., unless $\tilde{D}$ is itself of the form (2.9), the second part of the lemma is also proved. □

When the Laplacian Δ_h is used as a preconditioner for A_h, the condition number of the preconditioned system is bounded, independent of h:

$$\kappa(\Delta_h^{-1}A_h) \leqq \max\left\{\frac{a_1}{a_2}, \frac{a_2}{a_1}\right\}.$$

Theorem 2.2 shows that, for small h, the best diagonal scaling is close to the identity (or a scalar multiple of the identity) and this bound cannot be improved much. The wrong diagonal scaling can greatly increase this condition number. For larger values of h, however, an appropriate diagonal scaling can significantly reduce the condition number of the preconditioned system, as will be demonstrated in the following section.

A similar result holds for the two-dimensional problem

$$\begin{aligned} &-\left(\frac{\partial}{\partial x}\left(a\frac{\partial u}{\partial x}\right)+\frac{\partial}{\partial y}\left(a\frac{\partial u}{\partial y}\right)\right)=f, \qquad (x,y)\in(0,1)\times(0,1), \\ &u(x,0)=u(x,1)=u(0,y)=u(1,y)=0, \end{aligned} \tag{2.16}$$

where the coefficient $a(x, y)$ has the form

$$a(x,y)=\begin{cases} a_1, & \text{if } y<.5, \\ a_2, & \text{if } y>.5, \end{cases} \qquad a_1, a_2>0, \quad a_1\neq a_2. \tag{2.17}$$

Again, let A_h be the matrix arising from a continuous piecewise linear finite element approximation for this problem on a uniform triangular grid of size h, having a mesh line at the discontinuity, $y = .5$. If the natural ordering of nodes is used then A_h has the form

$$A_h=\begin{bmatrix} a_1T & -a_1I & & & & & & \\ -a_1I & & & & & & & \\ & & -a_1I & & & & & \\ & -a_1I & a_1T & -a_1I & & & & \\ & & -a_1I & \frac{a_1+a_2}{2} & -a_2I & & & \\ & & & -a_2I & a_2T & & & \\ & & & & & -a_2I & & \\ & & & & & -a_2I & a_2T & \end{bmatrix}, \tag{2.18}$$

where T = tridi $(-1, 4, -1)$ and I is the identity of order n for a grid of $n \times n$ interior nodes. Let Δ_h be the two-dimensional Laplacian matrix arising from a continuous piecewise linear finite element approximation for the problem

$$\begin{aligned} &-\left(\frac{\partial^2 u}{\partial x^2}+\frac{\partial^2 u}{\partial y^2}\right)=f, \qquad (x,y)\in(0,1)\times(0,1), \\ &u(x,0)=u(x,1)=u(0,y)=u(1,y)=0 \end{aligned} \tag{2.19}$$

on the same uniform grid. The matrix Δ_h is block tridi $(-I, T, -I)$. The following theorem is proved very similarly to the one-dimensional case.

THEOREM 2.3. *Let A_h and Δ_h be as defined above and let D_h be a positive definite diagonal matrix that satisfies* (1.4). *Then*

(i) *D_h has the form*

$$D_h = \begin{pmatrix} d_{1,h} I_h & & \\ & \bar{d}_h I_{.5,h} & \\ & & d_{2,h} I_h \end{pmatrix},$$

where $d_{1,h}$, $d_{2,h}$, and $\bar{d}_h$ are positive scalars, I_h is the identity of order $[n(n-1)]/2$, and $I_{.5,h}$ is the identity of order n, where $h = 1/(n+1)$.

(ii) *In the limit as $h \to 0$, these scalars approach each other; that is,*

$$\lim_{h\to 0} d_{1,h} = \lim_{h\to 0} d_{2,h} = \lim_{h\to 0} \bar{d}_h \equiv d,$$

and the condition number of the optimally preconditioned matrix approaches that of the matrix preconditioned by the simple Laplacian:

$$\lim_{h\to 0} \kappa((D_h \Delta_h D_h)^{-1} A_h) = \lim_{h\to 0} \kappa(\Delta_h^{-1} A_h) = \max\left\{\frac{a_1}{a_2}, \frac{a_2}{a_1}\right\}.$$

(iii) *If $\hat{D}_h$ is any matrix of the form*

$$\hat{D}_h = \begin{pmatrix} \hat{d}_{1,h} I_h & & \\ & \hat{d}_h I_{.5,h} & \\ & & \hat{d}_{2,h} I_h \end{pmatrix} \tag{2.20}$$

and the positive scalars $\hat{d}_{1,h}$, $\hat{d}_{2,h}$, and $\hat{d}_h$ approach different *limits as $h \to 0$ (more generally, if there exist positive constants ε and δ such that for all h less than δ either $|\hat{d}_{1,h} - \hat{d}_h| > \varepsilon$ or $|\hat{d}_{2,h} - \hat{d}_h| > \varepsilon$, then*

$$\kappa((\hat{D}_h \Delta_h \hat{D}_h)^{-1} A_h) \geq O(h^{-2}) \quad \text{as } h \to 0.$$

As in the one-dimensional case, we prove this theorem through a series of lemmas (Lemmas 2.5–2.11), dropping the subscript when it is clear which variables depend on h. The matrices considered in the two-dimensional case can be thought of as block matrices, with n blocks, each of order n. The subscript $m = (n+1)/2$ will denote the middle block, corresponding to the line of discontinuity in a. For any n^2-vector v, v_k will denote the kth block of v.

LEMMA 2.5. *Let D be any matrix of the form*

$$D = \begin{pmatrix} d_1 I & & \\ & \bar{d} I_{.5} & \\ & & d_2 I \end{pmatrix}, \tag{2.21}$$

where d_1, d_2, and $\bar{d}$ are positive scalars, I is the identity matrix of order $[n(n-1)]/2$, and $I_{.5}$ is the identity matrix of order n. Define M to be the matrix $D\Delta D$, where Δ is the Laplacian matrix. The vectors v for which the Rayleigh quotient $v^T Av / v^T Mv$ attains its extreme values satisfy

$$(Av)_i = 0, \qquad i = 1, \cdots, m-2, m+2, \cdots, n. \tag{2.22}$$

Proof. As in the one-dimensional case, the result is an immediate consequence of Theorem 2.1.

LEMMA 2.6. *Let v be a vector satisfying* (2.22) *and let v_{m-1} and v_{m+1} be eigenvectors of T = tridi $(-1, 4, -1)$. Then each block v_i can be written in the form*

$$
\begin{aligned}
v_i &= \gamma_i v_{m-1}, \qquad i = 1, \cdots, m-2, \\
v_i &= \gamma_i v_{m+1}, \qquad i = m+2, \cdots, n
\end{aligned} \tag{2.23}
$$

for some scalars γ_i. If $v_{m\pm1}$ is the eigenvector associated with the smallest eigenvalue of T, then $\gamma_{m\pm2} = 1 - O(h)$.

Proof. Equations (2.22) are equivalent to

$$
\begin{pmatrix} T & -I & & \\ -I & & & -I \\ & & -I & T \end{pmatrix}
\begin{pmatrix} v_{m-2} \\ v_{m-3} \\ \cdot \\ \cdot \\ v_1 \end{pmatrix}
= \begin{pmatrix} v_{m-1} \\ 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix},
$$

$$
\begin{pmatrix} T & -I & & \\ -I & T & & -I \\ & & -I & T \end{pmatrix}
\begin{pmatrix} v_{m+2} \\ v_{m+3} \\ \cdot \\ \cdot \\ v_n \end{pmatrix}
= \begin{pmatrix} v_{m+1} \\ 0 \\ \cdot \\ \cdot \\ 0 \end{pmatrix}. \tag{2.24}
$$

We will consider only the first set of equations, since the second is handled in exactly the same way. Assume there is a solution with $v_i = \gamma_i v_{m-1}$, $i = 1, \cdots, m-2$, for some scalars γ_i. Let v_{m-1} correspond to an eigenvalue μ of T. Then equations (2.24) become

$$
\begin{pmatrix} \mu & -1 & & \\ -1 & & & -1 \\ & & -1 & \mu \end{pmatrix}
\begin{pmatrix} \gamma_{m-2} \\ \vdots \\ \gamma_1 \end{pmatrix}
= \begin{pmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{2.25}
$$

This has a unique solution for $\gamma_1, \cdots, \gamma_{m-2}$ if $\mu \geqq 2$, and since all eigenvalues of T are greater than 2 this condition holds and the solution of (2.24) is, indeed, of the form (2.23).

Let T_k denote the tridiagonal matrix tridi $(-1, \mu, -1)$ of order k and let det (T_k) denote its determinant. Solving (2.25) using Cramer's rule gives

$$
\gamma_{m-2} = \frac{\det(T_{m-3})}{\det(T_{m-2})},
$$

where det (T_k) satisfies

$$
\begin{aligned}
&\det(T_0) = 1, \qquad \det(T_1) = \mu, \\
&\det(T_k) = \mu \det(T_{k-1}) - \det(T_{k-2}),
\end{aligned}
$$

and

$$
r_k \equiv \frac{\det(T_k)}{\det(T_{k-1})}
$$

satisfies

$$
\begin{aligned}
&r_1 = \mu, \\
&r_k = \mu - \frac{1}{r_{k-1}}, \qquad k = 2, \cdots, m-2.
\end{aligned}
$$

If carried out indefinitely, this recurrence converges to a solution of the equation

$$r = \mu - \frac{1}{r},$$

namely,

$$r = \frac{\mu + \sqrt{\mu^2 - 4}}{2},$$

and it is easy to check that after $m - 2 = O(h^{-1})$ steps, the ratio r_{m-2} is greater than this limit by $O(h)$. If μ is the smallest eigenvalue of T, then $\mu = 2 + O(h^2)$, and so $r_{m-2} = 1 + O(h)$ and $\gamma_{m-2} = 1/r_{m-2}$ is $1 - O(h)$. □

LEMMA 2.7 (Theorem 2.3(iii)). *Let D be any positive definite matrix of the form* (2.21). *If d_1, d_2, and $\bar{d}$ approach different limits as $h \to 0$, then*

$$\kappa(M^{-1}A) \geqq O(h^{-2}) \quad \text{as } h \to 0,$$

where $M = D\Delta D$.

Proof. Let v be a vector satisfying (2.22), with $v_{m-1} = v_m = v_{m+1}$ being the eigenvector of T, of unit norm, corresponding to the smallest eigenvalue $\mu = 2 + O(h^2)$. Then $v^T A v$ is given by

$$v^T A v = a_1(\mu - 1 - \gamma_{m-2}) + \frac{a_1 + a_2}{2}\mu - a_1 - a_2 + a_2(\mu - 1 - \gamma_{m+2}) = O(h),$$

while $v^T M v$ satisfies

$$v^T M v = d_1^2(\mu - \gamma_{m-2}) - 2d_1\bar{d} + \bar{d}^2\mu + d_2^2(\mu - \gamma_{m+2}) - 2\bar{d}d_2 > (d_1 - \bar{d})^2 + (\bar{d} - d_2)^2.$$

Hence the ratio satisfies

$$\frac{v^T M v}{v^T A v} \geqq O(h^{-1}).$$

If, instead of having unit length, the blocks v_{m-1} and v_{m+1} are taken to have lengths $\bar{d}/d_1$ and $\bar{d}/d_2$, respectively, then we find

$$v^T A v > a_1\left(\frac{\bar{d}}{d_1} - 1\right)^2 + a_2\left(\frac{\bar{d}}{d_3} - 1\right)^2,$$

$$v^T M v = \bar{d}^2[(\mu - \gamma_{m-2} - 1) + (\mu - 2) + (\mu - \gamma_{m+2} - 1)] = O(h).$$

In this case, then, we have

$$\frac{v^T A v}{v^T M v} \geqq O(h^{-1}),$$

and so the condition number satisfies

$$\kappa(M^{-1}A) = \left(\max_{v \neq 0} \frac{v^T A v}{v^T M v}\right) \cdot \left(\max_{v \neq 0} \frac{v^T M v}{v^T A v}\right) \geqq O(h^{-2}). \qquad \square$$

LEMMA 2.8 (Theorem 2.3(ii)). *If D is a matrix of the form* (2.21) *that satisfies*

$$\min_{\hat{D} \text{ of the form (2.21)}} \kappa((\hat{D}\Delta\hat{D})^{-1}A) = \kappa((D\Delta D)^{-1}A),$$

then d_1, d_2, and $\bar{d}$ approach the same limit as $h \to 0$, and

$$\kappa(M^{-1}A) \to \kappa(\Delta^{-1}A) \quad \text{as } h \to 0,$$

where $M = D\Delta D$.

Proof. The proof is the same as that of Lemma 2.3. □

LEMMA 2.9. *Let $\tilde{D}$ be any positive definite diagonal matrix whose $(m-1)$st, mth, and $(m+1)$st diagonal blocks are just scalar multiples of the identity. Let $\hat{D}$ be the matrix of the form* (2.21), *which matches $\tilde{D}$ in blocks $m-1$, m, and $m+1$. Then*

$$\kappa((\hat{D}\Delta\hat{D})^{-1}A) \leqq \kappa((\tilde{D}\Delta\tilde{D})^{-1}A).$$

Proof. The proof is analogous to that of Lemma 2.4 in the one-dimensional case. □

Lemma 2.9 shows that the matrix D of the form (2.21), which minimizes $\kappa((\hat{D}\Delta\hat{D})^{-1}A)$ over all matrices $\hat{D}$ of the form (2.21), also minimizes this quantity over all diagonal matrices whose $(m-1)$st, mth, and $(m+1)$st diagonal blocks are scalar multiples of the identity. To show that it minimizes this quantity over all diagonal matrices with possibly nonconstant elements in these blocks requires some additional work. To this end, we prove the following lemma.

LEMMA 2.10. *Let D be any positive definite matrix of the form* (2.21) *and let $M = D\Delta D$. The vectors v for which the Rayleigh quotient v^TAv/v^TMv attains its extreme values have blocks of the form*

$$v_i = \alpha_i s, \qquad i = 1, \cdots, n, \tag{2.26}$$

where $\alpha_1, \cdots, \alpha_n$ are scalars and s is the eigenvector of T corresponding to the smallest eigenvalue.

Proof. Let S be the matrix whose columns are the eigenvectors of T, and let $\Theta = \operatorname{diag}(\theta_1, \cdots, \theta_n)$ be the diagonal matrix of eigenvalues, so that we have $TS = S\Theta$, $S^TS = SS^T = I$. Define U to be the block diagonal matrix whose diagonal blocks are all equal to S. Then U^TAU is of the form (2.18) with T replaced by Θ, and U^TMU is $D(\text{block tridi}(-I, \Theta, -I))D$. Let P be a permutation matrix such that the jth column of the ith block of a matrix B is the ith column of the jth block of the matrix BP. Then the matrices P^TU^TAUP and P^TU^TMUP are block diagonal, with diagonal blocks given by

$$A_{ii} = \begin{pmatrix} a_1\theta_i & -a_1 & & & & & \\ -a_1 & & -a_1 & & & & \\ & -a_1 & a_1\theta_i & -a_1 & & & \\ & & -a_1 & \frac{a_1+a_2}{2}\theta_i & -a_2 & & \\ & & & -a_2 & a_2\theta_i & -a_2 & \\ & & & & -a_2 & & -a_2 \\ & & & & & -a_2 & a_2\theta_i \end{pmatrix}$$

and

$$M_{ii} = \begin{pmatrix} d_1^2\theta_i & -d_1^2 & & & & & \\ -d_1^2 & & -d_1^2 & & & & \\ & -d_1^2 & d_1^2\theta_i & -d_1\bar{d} & & & \\ & & -d_1\bar{d} & \bar{d}^2\theta_i & -\bar{d}d_2 & & \\ & & & -\bar{d}d_2 & d_2^2\theta_i & -d_2^2 & \\ & & & & -d_2^2 & & -d_2^2 \\ & & & & & -d_2^2 & d_2^2\theta_i \end{pmatrix},$$

respectively. Note that these blocks can be written in the form

$$A_{ii} = A_{1D} + \frac{(\theta_i - 2)}{2} \cdot \text{diag}\,(A_{1D}), \qquad M_{ii} = M_{1D} + \frac{(\theta_i - 2)}{2} \cdot \text{diag}\,(M_{1D}),$$

where A_{1D} and M_{1D} are the matrices that would arise from a one-dimensional problem. Since each eigenvalue θ_i of T is greater than 2, each block A_{ii} and M_{ii} has positive eigenvalues. Hence the largest (smallest) value of the ratio $w^T P^T U^T AUPw / w^T P_T U^T MUPw$ is attained for a vector w having one nonzero block w_i corresponding to the block $M_{ii}^{-1} A_{ii}$ with the largest (smallest) eigenvalue. Since the largest (smallest) eigenvalue of $M_{1D}^{-1} A_{1D}$ is greater (smaller) than that of $\text{diag}\,(M_{1D})^{-1}\,\text{diag}\,(A_{1D})^{-1}$, the block $M_{ii}^{-1} A_{ii}$ with both the largest and smallest eigenvalues is the one corresponding to the smallest value θ_i. Thus, the extreme values of the ratio $w^T P^T U^T AUPw / w^T P^T U^T MUPw$ are attained for vectors w with a single nonzero block, corresponding to the smallest eigenvalue θ_i. The vector $v = UPw$, then, is an extreme vector for the Rayleigh quotient $v^T Av / v^T Mv$, and each block is a scalar multiple of the eigenvector s of T corresponding to the smallest eigenvalue. □

LEMMA 2.11 (Theorem 2.3(i)). *If D is a matrix of the form* (2.21) *that satisfies*

$$\min_{\hat{D}\ \text{of the form}\ (2.21)} \kappa((\hat{D}\Delta\hat{D})^{-1}A) = \kappa((D\Delta D)^{-1}A), \tag{2.27}$$

then D also satisfies

$$\min_{\tilde{D} \in \{\text{positive definite diagonal matrices}\}} \kappa((\tilde{D}\Delta\tilde{D})^{-1}A) = \kappa((D\Delta D)^{-1}A). \tag{2.28}$$

Moreover, any positive definite diagonal matrix that satisfies (2.28) *is of the form* (2.21).

Proof. Let $\tilde{D} = \text{diag}\,(\tilde{D}_1, \cdots, \tilde{D}_n)$ be any positive definite diagonal matrix. Let $\hat{D}$ be the matrix of the form (2.21) whose $(m-1)$st, mth, and $(m+1)$st block coefficients are

$$d_1 = s^T \tilde{D}_{m-1} s, \quad \bar{d} = s^T \tilde{D}_m s, \quad d_2 = s^T \tilde{D}_{m+1} s,$$

where s is the eigenvector of T corresponding to the smallest eigenvalue μ. Define $\hat{M} = \hat{D}\Delta\hat{D}$ and $\tilde{M} = \tilde{D}\Delta\tilde{D}$. Let v be a vector satisfying (2.22) and (2.26). Then $v^T \tilde{M} v$ satisfies

$$v^T \tilde{M} v = v^T \tilde{D}\hat{D}^{-1}\hat{M}\hat{D}^{-1}\tilde{D}v = w^T \hat{M} w,$$

where $w = \hat{D}^{-1}\tilde{D}v$. The vector w can be written in the form $v + \hat{v}$, where $\hat{v} = (\hat{D}^{-1}\tilde{D} - I)v$. Because of the choice of d_1, d_2, and $\bar{d}$, we have

$$\begin{aligned}
\hat{v}^T \hat{M} v &= \alpha_{m-1} s^T (d_1^{-1}\tilde{D}_{m-1} - I)^T (d_1^2 \alpha_{m-1}\mu s - d_1^2 \alpha_{m-2} s - d_1 \bar{d}\alpha_m s) \\
&\quad + \alpha_m s^T (\bar{d}^{-1}\tilde{D}_m - I)^T (\bar{d}^2 \alpha_m \mu s - d_1 \bar{d}\alpha_{m-1} s - \bar{d} d_2 \alpha_{m+1} s) \\
&\quad + \alpha_{m+1} s^T (d_2^{-1}\tilde{D}_{m+1} - I)^T (d_2^2 \alpha_{m+1}\mu s - d_2^2 \alpha_{m+2} s - \bar{d} d_2 \alpha_m s) \\
&= 0.
\end{aligned}$$

It follows that

$$v^T \tilde{M} v = w^T \hat{M} w = v^T \hat{M} v + \hat{v}^T \hat{M} \hat{v} \geqq v^T \hat{M} v.$$

Since by Theorem 2.1 and Lemma 2.10 the largest value of the Rayleigh quotient $v^T \hat{M} v / v^T Av$ is obtained for a vector v satisfying (2.22) and (2.26), it follows that

$$\max_{v \neq 0} \frac{v^T \tilde{M} v}{v^T Av} \geqq \max_{v \neq 0} \frac{v^T \hat{M} v}{v^T Av}. \tag{2.29}$$

Now let a vector w be given by

$$w = \tilde{D}^{-1}\hat{D}v,$$

where v again satisfies (2.22) and (2.26). Then $w^T\tilde{M}w$ is equal to $v^T\hat{M}v$ and again we can write w in the form $w = v + \hat{v}$, where $\hat{v} = (\tilde{D}^{-1}\hat{D} - I)v$. We now have

$$\begin{aligned}\hat{v}^T Av = {} & \alpha_{m-1} s^T (d_1 \tilde{D}_{m-1}^{-1} - I)^T a_1 (\alpha_{m-1}\mu s - \alpha_{m-2} s - \alpha_m s) \\ & + \alpha_m s^T (\bar{d}\tilde{D}_m^{-1} - I)^T \left(\frac{a_1 + a_2}{2} \alpha_m \mu s - a_1 \alpha_{m-1} s - a_2 \alpha_{m+1} s \right) \\ & + \alpha_{m+1} s^T (d_2 \tilde{D}_{m+1}^{-1} - I)^T a_2 (\alpha_{m+1}\mu s - \alpha_m s - \alpha_{m+2} s),\end{aligned}$$

which can be written in the form

$$\hat{v}^T Av = v_{m-1}^T (CACv)_{m-1} + v_m^T (CACv)_m + v_{m+1}^T (CACv)_{m+1},$$

where C is a diagonal matrix whose diagonal elements are one except in blocks $m-1$, m, and $m+1$, where they are

$$c_1 = (d_1 s^T \tilde{D}_{m-1}^{-1} s - 1)^{1/2}, \quad \bar{c} = (\bar{d} s^T \tilde{D}_m^{-1} s - 1)^{1/2}, \quad c_2 = (d_2 s^T \tilde{D}_{m+1}^{-1} s - 1)^{1/2},$$

respectively. Because of the choice of d_1, d_2, and $\bar{d}$, we know that the quantities under the square roots are nonnegative, since

$$(s^T \tilde{D}_i s) \cdot (s^T \tilde{D}_i^{-1} s) = \left(\sum_{j=1}^n s_j^2 \tilde{d}_{i,j} \right) \cdot \left(\sum_{j=1}^n s_j^2 \tilde{d}_{i,j}^{-1} \right) = \sum_{j=1}^n s_j^4 + \sum_{j=1}^n \sum_{\substack{k=1 \\ k \neq j}}^n s_j^2 s_k^2 \left(\frac{\tilde{d}_{i,j}}{\tilde{d}_{i,k}} + \frac{\tilde{d}_{i,k}}{\tilde{d}_{i,j}} \right)$$

$$\geqq \sum_{j=1}^n s_j^4 + 2 \sum_{j=1}^n \sum_{\substack{k=1 \\ k \neq j}}^n s_j^2 s_k^2 = \left(\sum_{j=1}^n s_j^2 \right)^2 = 1.$$

Hence $\hat{v}^T Av$ is nonnegative and so we have

$$w^T Aw \geqq v^T Av + \hat{v}^T A\hat{v} \geqq v^T Av.$$

Since by Theorem 2.1 and Lemma 2.10 the Rayleigh quotient $v^T Av / v^T \hat{M} v$ obtains its largest value for some v satisfying (2.22) and (2.26), it follows that

$$\max_{w \neq 0} \frac{w^T Aw}{w^T \tilde{M} w} \geqq \max_{v \neq 0} \frac{v^T Av}{v^T \hat{M} v}. \tag{2.30}$$

Combining (2.29) and (2.30), we obtain the desired result:

$$\kappa(M^{-1}A) \leqq \kappa(\hat{M}^{-1}A) \leqq \kappa(\tilde{M}^{-1}A).$$

Since the inequalities in (2.29) and (2.30) are strict unless $\hat{v}$ is zero, i.e., unless $\tilde{D}$ is, itself, of the form (2.21), the second part of the lemma is also proved. □

While our primary interest has been in *diagonal* scalings of the Laplacian, it should be noted that the proofs of Lemmas 2.4 and 2.11 make no use of the assumption that D is diagonal outside of positions (blocks) $m-1$, m, and $m+1$. They can therefore be generalized to the following result.

COROLLARY. *For the one-dimensional problem, the matrix D_h of Theorem* 1.2 *minimizes $\kappa((E_h^T \Delta_h E_h)^{-1} A_h)$ over all matrices E_h whose three center rows and columns ($m-1$, m, and $m+1$) have nonzeros only on the diagonal. For the two-dimensional problem, the matrix D_h of Theorem* 2.1 *minimizes $\kappa((E_h^T \Delta_h E_h)^{-1} A_h)$ over all matrices E_h whose $3n$ center rows and columns ($n(m-1)$ through $n(m+1)$) have nonzeros only*

on the diagonal and these nonzeros are positive. More generally, it minimizes this quantity over all matrices whose $3n$ *center rows and columns have nonzeros only in the* $n \times n$ *diagonal blocks and for which these diagonal blocks,* E_{m-1}, E_m, *and* E_{m+1}, *have the property that*

$$(s^T E_i s) \cdot (s^T E_i^{-1} s) \geqq 1, \qquad i = m-1, m, m+1,$$

where s *is the eigenvector of* T *corresponding to the smallest eigenvalue.*

3. Numerical results. For a given matrix A_h an optimization code can be used to determine numerically the optimal preconditioner of the form (1.3). A particularly efficient technique for solving this type of optimization problem was developed by Overton [5]. Experiments with this code were reported in [2]. The code uses a variant of Newton's method to determine a matrix M of a specified form (e.g., form (1.3)) for which the spectral radius $\rho(I - M^{-1}A_h)$ is minimal. It was shown in [2] that this same matrix M (or any scalar multiple of M) also minimizes the condition number $\kappa(M^{-1}A_h)$, provided that the set over which the minimization is being performed contains all positive scalar multiples of its members, which it does in this case.

In the following experiment, the matrix A_h was taken to be the matrix arising from a continuous piecewise linear finite element approximation on a uniform grid of size h for the one-dimensional problem (2.4), (2.5), where

$$a_1 = 1, \qquad a_2 = 100. \tag{3.1}$$

The optimization code was run to determine the optimal preconditioner of the form (1.3). The diagonal matrix D_h determined by the code was always of the form (2.9), as Theorem 2.2 shows that it must be. (In fact, this observation of the numerical results led to the statement and proof of Theorem 2.2). The largest problem size that the optimization code was able to handle directly, however, was about $n = 225$, and the asymptotic behavior of the system cannot be deduced from results on grids of this size.

Using Theorem 2.2, however, the problem of finding the optimal matrix D_h can be reduced to a 3×3 eigenvalue optimization problem. From (2.12) and (2.13) it follows that for vectors v satisfying (2.10) and (2.11) we have

$$v^T A v = v_C^T A_C v_C, \qquad v^T M v = v_C^T M_C v_C,$$

where

$$A_C = \begin{bmatrix} \frac{n+1}{n-1}a_1 & -a_1 & 0 \\ -a_1 & a_1 + a_2 & -a_2 \\ 0 & -a_2 & \frac{n+1}{n-1}a_2 \end{bmatrix},$$

$$M_C = \begin{bmatrix} \frac{n+1}{n-1}d_1^2 & -d_1\bar{d} & 0 \\ -d_1\bar{d} & 2\bar{d}^2 & -\bar{d}d_2 \\ 0 & -\bar{d}d_2 & \frac{n+1}{n-1}d_2^2 \end{bmatrix}, \qquad v_C = \begin{bmatrix} v_{m-1} \\ v_m \\ v_{m+1} \end{bmatrix}.$$

It follows that $\kappa(M^{-1}A_h)$ is equal to $\kappa(M_C^{-1}A_C)$, and finding a matrix D_h that satisfies (1.4) is equivalent to determining d_1, d_2, and $\bar{d}$ to minimize $\kappa(M_C^{-1}A_C)$. While it appears

difficult to solve this problem analytically, it is a simple problem for the numerical optimization code.

Results from the optimization code are plotted in Figs. 1(a,b). In Fig. 1(a) the condition number of the optimally preconditioned matrix is plotted against h^{-1}. Fig. 1(b) shows the ratios $d_1/\bar{d}$ and $d_2/\bar{d}$ for various grid sizes. Note that it is not until the number of grid points n reaches about 10^4 that the asymptotic behavior of the system is approached: $\kappa(M^{-1}A_h) \approx 100$ and $d_1 \approx \bar{d} \approx d_2$. This leads us to question the relevance of asymptotic results such as those in Theorem 2.2 since for problems of typical size, they may not be approached. It is interesting to note, however, that for all problem sizes the scalar $\bar{d}$ is approximately equal to d_2, the diagonal element corresponding to the subregion with the larger diffusion coefficient.

Additional experiments were performed with larger jumps in the diffusion coefficient:

$$\frac{a_2}{a_1} = 10^3, 10^4, 10^6.$$

Results for the case $a_2/a_1 = 10^3$ are plotted in Figs. 1(c,d). The grid size n at which the condition number reaches some fixed fraction, say, .9 of its asymptotic value, seems to grow linearly with the size of the jump in $a(x)$. When a_2/a_1 is 100, a grid of size about 2000 is needed before the condition number of the optimally preconditioned matrix reaches 90, and so for a_2/a_1 equal to 10^k it is only for grid sizes $n \approx 2 \cdot 10^{k+1}$ that the condition number of the optimally preconditioned matrix reaches $.9 \cdot 10^k$. Since large jumps in a diffusion coefficient are usually handled with much coarser grids, the asymptotic results become even less important.

The optimization code was also applied to a similar two-dimensional problem:

$$-\left(\frac{\partial}{\partial x} a \frac{\partial u}{\partial x} + \frac{\partial}{\partial y} a \frac{\partial u}{\partial y}\right) = f \quad \text{in } (0,1)x(0,1),$$

$$u(x,0) = u(x,1) = u(0,y) = u(1,y) = 0,$$

where

$$a(x,y) = \begin{cases} 1, & y < .5, \\ 100, & y > .5. \end{cases} \tag{3.2}$$

The matrix A_h was again derived from a continuous piecewise linear finite element approximation on a uniform grid of size h. The optimization code was used to find the diagonal matrix D_h for which

$$\kappa((D_h \Delta_h D_h)^{-1} A_h)$$

was minimal, where Δ_h is the five-point Laplacian.

The matrix D_h returned by the optimization code was observed to have the form

$$D_h = \begin{pmatrix} d_1 I & & \\ & \bar{d} I_{.5} & \\ & & d_2 I \end{pmatrix},$$

where d_1, d_2, and $\bar{d}$ are scalars, I is the identity corresponding to the subregion $(0, 1) \times (0, .5)$ or $(0, 1) \times (.5, 1)$, and $I_{.5}$ is the identity on the dividing line, $y = .5$. Applying the code directly to this problem, we were not able to work with fine enough grid sizes to determine the asymptotic behavior of the system. Using Theorem 2.3, however,

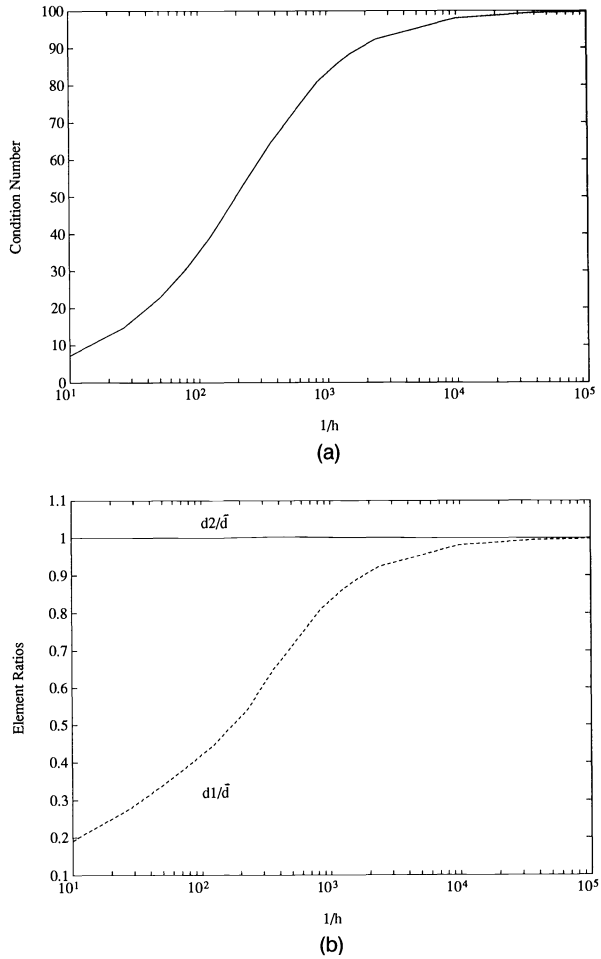


FIG. 1. (a) *Optimal diagonal scaling of the Laplacian as preconditioner (one-dimensional)* ($a2/a1 = 100$). (b) *Element ratios of optimal diagonal scaling (one-dimensional)* ($a2/a1 = 100$).

this problem can again be reduced to a 3×3 eigenvalue optimization problem. It is equivalent to finding scalars d_1, d_2, and $\bar{d}$ to minimize the condition number of $M_C^{-1}A_C$, where

$$A_C = \begin{bmatrix} a_1(\mu-\gamma) & -a_1 & 0 \\ -a_1 & \dfrac{a_1+a_2}{2}\mu & -a_2 \\ 0 & -a_2 & a_2(\mu-\gamma) \end{bmatrix}, \quad M_C = \begin{bmatrix} d_1^2(\mu-\gamma) & -d_1\bar{d} & 0 \\ -d_1\bar{d} & \bar{d}^2\mu & -\bar{d}d_2 \\ 0 & -\bar{d}d_2 & d_2^2(\mu-\gamma) \end{bmatrix}.$$

Here μ is the smallest eigenvalue of the $n \times n$ tridiagonal matrix $T = \text{tridi}(-1, 4, -1)$, and γ is the first component of the solution to the linear system (2.25).

Using this formulation of the problem, the numerical solution becomes very easy for the optimization code. The condition number of the optimally preconditioned matrix

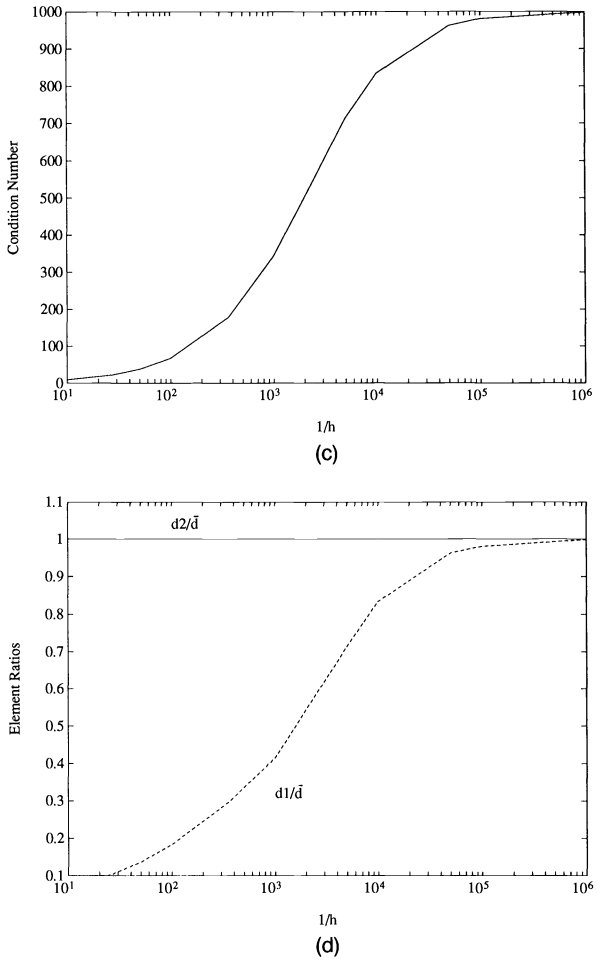


FIG. 1. (c) *Optimal diagonal scaling of the Laplacian as preconditioner (one-dimensional)* ($a2/a1 = 1000$). (d) *Element ratios of optimal diagonal scaling (one-dimensional)* ($a2/a1 = 1000$).

is plotted in Fig. 2(a) for different grid sizes, and the ratios $d_1/\bar{d}$ and $d_2/\bar{d}$ are plotted in Fig. 2(b). For the two-dimensional problem, an even finer grid is required before the asymptotic limit is closely approached. Again, however, even for relatively coarse grids, the scalar $\bar{d}$ was approximately equal to d_2, the diagonal element corresponding to the subregion with the larger diffusion coefficient.

In contrast to the above results, Table 1 shows results for the problem (2.1), where $a(x)$ is given by

$$a(x) = .01 + x^2. \tag{3.3}$$

Although the total variation in $a(x)$ over the interval (0, 1) is approximately the same as that in (3.1) ($a_{\max}/a_{\min} = 101$), it now varies smoothly. The diagonal matrix D_h returned by the optimization code is now very nearly equal to the square root of the diagonal of A_h. Table 1 shows the largest and smallest ratio between the square of a diagonal element of D_h and the corresponding element of A_h. D_h has been multiplied by

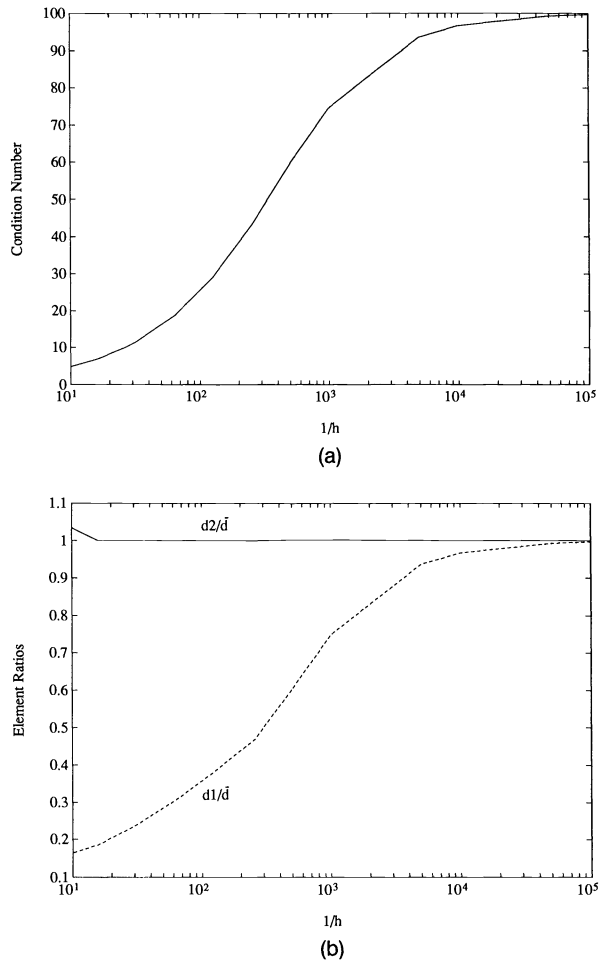


FIG. 2. (a) *Optimal diagonal scaling of the Laplacian as preconditioner (two-dimensional)* ($a2/a1 = 100$). (b) *Element ratios of optimal diagonal scaling (two-dimensional)* ($a2/a1 = 100$).

a scalar so that its center element is equal to the square root of the corresponding diagonal element of A_h. In this case, then, the optimal diagonal matrix D_h is not of the form (2.9) and it does not appear to approach the identity as $h \to 0$. Rather, it appears to approach the square root of the diagonal of A_h.

TABLE 1

Ratios of diagonal elements for the optimal preconditioner and condition number of the preconditioned system (3.3).

$1/h$	$\max_{i=1,\cdots,n} D_{ii}^2/A_{ii}$	$\min_{i=1,\cdots,n} D_{ii}^2/A_{ii}$	$\kappa(M^{-1}A_h)$
10	1.01	.98	1.17
26	1.00	.98	1.26
50	1.00	.99	1.28

4. Further discussion. In the case of a smoothly varying diffusion coefficient a, the differential operator $\nabla\cdot(a\nabla u)$ can be written in the form

$$a\Delta u+\nabla a\cdot\nabla u. \tag{4.1}$$

Consider the equation $\Delta u = -f$. Making a change of variable from u to $v = a^{-1/2}u$ and multiplying this equation by $a^{1/2}$ gives

$$a^{1/2}\Delta(a^{1/2}v)=a\Delta v+\nabla a\cdot\nabla v+a^{1/2}(\Delta a^{1/2})v=-a^{1/2}f. \tag{4.2}$$

The matrix $M = D\Delta_h D$, where D is the square root of the diagonal of A_h, represents the differential operator in (4.2), with the same homogeneous Dirichlet boundary conditions as the original problem. Since this is a second-order self-adjoint operator, it follows that using M as a preconditioner for A_h gives a condition number for the preconditioned system that is $O(1)$, independent of the mesh size [3]. Since the leading terms of the differential operator in (4.2) match those in (4.1), it is perhaps not surprising that this is a near-optimal diagonal scaling.

When the coefficient a is discontinuous or continuous but not differentiable, there is no such analogy between the preconditioner and a differential operator whose leading term(s) match those of the original equation. In this case, a discontinuous diagonal scaling of the Laplacian does not represent a second-order self-adjoint elliptic operator and, as Theorems 2.2 and 2.3 show for a specific problem class, the condition number of the matrix preconditioned in this way may become infinite as h goes to zero.

5. Conclusions. While the problems considered in this paper are very simple ones, the negative result—that, in the limit as $h \to 0$, scaling the Laplacian by a diagonal matrix cannot improve its performance as a preconditioner for a problem with a discontinuous diffusion coefficient—can be expected to hold for more complicated problems as well. Techniques similar to those used in proving Theorems 2.2 and 2.3 should also be applicable to problems defined in different domains, with different boundary conditions, and with multiple discontinuities in the diffusion coefficient.

An interesting question is whether the result that the optimal diagonal scaling (for a fixed-size grid) is piecewise constant would hold also for more complicated problems, e.g., for two-dimensional problems with four different diffusion coefficients in each of four quadrants. Numerical results indicate that in this case the optimal diagonal scaling of the Laplacian is *not* exactly constant in regions with constant diffusion coefficient but has a small amount of variation. Still, for problems with just a few such discontinuities, one might be able to find the optimal piecewise constant diagonal scaling (which is not far from the optimal diagonal scaling) by reducing the problem to a low-dimensional eigenvalue optimization problem and solving with a numerical optimization code. Since the size of the optimization problem is much much smaller than the size of the linear system to be solved, this could prove to be a practical procedure.

Another idea suggested by these results is that the class of diagonal scalings of the Laplacian is not a sufficiently general class to provide an effective preconditioner for problems with a discontinuous diffusion coefficient. A slight generalization to allow scaling the Laplacian by a low-rank update of a diagonal matrix might yield significantly better results for such problems. This idea is currently being investigated using the same optimization code.

REFERENCES

[1] P. CONCUS AND G. GOLUB, *Use of fast direct methods for the efficient numerical solution of nonseparable elliptic equations*, SIAM J. Numer. Anal., 10 (1973), pp. 1103–1120.

[2] A. GREENBAUM AND G. RODRIGUE, *Optimal preconditioners of a given sparsity pattern*, BIT, 29 (1989), pp. 610–634.

[3] T. MANTEUFFEL AND S. PARTER, *Preconditioning and boundary conditions*, SIAM J. Numer. Anal., 27 (1990), pp. 656–694.

[4] A. MAYO, *The fast solution of Poisson's and the biharmonic equations on irregular regions*, SIAM J. Numer. Anal., 21 (1984), pp. 285–299.

[5] M. OVERTON, *On minimizing the maximum eigenvalue of a symmetric matrix*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 256–268.

[6] O. WIDLUND, *On the use of fast methods for separable finite difference equations for the solution of general elliptic problems*, in Sparse Matrices and Their Applications, D. J. Rose and R. A. Willoughby, eds., Plenum Press, New York, 1972.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 847–862, July 1992

010

BOUNDS OF EIGENVALUES OF PRECONDITIONED MATRICES*

O. AXELSSON†

Abstract. Some methods to bound individual eigenvalues of a generalized eigenvalue problem $\lambda Cx = Ax$ are presented, both for general positive semidefinite matrices and for the special case where C is an incomplete factorization of A. This provides accurate estimates of the rate of convergence of preconditioned conjugate gradient methods to solve linear systems with A. In particular, methods are presented to actually numerically compute bounds of the extreme eigenvalues. The estimates enable us to compare modified and unmodified incomplete factorization methods.

Key words. generalized eigenvalue problem, local eigenvalue estimates, upper and lower bounds, preconditioned iterative methods, rate of convergence

AMS(MOS) subject classifications. 65F10, 65F15, 65F50

1. Introduction. Consider the solution of a linear system of algebraic equations, $Ax = b$, where A is symmetric and positive semidefinite using a preconditioning matrix C, which is symmetric and positive definite. In practice, A will be sparse, and C is frequently given as a product of sparse triangular matrices. It is well known that the rate of convergence of the preconditioned conjugate gradient (PCG) method to solve $Ax = b$ by iteration depends on the distribution of eigenvalues of the generalized eigenvalue problem,

$$\lambda Cx = Ax. \tag{1.1}$$

For details, see [2], [18], [15], [8], and [17]. In general, this distribution is not known. However, in this paper we consider some methods for deriving upper and lower bounds of individual eigenvalues of (1.1). Let

$$R = C - A \tag{1.2}$$

and let $\lambda_{\max}(R)$ and $\lambda_{\min}(R)$ denote the largest and smallest eigenvalues of R, respectively. We assume that $\lambda_{\max}(R)$ is nonnegative and $\lambda_{\min}(R)$ is nonpositive, i.e.,

$$\lambda_{\min}(R) \leqq 0 \leqq \lambda_{\max}(R).$$

Note that this can always be achieved by multiplying C with a proper scalar and that such a translation of the eigenvalues does not change the rate of convergence of the PCG method.

Estimates of the extreme eigenvalues can be found for the special case of incomplete factorization preconditioned matrices for difference matrices in [12], [13], [3], [16], and [5]. The present paper gives for the first time fairly complete general tools for estimating interior eigenvalues.

In § 2 we present some general methods to bound the eigenvalues, while in § 4 we derive upper bounds for the special case when C is computed as an incomplete factorization of A.

Such incomplete factorization methods are introduced in § 4. We show also the equivalence between the method of perturbations (see [3], [10], and [5]) and the method

* Received by the editors April 5, 1990; accepted for publication (in revised form) June 4, 1991. This work was supported in part by the Florida State University Supercomputer Computations Research Institute, which is partially funded by the U.S. Department of Energy through contract DE-FC05-85ER250000.

† Department of Mathematics and Computer Science, University of Nijmegen, NL-6525 ED Nijmegen, the Netherlands (u641007@hnykun11).

of relaxation (see [8]) used when computing incomplete factorizations. The derivation of lower eigenvalue bounds when the method of perturbations has been used is discussed in § 3. For alternative methods using graph theory methods, see [11], [7], and [20].

In § 5 we apply the bounds to show the distribution of eigenvalues and the order of condition numbers, in particular, for elliptic difference matrices. We show also that the bounds of the extreme eigenvalues are readily computable, which in particular implies that it can be efficient to use a Chebyshev acceleration iterative method as an alternative to the conjugate gradient method.

2. Bounds of eigenvalues of generalized eigenvalue problems. Consider the generalized eigenvalue problem

$$\lambda Cx = Ax, \tag{2.1}$$

where A is symmetric and positive semidefinite and C is symmetric and positive definite.[1]

We derive upper and lower bounds of the extreme eigenvalues and also of interior eigenvalues. If A is singular, we will consider the positive part of the spectrum. The first bound is elementary and is presented only for comparison with a more general method to follow.

LEMMA 2.1. *Consider the generalized eigenvalue problem* (2.1). *Then*

(a) $\lambda_{\min}(A)/\lambda_{\max}(C) \leqq \lambda_{\min}(C^{-1}A) \leqq \lambda_{\min}(A)/\lambda_{\min}(C)$;

(b) $\lambda_{\max}(A)/\lambda_{\max}(C) \leqq \lambda_{\max}(C^{-1}A) \leqq \lambda_{\max}(A)/\lambda_{\min}(C)$.

Proof. We have

$$\lambda_{\max}(C^{-1}A) = \max_{x \neq 0} \{x^TAx/x^TCx\} \geqq x^TAx/x^TCx$$

for any $x \neq 0$. Let $\hat{x}$ be the eigenvector of A corresponding to $\lambda_{\max}(A)$. Then

$$\lambda_{\max}(C^{-1}A) \geqq \hat{x}^TA\hat{x}/\hat{x}^TC\hat{x} = \lambda_{\max}(A)\hat{x}^T\hat{x}/\hat{x}^TC\hat{x} \geqq \lambda_{\max}(A)/\lambda_{\max}(C).$$

This shows the left-hand side part of Lemma 2.1(b). Clearly

$$\lambda_{\max}(C^{-1}A) \leqq \max_{x \neq 0} \{x^TAx/x^Tx\} / \min_{x \neq 0} \{x^TCx/x^Tx\} = \lambda_{\max}(A)/\lambda_{\min}(C),$$

which shows the right-hand side part. In a similar way, Lemma 2.1(a) follows. □

We now let the eigenvalues $\lambda_i = \lambda_i(C^{-1}A)$ of $C^{-1}A$ and of C, C^{-1}, and A, be ordered in an increasing order and consider bounds of λ_i. For this purpose we shall use the Courant–Fischer theorem to bound eigenvalues of sums and products of matrices.

LEMMA 2.2. *Let A and B be symmetric matrices and let $\lambda_i(A)$, $\lambda_i(A+B)$, and so forth, denote the ith eigenvalues ordered in an increasing order. Then*

(a) $\lambda_i(A) + \lambda_{\min}(B) \leqq \lambda_i(A+B) \leqq \lambda_i(A) + \lambda_{\max}(B)$;

(b) *If $\lambda_{\max}(B)$ is nonnegative and A is positive definite, then*

$$\lambda_i(AB) \leqq \lambda_i(A)\lambda_{\max}(B).$$

(c) *If $\lambda_{\min}(B)$ is nonnegative and A is positive definite, then $\lambda_i(AB) \geqq \lambda_i(A)\lambda_{\min}(B)$.*

Proof. The proof of Lemma 2.2(a) can be found in Wilkinson [23, p. 101], for instance. For completeness, we give the proof. Let $v_1, \cdots, v_n$ denote the eigenvectors of A. Then the Courant–Fischer theorem shows that for any x, $x^Tx = 1$,

$$\begin{aligned}\lambda_i(A+B) &\geqq \min_{x \perp v_1, \cdots, v_{i-1}} x^T(A+B)x \\ &\geqq \min_{x \perp v_1, \cdots, v_{i-1}} x^TAx + \min_{x^Tx=1} x^TBx = \lambda_i(A) + \lambda_{\min}(B),\end{aligned}$$

[1] Note that the eigenvalues of (2.1) are identical to the spectrum of $C^{-1}A$.

which shows the lower bound of Lemma 2.2(a). Similarly,

$$\lambda_i(AB) = \lambda_i(A^{1/2}BA^{1/2})$$

$$\leqq \max_{x \perp v_n, \cdots, v_{i+1}} \left\{ \frac{x^T A^{1/2} B A^{1/2} x}{x^T A x} \cdot \frac{x^T A x}{x^T x} \right\}$$

$$\leqq \max_{x \neq 0} \frac{(A^{1/2}x)^T B (A^{1/2}x)}{(A^{1/2}x)^T (A^{1/2}x)} \cdot \max_{x \perp v_n, \cdots, v_{i+1}} \frac{x^T A x}{x^T x} = \lambda_i(A)\lambda_{\max}(B),$$

which shows Lemma 2.2(b). The proofs of Lemma 2.2(c) and the upper bound of Lemma 2.2(a) follow the same lines. □

THEOREM 2.1. *Let* $R = C - A$ *and assume that* $\lambda_{\min}(R) \leqq 0$. *Then*

$$\lambda_i(C^{-1}A) \leqq 1 + \lambda_i(C^{-1})\lambda_{\max}(-R) \tag{2.2}$$

and, in particular,

$$\lambda_{\max}(C^{-1}A) \leqq 1 + \lambda_{\max}(-R)/\lambda_{\min}(C).$$

Proof. Since $A = C - R$ we have

$$C^{-1}A = I - C^{-1}R = I + C^{-1}(-R).$$

Hence Lemma 2.2(b) shows that

$$\lambda_i(C^{-1}A) = 1 + \lambda_i(C^{-1}(-R)) \leqq 1 + \lambda_i(C^{-1})\lambda_{\max}(-R),$$

where, to show that $\lambda_{\max}(-R)$ is nonnegative, we have used

$$\lambda_{\max}(-R) = -\lambda_{\min}(R),$$

which is nonnegative, by assumption. □

2.1. Alternative bounds. Consider the matrix pencil $\lambda C - A$. This can be written in the form

$$\lambda C - A = \left(\frac{\lambda}{\mu} - 1\right)A + \frac{\lambda}{\mu}R_\mu, \tag{2.3}$$

where $R_\mu = \mu C - A$ and where we let $0 < \lambda < \mu$. This decomposition for the matrix pencil $\lambda C - A$ can be used to show the following alternative bounds, which relate eigenvalues of the generalized eigenvalue problem to the eigenvalues of A.

THEOREM 2.2. *Assume that* μ *is sufficiently large so that* $\lambda_{\min}(\mu C - A) > -\lambda_{\min}(A)$. *Then*

$$\frac{\mu\lambda_i(A)}{\lambda_i(A) + \lambda_{\max}(\mu C - A)} \leqq \lambda_i(C^{-1}A) \leqq \frac{\mu\lambda_i(A)}{\lambda_i(A) + \lambda_{\min}(\mu C - A)}. \tag{2.4}$$

In particular, assuming that $\lambda_{\min}(R) > -\lambda_{\min}(A)$, *we have for* $\mu = 1$,

$$\frac{\lambda_i(A)}{\lambda_i(A) + \lambda_{\max}(R)} \leqq \lambda_i(C^{-1}A) \leqq \frac{\lambda_i(A)}{\lambda_i(A) + \lambda_{\min}(R)}.$$

Further, as $\mu \to \infty$, *we find that*

$$\lambda_i(A)/\lambda_{\max}(C) \leqq \lambda_i(C^{-1}A) \leqq \lambda_i(A)/\lambda_{\min}(C). \tag{2.5}$$

Proof. The upper bound of Lemma 2.2(a) applied to (2.3) shows that for any λ, $0 < \lambda < \mu$.

$$\begin{aligned}\lambda_{n-i+1}(\lambda C - A) &\leqq (1-\lambda/\mu)\lambda_{n-i+1}(-A) + \frac{\lambda}{\mu}\lambda_{\max}(R_\mu) \\ &= -(1-\lambda/\mu)\lambda_i(A) + \frac{\lambda}{\mu}\lambda_{\max}(R_\mu).\end{aligned} \tag{2.6}$$

Note now that $\lambda_{n-i+1}(\lambda C - A) = 0$ if and only if $\lambda = \lambda_i(C^{-1}A)$. See Fig. 1. (The curves for $\lambda_n(\lambda C - A)$ and $\lambda_{n-i+1}(\lambda C - A)$ are generally not straight lines. The unlabeled pair of lines shows the right-hand side function in (2.6) for $i = 1$ and for some $i > 1$, respectively.) Also note that the right-hand side of (2.6) has a zero $\underline{\lambda}_i$ in the interval $(0, \mu)$, where

$$\underline{\lambda}_i = \mu\lambda_i(A)[\lambda_i(A) + \lambda_{\max}(R_\mu)]^{-1}.$$

It is readily seen that this latter becomes a lower bound of $\lambda_i(C^{-1}A)$. Similarly,

$$\begin{aligned}\lambda_{n-i+1}(\lambda C - A) &\geqq (1-\lambda/\mu)\lambda_{n-i+1}(-A) + \frac{\lambda}{\mu}\lambda_{\min}(R_\mu) \\ &= -\left(1 - \frac{\lambda}{\mu}\right)\lambda_i(A) + \frac{\lambda}{\mu}\min(R_\mu)\end{aligned} \tag{2.7}$$

and the right-hand side of (2.6) has a zero $\bar{\lambda}_i$, where

$$\bar{\lambda}_i = \mu\lambda_i(A)/[\lambda_i(A) + \lambda_{\min}(R_\mu)],$$

which becomes an upper bound. The remaining statements follow at once. □

Remark 2.1. The special case of $\mu = 1$ of Theorem 2.2 was considered in [6]. The use of the Courant–Fischer theorem to bound eigenvalues of incomplete factorization preconditioned matrices was suggested by van der Vorst [21]. Note also that the estimates (2.5) for the extreme eigenvalues generate those in Lemma 2.1. In fact, it is readily seen that we could have derived such estimates in Lemma 2.1 for all eigenvalues using the Courant–Fischer theorem, but that these estimates are only a special case of (2.4).

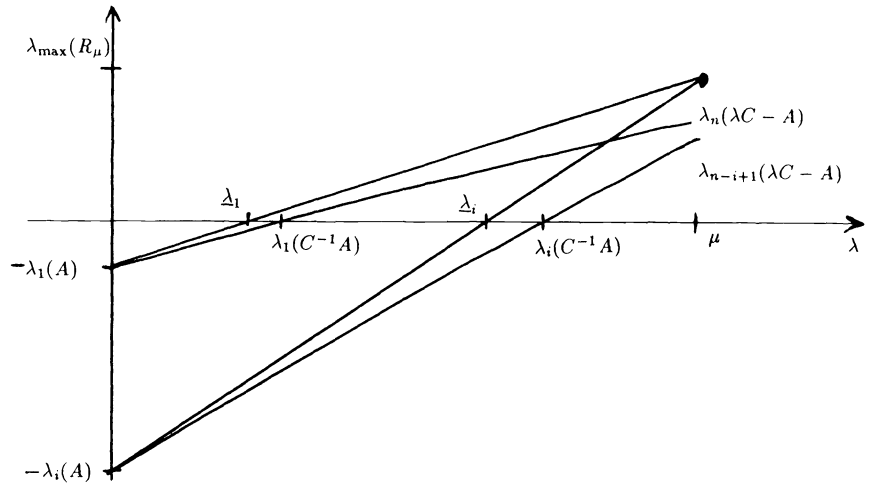


FIG. 1. *Lower eigenvalue bounds using the Courant–Fischer theorem.*

Note finally that the bounds in Theorem 2.2 for $\mu = 1$ converge to the optimal value 1, as $\|R\| \to 0$.

We conclude this section by presenting the following important result regarding a uniform upper bound.

COROLLARY 2.1. *For any μ such that $\lambda_{\min}(\mu C - A) \geqq 0$ we have*

$$\lambda_{\max}(C^{-1}A) \leqq \mu.$$

Proof. This follows from the upper bound of (2.4). Clearly, this can also be seen by the variational characterization

$$\lambda_{\max}(C^{-1}A) = \mu \max_{x, x^T x = 1} \frac{x^T A x}{x^T (\mu C) x},$$

because, by assumption, $x^T(\mu C)x \geqq x^T A x$ for any x. □

3. Computable lower eigenvalue bounds using the method of perturbations. For many years it has been recognized that a certain perturbation technique is a convenient tool to bound eigenvalues of preconditioned matrices; see [3], [16], [7], and also comments in [10]. The technique is to add certain small positive numbers to the diagonal of the original matrix when computing the preconditioning matrix to it. The purpose of this is to control the *upper* eigenvalue of the preconditioned matrix, as we shall see in § 4. For a recent exposition of this, see [5].

In this section we shall assume in addition that A and C are monotone matrices, and that R, to be defined below, is negative semidefinite. As has been shown in [16] and [4], a sufficient condition for this to hold is that A is an M-matrix, when we use the modified method of incomplete factorization (for the perturbed matrix $\tilde{A}$; see [3], [16], and [8]); i.e., let $Rv = 0$ (see below).

The perturbations decrease the *smallest* eigenvalue. How this can be bounded and estimated with a computable bound is shown here.

Let then the diagonal perturbation matrix Δ have nonnegative entries and let $\tilde{A} = A + \Delta$ be the perturbed matrix. Further, let C be a preconditioning to A, let

$$R = C - \tilde{A} \tag{3.1}$$

and assume that Δ is computed so that

$$Rv = 0 \tag{3.2}$$

for some $v > 0$ such that $Av \geqq 0$. Then

$$A + R = C - \Delta, \quad (C - \Delta)v \geqq 0, \quad C^{-1}\Delta \geqq 0.$$

Further, since R is symmetric and negative semidefinite, we have

$$\lambda_i(C^{-1}A) = 1 - \lambda_{n-i+1}(C^{-1}(\Delta + R)) \geqq 1 - \lambda_{\max}(C^{-1}\Delta) \tag{3.3}$$

and $C^{-1}\Delta \geqq 0$ and the Perron–Frobenius theorem shows that

$$\lambda_{\max}(C^{-1}\Delta) = \rho(C^{-1}\Delta).$$

THEOREM 3.1. *Let $C = \hat{A} + R$, $\hat{A} = A + \Delta$, where Δ is a positive semidefinite diagonal matrix, R is negative semidefinite, $Rv = 0$, $v > 0$, and let $\Delta = \operatorname{diag}(\delta_1, \delta_2, \cdots, \delta_n)$. Then*

(a) $\lambda_i(C^{-1}A) \geqq 1 - \min\{\max_i(v_i^{-1}(C^{-1}\Delta v)_i), \max_i(v_i\delta_i(C^{-1}v^{-1})_i)\}$;

(b) $\lambda_i(C^{-1}A) \geqq 1/[1 + \max\{\max_i(v_i^{-1}(A^{-1}\Delta v)_i), \max_i(v_i\delta_i(A^{-1}v^{-1})_i)\}]$,

where v^{-1} denotes the vector with components v_i^{-1} and $v > 0$ is such that $Av \geqq 0$.

Proof. Equation (3.3) shows that

$$\lambda_i(C^{-1}A) \geqq 1 - \rho(C^{-1}\Delta).$$

Further,

$$\rho(C^{-1}\Delta) = \rho(D_v^{-1}C^{-1}\Delta D_v),$$

where $D_v = \operatorname{diag}(v_1, v_2, \cdots, v_n)$. But for any matrix B and any natural norm, we find $\rho(B) \leqq \|B\|$. Hence using this and $\|B\|_1 = \|B^T\|_\infty$, we find that

$$\rho(C^{-1}\Delta) \leqq \min\{\|D_v^{-1}C^{-1}\Delta D_v\|_\infty, \|D_v^{-1}C^{-1}\Delta D_v\|_1\}$$

$$= \min\left\{\max_i\left(\frac{1}{v_i}(C^{-1}\Delta D_v)_i\right), \max_i(v_i\delta_i(C^{-1}v^{-1})_i)\right\},$$

which proves Theorem 3.1(a). Similarly, $C = A + \Delta + R$ shows that

$$\lambda_i(A^{-1}C) = 1 + \lambda_i(A^{-1}(\Delta + R)) \leqq 1 + \lambda_{\max}(A^{-1}\Delta),$$

or since $A^{-1}\Delta \geqq 0$,

$$\lambda_i(A^{-1}C) \leqq 1 + \rho(A^{-1}\Delta)$$

and

$$\lambda_i(C^{-1}A) \geqq 1/[1 + \rho(A^{-1}\Delta)]$$

$$\geqq 1 \Big/ \left[1 + \min\left\{\max_i\left(\frac{1}{v_i}(A^{-1}\Delta v)_i\right), \max_i(v_i\delta_i(A^{-1}v^{-1})_i)\right\}\right],$$

which shows Theorem 3.1(b). □

Remark 3.1. Both the bound in Theorem 3.1(a) and the bound in Theorem 3.1(b) are computable. The first requires only a solution with the preconditioning matrix C, which is usually cheap, while the latter requires a solution with a given matrix A itself. However, the latter bound is sharper, because

$$1/[1 + \rho(A^{-1}\Delta)] = 1/\lambda_{\max}(A^{-1}(A + \Delta)) = \lambda_{\min}((A + \Delta)^{-1}A)$$

$$= \lambda_{\min}((A + \Delta)^{-1}(A + \Delta - \Delta)) = \lambda_{\min}(I - (A + \Delta)^{-1}\Delta)$$

$$= 1 - \rho((A + \Delta)^{-1}\Delta) = 1 - \rho((C - R)^{-1}\Delta) \geqq 1 - \rho(C^{-1}\Delta).$$

An alternative method to derive lower eigenvalue bounds uses matrix graph theory based on the matrix graph corresponding to the nonzero pattern of A. For details, see [7] and for an extension of this method, see [20].

4. Upper eigenvalue bounds for incomplete factorization methods using the method of perturbations. Let A be split as

$$A = D - L - L^T,$$

where D is the (block) diagonal part of A and $(-L)$ is the (block) lower triangular part of A.

We consider now the case when the preconditioning matrix C has been computed as an incomplete factorization. The form of a *generalized symmetric successive overrelaxation method* (SSOR) *matrix* (see [3] and [5]) is

$$C = (X - L)X^{-1}(X - L^T), \tag{4.1}$$

where X is (block) diagonal with positive diagonal entries (or positive definite diagonal blocks) chosen as described below and, in the block matrix case, partitioned consistently with the partitioning of D. Equation (4.1) shows that

$$C = X + LX^{-1}L^T - L - L^T,$$

so

$$R \equiv C - A = X - D + LX^{-1}L^T.$$

Let R^o be defined by

$$(R^o)_{i,j} = \begin{cases} 0, & i = j, \\ (LX^{-1}L^T)_{i,j}, & i \neq j. \end{cases}$$

(In the block matrix case, $(R^o)_{i,j}$ denotes the i,jth block of R^o.) Hence R^o consists of the "fill-in" entries, i.e., the entries of the matrix $LX^{-1}L^T$ which fall outside the (block) diagonal. X is computed recursively from

$$X_i = D_i - (LX^{-1}L^T)_{i,j} - w_i(R^o e)_i, \qquad i = 1, 2, \cdots, \tag{4.2}$$

where D_i is the ith block of D, $e = (1, 1, \cdots, 1)^T$, and w_i are relaxation parameters. (In the block case, w_i is a diagonal matrix.) Note that $(R^o e)_i$ is a scalar if X and D are diagonal and a diagonal matrix, if X and D are block diagonal. (Hence in the latter case, the off-diagonal entries of X_i are determined so that they are equal to the corresponding entries of $D_i - (LX^{-1}L^T)_{i,i}$.) Hence X_i is uniquely determined by (4.2). Also, by choosing w_i properly (even negative, if necessary) we can guarantee that X_i becomes positive definite. The method of using a relaxation parameter was first introduced in [8]. It follows readily that for $w_i = 1$, $i = 1, 2, \cdots$, we have $Ce = Ae$, which is the rowsum criterion and a basis for the so-called modified incomplete factorization method of Gustafsson [16]. When $w_i = 0$, then $R = R^o$ and $R_{ii} = 0$. This is the unmodified method first considered by Meijerink and van der Vorst [19].

We show now that the relaxation method is equivalent to a method of perturbation, which latter type of method has been used in [3] and [5].

For a method of perturbations we compute C; i.e., the diagonal of X in (4.1), such that

$$Ce = (A + \Delta)e,$$

which implies that

$$(X - L - L^T + LX^{-1}L^T)e = (A + \Delta)e$$

or, since $A = D - L - L^T$,

$$(Xe)_i - (De)_i + (LX^{-1}L^T)_{i,i} + (R^o e)_i = \delta_i. \tag{4.3}$$

Comparing (4.2) and (4.3), we see that

$$\delta_i = (1 - w_i)(R^o e)_i. \tag{4.4}$$

Hence the method of relaxation is equivalent to the method of perturbations if the diagonal of A is perturbed by δ_i defined by (4.4). Note that δ_i is nonnegative if w_i is chosen properly.

Next we shall derive an upper bound for the largest eigenvalue of $C^{-1}A$. We extend then a method used in [2], [5], and [7]. It is readily seen that we can write C in (4.1) in the form

$$\mu C=\left[\left(1-\frac{1}{\mu}\right)X-L+\frac{1}{\mu}X\right]\left[\frac{1}{\mu}X\right]^{-1}\left[\left(1-\frac{1}{\mu}\right)X-L^T+\frac{1}{\mu}X\right]$$

or

$$\mu C-A=\mu VX^{-1}V^T+\left(2-\frac{1}{\mu}\right)X-D,$$

where $V=[1-(1/\mu)]X-L$. Hence, since $VX^{-1}V^T$ is positive semidefinite, for any positive μ we find that

$$\lambda_i(\mu C-A)\geqq\lambda_i\left(\left(2-\frac{1}{\mu}\right)X-D\right)\geqq\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right). \tag{4.5}$$

We assume that $2X-D$ is positive definite (which again can be achieved by a proper choice of w_i in (4.2)). Therefore there exists a positive μ for which

$$\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right)\geqq 0.$$

Theorem 2.2 (see the upper bound part of (2.4)) shows now that

$$\lambda_i(C^{-1}A)\leqq\mu\lambda_i(A)\Big/\left[\lambda_i(A)+\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right)\right]. \tag{4.6}$$

For the largest eigenvalue we have then that

$$\max_i\lambda_i(C^{-1}A)\leqq\mu\Big/\left[1+\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right)\Big/\max_i\lambda_i(A)\right].$$

Assume now that

$$\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right)\geqq\left(2-\frac{1}{\mu}\right)x-d\geqq 0, \tag{4.7}$$

where x and d are certain positive scalars. In the case X, D and $2X-D$ are M-matrices, then

$$\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right)\geqq\min_i\left\{\left(\left(2-\frac{1}{\mu}\right)X-D\right)e\right\}_i.$$

In particular, if X and D are diagonal and $D=dI$ has a constant diagonal, then

$$\lambda_{\min}\left(\left(2-\frac{1}{\mu}\right)X-D\right)\geqq\left(2-\frac{1}{\mu}\right)x-d,$$

where x is the smallest diagonal entry of X. Clearly, since $2X-D$ is positive definite, we have $2x-d>0$.

Note that one can always scale A prior to applying the generalized SSOR method; i.e., consider $D^{-1/2}AD^{-1/2}$, where the scaled matrix has a unit diagonal. In case D is block diagonal, for practical reasons, scaling is however viable only if the order of the blocks of D is small.

We now derive the best (i.e., smallest) upper bound of the form (4.6) by choosing the parameter μ properly. For an earlier presentation of this, see [6]. Equations (4.6) and (4.7) show that

$$\lambda_i(C^{-1}A) \leqq \mu\lambda_i(A) \Big/ \left[\lambda_i(A) + \left(2 - \frac{1}{\mu}\right)x - d\right] \tag{4.8}$$

and differentiation of the right-hand side with respect to μ shows the stationary point

$$\mu_i = 2x/[2x - d + \lambda_i(A)], \qquad \lambda_i(C^{-1}A) \leqq \bar{\lambda}_i = \frac{4x\lambda_i(A)}{[2x - d + \lambda_i(A)]^2}. \tag{4.9}$$

However, we require that $[2 - (1/\mu)]x - d \geqq 0$, i.e., that

$$\mu \geqq \frac{x}{2x - d}.$$

Hence (4.9) gives a minimum in the interval $x/(2x - d) \leqq \mu < \infty$ only if $\lambda_i(A) \leqq 2x - d$. If $\lambda_i(A) \geqq 2x - d$, then $\mu = x/(2x - d)$ gives the smallest upper bound in (4.8), and this bound is $x/(2x - d)$. We collect these results in the following theorem.

THEOREM 4.1. *Assume that* $\lambda_{\min}\{[2 - (1/\mu)]X - D\} \geqq [2 - (1/\mu)]x - d$ *for some positive scalars* x, d, *where* $2x - d < 0$. *Then*

$$\lambda_i(C^{-1}A) \leqq \bar{\lambda}_i = \begin{cases} \dfrac{4x\lambda_i(A)}{[2x - d + \lambda_i(A)]^2}, & \text{if } \lambda_i(A) \leqq 2x - d, \\ 1 \Big/ \left(2 - \dfrac{d}{x}\right), & \text{for any eigenvalue.} \end{cases}$$

Note that by choosing the parameters w_i properly, we can require that $2 - (d/x) \geqq \alpha^{-1}$, for any prescribed positive number α, in which case Corollary 2.1 shows that α becomes an upper bound of the spectrum of $C^{-1}A$.

COROLLARY 4.1. *If* $2 - (d/x) \geqq \alpha^{-1}$ *for a positive* α, *then* $\lambda_{\max}(C^{-1}A) \leqq \alpha$.

Note that we can choose first an α and then compute perturbations (δ_i) or relaxation parameters (w_i) such that $2 - (d/x) = \alpha^{-1}$. Hence the upper bound is under the control of the user of the incomplete factorization method.

5. Applications. Consider the selfadjoint problem

$$-\delta u_{xx} - u_{yy} = f \quad \text{in } [0, 1]^2, \tag{5.1}$$

where $\delta > 0$ and with Dirichlet boundary conditions, discretized by central difference approximations on a uniform mesh. Using a natural ordering, we find that

$$a_{i,i-n} = -1, a_{i,i-1} = -\delta, a_{i,i} = d, a_{i,i+1} = -\delta, a_{i,i+n} = -1,$$

where $d = 2(1 + \delta)$, and the mesh width is $h = 1/(n + 1)$. Consider first the choice of the relaxation parameters.

5.1. Optimal choice of the relaxation parameter to minimize the condition number. For the entries of the matrix X in (4.1) we find that

$$x_i = d_i - \sum_{i<j} l_{i,j}x_j^{-1}l_{j,i}^t - w(R^o e)_i, \qquad i = 1, 2, \cdots$$

or

$$x_i = 2(1 + \delta) - \delta^2 x_{i-1}^{-1} - x_{i-n}^{-1} - w\delta(x_{i-n}^{-1} + x_{i-1}^{-1})$$

(apart from corrections at points next to the boundary). We see readily that as $i \to \infty$ and $h \to 0$, x_i converges to a lower bound x, where

$$x = 2(1+\delta) - (1+2w\delta+\delta^2)/x$$

or

$$x = 1+\delta+\{2\delta(1-w)\}^{1/2}. \tag{5.2}$$

Note that

$$2x - d = 2\{2\delta(1-w)\}^{1/2} \tag{5.3}$$

and R has entries converging to

$$r_{i,i\pm(n-1)} = \delta x^{-1}, \qquad r_{i,i} = -2w\delta x^{-1}.$$

Hence

$$\lambda_{\max}(R) = 2\delta(1-w)/x, \qquad \lambda_{\max}(-R) = 2\delta(1+w)/x. \tag{5.4}$$

(To make this proof easier we consider periodic boundary conditions instead of the Dirichlet conditions. This is similar to the local analysis used in the finite difference theory for multigrid methods. In order not to burden the presentation with further such details we leave it to the interested reader to show this.)

Since we require that these numbers are nonnegative and that $(2x - d)$ is positive, we assume that $-1 \leqq w < 1$.

Note that $\lambda_1(A) = (1+\delta)(2\sin \pi h/2)^2$. Theorem 2.2 (with $\mu = 1$) shows that

$$\lambda_1(C^{-1}A)^{-1} \leqq 1 + \lambda_{\max}(R)/\lambda_1(A) = 1 + \frac{2\delta(1-w)}{x}\lambda_1(A)^{-1}.$$

Hence $\lambda_{\max}(C^{-1}A) \leqq 1 + \lambda_{\max}(C^{-1})\lambda_{\max}(-R)$ and Theorem 4.1 show that

$$\max_i \lambda_i(C^{-1}A) \leqq \min\left\{1 \Big/ \left(2 - \frac{d}{x}\right), 1 + \frac{2\delta(1+w)}{x} \cdot \frac{x}{(x-(1+\delta))^2}\right\},$$

where we have made use of

$$\lambda_{\max}(C^{-1}) = \max_v \frac{((X-L)^{-1}v)^T X (X-L)^{-1}v}{v^T v}$$

$$= \max_w \frac{w^T X w}{((X-L)w)^T (X-L)w} \leqq \frac{x}{[x-(1+\delta)]^2}.$$

Hence

$$\lambda_1^{-1} \leqq 1 + \frac{2\delta(1-w)}{1+\delta+\{2\delta(1-w)\}^{1/2}}\lambda_1(A)^{-1}$$

and

$$\max_i \lambda_i \leqq \min\left\{\frac{1}{2} + \frac{1+\delta}{2\{2\delta(1-w)\}^{1/2}}, \frac{2}{1-w}\right\}.$$

The condition number $\kappa(w) = \max_i \lambda_i/\lambda_1$ is therefore bounded above by

$$\kappa(w) \leqq \min\left\{\frac{1}{2} + \frac{1+\delta}{2\{2\delta(1-w)\}^{1/2}}, \frac{2}{1-w}\right\}\left[1 + \frac{2\delta(1-w)\lambda_1(A)^{-1}}{1+\delta+\{2\delta(1-w)\}^{1/2}}\right]$$

or

$$\kappa(w) \leqq \min \left\{ \frac{1}{2}\left[\frac{1+\delta}{\{2\delta(1-w)\}^{1/2}} + 1 + \{2\delta(1-w)\}^{1/2}\lambda_1(A)^{-1} \right], \right.$$
$$\left. \frac{2}{1-w} + 4\delta \frac{1}{1+\delta+\{2\delta(1-w)\}^{1/2}}\lambda_1(A)^{-1} \right\}, \qquad -1 \leqq w < 1.$$

To minimize $\kappa(w)$, we need to choose

$$w = w_{opt} = 1 - \frac{1+\delta}{2\delta}\lambda_1(A) \quad \text{and} \quad w = -1,$$

respectively, for the two functions in the outer bracket.

Hence

$$\min_w \kappa(w) = \min \left\{ \kappa(w_{opt}), 1 + \frac{4\delta}{1+\delta+2\delta^{1/2}}\lambda_1(A)^{-1} \right\}$$
$$= \min \left\{ \frac{1}{2} + \frac{1}{2\sin\frac{\pi h}{2}}, 1 + \frac{4\delta}{(1+\delta^{1/2})^2}\lambda_1(A)^{-1} \right\}$$

and we find that

$$\min_w \kappa(w) = \begin{cases} \dfrac{1}{2} + \dfrac{1}{2\sin\frac{\pi h}{2}}, \delta \gtrsim \dfrac{1}{4}\lambda_1(A)^{1/2}, & \text{for } w = 1 - \dfrac{1+\delta}{2\delta}\lambda_1(A), \\ 1 + \dfrac{4\delta}{(1+\delta^{1/2})^2}\lambda_1(A)^{-1}, \delta \lesssim \dfrac{1}{4}\lambda_1(A)^{1/2}, & \text{for } w = -1. \end{cases}$$

Note that as δ decreases, the optimal value of w switches for $\delta \simeq \lambda_1(A)^{1/2}$ from a value slightly less than unity to the value (-1).

We conclude that the spectral condition number is bounded above by

$$\frac{1}{2} + (\pi h)^{-1} \quad \text{for } w = 1 - \frac{1+\delta}{2\delta}\lambda_1(A) = 1 - O(h^2), \tag{5.5}$$

for any fixed value of δ, but for h fixed and δ significantly small,

$$1 + \frac{4\delta}{(1+\delta^{1/2})^2}\lambda_1(A)^{-1/2}, \quad \text{for } w = -1$$

gives a smaller bound.

The problem (5.1) is a singular perturbation problem for $\delta \ll 1$ and its solution has boundary layers (of width $O(\delta^{1/2})$) at $x = 0$ and $x = 1$ as $\delta \to 0$. To accurately resolve the solution, we need therefore to use a stretched mesh with varying mesh widths in the x direction, like $h_i = h\rho^{(n/2)-i+1}$, $i = 1, 2, \cdots$, where $\rho = \delta^{1/n}$ (i.e., $h_1 = \delta^{1/2}$). We now first scale the matrix A. For such a problem it is recommended to choose the relaxation parameters differently. For instance, we choose an upper bound α of the eigenvalues of $C^{-1}A$ and compute w_i from (4.2) such that $2 - (d/x_i) \geqq \alpha^{-1}$ for all i, where d is the diagonal of the scaled matrix A (i.e., $d = 1$).

5.2. A lower bound of the number of iterations of the preconditioned conjugate gradient method. Consider next the derivation of a lower bound for the number of iterations required to solve the discretization problem (5.1) using a fixed value of the

relaxation parameter. (From the remark above it follows that this means that we do not now consider the singular perturbation case in practice.)

We recall from (5.3) that

$$2x-d=2\{2\delta(1-w)\}^{1/2}$$

and Theorems 2.2 and 4.1 show that for i such that $\lambda_i(A) \leqq 2x - d$, the difference between the upper and lower bounds of $\lambda_i(C^{-1}A)$, is

$$\bar{\lambda}_i-\underline{\lambda}_i=\left(\frac{4x}{[2x-d+\lambda_i(A)]^2}-\frac{1}{\lambda_{\max}(R)+\lambda_i(A)}\right)\lambda_i(A)$$

or, by (5.2) and (5.4),

$$\bar{\lambda}_i-\underline{\lambda}_i=\frac{[2d-\lambda_i(A)](\lambda_i(A))^2}{\{[\lambda_i(A)+2\{2\delta(1-w)\}^{1/2}]^2\cdot[\lambda_i(A)+2\delta(1-w)/x]\}}. \tag{5.6}$$

Hence if $w < 1$ and $\lambda_i(A) \to 0$, $h \to 0$, then (5.5) implies that

$$\lambda_i(C^{-1}A)=\frac{x}{2\delta(1-w)}\lambda_i(A)+O(\lambda_i(A)^2), \qquad h\to 0,$$

which shows that for any fixed value of w, $w < 1$ (independent of the problem parameter h), the eigenvalues λ_i of $C^{-1}A$ are very close to the factor $x/(2\delta(1 - w))$ times the corresponding eigenvalues $\lambda_i(A)$ of A, for all $\lambda_i(A) = o(1)$, $h \to 0$.

We state this result.

LEMMA 5.1. *Consider the generalized* SSOR *incomplete factorization method C in* (4.1), (4.2) *for the difference matrix A for problem* (5.1), *discretized by finite differences with constant stepsize h. For any fixed value w, w < 1, independent of the problem parameter h, the eigenvalues* λ_i *of* $C^{-1}A$ *satisfy*

$$\lambda_i=\frac{x}{2\delta(1-w)}\lambda_i(A)+O(\lambda_i(A)^2), \qquad h\to 0,$$

where $x = 1 + \delta + \{2\delta(1 - w)\}^{1/2}$. *Hence apart from a constant factor, as* $h \to 0$ *the smallest eigenvalues of* $C^{-1}A$ *are distributed essentially as the smallest eigenvalues of A.*

We now estimate the rate of convergence of the preconditioned conjugate gradient method to solve an algebraic system $Ax = b$ when the matrix C defined in (4.1), (4.2) is used as a preconditioner. In particular, we estimate the number of iterations for the case when the smallest eigenvalues of $C^{-1}A$ are distributed essentially as those of A. This is used to discuss the rate of convergence of relaxed methods where $2x - d = O(1)$, and to compare them with relaxed methods where $2x - d = o(1)$, $h \to 0$.

Let then $e^{(k)} = x - x^{(k)}$ be the iteration error at the kth iteration step. As is well known (see [7], for instance) the iteration error satisfies

$$\|e^{(k)}\|_{A^{1/2}}=\min_{p_k\in\pi_k^1}\|p_k(C^{-1}A)e^{(0)}\|_{A^{1/2}}, \tag{5.7}$$

where π_k^1 denotes the set of polynomials of degree k, which are normalized to unity at the origin, i.e., for which $p_k(0) = 1$. By expanding $e^{(0)}$ in the eigenvectors of $C^{-1}A$, we find that

$$\|e^{(k)}\|_{A^{1/2}}\leqq\min_{p_k\in\pi_k^1}\max_{\lambda\in S(C^{-1}A)}|p_k(\lambda)|\,\|e^{(0)}\|_{A^{1/2}},$$

where $S(C^{-1}A)$ denotes the spectrum of $C^{-1}A$. Note also that, as has been shown in [15], there always exists an $e^{(0)}$ (i.e., $x^{(0)}$) for which we have equality in (5.7). Hence

to find the number of iterations required to iterate to a relative precision ε, $\varepsilon < 1$ in the worst-case initial guess, we must find the smallest polynomial of degree k for which

$$\min_{p_k \in \pi_k^1} \max_{\lambda \in S(C^{-1}A)} |p_k(\lambda)| \leqq \varepsilon.$$

Let a, b be the extreme eigenvalues of $C^{-1}A$. If the eigenvalues are distributed as the points, called here the Chebyshev set for $[a, b]$, where the shifted Chebyshev polynomial,

$$T_k\left(\frac{b+a-2\lambda}{b-a}\right), \qquad T_k(x) = \frac{1}{2}\left[(x+\sqrt{x^2-1})^k + (x-\sqrt{x^2-1})^k\right],$$

takes its maximum absolute value, we find that

$$\min_{p_k} \max_{\lambda \in S(C^{-1}A)} |p_k(\lambda)| = \frac{1}{T_k\left(\frac{b+a}{b-a}\right)} = 2\frac{\sigma^k}{1+\sigma^{2k}}, \tag{5.8}$$

where $\sigma = (1-\sqrt{a/b})/(1+\sqrt{a/b})$ is the asymptotic average reduction rate. Equations (5.7) and (5.8) now show that k is the smallest integer for which

$$k \geqq k^*(a,b,\varepsilon) \equiv \ln\left(\frac{1}{\varepsilon} + \sqrt{\frac{1}{\varepsilon^2}-1}\right) \Big/ \ln(\sigma^{-1}) \tag{5.9}$$

and hence

$$k^* \geqq \ln\left(\frac{1}{\varepsilon}\right) \Big/ \ln(\sigma^{-1}).$$

For $\varepsilon \ll 1$ and $b/a \gg 1$ we find an accurate upper bound

$$k^* \leqq \left\lceil \ln\left(\frac{2}{\varepsilon}\right) \Big/ \ln(\sigma^{-1}) \right\rceil \leqq \left\lceil \frac{1}{2}\left(\frac{b}{a}\right)^{1/2} \ln\frac{2}{\varepsilon} \right\rceil. \tag{5.10}$$

However, when the eigenvalues are not distributed as the Chebyshev set, we can find better estimates of k^*. Assume now that

$$S(C^{-1}A) = \bigcup_{i=1}^{p} \lambda \cup [a,b], \tag{5.11}$$

where $\lambda_i < a < b$, $i = 1, \cdots, p$ and that the eigenvalues in $[a, b]$ are distributed as the Chebyshev set for $[a, b]$.

Then we readily derive (see [1], [2], and [18]) the upper bound

$$k^* \leqq p + \left\lceil \ln\frac{2}{\varepsilon} + \sum_{i=1}^{p} \ln\left(\frac{b}{\lambda_i} - 1\right) \right\rceil \Big/ \ln \sigma^{-1}. \tag{5.12a}$$

For small values of p this bound can be quite accurate. If there are only a few eigenvalues $\lambda_1, \cdots, \lambda_p$ which are well separated from the remainder of the spectrum $[a, b]$ and from each other, and if the initial error contains large components in the direction of these eigenvectors, then it is reasonable to assume that these error components are annihilated early in the iteration. In that case we can expect the number of iterations k^* to satisfy

$$k^* \geqq p \left\lfloor \ln\frac{1}{\varepsilon} + \sum_{i=1}^{p} \ln\left(\frac{a}{\lambda_i} - 1\right) \right\rfloor \Big/ \ln \sigma^{-1}. \tag{5.12b}$$

We obtain such an initial distribution of the Fourier components if we use the CGT method (see [14]), where the first iterations consist of Chebyshev iterations with iteration parameters chosen to construct the Chebyshev polynomials for the interval $[a, b]$. These iterations make the Fourier components in $[a, b]$ small, but leave the components for λ_i, $1 \leqq i \leqq p$, essentially unaffected. After these iterations the conjugate gradient method is applied, and (5.12a) and (5.12b) indicate that after some steps the components of the current residual become more uniformly distributed resulting in a number of iterations shown by the bounds in (5.12a), (5.12b). However, to estimate the number of iterations more accurately is outside the scope of this paper, but numerical tests in [8] and [9] indicate that for small values of p the penalty for using an asymptotic average reduction rate based on the interval $[a, b]$ (instead of the interval $[\lambda_1, b]$) contains the logarithmic function terms as shown in (5.12a), (5.12b). At any rate the lower bound $\ln 1/\varepsilon / \ln \sigma^{-1}$ is clearly too optimistic in general. For further discussions of the influence on the rate of convergence of the conjugate gradient method of small, well-separated eigenvalues, see [22].

In practice, the eigenvalues in the interval $[a, b]$ may not be distributed as assumed here, but it is known from numerical evidence that as they become increasingly densely distributed in such an interval, this assumption can be made.

For the estimate of the number of iterations for the relaxed preconditioning method in (4.1), (4.2) we need also the following lemma, which is an obvious corollary of (5.7) and (5.8).

LEMMA 5.2. (a) *Let* S^1 *be a subset of* $S(C^{-1}A)$ *or* (b) *let*

$$S = \bigcup_{s=1}^{p} \lambda_s \cup [a, b], \qquad S^1 = \bigcup_{s=1}^{p} \lambda_s \cup [a', b],$$

where $a' > a$.

Then the conjugate gradient method for S^1 converges to a given precision (ε) with a number of iterations at most equal to that for S, when the initial iteration error in both cases is the worst case.

We consider now the case where the smallest eigenvalues of $C^{-1}A$ are assumed to be distributed in the same way as the corresponding eigenvalues of A. As Lemma 2.1 shows, this is a valid assumption when $2X - D$ is positive definite, uniformly in the problem parameter h.

We assume further that the eigenvalues of A are distributed as for the central difference operator. In fact, Lemma 5.2 shows that it suffices to consider the one-dimensional case, where $\lambda_s(A) = \mu_s = [2 \sin (s\pi h/2)]^2$, $s = 1, 2, \cdots, n$, $h = 1/(n + 1)$ because for the multidimensional case (an $n \times n$ or an $n \times n \times n$ mesh) the spectrum contains these eigenvalues (apart from a constant factor) as a subset. Note also that the set $\{\mu_s\}$ (and $\{\lambda_s\}$) becomes very dense near the right endpoint b, hence the approximation with a Chebyshev set is a valid one for intervals near b. Therefore the estimate we derive can, for practical purposes, be used as a bound for the actual number of iterations and for elliptic problems in any space dimension.

Note first that the Chebyshev set of points for the interval $[a, b]$ is

$$\frac{b+a}{2} - \frac{b-a}{2} \cos \frac{l\pi}{k}, \qquad l = 0, 1, \cdots, k.$$

Hence if we use a Chebyshev polynomial of degree $k = n + 1$, we find that these points are very close to the eigenvalues

$$\lambda_s(A) = \mu_s = 2\left(1 - \cos \frac{s\pi}{n+1}\right), \qquad s = 1, 2, \cdots, n.$$

This equation and (5.7) show that if the initial errors are distributed uniformly (with respect to the norm $\|\cdot\|_{A^{1/2}}$) and if ε is sufficiently small we always need $O(n) = O(h^{-1})$ iterations; i.e., the same order as predicted by the upper bound (5.10). (Naturally, in the absence of round-off errors, we never need more than n iterations when there are n disjoint eigenvalues.) Actually, numerical tests of the conjugate gradient method to solve a system with matrix $A = \text{tridiag}(-1, 2, -1)$ shows that the residuals decay slowly for the first $n - 1$ iterations and then drop significantly for the next iteration, which shows that for ε sufficiently small, we generally need (at least) n iterations.

The above indicates that the number of iterations of the relaxed method with w fixed, $w < 1$, are always $O(h^{-1})$ for proper initial distributions of the Fourier coefficients of the error. Another indication of this is the following argument.

With $a = (2 \sin (p + 1)\pi h/2)^2$, we choose the largest p^* eigenvalues for which

$$\ln\left(\frac{a}{\mu_s} - 1\right) \geqq \ln(1 + \delta'), \qquad 1 \leqq s \leqq p^*,$$

for some positive δ'. Again, Lemma 5.2 shows that the estimate of the corresponding number of iterations will be a lower bound of the actual number of iterations because we base it on a subset of the actual spectrum. We find then that

$$\frac{a}{\mu_s} \geqq 2 + \delta', \qquad 1 \leqq s \leqq p^*.$$

Using the elementary inequality

$$\frac{2}{\pi} x \leqq \sin(x) \leqq x, \qquad 0 \leqq x \leqq \frac{\pi}{2},$$

we find that

$$\frac{a}{\mu_s} \geqq \left(\frac{2}{\pi}\right)^2 \left(\frac{p+1}{s}\right)^2 \geqq 2 + \delta'$$

if

$$s \leqq p^* = \left[\frac{2}{\pi(2 + \delta')^{1/2}}(p+1)\right].$$

The above and (5.12b) show then that (assuming the lower bound holds)

$$k \geqq p^* + \left[\ln\frac{1}{\varepsilon} + \sum_{1}^{p^*} \ln(1 + \delta')\right] \Big/ \ln(\sigma^{-1}), \tag{5.13}$$

where $\sigma = (1 - \sqrt{a/b})/(1 + \sqrt{a/b})$ and $\sqrt{a/b} \leqq \text{const}\,(p + 1)h$, where const does not depend on p and h. Equation (5.13) shows then that

$$k \geqq \frac{p^* \ln(1 + \delta')}{\ln(\sigma^{-1})} \sim O(h^{-1}), \qquad h \to 0,$$

either if $a/b = O(1)$ where $p = O(h^{-1})$ or if $a/b = O(h^{2\nu})$, $0 < \nu \leqq 1$, where $p = O(h^{\nu - 1})$.

If $2x - d = O(1)$, $h \to 0$ (i.e., if $w < 1$ and w independent of h), (5.13) shows that the number of iterations is $k^* \geqq O(h^{-1})$. Hence we have presented arguments for two entirely different distributions of initial errors to show that $k^* \geqq O(h^{-1})$. On the other hand, if $1 - w = O(\lambda_1(A)) = O(h^2)$, (5.6) shows that the spectral condition num-

ber of $C^{-1}A$ is $O(h^{-1})$ and the upper bound (5.10) shows the number of iterations $O(h^{-1/2}) \ln (2/\varepsilon)$ in this case.

Clearly, for special choices of the initial residual (or initial vector), it may happen that the unmodified method converges faster than the modified. In particular, this seems to be the case in the presence of round-off errors of significant size unless, for instance, a method of perturbation in [3] or the method of relaxation in [8] is used. For some tests indicating this, see [21]. See also comments in [20].

Acknowledgments. Anne Greenbaum's comments on an earlier version of this paper helped to improve the presentation of parts of the paper.

Theorem 3.1(a) was derived by L. Yu. Kolotilina, which is also gratefully acknowledged.

REFERENCES

[1] L. ANDERSON, SSOR *preconditioning of Toeplitz matrices*, Ph.D. thesis, Department of Computer Sciences, Chalmers University of Technology, Göteborg, Sweden, 1976.

[2] O. AXELSSON, *A class of iterative methods for finite element equations*, Comput. Methods Appl. Mech. Engrg., 9 (1976), pp. 123–137.

[3] ———, *A generalized* SSOR *method*, BIT, 12 (1972), pp. 443–467.

[4] ———, *A survey of preconditioned iterative methods for linear systems of algebraic equations*, BIT, 25 (1985), pp. 166–187.

[5] ———, *On iterative solution of elliptic difference equations on mesh-connected array of processors*, Internat. J. High Speed Comput., 1 (1989), pp. 165–183.

[6] ———, *Condition number estimates for elliptic difference problems with anisotropy*, Equadiff 7, Teubner-Texte zur Mathematik, Band 118, Teubner-Verlag, Leipzig, 1990, pp. 218–224.

[7] O. AXELSSON AND V. A. BARKER, *Finite Element Solution of Boundary Value Problems, Theory and Computation*, Academic Press, Orlando, FL, 1984.

[8] O. AXELSSON AND G. LINDSKOG, *On the eigenvalue distribution of a class of preconditioning methods*, Numer. Math., 48 (1986), pp. 479–498.

[9] ———, *On the rate of convergence of the preconditioned conjugate gradient method*, Numer. Math., 48 (1986), pp. 499–523.

[10] R. BEAUWENS, *On Axelsson's perturbations*, Linear Algebra Appl., 68 (1985), pp. 221–242.

[11] ———, *Lower eigenvalue bounds for pencils of matrices*, Linear Algebra Appl., 85 (1987), pp. 101–119.

[12] T. DUPONT, R. KENDALL, AND H. H. RACHFORD, *An approximate factorization procedure for solving selfadjoint elliptic difference equations*, SIAM J. Numer. Anal., 5 (1968), pp. 559–573.

[13] T. DUPONT, *A factorization procedure for the solution of elliptic difference equations*, SIAM J. Numer. Anal., 5 (1968), pp. 753–782.

[14] M. ENGELI, T. GINSBURG, H. RUTISHAUSER, AND E. STIEFEL, *Refined iterative methods for computation of the solution and the eigenvalues of selfadjoint boundary value problems*, Mitteilungen aus dem Institut für angewandte Mathematik, no. 8, ETH, Zürich, Switzerland, 1959.

[15] A. GREENBAUM, *Comparison of splittings used with the conjugate gradient algorithm*, Numer. Math., 33 (1979), pp. 181–194.

[16] I. GUSTAFSSON, *A class of first order factorization methods*, BIT, 18 (1978), pp. 142–156.

[17] V. P. IL'IN, *Incomplete factorization methods*, Soviet J. Numer. Anal. Math. Modelling, 3 (1988), pp. 179–198.

[18] A. JENNINGS, *Influence of the eigenvalue spectrum of the convergence rate of the conjugate gradient method*, J. Inst. Math. Appl., 20 (1977), pp. 61–72.

[19] J. A. MEIJERINK AND H. A. VAN DER VORST, *An iterative solution method for linear systems of which the coefficient matrix is a symmetic M-matrix*, Math. Comp., 31 (1977), pp. 148–152.

[20] Y. NOTAY, *Solving positive (semi) definite linear systems by preconditioned iterative methods*, in Preconditioned Conjugate Gradient Methods, O. Axelsson and L. Yu. Kolotilina, eds., Lecture Notes in Mathematics, 1457, Springer-Verlag, Berlin, 1990, pp. 105–125.

[21] H. A. VAN DER VORST, *The convergence behaviour of preconditioned conjugate gradient and conjugate gradient square methods*, talk presented at the Conference on Preconditioned Conjugate Gradient Methods, June 19–21, 1989, University of Nijmegen, the Netherlands.

[22] H. A. VAN DER VORST AND A. VAN DER SLUIS, *The rate of convergence of conjugate gradients*, Preprint No. 354, Department of Mathematics, University of Utrecht, Utrecht, the Netherlands, 1984.

[23] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 863–886, July 1992

011

ITERATIVE SOLUTION METHODS AND PRECONDITIONERS FOR BLOCK-TRIDIAGONAL SYSTEMS OF EQUATIONS*

S. HOLMGREN† AND K. OTTO†

Abstract. Systems of equations arising from implicit time discretizations and finite difference space discretizations of systems of partial differential equations in two space dimensions are considered. The nonsymmetric linear systems are solved using a preconditioned CG-like iterative method. A class of preconditioners, referred to as semicirculant, is examined. For a scalar hyperbolic model problem, it is shown that the number of iterations required using a preconditioner in the semicirculant class is independent of the number of unknowns, provided that the quotient κ between the time- and space-step is held constant. Also, it is shown that the number of iterations grows no faster than $\sqrt{\kappa}$. This type of favorable convergence property is also observed in numerical experiments solving more complicated problems.

Key words. hyperbolic PDE, difference equations, CG-like methods, circulant preconditioners, spectra

AMS(MOS) subject classifications. 65F10, 65N20

1. Introduction.

In this report we study conjugate gradient–like iterative methods and preconditioners for an important class of systems of equations. We consider iterations for nonsymmetric coefficient matrices and semicirculant preconditioners. Some theoretical results for the spectrum of the preconditioned coefficient matrix are presented for a scalar model problem.

The systems of equations arise from implicit time discretizations of systems of time-dependent partial differential equations in two space dimensions. We only consider first-order partial differential equations (PDEs), but the solution procedure is not limited to this class of equations. Our aim is to develop efficient solution procedures for some problems involving the Euler or Navier–Stokes equations. Specifically, we are interested in problems where the systems of PDEs have time-scales of different orders of magnitude, and where the fast time-scales do not have to be resolved. An important application is almost incompressible flow [14], [16], where the sound waves in the medium are much faster than the motion of the fluid. Here it is possible to calculate an accurate solution even if the fast sound oscillations are not resolved by the time-marching method, provided these oscillations are not present in the initial solution. For an explicit time-marching method the time-step would, due to the stability criterion, be restricted by the fastest time-scale. Thus, an explicit time-marching method is "too accurate" resolving fast oscillations that are not present in the problem, and it is preferable to use an implicit time-marching scheme.

1.1. Notation.

All matrices used in the presentation are square matrices. $I_{(n)}$ denotes the identity matrix of order n. We define the matrices $\mathrm{diag}_{(n)}(\alpha_i)$, $\mathrm{trip}_{(n)}(\alpha_i, \beta_i, \gamma_i)$, and $\mathrm{circ}_{(n)}(\alpha_1, \alpha_2, \cdots, \alpha_n)$ in the following way:

$$\mathrm{diag}_{(n)}(\alpha_i) = \begin{pmatrix} \alpha_1 & & \\ & \ddots & \\ & & \alpha_n \end{pmatrix},$$

* Received by the editors April 5, 1990; accepted for publication (in revised form) July 25, 1991. This work was supported by the Swedish Board for Technical Development (STU).

† Department of Scientific Computing, Uppsala University, Sturegatan 4B, S-753 14 Uppsala, Sweden (sverker@tdb.uu.se and kurt@tdb.uu.se).

$$\mathrm{trip}_{(n)}(\alpha_i, \beta_i, \gamma_i) = \begin{pmatrix} \beta_1 & \gamma_1 & & & \alpha_1 \\ \alpha_2 & \beta_2 & \gamma_2 & & \\ & \ddots & \ddots & \ddots & \\ & & \alpha_{n-1} & \beta_{n-1} & \gamma_{n-1} \\ \gamma_n & & & \alpha_n & \beta_n \end{pmatrix},$$

$$\mathrm{circ}_{(n)}(\alpha_1, \alpha_2, \cdots, \alpha_n) = \begin{pmatrix} \alpha_1 & \alpha_2 & \cdots & \alpha_{n-1} & \alpha_n \\ \alpha_n & \alpha_1 & \alpha_2 & \cdots & \alpha_{n-1} \\ \ddots & \ddots & \ddots & \ddots & \ddots \\ \alpha_3 & \cdots & \alpha_n & \alpha_1 & \alpha_2 \\ \alpha_2 & \alpha_3 & \cdots & \alpha_n & \alpha_1 \end{pmatrix}.$$

1.2. The structure of the coefficient matrices. We study the solution of linear systems of equations, where the structure of the coefficient matrices is block-tridiagonal periodic. The blocks on the main diagonal are tridiagonal periodic and each entry in the blocks is a small matrix with arbitrary structure. We define the B-class of matrices by

$$B = \mathrm{trip}_{(m_2)}(A_j, B_j^{(1)}, C_j), \tag{1.1}$$

where

$$A_j = \mathrm{diag}_{(m_1)}(\nu_{i,j,-2}),$$

$$B_j^{(1)} = \mathrm{trip}_{(m_1)}(\nu_{i,j,-1}, \nu_{i,j,0}, \nu_{i,j,1}),$$

$$C_j = \mathrm{diag}_{(m_1)}(\nu_{i,j,2}).$$

Here $\nu_{i,j,l}$, $l = -2, \cdots, 2$, are $n_c \times n_c$ matrices. In the definitions above, we have used periodic tridiagonal matrices in order to allow periodic boundary conditions in the PDE problems.

We consider linear systems of n_c PDEs in two space dimensions $x = (x_1, x_2)$ of the form

$$\frac{\partial u}{\partial t} = \mathscr{P} u, \tag{1.2}$$

where u is an n_c-vector containing the unknown functions and $\mathscr{P}$ is a spatial differential operator. We only consider first-order operators $\mathscr{P}$, where the spatial derivatives are approximated with finite difference operators using a five-point stencil. Using single-step implicit time-marching methods demands that a system of equations with a coefficient matrix belonging to the B-class is solved for the unknowns at each time level. Also, for the nonlinear operators in the Euler and Navier–Stokes equations, we obtain similar matrices by using a linearization [4], yielding a second-order accurate noniterative time-marching scheme, or an operator splitting [14], where implicit time-marching is used only for a part of the operator.

The structure of the coefficient matrices is relatively independent of the class of PDEs, but the properties of the matrices and therefore the "best" solution procedures vary with the properties of the underlying differential equations. Since the solution methods described in this report basically depend only on the structure of the coefficient matrix, it is possible to define these procedures for a large number of PDE problems.

However, for specific classes of PDEs there are probably more efficient solution algorithms available.

1.3. Properties of the systems of equations. We now make some important observations regarding the properties of the matrices in the B-class.

—The matrices B are very sparse, even inside the band. For an $m \times m$-grid, B is $(n_c m^2) \times (n_c m^2)$ and the bandwidth is $n_c m$, but less than $5n_c^2 m^2$ elements are nonzero. If standard Gaussian elimination is used practically all of the band will be filled, and the memory requirement will be $\mathcal{O}(n_c^2 m^3)$. For large grids the memory requirements effectively prohibit the use of standard direct solution methods. For a CG-like iterative method the memory requirement will be $\mathcal{O}(n_c^2 m^2)$. It is important that the memory requirement for a preconditioner used in conjunction with the iteration also is not larger than $\mathcal{O}(n_c^2 m^2)$.

—The matrices B are generally nonsymmetric. First-order derivatives in $\mathcal{P}$ approximated with centered differences contribute to the total matrix in an unsymmetric fashion, with elements like $\pm k/(2h)$ on the off-diagonals corresponding to the nearest neighbor gridpoints in the direction of differentiation.

—The matrices B are also generally not diagonally dominant. Assume that the time-step k is large compared to the space-step h. If $\mathcal{P}$ contains only first-order derivatives and centered differences are used, the nondiagonal dominance (and skew-symmetry) increases linearly with the time-step. In terms of the spectrum of B this implies that the largest imaginary parts of the spectrum grow approximately linearly with the quotient of the time- and space-step. Despite these complications it is often preferable to use centered differences. This avoids keeping track of the sign of the characteristic variables in each space- and time-point, and makes it easier to achieve high accuracy in space. Hence, we will use centered differences for our model problems.

—For hyperbolic and incompletely parabolic PDE problems with analytical Dirichlet boundary conditions, a number of numerical boundary conditions will also be required.

We now introduce the notion of a $(\cdot, \cdot)$ PDE problem, where "$\cdot$" is either "P," denoting "periodic," or "D," denoting "Dirichlet." The first position in the parenthesis holds the symbol describing the type of boundary condition in the x_1-direction and so on. For instance, a (P, D)-problem is a PDE problem with periodic boundary conditions in the x_1-direction, and analytical Dirichlet boundary conditions in the x_2-direction. A typical realistic problem is of (D, D) type.

1.4. CG-like iterative methods. As mentioned in § 1.3 the coefficient matrices B are nonsymmetric. There is a large number of generalizations of CG-like algorithms for nonsymmetric systems of equations, e.g., generalized conjugate gradients (GCG) [1], [3]; generalized conjugate residuals (GCR) [11], [12]; ORTHOMIN [23]; ORTHODIR and ORTHORES [25]; generalized minimum residual (GMRES) [20]; the Axelsson least-squares method (Axel-LS) [2]; biconjugate gradients (BICG) [13]; and conjugate gradients squared (CGS) [21]. We believe that the most critical choice is to be made among different preconditioners, and not among different iterative methods. We have chosen one representative method: restarted generalized conjugate residuals, GCR(l).

We use left preconditioning, i.e., the iterative method is applied to the system

$$M^{-1}Bu^{n+1} = M^{-1}b, \tag{1.3}$$

where M is a nonsingular matrix. The preconditioner M should be chosen so that the system $Mw = y$ is easy to solve and the number of iterations is significantly reduced.

Assume that it can be established that the spectrum of $M^{-1}B$ is contained in an ellipse centered on the real axis with center in c, foci in $c + e$ and $c - e$, and semimajor axis a, and that this ellipse does not contain the origin. Then it is proved [19], [3] that the asymptotic convergence factor ρ for the GCR(∞)-iterations is given by

$$\rho = \left| \frac{a + \sqrt{a^2 - e^2}}{c + \sqrt{c^2 - e^2}} \right| < 1. \tag{1.4}$$

Using further knowledge of the eigenvalue distribution it is possible to derive sharper bounds on the convergence rate. For CG-like methods it is important that the eigenvalues of $M^{-1}B$ are highly multiple or cluster near a small number of points in the complex plane. The latter property may be more important than the condition number or the size of the enclosing ellipse.

All iterations are considered to have converged when the following condition holds:

$$\frac{\| M^{-1}(b - Bx_k) \|_2}{\| M^{-1} b \|_2} < \varepsilon. \tag{1.5}$$

2. Semicirculant preconditioners. In this section we define a class of preconditioners for coefficient matrices in the B-class. We refer to preconditioners in this class as being semicirculant. Let $\otimes$ denote the Kronecker product. A semicirculant preconditioner M for B is then defined by

$$M = \mathrm{trip}_{(m_2)}(L_j, M_j^{(1)}, U_j), \tag{2.1}$$

where

$$L_j = I_{(m_1)} \otimes \rho_{j,-2},$$

$$M_j^{(1)} = \mathrm{circ}_{(m_1)}(\rho_{j,0}, \rho_{j,1}, 0, \cdots, 0, \rho_{j,-1}),$$

$$U_j = I_{(m_1)} \otimes \rho_{j,2}.$$

Here $\rho_{j,l}$, $l = -2, \cdots, 2$, are $n_c \times n_c$ matrices. We discuss the choice of these parameters in § 2.2. From the definition it is clear that M is formed by exchanging the periodic tridiagonal matrices $B_j^{(1)}$ in the definition of B by circulant matrices. Not requiring the outermost block-level of the preconditioner to be circulant makes it possible to retain information from variable coefficients and numerical boundary conditions in the space direction associated with m_2. The coefficients in the other space direction are reduced to constants, since circulant matrices are Toeplitz. We therefore expect that the best performance, using semicirculant preconditioners, is achieved when the difference approximation of the PDE has constant or almost constant coefficients in the space direction associated with m_1. Observe that even if the approximation of the PDE has constant coefficients in this space direction the matrices $B_j^{(1)}$ will generally not be Toeplitz, because of the entries added by the numerical boundary conditions. However, these entries only disturb the Toeplitz structure in the first and last row of the matrices. Also observe that the preconditioning matrix M is as sparse as B, and it is completely described by the $5m_2$ $n_c \times n_c$ matrices $\rho_{j,l}$, $l = -2, \cdots, 2, j = 1, \cdots, m_2$.

2.1. Preconditioner solve. The reason for introducing circulant matrices in the preconditioner is that such matrices are diagonalized by using discrete Fourier transforms (DFTs), which are performed by the fast Fourier transform (FFT) algorithm. Below we

derive the exact procedure for the preconditioner solve $w = M^{-1}y$. First we introduce some notation. Let

$$\omega_n = \exp(i2\pi/n) \tag{2.2}$$

and define $F_{(n)}$ as the nth-order Fourier matrix:

$$[F_{(n)}]_{j,k} = n^{-1/2} \cdot \omega_n^{(j-1)(k-1)}, \qquad j,k = 1, \cdots, n. \tag{2.3}$$

The algebraic relationships (2.4a)–(2.4f) are proved in [9]:

$$A \otimes (B \otimes C) = (A \otimes B) \otimes C, \tag{2.4a}$$

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD), \tag{2.4b}$$

$$(A \otimes B)^* = A^* \otimes B^*, \tag{2.4c}$$

$$F_{(n)}^* F_{(n)} = I_{(n)}. \tag{2.4d}$$

If A is $m \times m$ and B is $n \times n$ then

$$A \otimes B = (A \otimes I_{(n)})(I_{(m)} \otimes B) = (I_{(m)} \otimes B)(A \otimes I_{(n)}). \tag{2.4e}$$

Let the matrices $G_0, G_1, \cdots, G_{m-1}$ be $n \times n$. Then

$$(F_{(m)}^* \otimes I_{(n)})\, \mathrm{circ}_{(m)}(G_0, G_1, \cdots, G_{m-1})(F_{(m)} \otimes I_{(n)}) = \mathrm{diag}_{(m)}(\Lambda_j), \tag{2.4f}$$

where

$$\Lambda_j = \sum_{k=0}^{m-1} \omega_m^{(j-1)k} G_k.$$

Now consider the transformation

$$\tilde{M} \equiv (I_{(m_2)} \otimes Q^*) M (I_{(m_2)} \otimes Q), \tag{2.5}$$

where

$$Q \equiv F_{(m_1)} \otimes I_{(n_c)}. \tag{2.6}$$

This transformation yields

$$M^{-1} = (I_{(m_2)} \otimes Q)\, \tilde{M}^{-1} (I_{(m_2)} \otimes Q^*). \tag{2.7}$$

THEOREM A.

$$\tilde{M} = \mathrm{trip}_{(m_2)}(L_j, \Lambda_j^{(1)}, U_j), \tag{2.8}$$

where

$$\Lambda_j^{(1)} = \mathrm{diag}_{(m_1)}(\Lambda_{i,j}^{(0)}),$$

and

$$\Lambda_{i,j}^{(0)} = \rho_{j,0} + \omega_{m_1}^{(i-1)} \rho_{j,1} + \omega_{m_1}^{-(i-1)} \rho_{j,-1}.$$

Proof. From (2.5) and (2.1) it is clear that

$$\tilde{M} = \mathrm{trip}_{(m_2)}(Q^* L_j Q, Q^* M_j^{(1)} Q, Q^* U_j Q).$$

(i) Consider $Q^* L_j Q$ and $Q^* U_j Q$: Using (2.4b) twice and then (2.4d) we have

$$\begin{aligned} Q^* L_j Q &= (F^*_{(m_1)} \otimes I_{(n_c)})(I_{(m_1)} \otimes \rho_{j,-2})(F_{(m_1)} \otimes I_{(n_c)}) \\ &= (F^*_{(m_1)} F_{(m_1)}) \otimes \rho_{j,-2} \\ &= I_{(m_1)} \otimes \rho_{j,-2} \\ &= L_j. \end{aligned}$$

The same technique yields

$$Q^* U_j Q = I_{(m_1)} \otimes \rho_{j,2} = U_j.$$

(ii) Consider $Q^* M_j^{(1)} Q$: Using (2.4f) we have

$$\begin{aligned} Q^* M_j^{(1)} Q = {} & (F^*_{(m_1)} \otimes I_{(n_c)}) \\ & \times \operatorname{circ}_{(m_1)} (\rho_{j,0}, \rho_{j,1}, 0, \cdots, 0, \rho_{j,-1}) \\ & \times (F_{(m_1)} \otimes I_{(n_c)}) \\ = {} & \operatorname{diag}_{(m_1)} (\Lambda^{(0)}_{i,j}), \end{aligned}$$

where

$$\Lambda^{(0)}_{i,j} = \rho_{j,0} + \omega_{m_1}^{(i-1)} \rho_{j,1} + \omega_{m_1}^{-(i-1)} \rho_{j,-1}. \qquad \square$$

Using Theorem A, we see that a preconditioner solve $w = M^{-1} y$ can be performed according to:

$$\begin{aligned} & v := (I_{(m_2)} \otimes (F^*_{(m_1)} \otimes I_{(n_c)}))y, \\ & \text{Solve } \tilde{M} z = v, \\ & w := (I_{(m_2)} \otimes (F_{(m_1)} \otimes I_{(n_c)}))z. \end{aligned}$$

A more algorithmic formulation of the preconditioner solve follows.

TWO-DIMENSIONAL PRECONDITIONER SOLVE.

for $j = 1$ to $m_2 n_c$ do (in parallel):
    Perform an FFT of length m_1.
endfor
for $i = 1$ to m_1 do (in parallel):
    Solve a (possibly periodic) block-tridiagonal system of equations with blocksize n_c and $m_2 n_c$ unknowns.
endfor
for $j = 1$ to $m_2 n_c$ do (in parallel):
    Perform an FFT of length m_1.
endfor

The semicirculant framework can be extended to matrices in the three-dimensional B-class [18], which arises in space discretizations using a seven-point stencil. The three-dimensional preconditioner solve is based on FFT methods in two dimensions and the solution of block-tridiagonal systems in the third dimension.

2.2. Choice of parameters. Different choices of the parameters $\rho_{j,l}$, $l = -2, \cdots, 2$, $j = 1, \cdots, m_2$, result in different preconditioners. We first make the observation that if $\rho_{1,-2}$ and $\rho_{m_2,2}$ are zero, the matrix M will be nonperiodic block-tridiagonal at the outermost

level. For a $(\cdot, D)$-problem there is usually no reason for introducing a periodic structure at the outermost block-level. Still we have chosen to define the semicirculant preconditioners as periodic tridiagonal matrices, since this definition allows us to include some block-circulant preconditioners [17] in this more general framework. The methods described below for calculating $\rho_{j,l}$ are partially motivated by the following theorem, presented in [7].

THEOREM B. *Let A be an $n \times n$ matrix with elements $[A]_{i,j}$. Then the circulant matrix*

$$C = \mathrm{circ}_{(n)}(c_1, c_2, \cdots, c_n),$$

where

$$c_i = \frac{1}{n} \sum_{j=1}^{n} [A]_{j,k}, \qquad k = (j+i-2) \bmod (n) + 1,$$

is an optimal circulant preconditioner for A in the sense that C solves

$$\min_{\tilde{C} \in \mathscr{C}_n} \|\tilde{C} - A\|_F,$$

where $\mathscr{C}_n$ is the family of all circulant matrices of size n, and $\|\cdot\|_F$ denotes the Frobenius norm.

Observe that the entries in C are formed by averaging the diagonals of A, extended by wraparound. This implies that the number of nonzero diagonals in C will equal the number of nonzero diagonals in A, i.e., C will have almost the same sparsity structure as A. Another possible choice for the circulant matrix C is given in [22]. Here it is shown that it is possible to calculate the truly optimal circulant preconditioner, i.e., the matrix that solves

$$\min_{\tilde{C} \in \mathscr{C}_n} \|I - \tilde{C}^{-1} A\|_F,$$

using $\mathcal{O}(n \log_2 n)$ arithmetic operations. This preconditioner will be a dense circulant matrix, and we have not yet developed this approach.

We now define a number of semicirculant preconditioners for the problem $Bu^{n+1} = b$. We refer to the PDE problem resulting in this system of equations as the original problem. To describe the preconditioners we use the notation $M(\cdot, \cdot, \times)$. Here "$\cdot$" again denotes the type of analytical boundary conditions for a PDE problem, i.e., "$\cdot$" is either "P" ("periodic") or "D" ("Dirichlet"). "$\times$" denotes the "number of averaged levels," which is either "1" or "2." The value "1" corresponds to a semicirculant preconditioner described by $\rho_{j,l}$, $l = -2, \cdots, 2, j = 1, \cdots, m_2$, and the value "2" corresponds to a block-circulant preconditioner described by ρ_l, $l = -2, \cdots, 2$.

Now form a matrix $\tilde{B}$ using the difference approximation of the original problem, but imposing boundary conditions given by $(\cdot, \cdot)$, i.e., not necessarily the same type of boundary conditions as for the original problem. This is equivalent to determining a set of matrix elements $\tilde{v}_{i,j,l}$, $i = 1, \cdots, m_1$, $j = 1, \cdots, m_2$, $l = -2, \cdots, 2$, forming the matrix $\tilde{B}$. Most of the elements $\tilde{v}_{i,j,l}$ will be equal to the elements $v_{i,j,l}$ in the matrix B, but since the boundary conditions may be altered a small number of elements in each block may be different. The matrix $M(\cdot, \cdot, 1)$ is then formed using

$$\rho_{j,l} = \frac{1}{m_1} \sum_{i=1}^{m_1} \tilde{v}_{i,j,l}, \quad j = 1, \cdots, m_2, \quad l = -2, \cdots, 2, \tag{2.9}$$

and the block-circulant matrix $M(\cdot, \cdot, 2)$ is formed using

$$\rho_l = \frac{1}{m_2 m_1} \sum_{j=1}^{m_2} \sum_{i=1}^{m_1} \tilde{\nu}_{i,j,l}, \qquad l = 2, \cdots, 2. \tag{2.10}$$

2.3. Problems with constant coefficients. If a differential operator with constant coefficients is discretized on a uniform grid, the coefficient matrix will be Toeplitz, ignoring the effect of boundary conditions. Suppose that a difference approximation with constant coefficients is given. Then, if the boundary conditions are periodic in one or more space directions, the coefficient matrix will have one or more circulant levels. In some cases, this implies that a specific semicirculant preconditioner M^{-1} will be the exact inverse to B. For a problem in which the coefficients in the difference approximation of $\mathscr{P}$ given by (1.2) are independent of x_1, i.e.,

$$\nu_{i,j,l} = \nu_{j,l}, \quad j = 1, \cdots, m_2, \quad l = -2, \cdots, 2,$$

the following relations are valid:

$$M^{-1}(P, P, 1)B(P, P) = I_{(n_c m_1 m_2)}, \tag{2.11}$$

$$M^{-1}(P, D, 1)B(P, D) = I_{(n_c m_1 m_2)}. \tag{2.12}$$

If the coefficients are also independent of x_2, i.e.,

$$\nu_{i,j,l} = \nu_l, \qquad l = -2, \cdots, 2,$$

we have

$$M(P, P, 1) \equiv M(P, P, 2). \tag{2.13}$$

However, note that $M(\cdot, \cdot, \times)$ *always* has at least one circulant level and therefore $M^{-1}(\cdot, \cdot, \times)B(D, D) \neq I_{(n_c m_1 m_2)}$.

3. Model problems. In this section we define the model problems that we use for the study of the spectra in § 4 and the numerical experiments presented in § 5. The first model problem arises from the implicit time discretization of a two-dimensional scalar hyperbolic equation, and the second model problem arises from the implicit part of a semi-implicit time discretization of the two-dimensional Navier–Stokes or Euler equations.

3.1. Model problem 1. We consider the following two-dimensional (D, D)-problem [17], which we refer to as $MP1(D, D)$:

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x_1} + \frac{\partial u}{\partial x_2} = g, \tag{3.1}$$

and

$$0 < x_i \leqq a_i, \quad i = 1, 2, \quad t > 0,$$

$$u(x_1, 0, t) = f(x_1 - vt), \tag{3.2}$$

$$u(0, x_2, t) = f(x_2 - vt), \tag{3.3}$$

$$u(x_1, x_2, 0) = f(x_1 + x_2), \tag{3.4}$$

$$g = (2 - v)f', \qquad v > 0. \tag{3.5}$$

Here a_1 and a_2 are positive constants. f is a scalar function with derivative f'. The analytical solution of this problem is a wave of shape f moving in the direction $x_1 = x_2$

with velocity v:

$$u = f(x_1 + x_2 - vt). \tag{3.6}$$

Later we consider the corresponding $MP1\,(P, D)$-problem, which arises if boundary condition (3.2) is changed to

$$u(x_1, 0, t) = u(x_1, 1, t). \tag{3.7}$$

To model nonuniform space meshes or variable coefficients in the PDE, we introduce the following transformation from the computational to the physical domain:

$$x_i = a_i\left(\frac{\tanh(s_i(2\xi_i - 1))}{2\tanh(s_i)} + \frac{1}{2}\right), \quad 0 \leqq \xi_i \leqq 1, \quad i = 1, 2. \tag{3.8}$$

The parameters s_1 and s_2 are positive constants, which control the stretching of the grid. When s_i increases, the distribution of gridpoints in the ith space direction of the physical domain becomes more dense near the boundaries. In the limit $s_1, s_2 \to 0$ the grid is unstretched and the transformation defined by (3.8) is the identity transformation. Using (3.8), (3.1) transforms to:

$$\frac{\partial u}{\partial t} + \sigma_1(\xi_1)\frac{\partial u}{\partial \xi_1} + \sigma_2(\xi_2)\frac{\partial u}{\partial \xi_2} = g, \tag{3.9}$$

where

$$\sigma_i(\xi_i) \equiv a_i^{-1}\cosh^2(s_i(2\xi_i - 1))\frac{\tanh(s_i)}{s_i}, \qquad i = 1, 2. \tag{3.10}$$

The time discretization of (3.9) is performed using the second-order accurate trapezoidal rule. The space discretization is performed on a grid that is uniform in the computational domain with $(m_1 + 1) \times (m_2 + 1)$ gridpoints. The space-steps are given by $h_i = 1/m_i$, $i = 1, 2$. Let $u_{i,j}$ denote the approximative solution at the point (ih_1, jh_2), $i = 0, \cdots, m_1, j = 0, \cdots, m_2$. Observe that $u_{i,0}$ for $i = 0, \cdots, m_1$ and $u_{0,j}$ for $j = 0, \cdots, m_2$ are given directly by the boundary conditions (3.2) and (3.3) for the (D, D)-problem, and by (3.7) and (3.3) for the (P, D)-problem. This implies that we have to solve for $m_1 \times m_2$ unknowns in each time-step.

The spatial derivatives in the PDE are approximated using centered differences in the interior of the domain. For the numerical boundary condition required at the outflow boundaries we use one-sided differences. Using this type of numerical boundary conditions locally reduces the order of accuracy from two to one. This does not affect the order of accuracy inside the domain [15].

We now define the following quantities:

$$\begin{aligned} &\nu_{i,1} \equiv \kappa_1\sigma_{i,1} \equiv \frac{k}{h_1}\sigma_{i,1}, \qquad i = 1, \cdots, m_1, \\ &\nu_{j,2} \equiv \kappa_2\sigma_{j,2} \equiv \frac{k}{h_2}\sigma_{j,2}, \qquad j = 1, \cdots, m_2; \\ &u^n = (u^n_{1,1}u^n_{2,1}\cdots u^n_{m_1,1}u^n_{1,2}\cdots u^n_{m_1,m_2})^T. \end{aligned} \tag{3.11}$$

Introducing the discretizations in (3.9) yields the following system of equations for the unknowns at time level $n + 1$:

$$B(\cdot, D)u^{n+1} = b. \tag{3.12}$$

Here b contains known quantities and $B(\cdot, D)$ is a member of the B-class:

$$B(\cdot,D)=\begin{pmatrix} B^{(1)} & C_1 & & & & & \\ A_2 & B^{(1)} & C_2 & & & & \\ & A_3 & \ddots & \ddots & & & \\ & & \ddots & \ddots & C_{m_2-2} & & \\ & & & A_{m_2-1} & B^{(1)} & C_{m_2-1} & \\ & & & & 2A_{m_2} & B^{(1)}+2C_{m_2} \end{pmatrix}, \tag{3.13}$$

where

$$B^{(1)}(D,D)=\begin{pmatrix} 4 & \nu_{1,1} & & & & & \\ -\nu_{2,1} & 4 & \nu_{2,1} & & & & \\ & -\nu_{3,1} & \ddots & \ddots & & & \\ & & \ddots & \ddots & \nu_{m_1-2,1} & & \\ & & & -\nu_{m_1-1,1} & 4 & \nu_{m_1-1,1} & \\ & & & & -2\nu_{m_1,1} & 4+2\nu_{m_1,1} \end{pmatrix},$$

$$B^{(1)}(P,D)=\begin{pmatrix} 4 & \nu_{1,1} & & & & & -\nu_{1,1} \\ -\nu_{2,1} & 4 & \nu_{2,1} & & & & \\ & -\nu_{3,1} & \ddots & \ddots & & & \\ & & \ddots & \ddots & \nu_{m_1-2,1} & & \\ & & & -\nu_{m_1-1,1} & 4 & \nu_{m_1-1,1} & \\ \nu_{m_1,1} & & & & -\nu_{m_1,1} & 4 & \end{pmatrix},$$

and

$$A_j=I_{(m_1)}\otimes(-\nu_{j,2}), \qquad C_j=I_{(m_1)}\otimes\nu_{j,2}.$$

We consider $MP1(D, D)$ on the unit square using a uniform grid with m^2 gridpoints, where $m = m_1 = m_2$. We use a time-step that is κ times larger than the space-step, which implies that $\nu_{i,1} = \nu_{j,2} = \kappa_1 = \kappa_2 = \kappa$, $i, j = 1, \cdots, m$. We refer to this problem as $P1(m, \kappa)$.

We also consider $MP1(D, D)$ on a nonuniform grid with $a_1 = 50$ and $a_2 = 1$, and the stretching given by $s_1 = 0$ and $s_2 = 2$. The grid has m^2 gridpoints, where $m = m_1 = m_2$. This problem will be referred to as $P2(m, \kappa)$. We use this type of grid to model a situation where the solution changes rapidly in the x_2-direction near the boundaries $x_2 = 0$ and $x_2 = 1$. The solution is assumed to change slowly with x_1, and hence the grid is coarse in this direction.

3.2. Model problem 2. Model problem 2 arises in the solution of the isentropic two-dimensional Navier–Stokes and Euler equations using a semi-implicit time-marching scheme of a type described in [14]. The specific problem studied here is the calculation of the flow in a driven cavity using Navier–Stokes equations [16]. The original equations are symmetrized by a transformation presented in [14], yielding

$$u_t+(\varepsilon^{-1}\mathscr{P}_0+\mathscr{P}_1(u))u=\mathscr{P}_2(u)u. \tag{3.14}$$

Here ε is the Mach number and u is a vector with three components. The first component is a "transformed pressure," and the second and third components are the velocities in the x_1- and x_2-directions. $\mathscr{P}_2$ contains the nonlinear second-order terms in the Navier–Stokes equations corresponding to viscosity. $\mathscr{P}_1$ is a nonlinear first-order operator and $\varepsilon^{-1}\mathscr{P}_0$ is a linear first-order operator with constant coefficients. The time-marching method exploits the implicit Euler backwards scheme for $\varepsilon^{-1}\mathscr{P}_0$ and the explicit leap-frog scheme for $\mathscr{P}_1$ and $\mathscr{P}_2$. The discretization of spatial derivatives is performed on a uniform grid

with $m \times m$ internal gridpoints and space-step h using centered differences. Introducing the discretizations, the analytical and numerical boundary conditions, we arrive at a system of equations with $3m_1m_2$ unknowns. The coefficient matrix is a member of the B-class given by:

$$B(D,D)=\begin{pmatrix} B^{(1)}-AZ_1 & C-AZ_2 & & & \\ A & B^{(1)} & C & & \\ & \ddots & \ddots & \ddots & \\ & & A & B^{(1)} & C \\ & & & A-CZ_2 & B^{(1)}-CZ_1 \end{pmatrix}, \tag{3.15}$$

where

$$Z_1 = I_{(m_1)} \otimes (-2\zeta), \quad Z_2 = I_{(m_1)} \otimes \zeta, \quad A = I_{(m_1)} \otimes (-\nu_2),$$

$$B^{(1)}=\begin{pmatrix} I_{(3)}-2\nu_1\zeta & \nu_1+\nu_1\zeta & & & \\ -\nu_1 & I_{(3)} & \nu_1 & & \\ & \ddots & \ddots & \ddots & \\ & & -\nu_1 & I_{(3)} & \nu_1 \\ & & & -\nu_1-\nu_1\zeta & I_{(3)}+2\nu_1\zeta \end{pmatrix},$$

$$C = I_{(m_1)} \otimes \nu_2,$$

and

$$\nu_1=\begin{pmatrix} 0 & \kappa & 0 \\ \kappa & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}, \quad \nu_2=\begin{pmatrix} 0 & 0 & \kappa \\ 0 & 0 & 0 \\ \kappa & 0 & 0 \end{pmatrix}, \quad \zeta=\begin{pmatrix} 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},$$

and finally

$$\kappa = \frac{k}{\varepsilon h}.$$

We refer to this problem as $P3(m, \kappa)$.

4. Spectra of the preconditioned coefficient matrices. Here we study spectra of the preconditioned coefficient matrices. First we review some theoretical results reported for Toeplitz matrices preconditioned by circulant matrices. Then we present an analysis of the eigenvalues of the matrix $M^{-1}(P, P, 2)B(P, D)$ for $MP1$. Finally, we discuss the qualitative properties of different matrices $M^{-1}(\cdot, \cdot, \times)B(D, D)$ for $MP1$, by studying plots of numerically calculated spectra.

Circulant preconditioners for symmetric positive definite Toeplitz systems have been studied in, e.g., [6] and [5], and are found to be effective. For large systems it is proved that the number of CG-iterations required to solve the system is independent of the number of unknowns. The preconditioned coefficient matrix has only a finite number of eigenvalues outside an arbitrary small interval of the real axis enclosing 1. The authors of [6] also report on numerical experiments, where the same type of favorable convergence properties are observed for nondiagonally dominant symmetric Toeplitz matrices. In [5] and [17] the use of circulant preconditioners for near-Toeplitz systems of equations is proposed. Numerical experiments in [17] show that the number of iterations required are independent of the number of unknowns even for strongly nondiagonally dominant unsymmetric near-block-Toeplitz systems, where the coefficient matrix $B(D, D)$ is preconditioned with $M(D, D, 2)$ or $M(P, P, 2)$.

4.1. A theoretical study. In this section we calculate the eigenvalues of $M^{-1}(P, P, 2)B(P, D)$, i.e., we study $MP1$ with periodic boundary conditions in the x_1-direction and analytical Dirichlet boundary conditions in the x_2-direction. We have not yet performed the corresponding analysis for the more interesting matrix $M^{-1}(P, D, 1)B(D, D)$, since it is considerably more complicated. We show that the resulting matrix has $2m_1$ eigenvalues different from one, and present explicit formulas for the calculation of them. Furthermore, we show that if m_2 is large, the eigenvalues lie on two curve segments in the complex plane. These curves are independent of m_1 and m_2 and well separated from zero.

The matrix B considered here is a nearly block-Toeplitz tridiagonal matrix with m_2 blocks:

$$B=\begin{pmatrix} B^{(1)} & C & & & & & \\ A & B^{(1)} & C & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & \ddots & \ddots & \ddots & & \\ & & & A & B^{(1)} & C & \\ & & & & A-D_1 & B^{(1)}+D_0 \end{pmatrix}, \tag{4.1}$$

where $A = I_{(m_1)} \otimes b_{-2}$, $C = I_{(m_1)} \otimes b_2$, $D_1 = I_{(m_1)} \otimes d_1$, $D_0 = I_{(m_1)} \otimes d_0$, and $B^{(1)} = \operatorname{circ}_{(m_1)}(b_0, b_1, 0, \cdots, 0, b_{-1})$. The preconditioner $M(P, P, 2)$ is block-circulant:

$$M = M(P,P,2) = \operatorname{circ}_{(m_2)}(B^{(1)}, C, 0, \cdots, 0, A). \tag{4.2}$$

We define the error matrix E, given by

$$B = M + E. \tag{4.3}$$

This matrix has only four nonzero blocks:

$$E=\begin{pmatrix} 0 & 0 & \cdots & 0 & 0 & -A \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ & & & & & \\ 0 & 0 & \cdots & 0 & 0 & 0 \\ -C & 0 & \cdots & 0 & -D_1 & D_0 \end{pmatrix}. \tag{4.4}$$

From (4.3) we obtain

$$M^{-1}B = M^{-1}(M+E) = I_{(m_2 m_1)} + M^{-1}E. \tag{4.5}$$

It is possible to calculate the eigenvalues of $M^{-1}E$. We will not show the rather tedious algebra here. In the calculation we use the spectral decomposition of M^{-1}, observe that $M^{-1}E$ has only three nonzero block columns, and solve the characteristic equation

$$\det(\mu I_{(m_2 m_1)} - M^{-1}E) = 0. \tag{4.6}$$

The result of this calculation is that $M^{-1}E$ has $2m_1$ nonzero eigenvalues μ_k given by:

$$\mu_k = \frac{\alpha_k}{2} + \sqrt{\frac{\alpha_k^2}{4} - \beta_k}, \qquad k = 1, \cdots, m_1, \tag{4.7a}$$

$$\mu_{k+m_1} = \frac{\alpha_k}{2} - \sqrt{\frac{\alpha_k^2}{4} - \beta_k}, \qquad k = 1, \cdots, m_1, \tag{4.7b}$$

where

$$\alpha_k = -(d_1 + b_{-2})\tilde{\lambda}_{k,-1} + d_0\tilde{\lambda}_{k,0} - b_2\tilde{\lambda}_{k,1},$$

$$\beta_k = d_1 b_{-2}(\tilde{\lambda}_{k,-1}^2 - \tilde{\lambda}_{k,-2}\tilde{\lambda}_{k,0}) + b_2 b_{-2}(\tilde{\lambda}_{k,1}\tilde{\lambda}_{k,-1} - \tilde{\lambda}_{k,0}^2),$$

and

$$\tilde{\lambda}_{k,j} = \sum_{l=1}^{m_2} \frac{m_2^{-1}\omega_{m_2}^{j(l-1)}}{\lambda_k + \omega_{m_2}^{(l-1)}b_2 + \omega_{m_2}^{-(l-1)}b_{-2}},$$

$$\lambda_k = b_0 + \omega_{m_1}^{(k-1)}b_1 + \omega_{m_1}^{-(k-1)}b_{-1}.$$

We now study the discretization of $MP1(P, D)$ described in § 3.1, yielding $b_0 = 4$, $b_1 = -b_{-1} = \kappa_1$, and $b_2 = -b_{-2} = d_0/2 = d_1 = \kappa_2$. The quantities $\tilde{\lambda}_{k,j}$ can be regarded as Riemann sums. We introduce the function F:

$$F(\theta) = \frac{\exp(ij\theta)}{\lambda_k + \kappa_2 \exp(i\theta) - \kappa_2 \exp(-i\theta)}. \tag{4.8}$$

Exploiting the trapezoidal rule and the periodicity of F, $F(2\pi) = F(0)$, yields

$$\tilde{\lambda}_{k,j} = \frac{1}{2\pi}\int_0^{2\pi} F(\theta)\, d\theta + \mathcal{O}(m_2^{-2}). \tag{4.9}$$

By a change of variables, $z = \exp(i\theta)$, we obtain

$$\frac{1}{2\pi}\int_0^{2\pi} F(\theta)\, d\theta = \frac{1}{2\pi i}\int_C f(z)\, dz, \tag{4.10}$$

where C is the positively oriented unit circle. We now use the residue theorem to calculate the integrals (4.10). Summarizing the results, we find that in the limit $m_2 \to \infty$, all eigenvalues lie on the curve segments $\mu_1(\theta)$ and $\mu_2(\theta)$, where

$$\mu_1(\theta) = \hat{\lambda}_0 + \frac{\hat{\lambda}_{-1}}{2} + \sqrt{\frac{\hat{\lambda}_{-1}^2}{4} + \hat{\lambda}_0\hat{\lambda}_{-1} - \hat{\lambda}_0\hat{\lambda}_{-2}}, \tag{4.11a}$$

$$\mu_2(\theta) = \hat{\lambda}_0 + \frac{\hat{\lambda}_{-1}}{2} - \sqrt{\frac{\hat{\lambda}_{-1}^2}{4} + \hat{\lambda}_0\hat{\lambda}_{-1} - \hat{\lambda}_0\hat{\lambda}_{-2}}, \tag{4.11b}$$

and

$$\hat{\lambda}_j = \frac{(\zeta - \sqrt{\zeta^2+1})^{-j}}{2\sqrt{\zeta^2+1}}, \qquad j = -2, -1, 0,$$

where

$$\zeta = \frac{2}{\kappa_2} - i\frac{\kappa_1}{\kappa_2}\sin(\theta), \qquad 0 \leqq \theta \leqq 2\pi.$$

Observe that $\mu_1(\theta)$ and $\mu_2(\theta)$ are independent of m_1. Fig. 1 shows the spectrum of $I + M^{-1}E$, calculated from formulas (4.11a)–(4.11b), for a problem with $\kappa_1 = \kappa_2 = 100$ and $m_1 = 128$. Six eigenvalues are specially marked by stars, and given a capital letter label. The curve μ_1 is given by the segments $A^+ \leftrightarrow C^+$ and $A^- \leftrightarrow C^-$. The discontinuity is caused by a change of branch of the square root in (4.11a). The curve μ_2 is given by

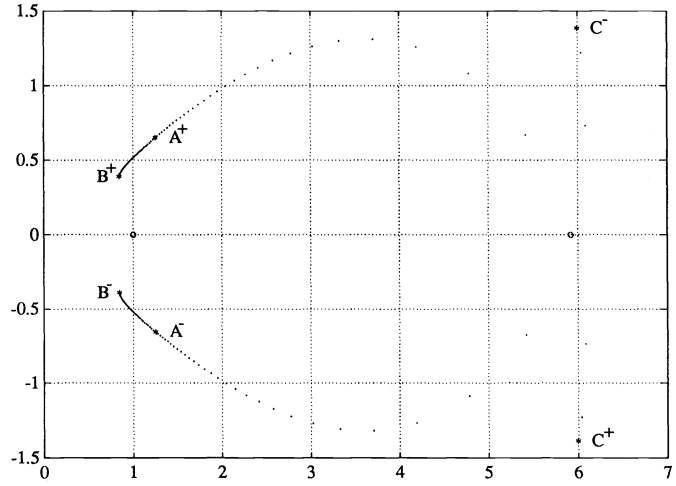


FIG. 1. *The spectrum for* $I + M^{-1}E$. $\kappa_1 = \kappa_2 = 100$, $m_1 = 128$, *and* $m_2 \to \infty$.

the segments $A^+ \leftrightarrow B^+$ and $A^- \leftrightarrow B^-$. The curve indicated by A^+, B^+, and C^+ is formed in the following way:

Curve segment	θ^{start}	θ^{stop}	Curve
μ_1	0	$\pi/2$	$A^+ \mapsto C^+$
μ_1	$\pi/2$	π	$C^+ \mapsto A^+$
μ_2	π	$3\pi/2$	$A^+ \mapsto B^+$
μ_2	$3\pi/2$	2π	$B^+ \mapsto A^+$

We now put $\kappa_1 = \kappa_2 = \kappa$. Asymptotically for large values of κ we have:

$$A^{\pm} = \tfrac{1}{4}(5 \pm i\sqrt{7}), \tag{4.12}$$

$$B^{\pm} = \tfrac{1}{5}(4 \pm 2i), \tag{4.13}$$

$$C^{\pm} = \frac{\sqrt{2}}{8}(3 \pm i)\sqrt{\kappa}. \tag{4.14}$$

Further algebra shows that, for large values of κ, the *spectral quotient* $\gamma \equiv \max_\theta |\mu_{1,2}| / \min_\theta |\mu_{1,2}|$ is given by

$$\gamma = \tfrac{5}{4}\sqrt{\kappa}. \tag{4.15}$$

According to the discussion in § 1.4, these results imply that if the GCR(∞)-iteration is used to solve a problem with coefficient matrix $M^{-1}B$, the asymptotic convergence factor is independent of the size of the problem. It is also clear that when κ grows, the asymptotic convergence factor does not grow faster than $\sqrt{\kappa}$.

Observe that half of the eigenvalues lie on the curve segments $A^{\pm} \leftrightarrow B^{\pm}$, which are independent of κ for large values of this parameter. The eigenvalues are also quite densely clustered at the points $B^{\pm}$. As mentioned in § 1.4, this favors the use of CG-like iterative methods.

4.2. Numerically calculated spectra. Here we show plots of the spectra of the matrices $M^{-1}(\cdot, \cdot, \times)B$ for the problems $P1(16, 100)$ and $P2(16, 100)$. The plots are

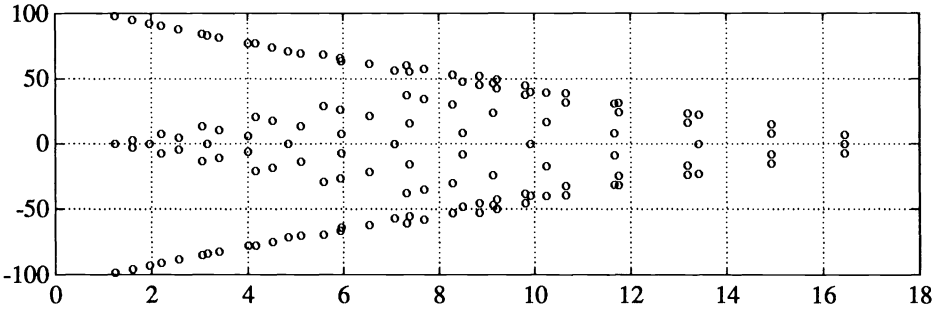


FIG. 2. *The spectrum of B for* $P1(16, 100)$.

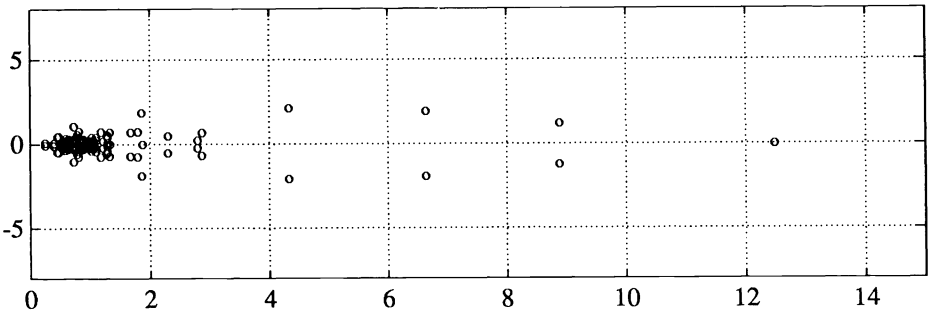


FIG. 3. *The spectrum of* $M^{-1}(D, D, 1)B$ for $P1(16, 100)$.

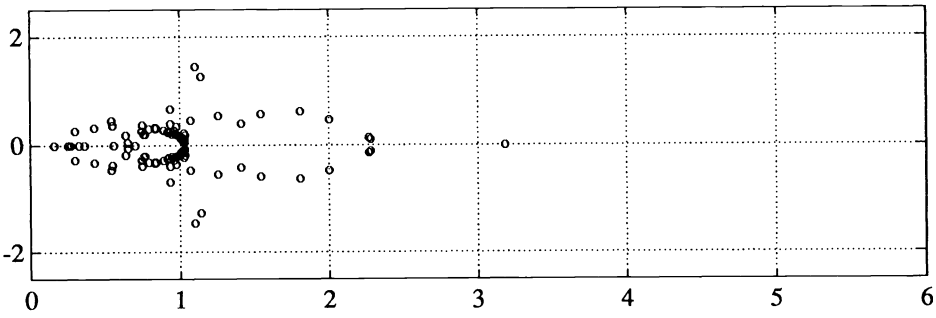


FIG. 4. *The spectrum of* $M^{-1}(D, D, 2)B$ *for* $P1(16, 100)$.

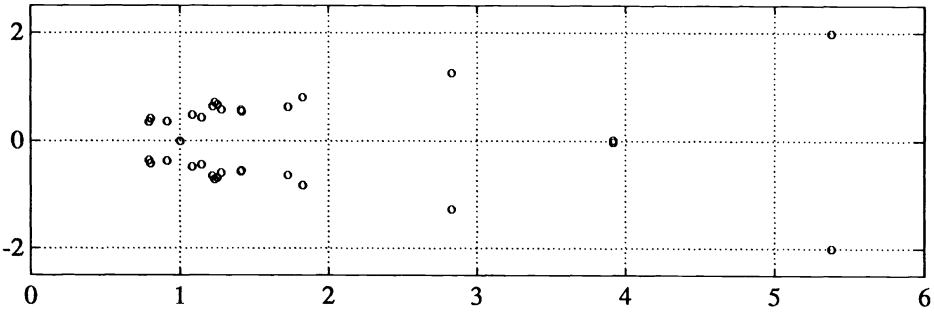


FIG. 5. *The spectrum of* $M^{-1}(P, D, 1)B$ *for* $P1(16, 100)$.

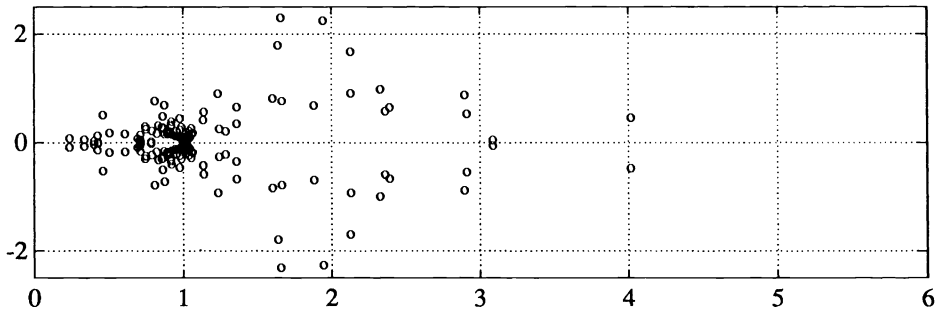


FIG. 6. *The spectrum of* $M^{-1}(P, D, 2)B$ *for* $P1(16, 100)$.

generated by the software tool PRO-MATLAB. Observe that the problems studied are small compared to realistic problems.

We first study $P1(16, 100)$. Note that the resulting coefficient matrix B is strongly nondiagonally dominant. The only disturbances of the block-Toeplitz structure in this matrix are caused by the numerical boundary conditions. This specific problem is also studied in [17]. There spectra of the preconditioned coefficient matrix using different incomplete LU factorizations and block-incomplete LU factorizations are shown.

Figure 2 shows the spectrum of the matrix $\frac{1}{4}B$, and Figs. 3–7 show the spectra of $M^{-1}(\cdot, \cdot, \times)B$ for different semicirculant preconditioners. The eigenvalues are labeled by μ_i, $i = 1, \cdots, 256$. In Table 1 some characteristics regarding the spectra are listed. The entries in the row labeled # ($\mu_i \neq 1$) represent the number of eigenvalues that are not numerically equal to one. $\gamma = \max_i |\mu_i| / \min_i |\mu_i|$. Observe that for $P1(m, \kappa)$, $M(P, P, 1) \equiv M(P, P, 2)$ according to (2.13).

Most of the spectra shown in Figs. 3–7 are similar in the sense that many of the eigenvalues are clustered near two points and the rest of the eigenvalues are scattered in a relatively small region in the complex plane. Further numerical calculations of the spectra for different problem sizes confirm that these regions are independent of the number of unknowns. Intuitively, the spectrum of $M^{-1}(P, D, 1)B$ shown in Fig. 5 is the most favorable for a CG-like iterative method, since this spectrum has only $32 = 2m_1$ eigenvalues different from one. This assumption is also verified by the numerical experiments presented in § 5.1.

Observe that, for all preconditioned coefficient matrices shown above, all eigenvalues have positive real parts. Also observe that the spectra are well separated from zero. Further numerical experiments show that for large values of κ, the smallest and largest eigenvalues

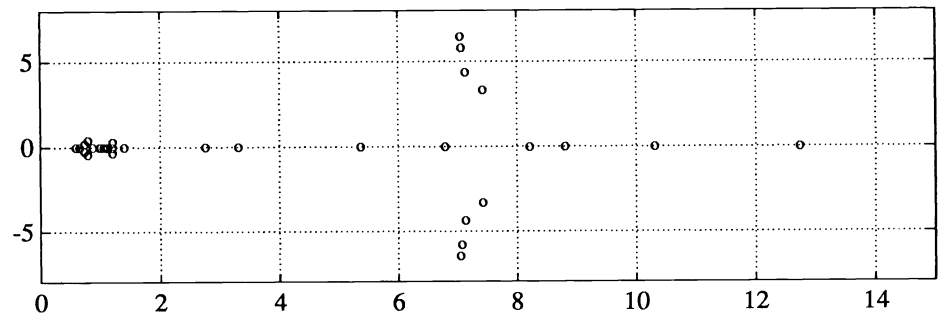


FIG. 7. *The spectrum of* $M^{-1}(P, P, 2)B$ *for* $P1(16, 100)$.

TABLE 1
Properties of the spectra for $P1(16, 100)$.

Preconditioner	None	$(D, D, 1)$	$(D, D, 2)$	$(P, D, 1)$	$(P, D, 2)$	$(P, P, 2)$
$\max_i \Re e(\mu_i)$	16.4	12.5	3.18	5.28	4.02	12.7
$\min_i \Re e(\mu_i)$	1.24	.246	.158	.790	.237	.592
$\max_i \Im m(\mu_i)$	98.1	2.10	1.45	1.99	2.30	6.47
$\max_i \lvert\mu_i\rvert$	98.1	12.5	3.18	5.73	4.04	12.7
$\min_i \lvert\mu_i\rvert$	1.24	.259	.158	.867	.250	.592
γ	79.2	48.2	20.2	6.61	16.2	21.5
$\#(\mu_i \neq 1)$	256	256	256	32	256	60

of $M^{-1}(P, D, 1)B$ behave in the same way as the corresponding eigenvalues of the problem theoretically analyzed in § 4.1.

We now study the spectrum of the preconditioned coefficient matrix for the problem $P2(16, 100)$. Fig. 8 shows the spectrum of the matrix $\frac{1}{4}B$ for this problem and Figs. 9 and 10 show the spectra of $M^{-1}(\cdot, \cdot, \times)B$ for different semicirculant preconditioners. In Table 2, properties of these spectra are listed.

It is clear from Figs. 9 and 10 that it is vital to retain the information of the variable coefficients in the x_2-direction. It is also probable that the preconditioner $M(P, D, 1)$ is a really good preconditioner for this problem. Numerical experiments presented in § 5.2 confirm this assumption.

5. Numerical experiments. In this section we present the results of some numerical experiments, where we solve the system of equations arising from the problems $P1(m, \kappa)$, $P2(m, \kappa)$, and $P3(m, \kappa)$. We use the GCR(l) iterative method combined with different semicirculant preconditioners. The implementations are made in FORTRAN 77, using 64 bit arithmetics for both the iterative methods and the preconditioners. In § 6 we discuss the implementations and algorithms in more detail. The results presented in this section are obtained using an Alliant FX/8 and a Cray X-MP/48. The calculations are performed by fixing an a priori defined solution, and multiplying the coefficient matrix B with this solution vector to determine the right-hand side. As an initial approximation, we use $u^{n+1} = 0$. In the stopping criterion (1.5), we use $\varepsilon = 10^{-6}$. Below, the term efficiency refers to the number of arithmetic operations required.

5.1. Problem 1. In [17] the solution of $P1(m, \kappa)$ is also considered. There the preconditioners $M(D, D, 2)$ and $M(P, P, 2)$ are compared to standard incomplete LU-

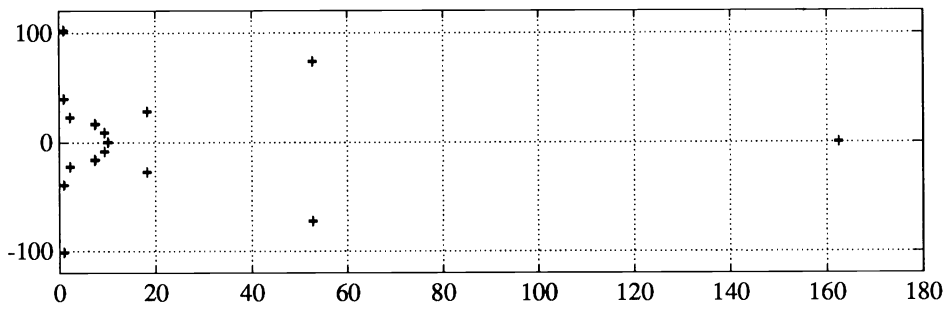


FIG. 8. *The spectrum of B for* $P2(16, 100)$.

TABLE 2
Properties of the spectra for $P2(16, 100)$.

Preconditioner	None	$(P, D, 1)$	$(P, D, 2)$	$(P, P, 2)$
$\max_i \Re e(\mu_i)$	163	1.05	4.02	24.6
$\min_i \Re e(\mu_i)$	1.00	.999	.207	.248
$\max_i \Im m(\mu_i)$	103	.0289	.771	16.8
$\max_i \lvert\mu_i\rvert$	163	1.05	4.09	25.9
$\min_i \lvert\mu_i\rvert$	10.2	.999	.211	.248
γ	15.9	1.05	19.4	104
$\#(\mu_i \neq 1)$	256	32	256	256

factorization preconditioners of different types [24], [10], [8]. Numerical experiments show that for large values of κ ($\kappa \geqq 100$) and large problems, the block-circulant preconditioners are considerably more efficient than the ILU and block-ILU factorizations. It is empirically observed that the number of iterations required using the block-circulant preconditioners is independent of the problem size. For very large values of κ ($\kappa \sim 1000$), the ILU and block-ILU preconditioners are not usable.

In Figs. 11–13 below we show the number of iterations required to solve $P1(m, \kappa)$, using the GCR(5)-iteration and semicirculant preconditioners. Note that in all figures Log means Log_2.

Observe that the largest problems solved are quite massive, containing $256 \times 256 = 65{,}536$ unknowns. The difference in efficiency between the different preconditioners decreases when the problem size increases. This is natural, since the influence of the different boundary conditions on the parameters $\rho_{j,l}$ decreases when the problems get larger.

For all three examined values of κ the preconditioner $M(P, D, 1)$ is the most efficient, at least for large problems. Figs. 11–13 indicate that the number of iterations required using this preconditioner for large problems approaches ~ 14 for $\kappa = 10$, ~ 28 for $\kappa = 100$, and ~ 60 for $\kappa = 1000$. These results indicate that the asymptotic convergence factor for the GCR(5)-iteration does not grow very quickly with κ. In fact, the growth is slower than $\mathcal{O}(\sqrt{\kappa})$, which was the result of the theoretical analysis in § 4.1 and the empirical observations in § 4.2. The reason for this is probably that the eigenvalues of $M^{-1}(P, D, 1)B$ are clustered.

5.2. Problem 2. We here study the solution of $P2(m, 100)$ presented in § 3.1. Figure 14 shows the number of GCR(5)-iterations required to solve this problem ex-

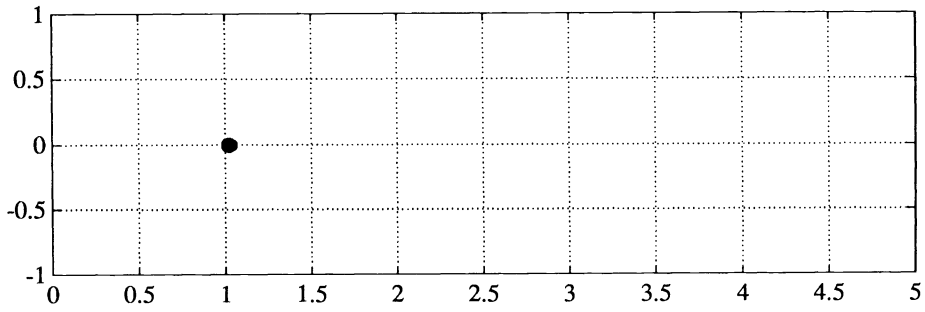


FIG. 9. *The spectrum of* $M^{-1}(P, D, 1)B$ *for* $P2(16, 100)$.

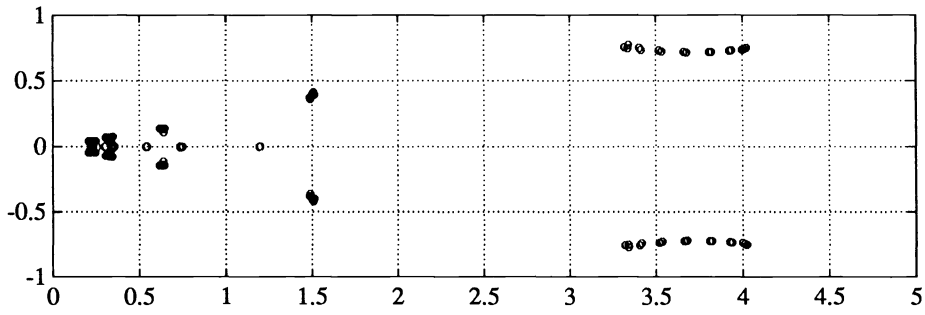


FIG. 10. *The spectrum of* $M^{-1}(P, D, 2)B$ *for* $P2(16, 100)$.

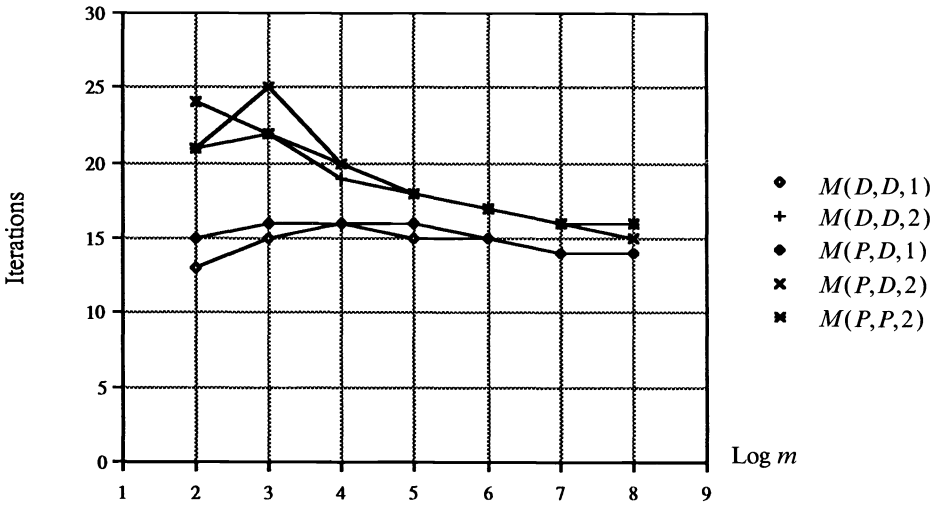


FIG. 11. *The number of iterations required for* $P1(m, 10)$.

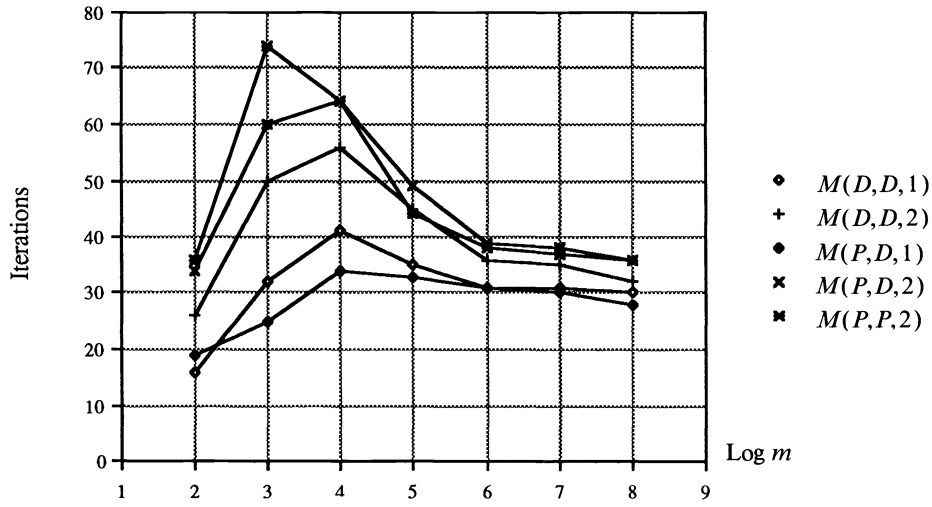


FIG. 12. *The number of iterations required for* $P1(m, 100)$.

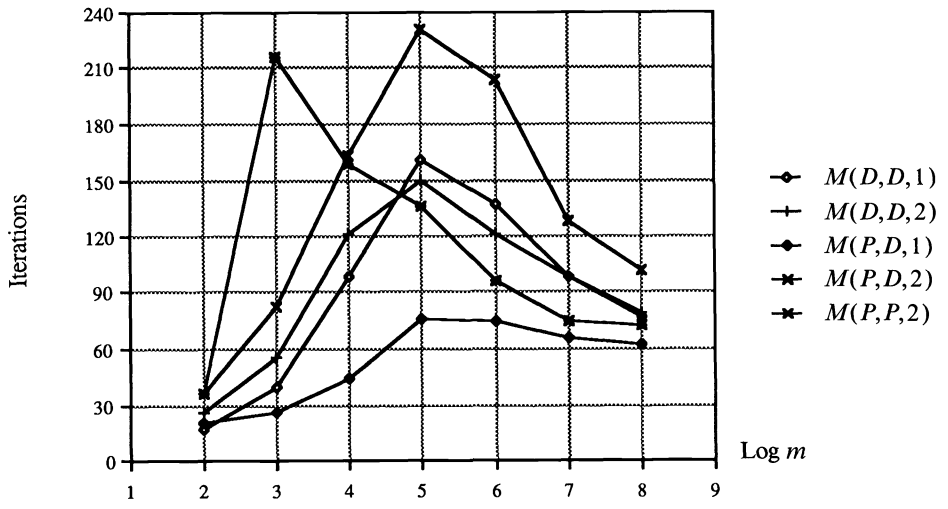


FIG. 13. *The number of iterations required for* $P1(m, 1000)$.

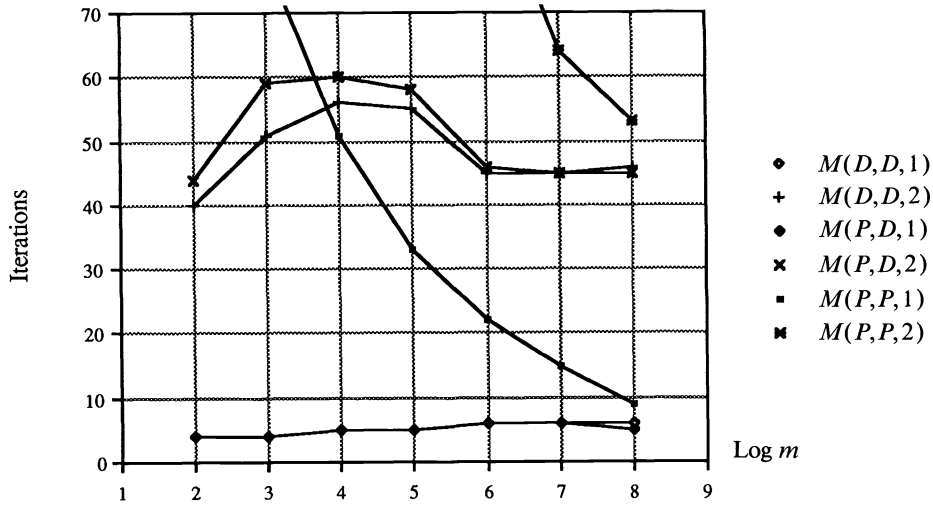


FIG. 14. *The number of iterations required for* $P2(m, 100)$.

ploiting different semicirculant preconditioners. As anticipated in § 4.2, the preconditioner $M(P, D, 1)$ is efficient for this problem, but the preconditioner $M(D, D, 1)$ is almost as good. The $M(P, D, 1)$- and $M(D, D, 1)$-preconditioners are probably quite useful for problems that are "nice" in one space direction and "nasty" in the other direction.

5.3. Problem 3. Finally, we study $P3(m, 100)$ presented in § 3.2. To solve this problem, we have used the GCR(15)-iteration combined with the preconditioners $M(P, D, \times)$. The GCR(15)-iteration performs slightly more efficiently than the GCR(5)-iteration for this problem. In Fig. 15 the results for the problem $P3(m, 100)$ are shown.

Observe that the total number of unknowns is given by $3m^2$ since $n_c = 3$. Once again, the preconditioner $M(P, D, 1)$ performs better than the preconditioner

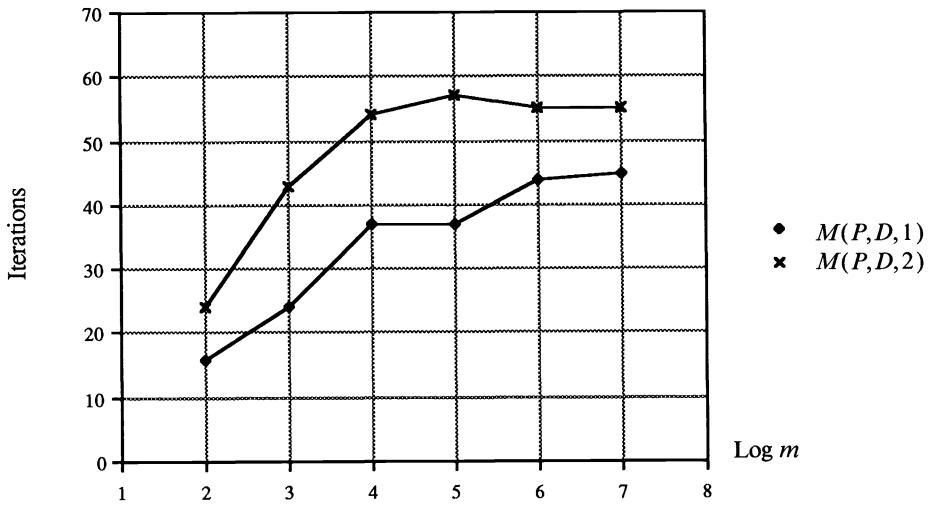


FIG. 15. *The number of iterations required for* $P3(m, 100)$.

$M(P, D, 2)$. Also for this problem, arising from the solution of a system of PDEs, the numerical results indicate that for large problems the number of iterations required is independent of problem size. Further numerical experiments show that the increase with κ in the number of iterations required is moderate (slower than $\sqrt{\kappa}$).

6. Arithmetic complexities and memory requirements. In this section we discuss the algorithms used in the CG-like iterations and the semicirculant preconditioner solves, as well as their scalar arithmetic complexities and memory requirements. Observe that the number of arithmetic operations given only includes the highest-order terms. Also observe that a complex addition is considered as two arithmetic operations, a complex multiplication as 6 arithmetic operations, and a complex division as 11 arithmetic operations.

The computational work required for one iteration using the GCR(l) iterative method is given by $a_{it} = 3l + 13 + a_m + a_{ps}$. In this context, the quantity a_m denotes the number of arithmetic operations per unknown needed to perform a matrix-vector product Bp, and a_{ps} denotes the number of arithmetic operations per unknown required for one preconditioner solve $w = M^{-1}y$. The arithmetic work for GCR(l) is an average value taken over a complete cycle of l iterations. The memory requirement for the GCR(l)-iteration is $m_{it} = 7 + 2(l + 1) + m_m + m_{ps}$. Here m_m denotes the amount of memory required to hold the coefficient matrix B, and m_{ps} denotes the memory requirement for the preconditioner. The memory requirement is normalized by the number of unknowns. For a matrix in the B-class $a_m = 10n_c$ and $m_m = 5n_c$. Note that the values of a_{it} and m_{it} are also valid if the system of equations arises from a three-dimensional problem.

To perform a semicirculant preconditioner solve, a number of FFTs have to be executed. An FFT of length m used in the preconditioner solve requires $5m \log_2 m$ arithmetic operations. This implies that for a preconditioner solve, $10 \log_2 m_1$ arithmetic operations per unknown are required for the FFTs.

Furthermore, a number of possibly periodic block-tridiagonal systems of equations with complex entries has to be solved. These systems are periodic if the preconditioner is of the $M(\cdot, P, \times)$ or $M(\cdot, \cdot, 2)$ type. In our implementation we assume that the block-tridiagonal systems of equations are periodic, and we solve them by block-cyclic

TABLE 3
Arithmetic complexities and memory requirements for the semicirculant preconditioners.

	Two-dimensional preconditioner	Three-dimensional preconditioner
a_{pf}	$60n_c^2 + 22n_c$	$60n_c^2 + 28n_c$
a_{ps}	$10 \log_2 m_1 + 48n_c$	$10(\log_2 m_1 + \log_2 m_2) + 48n_c$
m_{ps}	$13n_c + 12$	$13n_c + 12$

reduction. This algorithm is equivalent to block-Gaussian elimination without pivoting. Note that some of the block-tridiagonal systems may be ill conditioned to the same extent as the original system of equations. However, the block-tridiagonal systems can probably be solved without pivoting, since these systems are much smaller than the original system and the solution of them is part of a preconditioner algorithm. For preconditioners of $M(\cdot, D, 1)$ type, the block-tridiagonal systems are nonperiodic, and a solver using ordinary banded Gaussian elimination with pivoting would be simple to implement.

The block-cyclic reduction is divided into a factorizing and a backsubstitution phase. In the factorizing phase, the diagonal blocks of the systems of equations are inverted, and the elimination phase of the cyclic reduction is performed. This phase has to be performed only when the parameters of the preconditioner are changed. In the backsubstitution phase, the information from the factorization is used to solve the block-tridiagonal systems of equations for new right-hand sides. In Table 3 above, we summarize the different contributions to the total work required for the semicirculant preconditioners. a_{pf} denotes the work required to form the preconditioner and to factorize the small block-tridiagonal systems. In this table we also give counts for the corresponding three-dimensional semicirculant preconditioner.

The efficiency of the semicirculant preconditioners is investigated by solving $P1(m, \kappa)$ using GCR(5). The total arithmetic work without preconditioner is compared

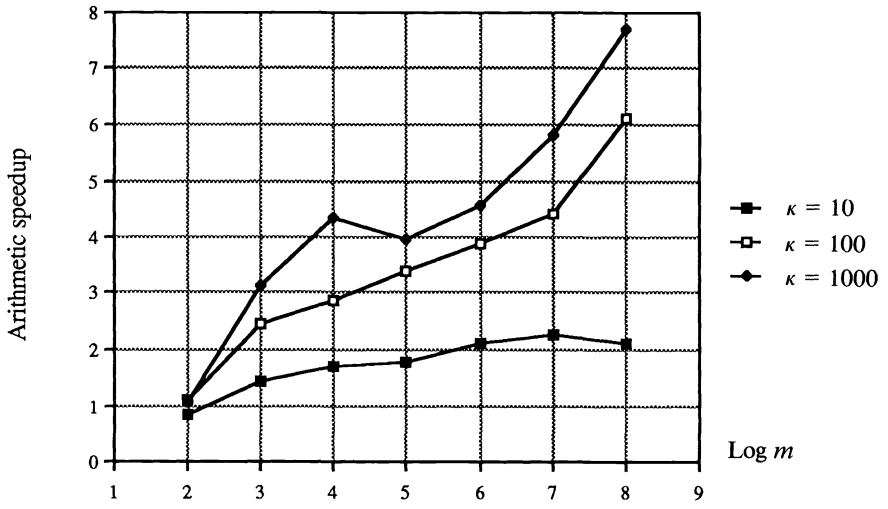


FIG. 16. *The arithmetic speedup for* GCR(5) + $M(P, D, 1)$ *compared to* GCR(5).

to the work when a semicirculant preconditioner is employed. The total arithmetic work is given by

$$W = a_{it} \cdot N_{it},$$

where N_{it} is the number of iterations.

In Fig. 16, $W(\text{GCR}(5))/W(\text{GCR}(5) + M(P, D, 1))$ is given for different problem sizes and different values of κ. Note that the number of arithmetic operations required to solve the problems is always smaller when using the preconditioner, except for one 4×4-problem. For large values of κ and relevant problem sizes, the speedup is significant.

7. Conclusions. We have studied two model problems, one two-dimensional scalar hyperbolic PDE employing the trapezoidal rule in time, and Navier–Stokes equations for the flow in a two-dimensional driven cavity exploiting a semi-implicit time discretization. The systems of equations arising from the space discretizations are solved using CG-like iterative methods combined with semicirculant preconditioners.

A study of the spectrum of the preconditioned coefficient matrix, using one of the semicirculant preconditioners for a scalar model problem with periodic boundary conditions in one space direction, shows that the number of eigenvalues different from one is $\mathcal{O}(\sqrt{n})$, where n is the total number of unknowns. Furthermore, these eigenvalues lie on two curve segments in the complex plane, which are independent of n. Hence, the asymptotic convergence rate for minimizing iterative methods like GCR is independent of n. Also, we show that the number of iterations does not increase faster than $\mathcal{O}(\sqrt{\kappa})$, where κ is the quotient between the time- and space-steps. Since the number of arithmetic operations required to perform a semicirculant preconditioner solve is $\mathcal{O}(n \log_2 n)$, the total number of arithmetic operations required to solve the system of equations will also be $\mathcal{O}(n \log_2 n)$.

For some problems using semicirculant preconditioners, spectra of the preconditioned coeffcient matrices were calculated. A representative CG-like iterative method and a number of semicirculant preconditioners were also implemented and numerically tested. These results show that for the model problems studied, the number of iterations required to solve the systems of equations approaches a constant number when the size of the problems is increased. Also, the increase in the number of iterations required when κ grows is smaller than $\sqrt{\kappa}$. Hence, the favorable convergence properties theoretically proved for the simple scalar model problem seem to be valid also for more complicated problems. The assumption that, for some of the semicirculant preconditioners, variable coefficients in the difference approximation are allowed in one space dimension without any decrease in performance was empirically verified.

REFERENCES

[1] O. AXELSSON, *Conjugate gradient type methods for unsymmetric and inconsistent systems of linear equations*, Linear Algebra Appl., 29 (1980), pp. 1–16.

[2] ———, *A generalized conjugate gradient, least square method*, Numer. Math., 51 (1987), pp. 209–227.

[3] ———, *A restarted version of a generalized preconditioned conjugate gradient method*, Report No. 8710, University of Nijmegen, Nijmegen, the Netherlands, 1987.

[4] R. M. BEAM AND R. F. WARMING, *On the construction and application of implicit factored schemes for conservation laws*, SIAM-AMS Proc., 11 (1978), pp. 85–129.

[5] R. H. CHAN, *The spectrum of a family of circulant preconditioned Toeplitz systems*, SIAM J. Numer. Anal., 26 (1989), pp. 503–506.

[6] R. H. CHAN AND G. STRANG, *Toeplitz equations by conjugate gradients with circulant preconditioner*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 104–119.

[7] T. F. CHAN, *An optimal circulant preconditioner for Toeplitz systems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 766–771.

[8] P. CONCUS, G. H. GOLUB, AND G. MEURANT, *Block preconditioning for the conjugate gradient method*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 220–252.

[9] P. J. DAVIS, *Circulant Matrices*, John Wiley and Sons, New York, 1979.

[10] T. DUPONT, R. P. KENDALL, AND H. H. RACHFORD, *An approximate factorization procedure for solving self-adjoint elliptic difference equations*, SIAM J. Numer. Anal., 5 (1968), pp. 559–573.

[11] S. C. EISENSTAT, H. C. ELMAN, AND M. H. SCHULTZ, *Variational iterative methods for nonsymmetric systems of linear equations*, SIAM J. Numer. Anal., 20 (1983), pp. 345–357.

[12] H. C. ELMAN, *Iterative methods for large, sparse, nonsymmetric systems of linear equations*, Ph.D. thesis and Report No. RR-229, Department of Computer Science, Yale University, New Haven, CT, 1982.

[13] R. FLETCHER, *Conjugate gradient methods for indefinite systems*, Lecture Notes in Mathematics 506, Springer-Verlag, Berlin, Heidelberg, New York, 1976.

[14] J. GUERRA AND B. GUSTAFSSON, *A semi-implicit method for hyperbolic problems with different time-scales*, SIAM J. Numer. Anal., 23 (1986), pp. 734–749.

[15] B. GUSTAFSSON, *The convergence rate for difference approximations to general mixed initial boundary value problems*, SIAM J. Numer. Anal., 18 (1981), pp. 179–190.

[16] B. GUSTAFSSON AND H. STOOR, *The Navier–Stokes equations for almost incompressible flow*, Report No. 122, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, 1989.

[17] S. HOLMGREN AND K. OTTO, *Iterative solution methods and preconditioners for nonsymmetric, non-diagonally dominant block-tridiagonal systems of equations*, Report No. 123, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, 1989.

[18] ———, *Iterative solution methods and preconditioners for block-tridiagonal systems of equations*, Report No. 127, Department of Scientific Computing, Uppsala University, Uppsala, Sweden, 1990.

[19] Y. SAAD, *Krylov subspace methods for solving large unsymmetric linear systems*, Math. Comp., 37 (1981), pp. 105–126.

[20] Y. SAAD AND M. H. SCHULTZ, GMRES: *A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[21] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[22] E. E. TYRTYSHNIKOV, *Optimal and superoptimal circulant preconditioners*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 459–473.

[23] P. K. W. VINSOME, ORTHOMIN, *an iterative method for solving sparse sets of simultaneous linear equations*, in Proc. Fourth Symposium on Reservoir Simulation, Society of Petroleum Engineers of AMIE, 1976.

[24] H. A. VAN DER VORST, *Preconditioning by incomplete decompositions*, Ph.D. thesis, Rijksuniversiteit Utrecht, Utrecht, the Netherlands, 1982.

[25] D. M. YOUNG AND K. C. JEA, *Generalized conjugate-gradient acceleration of nonsymmetrizable iterative methods*, Linear Algebra Appl., 34 (1980), pp. 159–194.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 887–904, July 1992

012

A PRECONDITIONED ITERATIVE METHOD FOR SADDLEPOINT PROBLEMS*

TORGEIR RUSTEN† AND RAGNAR WINTHER‡

Abstract. A preconditioned iterative method for indefinite linear systems corresponding to certain saddlepoint problems is suggested. The block structure of the systems is utilized in order to design effective preconditioners, while the governing iterative solver is a standard minimum residual method. The method is applied to systems derived from discretizations of the Stokes problem and mixed formulations of second-order elliptic problems.

Key words. minimum residual methods, indefinite systems, preconditioning

AMS(MOS) subject classifications. 65F10, 65N20

1. Introduction.

The purpose of this paper is to present a preconditioned iterative method for linear systems corresponding to certain saddlepoint problems. These systems arise frequently from discretizations of partial differential equations. Important examples are the system of discrete equations that results from the approximation of the equations of elasticity and the Stokes problem (cf. [15], [16], [19]). Other examples result from mixed formulations of second-order elliptic problems (cf. [12], [25], [30]). Since these discrete systems can become very large, it is of great interest to develop efficient iterative methods for such problems.

We shall consider linear systems of the form

$$\begin{aligned} Mx + By &= b, \\ B^T x &= c, \end{aligned} \tag{1.1}$$

where $M \in \mathbb{R}^{n\times n}$ is symmetric and positive definite, $B \in \mathbb{R}^{n\times m}$ with $m \leq n$ and B has full rank; i.e., $\operatorname{rank}(B) = m$.

It is well known that systems of the form (1.1) have a unique solution $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$. Furthermore, the system is equivalent to the constrained minimization problem:

$$\min \tfrac{1}{2}\langle Mx, x\rangle - \langle b, x\rangle \quad \text{subject to} \quad B^T x = c,$$

where $\langle \cdot, \cdot \rangle$ denotes the Euclidean inner product of $\mathbb{R}^n$. Alternatively, (1.1) is equivalent to the saddlepoint problem for the Lagrangian

$$\mathcal{L}(x, y) = \tfrac{1}{2}\langle Mx, x\rangle - \langle b, x\rangle + \langle y, B^T x - c\rangle.$$

For a review of these simple results for the system (1.1) we refer, for example, to [18].

The coefficient matrix A of system (1.1), given by

$$A = \begin{pmatrix} M & B \\ B^T & 0 \end{pmatrix}, \tag{1.2}$$

* Received by the editors April 5, 1990; accepted for publication (in revised form) August 12, 1991.

† Institute of Mathematics and Its Applications, University of Minnesota, 514 Vincent Hall, 206 Church Street S. E., Minneapolis, Minnesota 55455 (rusten@ima.umn.edu).

‡ Department of Informatics, University of Oslo, P. O. Box 1080 Blindern, N-0316 Oslo 3, Norway (ragnar@ifi.uio.no). This author's research was supported by VISTA, a research cooperation between the Norwegian Academy of Science and Letters and Den norske stats oljeselskap a.s. (STATOIL).

is symmetric, nonsingular, and indefinite. The preconditioned iterative method presented in this paper will be based on the minimum residual method. Application of this method to symmetric, indefinite systems is discussed by Paige and Saunders [23] and Chandra [10] (cf. Stoer [27]). They develop an algorithm that can be applied to any symmetric, nonsingular system. The main contribution of this paper is a discussion on how systems of the form (1.1) can be preconditioned in order to increase the convergence rate of the algorithm. Hence, the block structure of the system (1.1) will only be utilized in order to derive efficient preconditioners for the minimum residual method.

The convergence rate of the minimum residual method depends on the location of eigenvalues of the coefficient matrix. Since the coefficient matrix of (1.1) is indefinite, the spectrum of this matrix contains both positive and negative eigenvalues. We will show that an upper bound for the convergence rate of the minimum residual method applied to (1.1) can be related to the condition numbers of the matrices M and B. In addition, this convergence rate will depend on the relative scaling between M and B. The purpose of a preconditioner for the system (1.1) is therefore to improve the conditioning of the matrices M and B, while the relative scaling between them is kept roughly fixed.

Most iterative methods that have been suggested for systems of the form (1.1) are modifications of Uzawa's method (cf. [18]). This algorithm requires that a system of the form

$$(M + \rho BB^T)z = d, \tag{1.3}$$

where z is the unknown and $\rho \geq 0$ is a parameter, must be solved for each iteration. Hence, if an iterative method is used to solve the system (1.3), we obtain a solver with an inner and outer iteration. The problem with this approach is that in order to ensure convergence of the outer iteration, it might be necessary to iterate the inner iteration until it converges within computer precision, thus making the overall process somewhat costly. Also, certain iteration parameters have to be selected in order to speed up the convergence of Uzawa-type algorithms. In contrast, the minimum residual method does not require the choice of any parameters and leads to an "optimally" converging scheme. For the application of Uzawa-type methods to systems of the form (1.1) we refer to [11], [18], and [24]. Discussions of iterative methods for linear systems using an inner and outer iteration can also be found in [2], [4], [13], and [20]. Iterative methods for the system (1.1), based on positive definite reformulations of the system, are suggested by Bramble and Pasciak in [5] and [6].

In §2 we will discuss some elementary spectral properties of matrices of the form (1.2). These properties are related to the convergence rate of the minimum residual method in §3. In §4 we propose a general preconditioning strategy for systems of the form (1.1), while §5 reports on some numerical examples. We discuss the application of a fast Poisson solver and an incomplete Cholesky factorization to the discrete systems arising from mixed finite element methods for some second-order elliptic problems and to discrete Stokes problems.

2. The saddlepoint problem. The coefficient matrix A of (1.1) is symmetric, nonsingular, and indefinite. Thus the spectrum of A, $\Lambda(A)$, contains both positive and negative eigenvalues. In this section we establish a simple result relating $\Lambda(A)$ to properties of the matrices M and B. In the next section these properties will be related to the convergence rate of the minimum residual method.

Let M have eigenvalues $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n > 0$. Then

$$\mu_n|x|^2 \leq \langle Mx, x\rangle \leq \mu_1|x|^2 \tag{2.1}$$

for all $x \in \mathbb{R}^n$. Here $|\cdot|$ denotes the Euclidean norm on $\mathbb{R}^n$. Since $\operatorname{rank}(B) = m$, B has singular values $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m > 0$, and

$$\sigma_m|y| \leq |By| \leq \sigma_1|y| \tag{2.2}$$

for all $y \in \mathbb{R}^m$. We also have

$$\sigma_m|x| \leq |B^T x| \leq \sigma_1|x| \tag{2.3}$$

for all $x \in N(B^T)^\perp$. Note that this implies that the right inequality is valid for all $x \in \mathbb{R}^n$.

The following simple result relates the spectrum of the coefficient matrix A to these properties of M and B.

LEMMA 2.1. *Let $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_n > 0$ be the eigenvalues of M, $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m > 0$ the singular values of B, and denote by $\Lambda(A)$ the spectrum of A. Then*

$$\Lambda(A) \subset I \equiv I^- \cup I^+,$$

where

$$I^- = \left[\frac{1}{2}(\mu_n - \sqrt{\mu_n^2 + 4\sigma_1^2}), \frac{1}{2}(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_m^2})\right]$$

and

$$I^+ = \left[\mu_n, \frac{1}{2}(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2})\right].$$

Proof. The bounds are established by energy arguments. Let $\lambda \in \Lambda(A)$ and let (x, y) be the corresponding eigenvector, i.e.,

$$Mx + By = \lambda x, \tag{2.4}$$

$$B^T x = \lambda y. \tag{2.5}$$

Note that if $x \equiv 0$ then $y \equiv 0$ by (2.4), since $\operatorname{rank}(B) = m$. This is impossible when (x, y) is an eigenvector. Hence $x \not\equiv 0$. The proof is now divided into two parts. First we bound the positive eigenvalues, then the negative eigenvalues.

Let λ be a positive eigenvalue. By combining the inner product of (2.4) with x and the inner product of (2.5) with y we get

$$\langle Mx, x\rangle + \lambda|y|^2 = \lambda|x|^2.$$

Next, we use (2.1) to obtain

$$(\mu_n - \lambda)|x|^2 \leq -\lambda|y|^2 \leq 0,$$

and since $|x|$ is positive, we have $\lambda \geq \mu_n$. To derive an upper bound, we use (2.5) and the inner product of (2.4) with x to obtain

$$\langle Mx, x\rangle + \frac{1}{\lambda}|B^T x|^2 = \lambda|x|^2.$$

By (2.1) and (2.3) this implies the inequality

$$(\lambda^2 - \mu_1\lambda - \sigma_1^2)|x|^2 \leq 0.$$

Since $|x|$ is positive, we therefore conclude that $\lambda \leq \frac{1}{2}(\mu_1 + \sqrt{\mu_1^2 + 4\sigma_1^2})$. Hence, we have derived the desired bounds for the positive eigenvalues.

Consider next a negative eigenvalue λ. The derivation of the lower bound of I^- is similar to the derivation of the upper bound of I^+. Finally, we derive the upper bound for the negative eigenvalues. To do this, let $x = v + w$, where $v \in N(B^T)^\perp$ and $w \in N(B^T)$. By (2.5) and the inner product of (2.4) with v we get

$$\langle Mw, v\rangle = \lambda|v|^2 - \langle Mv, v\rangle - \frac{1}{\lambda}|B^T v|^2.$$

Applying (2.1) and (2.3) we obtain

$$\langle Mw, v\rangle \geq (\lambda - \mu_1 - \sigma_m^2/\lambda)|v|^2. \tag{2.6}$$

To proceed we must bound $\langle Mw, v\rangle$. This is achieved by forming the inner product of (2.4) with w and using (2.1). Since $w \in N(B^T)$ and $\lambda - \mu_n$ is negative we get

$$\langle Mv, w\rangle \leq (\lambda - \mu_n)|w|^2 \leq 0.$$

Together with (2.6) and the symmetry of M we obtain

$$0 \leq (\lambda^2 - \mu_1\lambda - \sigma_m^2)|v|^2.$$

If $v \equiv 0$, (2.5) implies $y \equiv 0$ and (2.4) reduces to $Mw = \lambda w$. Since λ is negative this is a contradiction, and it follows that $\lambda^2 - \mu_1\lambda - \sigma_m^2 \geq 0$. Hence,

$$\lambda \leq \frac{1}{2}(\mu_1 - \sqrt{\mu_1^2 + 4\sigma_m^2}).$$ □

We remark that the estimates given in Lemma 2.1 are indeed sharp. In order to see this, consider a 3×3 matrix of the form

$$\begin{pmatrix} \mu & 0 & 0 \\ 0 & \mu & \sigma \\ 0 & \sigma & 0 \end{pmatrix}, \tag{2.7}$$

where $\mu > 0$. This matrix is of the form (1.2) and has the eigenvalues

$$\mu, \quad \tfrac{1}{2}\left(\mu + \sqrt{\mu^2 + 4\sigma^2}\right) \quad \text{and} \quad \tfrac{1}{2}\left(\mu - \sqrt{\mu^2 + 4\sigma^2}\right). \tag{2.8}$$

We also observe that a block diagonal matrix, with diagonal blocks of the form (2.7), can be transformed to a matrix with the desired structure (1.2) by renumbering the variables. Hence, by varying μ and σ along the diagonal, we conclude from (2.8) that the bounds of Lemma 2.1 are sharp.

The result above will be used in §3 when we discuss the convergence properties of the minimum residual method applied to the system (1.1). We observe that the two intervals I^- and I^+ are located on each side of the origin. Let $\kappa(M)$ and $\kappa(B)$ be the spectral condition numbers of M and B, respectively; i.e.,

$$\kappa(M) = \mu_1/\mu_n \quad \text{and} \quad \kappa(B) = \sigma_1/\sigma_m.$$

From the expressions above we can easily deduce that if $\kappa(M)$, $\kappa(B) \to 1$, then the interval I^- degenerates to a single point and the length of the interval I^+ tends to a value $\leq \sigma_1$.

3. The minimum residual method. The purpose of this section is to discuss the minimum residual method for symmetric, indefinite linear systems. In particular, we shall discuss the application of this method to systems of the form (1.1).

Consider first a system of the form

$$Ax^* = b, \tag{3.1}$$

where A is a square, symmetric, and nonsingular matrix, while x^* and b are vectors. If A also is positive definite then (3.1) can be solved by the conjugate gradient method. However, the minimum residual method can also be used when A is indefinite.

The minimum residual method for indefinite systems has been studied by Paige and Saunders [23] and Chandra [10] (cf. Stoer [27]). We first give a brief derivation of the method.

Let $V_k = \text{span}\{b, Ab, \cdots, A^{k-1}b\}$. The approximation $x_k \in V_k$ of x^* is determined by

$$|Ax_k - b|^2 = \inf_{x \in V_k} |Ax - b|^2. \tag{3.2}$$

This solution $x_k \in V_k$ is unique and satisfies the system

$$\langle Ax_k, Av \rangle = \langle b, Av \rangle \quad \forall v \in V_k. \tag{3.3}$$

The iterative method for the calculation of the vectors $\{x_k\}$, originally proposed in [23], was derived from the Lanczos process stated in terms of the Euclidean inner product. An alternative algorithm can be derived from the Lanczos process using the "energy" inner product $\langle \cdot, \cdot \rangle_A$ given by $\langle p, q \rangle_A = \langle Ap, Aq \rangle$. In this case a sequence of vectors $\{p_j\}$ is generated, such that $\{p_1, p_2, \cdots, p_k\}$ is an orthonormal basis for V_k with respect to the inner product $\langle \cdot, \cdot \rangle_A$.

Using this notation, it follows from (3.3) that

$$\langle x_{k+1} - x_k, v \rangle_A = 0 \quad \forall v \in V_k.$$

Hence

$$x_{k+1} = x_k + \alpha_{k+1} p_{k+1}$$

for a suitable scalar α_{k+1}. Furthermore, from the equation

$$\langle A(x_k + \alpha_{k+1} p_{k+1}), Ap_{k+1} \rangle = \langle b, Ap_{k+1} \rangle$$

and the properties of the vectors $\{p_k\}$, we derive the expression

$$\alpha_{k+1} = \frac{\langle r_k, Ap_{k+1} \rangle}{\langle p_{k+1}, p_{k+1} \rangle_A} = \langle r_k, Ap_{k+1} \rangle.$$

Here r_k is the residual vector defined by $r_k = b - Ax_k$. The complete iteration for the vectors $\{p_k\}$ and $\{x_k\}$ is therefore given by the algorithm below.

ALGORITHM 3.1. *The minimum residual method:*

$p_0 := x_0 := 0$
$\beta_1 p_1 := x_1 := b$
$k := 1$
while $r_k \not\equiv 0$ do
  $\beta_{k+1} p_{k+1} := Ap_k - \langle Ap_k, p_k \rangle_A p_k - \langle Ap_k, p_{k-1} \rangle_A p_{k-1}$
  $x_{k+1} := x_k + \langle r_k, Ap_{k+1} \rangle p_{k+1}$
  $k := k + 1$

The constants $\beta_k > 0$ are chosen such that $\langle p_k, p_k \rangle_A = 1$. For exact arithmetic this is always possible, since $r_k \neq 0$ implies that $Ap_k \notin V_k$; i.e., $\beta_{k+1} p_{k+1} \neq 0$.

From the optimality property (3.2) we can derive a convergence estimate for the method. Recall that $r_k = b - Ax_k$ satisfies

$$|r_k|^2 = |b - Ax_k|^2 \leq |b - Ax|^2 \quad \forall x \in V_k. \tag{3.4}$$

Using spectral theory we easily obtain from (3.4) that

$$|r_k| \leq \delta_k(\Lambda(A))|b|, \tag{3.5}$$

where

$$\delta_k(\Lambda) = \inf_{p \in P_k} \sup_{\lambda \in \Lambda} |p(\lambda)|$$

and $P_k = \{p \mid p \text{ is a polynomial of degree less than or equal } k \text{ and } p(0) = 1\}$.

It is clear from (3.5) that we can estimate the convergence rate of the minimum residual method by estimating the quantity $\delta_k(\Lambda)$. If all the eigenvalues are positive, we can embed Λ in a positive interval $I = [a, b]$. In this case it is possible to derive an analytical expression for $p^* \in P_k$ such that

$$\sup_{\lambda \in I} |p^*(\lambda)| = \inf_{p \in P_k} \sup_{\lambda \in I} |p(\lambda)|.$$

It is well known that p^* can be expressed in terms of Chebyshev polynomials, and that this leads to the estimate (cf., e.g., [21])

$$\delta_k(\Lambda) \leq \delta_k(I) \leq 2e^{-2k/\sqrt{\kappa(A)}},$$

where $\kappa(A)$ denotes the spectral condition number of A. Furthermore, if the eigenvalues of A have a relatively uniform and dense distribution in I, this convergence estimate is rather sharp.

Consider the case when

$$\Lambda(A) \subset I \equiv I^- \cup I^+,$$

where the intervals I^- and I^+ are located to the left and to the right of the origin, respectively. In this case it seems to be harder to obtain an analytical expression of the optimal polynomial p^* (cf. the discussion given in Atlestam [1]). However, we observe that even if it is hard to derive sharp estimates for the quantity $\delta_k(I)$ in the case of two intervals, some general properties can easily be seen. First of all, by a proper scaling of the polynomials in P_k, we obtain that $\delta_k(\alpha I) = \delta_k(I)$ for any $\alpha \neq 0$. Also, if $\tilde{I} \subset I$ then obviously $\delta_k(\tilde{I}) \leq \delta_k(I)$. The estimate (3.5) therefore clearly indicates that the minimum residual method converges faster when we shrink the intervals. This phenomenon is illustrated by the following numerical example.

Example 3.1. We fix $n = 600$ and let A be the diagonal matrix with 300 diagonal elements equally distributed in each of the intervals $[-a, -1]$ and $[2, d]$. All the elements on the right-hand side b are taken to be 1. The linear system $Ax = b$ is solved by the minimum residual method for different values of a and d. The iterations are terminated when $\|r_k\| \leq 10^{-4}$. The number of iterations required is given in Table 3.1.

TABLE 3.1
Number of iterations for Example 3.1.

$a\backslash d$	50	10	5
20	194	89	62
4	92	42	30
1	28	12	9

Assume that $\Lambda(A) \subset I \equiv I^- \cup I^+$, where I^- and I^+ are given by Lemma 2.1. By scaling I as indicated above, such that the lower limit of I^+ is unity, we can assume that I^- and I^+ are given by

$$\begin{aligned} I^- &= \left[\tfrac{1}{2}(1 - \sqrt{1 + 4\rho^2\kappa^2(B)}), \tfrac{1}{2}(\kappa(M) - \sqrt{\kappa^2(M) + 4\rho^2})\right], \\ I^+ &= \left[1, \tfrac{1}{2}(\kappa(M) + \sqrt{\kappa^2(M) + 4\rho^2\kappa^2(B)})\right]. \end{aligned} \tag{3.6}$$

Here $\rho = \rho(B, M)$ denotes the relative scaling between the matrices B and M given by the ratio between the smallest singular value of B and the smallest eigenvalue of M; i.e.,

$$\rho(B, M) = \sigma_m/\mu_n.$$

The expressions for I^- and I^+ given by (3.6) indicates that the convergence rate of the minimum residual method can be estimated by the three parameters $\kappa(B)$, $\kappa(M)$, and $\rho(B, M)$. Note that if $\kappa(B)$ and $\kappa(M)$ are fixed, the upper limit of I^+ will increase as a function of ρ. On the other hand, if $\rho \to 0$ the upper limit of I^- will approach zero. Since the polynomials in P_k must satisfy $p(0) = 1$, this is obviously not desirable in order to make $\delta_k(I)$ small. The scaling factor $\rho(B, M)$ should therefore neither be too small nor too large; i.e., the matrices B and M should be properly balanced.

However, if the scaling factor $\rho(B, M)$ is fixed, it is easy to see that a reduction in one of the condition numbers $\kappa(B)$ or $\kappa(M)$ will lead to a proper shrinking of the intervals I^- and I^+ given by (3.6). As a conclusion of this section we therefore obtain that if the matrices B and M are well conditioned and properly balanced, the minimum residual method applied to the system (1.1) will converge fast.

4. Preconditioning of saddlepoint problems. The purpose of this paper is to discuss the application of the minimum residual method to systems of the form (1.1) arising from the discretization of partial differential equations. From the previous section we know that the convergence rate of this method is in some sense governed by three parameters measuring the conditioning of the matrices M and B and the relative scaling between them; i.e., $\kappa(M)$, $\kappa(B)$, and $\rho(B, M)$. Furthermore, if these three parameters can be bounded independently of the discretization parameter h (cf. §5), the convergence rate is bounded independently of the number of unknowns.

However, for most interesting examples of systems arising from discretization of partial differential equations, at least one of the condition numbers will increase when the discretization is refined. A direct application of the minimum residual method is therefore usually not practical, since the rate of convergence will be too slow. In order to speed up the convergence we will therefore consider a preconditioned version of the method.

From the discussion above we know that the convergence of the method usually will be accelerated if we can properly shrink the two intervals given by (3.6). Furthermore, if the scaling parameter $\rho(B, M)$ is fixed, this can be obtained by reducing the

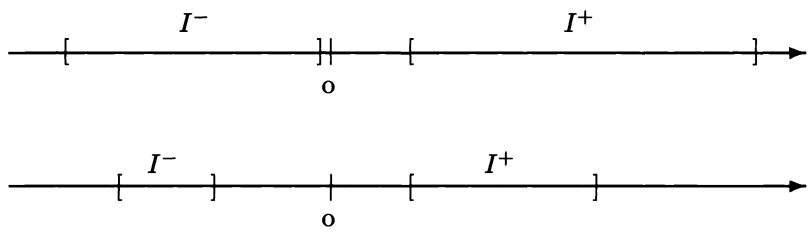


FIG. 4.1. *Effect of reducing $\kappa(M)$ or $\kappa(B)$ when $\rho(B, M)$ is fixed.*

condition number of M or B (cf. Fig. 4.1). The preconditioned system will therefore be designed such that the conditioning of at least one of the matrices is reduced, while the relative scaling between them should be kept approximately fixed.

The preconditioned minimum residual method for systems of the form (1.1) will be presented in general terms in this section. In §5 we apply the method to some model problems arising from the discretization of the Stokes problem and mixed formulations of second-order elliptic equations.

We will use a block diagonal preconditioner for the system (1.1). Such preconditioners for these systems have earlier been suggested by Fortin [17]. The advantage of block diagonal preconditioners is that, through such a setup, the properties of the matrices M and B, which are discretizations of differential operators, can then be easily utilized (cf. §5 below). In general, a linear system is preconditioned if it is replaced by an equivalent system with the property that the convergence rate of the iterative method will be significantly improved. Furthermore, this modification of the system should not significantly increase the necessary work required in each iteration. In addition, the preconditioned system should have the same structure as the original system.

Let $L \in \mathbb{R}^{n\times n}$ and $R \in \mathbb{R}^{m\times m}$ be nonsingular matrices, and let $S = \operatorname{diag}(L, R^T)$. Then the matrix $S^{-1}AS^{-T}$ is given by

$$S^{-1}AS^{-T} = \begin{pmatrix} L^{-1}ML^{-T} & L^{-1}BR^{-1} \\ (L^{-1}BR^{-1})^T & 0 \end{pmatrix}.$$

Hence, the system (1.1) is equivalent to the system

$$\begin{pmatrix} L^{-1}ML^{-T} & L^{-1}BR^{-1} \\ (L^{-1}BR^{-1})^T & 0 \end{pmatrix} \begin{pmatrix} v \\ w \end{pmatrix} = \begin{pmatrix} L^{-1}b \\ R^{-T}c \end{pmatrix}, \tag{4.1}$$

where $v = L^T x$ and $w = Ry$. The system (4.1) has the same structure as (1.1). In fact, the problem (4.1) is equivalent to the constrained minimization problem

$$\min \tfrac{1}{2}\langle L^{-1}ML^{-T}v, v\rangle - \langle L^{-1}b, v\rangle \quad \text{subject to} \quad (L^{-1}BR^{-1})^T v = R^{-T}c.$$

If L and R are such that

$$\kappa(L^{-1}ML^{-T}) \ll \kappa(M) \quad \text{and/or} \quad \kappa(L^{-1}BR^{-1}) \ll \kappa(B),$$

while

$$\rho(L^{-1}BR^{-1}, L^{-1}ML^{-T}) \approx \rho(B, M),$$

then, according to the discussion above, the minimum residual method applied to (4.1) will usually converge faster than if the method is applied to the original system

(1.1). In addition, the matrices L and R should have the property that linear systems with coefficient matrices given by LL^T or R^TR can be solved by a fast solver. The last requirement is necessary since such linear systems have to be solved once in each iteration of the preconditioned minimum residual method.

The choice of the preconditioning matrices L and R is, of course, problem dependent. We shall make only a few general remarks. Since M is symmetric and positive definite and the preconditioning of such systems has been intensively studied during the last years, we often know how to find a matrix L with the required property that $\kappa(L^{-1}ML^{-T})$ is sufficiently small. In such cases, the remaining problem is therefore partly to find a matrix R such that $L^{-1}BR^{-1}$ is well conditioned; i.e., the matrix R should be chosen such that $L^{-1}BR^{-1}$ is approximately a unitary matrix. We observe, furthermore, that this is equivalent to the requirement

$$B^T(LL^T)B \approx R^TR.$$

Hence the matrix R should be chosen such that R^TR is a good preconditioner for the positive definite matrix $B^T(LL^T)B$.

The problem of designing an effective preconditioner of the form discussed above for the system (1.1) is, in some sense, equivalent to the problem of designing effective preconditioners for the symmetric and positive definite matrices M and $B^T(LL^T)^{-1}B$. Furthermore, if the original system is properly scaled, the scaling factor ρ should not be significantly changed by the preconditioning of the system.

5. Numerical examples. In this section we will present results from numerical experiments with the preconditioned minimum residual method. The linear systems used in these experiments result from mixed finite element discretizations of second-order elliptic equations or from the Stokes equations. The matrix M in (1.2) is well conditioned in the first case, but not in the second case.

The preconditioned minimum residual method will be compared with a two-level conjugate gradient method (cf. [29]). Before describing the numerical examples we therefore briefly present this two-level method. The method is closely related to Uzawa's method, except that no acceleration parameter has to be chosen.

Consider the general system (1.1). Since M is positive definite, it follows from the first equation of (1.1) that

$$x = M^{-1}(b - By). \tag{5.1}$$

Substituting this into the second equation of (1.1), we obtain

$$B^TM^{-1}By = B^TM^{-1}b - c. \tag{5.2}$$

Since B has full rank and M is symmetric and positive definite, the matrix $B^TM^{-1}B$ is also symmetric and positive definite. We may therefore solve (5.2) by the conjugate gradient method. Once y is known, x is calculated from (5.1) by the conjugate gradient method.

The conjugate gradient method involves calculating a matrix-vector product in each iteration. Calculating the action of $B^TM^{-1}B$ on a vector involves inverting the matrix M. In some applications it is possible to solve linear systems of the form

$$Mz = d, \tag{5.3}$$

where z is the unknown, by a fast direct method. However, in general this may be too expensive. In practice we may therefore be forced to solve (5.3) by the conjugate gradient method. This results in a two-level procedure, with an inner and an outer iteration, for the solution of the system (5.2).

5.1. Mixed methods for second-order elliptic problems. We consider preconditioning of a mixed finite element discretization of a second-order elliptic partial differential equation with variable coefficients. The resulting linear system has the form (1.1). If the coefficients are mildly varying the matrix M is well conditioned independently of the grid. For this model problem we therefore choose $L = I$. However, the condition number of the matrix B grows with the number of unknowns. Below we will consider two possible choices for generation of the matrix R.

Let $\Omega \subset \mathbb{R}^2$ be a bounded domain with boundary Γ and unit outward normal vector ν. In the numerical examples we consider the following partial differential equation:

$$\begin{aligned} -\nabla \cdot a(x)\nabla p &= f \quad \text{in } \Omega, \\ p &= g, \quad \text{on } \Gamma, \end{aligned} \tag{5.4}$$

where $f \in L^2(\Omega)$ and $g \in L^2(\Gamma)$ are given functions. The 2×2 matrix $a(x)$ is symmetric, and we assume that there exist positive constants α_1 and α_2 such that the inequalities

$$\alpha_1 |\xi|^2 \leq a(x)\xi \cdot \xi \leq \alpha_2 |\xi|^2 \tag{5.5}$$

hold for almost all $x \in \Omega$ and for all $\xi \in \mathbb{R}^2$.

In many applications the variable $u = -a\nabla p$ has physical significance, and may even be of primary interest. For example, if (5.4) is the pressure equation in reservoir simulation, then u is the Darcy velocity. If we solve for the pressure p with a standard finite element method we must differentiate p numerically and multiply the discrete gradient of p by a possibly rough coefficient to obtain u. This may lead to unsatisfactory results. Alternatively, we can use a mixed finite element method, where we solve for u and p simultaneously. The advantage with this approach is that we can approximate u and p with the same order of convergence. However, instead of obtaining a positive-definite system for the discrete solution we obtain a saddlepoint problem of the form (1.1).

In this section $(\cdot, \cdot)$ denotes the inner product on $L^2(\Omega)$ and on $(L^2(\Omega))^2$, with the corresponding norm $\|\cdot\|$, and $\langle \cdot, \cdot \rangle$ is the inner product on $L^2(\Gamma)$. Furthermore, $H(\mathrm{div}, \Omega)$ denotes the set of functions $v \in (L^2(\Omega))^2$ with $\nabla \cdot v \in L^2(\Omega)$, and with norm $\|\cdot\|_{\mathrm{div}}$ given by $\|v\|_{\mathrm{div}}^2 = \|v\|^2 + \|\nabla \cdot v\|^2$. We refer to [19] for more details.

Let $V_h \subset H(\mathrm{div}, \Omega)$ and $Q_h \subset L^2(\Omega)$ be finite-dimensional spaces. The mixed finite element method for the equation (5.4) is given by: Find $(u_h, p_h) \in V_h \times Q_h$ such that

$$\begin{aligned} (a^{-1}u_h, v) - (p_h, \nabla \cdot v) &= -\langle g, v \cdot \nu \rangle \quad \forall v \in V_h, \\ -(q, \nabla \cdot u_h) &= -(f, q) \quad \forall q \in Q_h. \end{aligned} \tag{5.6}$$

Here u_h is an approximation of the variable u introduced above. The discrete problem (5.6) leads to a linear system of the form (1.1). In order to guarantee that the matrix B of (1.1) has full rank, and in order to obtain numerical stability, the spaces must satisfy the so-called inf-sup condition; i.e., there exists a constant β, independent of h, such that

$$\inf_{q \in Q_h} \sup_{v \in V_h} \frac{(q, \nabla \cdot v)}{\|q\| \|v\|_{\mathrm{div}}} \geq \beta > 0.$$

In addition, the spaces V_h and Q_h should be chosen such that

$$\sup_{q \in Q_h} (q, \nabla \cdot v) > 0$$

for all $v \in V_h$ with $\nabla \cdot v \neq 0$. There exist several stable mixed finite elements for the discretization of the system (5.4) (cf. [7], [8], [25]). We have used the rectangular Raviart–Thomas element of order zero (see [25]) to construct the finite-dimensional subspaces V_h and Q_h. We briefly review the construction of these spaces.

For simplicity we shall consider only the case when Ω is the unit square. For a positive integer N let $h = 1/N$ and define the piecewise polynomial space

$$M_s^r = \{w \in C^s([0,1]) \mid w|_{[(i-1)h,ih]} \in P_r, i = 1, 2, \cdots, N\},$$

where $s = -1$ refers to discontinuous functions and P_r is the set of polynomials of degree less than or equal to r. Using this notation, we define

$$Q_h = M_{-1}^0 \otimes M_{-1}^0 \quad \text{and} \quad V_h = [M_0^1 \otimes M_{-1}^0] \times [M_{-1}^0 \otimes M_0^1],$$

where $\otimes$ denotes the tensor product. With these discrete spaces the convergence rate of the finite element approximations is $O(h)$ for both the pressure and the velocity in their associated norms (cf. [25]).

If a nodal basis for V_h is introduced and the unknown velocity coefficients corresponding to the x_1 direction are ordered before those related to the x_2 direction, the matrix M has the block structure

$$M = \begin{pmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{pmatrix}.$$

We have evaluated the integrals in (5.6) by using a certain combination of the midpoint and trapezoid rule (cf. [26]). With a natural numbering of the unknowns the matrices M_{11} and M_{22} are then diagonal. If exact integration had been used they would have been tridiagonal. When a is diagonal the matrix $M_{12} = 0$, and M may be inverted effectively using direct elimination. However, if a is not diagonal then $M_{12} \neq 0$. In this case a direct elimination method applied to M may be too costly (cf. [14]). An iterative method that requires the inversion of M will therefore lead to a two-level iterative method. However, the preconditioned minimum residual method uses the matrix M only in matrix-vector products. The efficiency of this method is therefore independent of the possible diagonal structure of a.

In order to use the preconditioned minimum residual method, we must choose the matrices L and R. Note that the condition number of M is dominated by α_2/α_1 and is independent of h. Thus we set $L = I$ and it remains to choose the matrix R. Since the condition number of the matrix B, which is a discrete gradient operator, is $O(h^{-1})$, it is essential for the efficiency of the method that R be properly chosen. Since $L = I$, it follows that R should be chosen such that $R^T R$ is a preconditioner for the discrete Laplacian $B^T B$. We may therefore use well-known preconditioners for the Poisson equation in order to find a suitable preconditioning matrix R.

Incomplete factorization methods have frequently been used as a preconditioning technique for the Poisson equation. These methods are easy to implement and usually result in a significant reduction of the condition number of the coefficient matrix. The basic idea is to use Cholesky factorization or Gaussian elimination and neglect all the fill-in generated by the elimination process. There are also more effective variants

TABLE 5.1
Number of iterations for Example 5.1.

h (number of iterations)	1/8	1/16	1/32	1/64	1/128
No preconditioner	78	146	281	530	>1000
Incomplete Cholesky preconditioner	30	49	85	165	335
Fast Poisson	23	23	22	22	21

TABLE 5.2
Number of iterations for Example 5.2.

h (number of iterations)	1/8	1/16	1/32	1/64	1/128
No preconditioner	147	318	625	>1000	>1000
Incomplete Cholesky preconditioner	64	90	147	274	576
Fast Poisson	54	61	62	61	61

where more nonzero elements are allowed in the factorization, or where fill-in terms are added to the diagonal (cf. [3], [9]). In our calculations we have used the incomplete Cholesky factorization algorithm in [21] to obtain an approximate factorization R^TR of B^TB. Note that if we set $Q = BR^{-1}$, we have $Q^TQ = R^{-T}B^TBR^{-1} \approx I$. Hence we may say that QR is an approximate "QR-factorization" of B. We emphasize that the preconditioned minimum residual method uses only the matrix R. The matrix Q is not computed.

The incomplete Cholesky factorization can be used for problems on a general domain Ω. However, when Ω is a square we can also apply a fast solver as a preconditioner for B^TB. If we use a regular node numbering, B^TB is equal to the matrix obtained from the five-point discretization of the Poisson equation. In this case we can use a fast Poisson solver (cf. [28]) to solve systems with coefficient matrix B^TB. We may think of this as having an exact factorization R^TR of B^TB. Letting $Q = BR^{-1}$, Q is orthogonal and QR is an exact "QR-factorization" of B. Hence, $\kappa(BR^{-1}) = \kappa(Q) = 1$, and the conditioning of BR^{-1} is best possible. We note again that Q is not computed.

In all the numerical experiments $g = 0$ and $f = 2$, and the minimum residual method is terminated when $|S^{-1}r_k|/|S^{-1}r_0| \leq 10^{-5}$. As above, $|\cdot|$ denotes the Euclidean norm and S is the preconditioner.

Example 5.1. In the first example we let $a(x) = \text{diag}(1 + x_1x_2, 1)$, and we solve the discrete systems with the preconditioned minimum residual method. In Table 5.1 we list the number of iterations for the different preconditioners. For comparison we have also added the number of iterations without preconditioning. When the fast Poisson preconditioner is used, $\kappa(BR^{-1}) = 1$ and we expect the number of iterations to be independent of h. Furthermore, in this example the inequalities in (5.5) hold with $\alpha_1 = 1$ and $\alpha_2 = 2$. The condition number of the matrix M in (1.2) is therefore small and the minimum residual method should converge in a few iterations. It is clear from Table 5.1 that this is actually the case. We also observe that the incomplete Cholesky factorization leads to a preconditioner which reduces the number of iterations significantly compared to the case with no preconditioner, but which is considerably less effective than a fast Poisson solver. Recall, however, that the incomplete Cholesky factorization can be used on general domains.

Example 5.2. In our next example we consider a nondiagonal matrix a, given by

$$a(x) = \begin{pmatrix} 1 + 4(x_1^2 + x_2^2) & 3x_1x_2 \\ 3x_1x_2 & 1 + 11(x_1^2 + x_2^2) \end{pmatrix}. \tag{5.7}$$

TABLE 5.3
Number of iterations for Example 5.3.

h (number of iterations)	1/8	1/16	1/32	1/64	1/128
No preconditioner	149	320	619	>1000	>1000
Incomplete Cholesky preconditioner	66	93	147	278	569
Fast Poisson	56	62	61	61	61

TABLE 5.4
Number of iterations for Example 5.4. *The tolerance* ϵ_i *of the inner iteration is* 10^{-7}.

h (number of iterations)	1/8	1/16	1/32	1/64	1/128
No preconditioner	32	69	*	*	*
Incomplete Cholesky preconditioner	21	32	*	*	*
Fast Poisson	20	26	29	30	*

The eigenvalues of this matrix are

$$\lambda_1(x) = \tfrac{1}{2}(2 + 15(x_1^2 + x_2^2) + (49(x_1^2 + x_2^2)^2 + 36x_1^2x_2^2)^{1/2})$$

and

$$\lambda_2(x) = \tfrac{1}{2}(2 + 15(x_1^2 + x_2^2) - (49(x_1^2 + x_2^2)^2 + 36x_1^2x_2^2)^{1/2}).$$

Since $1 \leq \lambda_1(x) \leq \lambda_2(x) \leq 25$ the inequalities (5.5) hold with $\alpha_1 = 1$ and $\alpha_2 = 25$. Again we solve the linear systems with the minimum residual method. The number of iterations for different values of h, and for different choices of preconditioners, are listed in Table 5.2. Comparing this with Example 5.1, we note that the minimum residual method requires more iterations to converge for any choice of preconditioner compared to the corresponding results of Example 5.1. This is what we should expect, since the condition number of the matrix M in (1.2) is larger than in Example 5.1. However, the nondiagonal matrix a poses no difficulties in the application of the minimum residual method.

Example 5.3. In the last example, with the preconditioned minimum residual method we choose a to be the diagonal matrix $a(x) = \operatorname{diag}(\lambda_1(x), \lambda_2(x))$. Here $\lambda_1(x)$ and $\lambda_2(x)$ are the eigenvalues of the matrix (5.7) given in Example 5.2. The number of iterations for different values of h, and for different choices of preconditioners, are listed in Table 5.3. We note that the results of Examples 5.2 and 5.3 are almost identical. Hence this example demonstrates that the convergence rate depends on the conditioning of the matrix a and not on the possible diagonal structure of a.

Example 5.4. We again choose the variable coefficient matrix a to be the full matrix given by (5.7). The solution (u_h, p_h) of (5.6) is calculated by solving (5.2) and (5.1). Note that the symmetric, positive-definite coefficient matrix in (5.2) is not well conditioned. We solve this equation with the preconditioned conjugate gradient method and we choose the incomplete Cholesky factorization or the fast Poisson solver used above as preconditioners. Once in every iteration we must invert M; i.e., solve an equation of the form (5.3). Since the matrix a is full, it is impractical to do this with a direct elimination method. We therefore must solve equations of the form (5.3) with the conjugate gradient method. The matrix M is well conditioned and no preconditioner is necessary. This results in a two-level method. The outer iteration is terminated when $|R^{-T}r_k|/|R^{-T}r_0| < 10^{-5}$. In Tables 5.4–5.6 the number of outer iterations is reported, when the inner iteration is terminated when $|r_k|/|r_0| < \epsilon_i$ for $\epsilon_i = 10^{-7}, 10^{-8}, 10^{-9}$. The *'s in Tables 5.4–5.6 signal a breakdown of the conjugate

TABLE 5.5
Number of iterations for Example 5.4. *The tolerance* ϵ_i *of the inner iteration is* 10^{-8}.

h (number of iterations)	1/8	1/16	1/32	1/64	1/128
No preconditioner	32	69	148	313	*
Incomplete Cholesky preconditioner	21	32	57	*	*
Fast Poisson	20	25	28	30	32

TABLE 5.6
Number of iterations for Example 5.4. *The tolerance* ϵ_i *of the inner iteration is* 10^{-9}.

h (number of iterations)	1/8	1/16	1/32	1/64	1/128
No preconditioner	32	68	147	311	661
Incomplete Cholesky preconditioner	21	32	57	118	245
Fast Poisson	20	25	28	30	33

gradient algorithm. Note that since only the outer iterations are listed, these numbers are not directly comparable to the numbers in the previous examples. This experiment demonstrates that it is necessary to solve (5.3) with high-order accuracy to ensure convergence of the outer iteration.

When the variable coefficient a is diagonal, the matrix M is also diagonal. Hence, systems of the form (5.3) are easy to solve. In this case (5.2) may therefore be solved efficiently by the conjugate gradient method without a two-level iteration. Recall that in the pressure equation of reservoir simulation, where the matrix a corresponds to a mobility matrix, problems with a nondiagonal matrix a occur frequently. When a is a full matrix, M in (1.2) is not block diagonal and a direct solution of (5.3) is impractical. In this case (5.2) must be solved by a two-level iterative method, which is very sensitive to the accuracy of the inner iteration. Examples 5.2 and 5.3 clearly indicate that the performance of the preconditioned minimum residual method is independent of a being a diagonal or a full matrix. It is only dictated by the variation of a. Therefore, our opinion is that the minimum residual method, which has only one level of iteration, is preferable for these problems.

5.2. The Stokes problem. Our second model problem is a mixed finite element discretization of the stationary Stokes problem. The resulting linear system is again of the form (1.1). In this case none of the matrices M and B are well conditioned.

Let $\Omega \subset \mathbb{R}^2$ again be a bounded domain with boundary Γ. Define by $L_0^2(\Omega)$ the set of $L^2(\Omega)$ functions with zero integral over Ω, and let $H_0^1(\Omega)$ be the set of functions in $L^2(\Omega)$, with first-order partial derivatives in $L^2(\Omega)$ and with vanishing trace on Γ. Furthermore, denote by $\|\cdot\|_1$ the usual norm on $H_0^1(\Omega)$.

The variational formulation of a generalized Stokes problem reads: Find $(u,p) \in (H_0^1(\Omega))^2 \times L_0^2(\Omega)$ such that

$$
\begin{aligned}
(\mu(x)\nabla u, \nabla v) - (p, \nabla \cdot v) &= (f,v) \quad \forall v \in (H_0^1(\Omega))^2, \\
(q, \nabla \cdot u) &= 0, \qquad \forall q \in L_0^2(\Omega),
\end{aligned} \tag{5.8}
$$

where $f \in L^2(\Omega)$ and $\mu \in L^\infty(\Omega)$ with $\mu(x) \geq \alpha > 0$ almost everywhere on Ω. In the Stokes problem μ is a constant. The existence of a unique solution in the Stokes case is proved in [19], and the variable coefficient case is an easy consequence of this result. We add the variable coefficient to the problem in order to test the preconditioner on a more difficult problem (cf. [5]).

Let $V_h \subset H_0^1(\Omega)$ and $Q_h \subset L_0^2(\Omega)$ be finite-dimensional spaces, and approximate

TABLE 5.7
Number of iterations for Example 5.5.

h (number of iterations)	1/16	1/32	1/64	1/128	1/256
No preconditioner	132	243	450	815	>1000
Incomplete Cholesky preconditioner	53	92	175	366	906
Fast Poisson	21	21	21	21	21

(5.8) by the following discrete problem: Find $(u_h, p_h) \in V_h^2 \times Q_h$ such that

$$\begin{aligned}(\mu(x)\nabla u_h, \nabla v) - (p_h, \nabla \cdot v) &= (f, v) \quad \forall v \in V_h^2, \\ -(q, \nabla \cdot u_h) &= 0 \quad \forall q \in Q_h.\end{aligned} \tag{5.9}$$

As above, the finite-dimensional spaces V_h^2 and Q_h should satisfy the inf-sup condition

$$\inf_{q \in Q_h} \sup_{v \in V_h^2} \frac{(q, \nabla \cdot v)}{\|q\| \|v\|_1} \geq \beta > 0, \tag{5.10}$$

where β is a constant independent of the mesh parameter h. In our experiments we use the subspaces described in [5]. We briefly review the construction of these spaces when Ω is the unit square.

First, choose a positive integer N, set $h = \frac{1}{2}N$, and divide Ω into $2N \times 2N$ subsquares. Then, subdivide all these subsquares into two triangles, and use the usual continuous linear finite element on this triangulation to construct V_h. Next let $\tilde{Q}_h$ be the set of discontinuous functions, constant on each subsquare. It can easily be seen that the spaces V_h^2 and $\tilde{Q}_h$ do not satisfy the inf-sup condition (5.10). Therefore, a subspace Q_h of $\tilde{Q}_h$ is used. Define for $k, l = 1, 2 \cdots 2N$ the function $\theta_{k,l}$ that takes the value one on the subsquare $[(k-1)h, kh] \times [(l-1)h, lh]$, and vanishes elsewhere. Then define the function $\phi_{i,j} \in \tilde{Q}_h$ by

$$\phi_{i,j} = \theta_{2i-1,2j-1} - \theta_{2i,2j-1} - \theta_{2i-1,2j} + \theta_{2i,2j}$$

for $i, j = 1, \cdots, N$, and the space Q_h by

$$Q_h = \{q \in \tilde{Q}_h \mid (q, \phi_{i,j}) = 0, \quad i, j = 1, \cdots, N\}.$$

It is verified in [22] that the inf-sup condition (5.10) holds for this pair of subspaces.

The system (5.9) has the same structure as the system (1.1). In the Stokes case the matrix M is block diagonal with two copies of a discrete Laplace operator on the diagonal. Obviously, we choose L to be block diagonal, with two Poisson preconditioners on the diagonal. In our calculations we use the preconditioners from §5.1. It remains to choose R; i.e., a preconditioner for $B^T(LL^T)^{-1}B$. However, in this model, $B^T M^{-1} B$ is well conditioned (cf. [5]). Hence, if LL^T is a preconditioner for M, $B^T(LL^T)^{-1}B$ will also be reasonably well conditioned. We therefore choose $R = I$.

In the numerical examples below, the iterations are terminated when the weighted Euclidean norm of the residual, $|S^{-1}r_k|$, is reduced by a factor of 10^{-5}.

Example 5.5. In this example we choose $\mu(x) = 1$, and solve the system (5.9) with the minimum residual method. The effect of the incomplete Cholesky and the fast Poisson preconditioners are reported in Table 5.7. In this example the fast Poisson solver actually inverts M. Hence, both $L^{-1}ML^T$ and $L^{-1}B$ are well conditioned

TABLE 5.8
Number of iterations for Example 5.6.

h (number of iterations)	1/16	1/32	1/64	1/128	1/256
No preconditioner	280	605	>1000	>1000	>1000
Incomplete Cholesky preconditioner	75	143	304	640	>1000
Fast Poisson	48	54	55	55	52

TABLE 5.9
Number of iterations for Example 5.7. *The tolerance* ϵ_i *of the inner iteration is* 10^{-8}.

h (number of iterations)	1/16	1/32	1/64	1/128
No preconditioner	20	22	*	*
Incomplete Cholesky preconditioner	20	22	*	*
Fast Poisson	20	22	*	*

independent of h. From Table 5.7 we observe, as expected, that the minimum residual method converges in a few iterations. Also for this problem we observe that the results obtained with the incomplete Cholesky factorization rank somewhere between the results obtained with no preconditioning and with preconditioning by the fast Poisson solver.

Example 5.6. In this example we use the minimum residual method to solve the system (5.9) with $\mu(x) = 1+x_1x_2+x_1^2-x_2^2/2$. Again we use the incomplete Cholesky and the fast Poisson preconditioners. The number of iterations is listed in Table 5.8. In this case the fast Poisson preconditioner is not an exact inverse of M, and we expect more iterations than in Example 5.5. However, we observe, as expected, that the number of iterations appears to be independent of h when the Poisson solver is used.

Example 5.7. In this last example we choose $\mu(x) = 1 + x_1x_2 + x_1^2 - x_2^2/2$ and solve the linear system with the two-level conjugate gradient method. In this case the coefficient matrix of the linear system (5.2) is well conditioned, but M is not. It is therefore unnecessary to precondition the outer iteration. In the inner iteration we use the preconditioned conjugate gradient method to solve the Poisson problem for each component of u. As preconditioners we still use the incomplete Cholesky factorization and the fast Poisson solver. The outer iteration is terminated when the Euclidean norm of the residual is reduced by a factor of 10^{-5}. The inner iteration is terminated when the weighted Euclidean norm of the residual is redused by a factor of ϵ_i, for $\epsilon_i = 10^{-8}, 10^{-9}$. The number of outer iterations is given in Tables 5.9 and 5.10. The $*$'s in these tables signal a breakdown of the conjugate gradient algorithm due to division by zero. The results show that the method is sensitive with respect to the accuracy of the inner iteration. Note that only the number of outer iterations is given.

We conclude this subsection with some comments. In Example 5.5, the fast Poisson solver may be used to solve (5.3). Equation (5.2) may therefore be solved in an efficient way by the conjugate gradient method without a two-level iteration. However, when μ is not constant or Ω is not square a direct solution of (5.3) may be impractical. Since the two-level iteration is sensitive to the accuracy of the inner iteration, the minimum residual method, which has only one iteration level, seems preferable.

5.3. Conclusion. The numerical examples above indicate that the preconditioned minimum residual method is an effective iterative procedure for saddlepoint

TABLE 5.10
Number of iterations for Example 5.7. *The tolerance* ϵ_i *of the inner iteration is* 10^{-9}.

h (number of iterations)	1/16	1/32	1/64	1/128
No preconditioner	20	22	24	26
Incomplete Cholesky preconditioner	20	22	24	26
Fast Poisson	20	22	24	26

problems of the form (1.1) arising as discretizations of partial differential equations. In particular, we have demonstrated that for properly scaled matrices M and B the convergence of the minimum residual method depends on the condition numbers of M and B, but not on the sparsity structure of these matrices. Furthermore, we have demonstrated that the performance of the two-level conjugate gradient method is sensitive to the accuracy of the inner iteration.

REFERENCES

[1] B. ATLESTAM, *Tschebysceff-polynomials for sets consisting of two disjoint intervals with application to convergence estimates for the conjugate gradient method*, Res. Report 77.06R, Department of Computer Sciences, Chalmers University of Technology and the University of Göteborg, Göteborg, Sweden, 1977.

[2] O. AXELSSON, *Numerical algorithms for indefinite problems*, in Elliptic Problem Solvers, Academic Press, New York, 1984, pp. 219–232.

[3] O. AXELSSON AND V. A. BARKER, *Finite element solution of boundary value problems*, in Computer Science and Applied Mathematics, Academic Press, New York, 1984.

[4] R. E. BANK, B. D. WELFERT, AND H. YSERENTANT, *A class of iterative methods for solving saddle point problems*, Numer. Math., 56 (1990), pp. 645–666.

[5] J. H. BRAMBLE AND J. E. PASCIAK, *A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems*, Math. Comp., 50 (1988), pp. 1–17.

[6] ———, *A domain decomposition technique for Stokes problems*, Appl. Numer. Math., 6 (1990), pp. 251–261.

[7] F. BREZZI, J. DOUGLAS, JR., R. DURÀN, AND M. FORTIN, *Mixed finite elements for second order elliptic problems in three variables*, Numer. Math., 51 (1987), pp. 237–250.

[8] F. BREZZI, J. DOUGLAS, JR., M. FORTIN, AND L. D. MARINI, *Efficient rectangular mixed finite elements in two and three space variables*, R.A.I.R.O. Math. Model. Numer. Anal., 21 (1987), pp. 581–604.

[9] T. F. CHAN AND H. C. ELMAN, *Fourier analysis of iterative methods for elliptic problems*, SIAM Rev., 31 (1989), pp. 20–49.

[10] R. CHANDRA, *Conjugate gradient methods for partial differential equations*, Report 129, Computer Science Department, Yale University, New Haven, CT, 1978.

[11] J. DOUGLAS, JR. AND P. PIETRA, *A description of some alternating-direction techniques for mixed finite element methods*, in Mathematical and Computational Methods in Seismic Exploration and Reservoir Modeling, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1985, pp. 37–53.

[12] J. DOUGLAS, JR. AND J. E. ROBERTS, *Global estimates for mixed methods for second order elliptic equations*, Math. Comp., 44 (1985), pp. 39–52.

[13] M. EIERMANN, X. LI, AND R. S. VARGA, *On hybrid semi-iterative methods*, SIAM J. Numer. Anal., 23 (1989), pp. 152–168.

[14] R. E. EWING, R. D. LAZAROV, P. LU, AND P. S. VASSILEVSKI, *Preconditioning indefinite systems arising from the mixed finite element discretization of second-order elliptic systems*, in Preconditioned Conjugate Gradient Methods, O. Axelsson and L. Kolotilina, eds., Lecture Notes in Mathematics 1457, Springer-Verlag, Berlin, 1990, pp. 28–43.

[15] R. S. FALK, *An analysis of the finite element method using Lagrangian multipliers for the stationary Stokes equations*, Math. Comp., 30 (1976), pp. 241–269.

[16] R. S. FALK AND J. E. OSBORN, *Error estimates for mixed methods*, R.A.I.R.O. Numer. Anal., 14 (1980), pp. 249–277.

[17] M. FORTIN, *Some iterative methods for incompressible flow problems*, Comput. Phys. Comm., 53 (1989), pp. 393–399.

[18] M. FORTIN AND R. GLOWINSKI, *Augmented Lagrangian methods: Application to the numerical*

solution of boundary-value problems, Stud. Math. Appl., Vol. 15, North–Holland, Amsterdam, 1983.

[19] V. GIRAULT AND P. A. RAVIART, *Finite element methods for the Navier–Stokes equations*, Springer Series in Computational Mathematics, Vol. 5, Springer-Verlag, Belin, New York, 1986.

[20] G. H. GOLUB AND M. L. OVERTON, *The convergence of inexact Chebyshev and Richardson iterative methods for solving linear systems*, Numer. Math., 53 (1988), pp. 571–593.

[21] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.

[22] C. JOHNSON AND J. PITKÄRANTA, *Analysis of some mixed finite element methods related to reduced integration*, Math. Comp., 38 (1982), pp. 375–400.

[23] C. C. PAIGE AND M. A. SAUNDERS, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629.

[24] W. QUECK, *The convergence factor of preconditioned algorithms of the Arrow–Hurwich type*, SIAM J. Numer. Anal., 26 (1989), pp. 1016–1030.

[25] P. A. RAVIART AND J. M. THOMAS, *A mixed finite element method for 2-order elliptic problems*, in Mathematical Aspects of Finite Element Methods, Lecture Notes in Mathematics 606, I. Galligani and E. Magenes, eds., Springer-Verlag, Berlin, New York, 1977, pp. 295–315.

[26] T. F. RUSSEL AND M. F. WHEELER, *Finite element and finite difference methods for continous flow in porous media*, in The Mathematics of Reservoir Simulation, R. E. Ewing, ed., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1983.

[27] J. STOER, *Solution of large linear systems of equations by conjugate gradient type methods*, in Mathematical Programming—The State of the Art, A. Bachem, M. Grötschel, and B. Korte, eds., Springer-Verlag, Berlin, New York, 1983, pp. 540–565.

[28] P. N. SWARZTRAUBER, *The methods of cyclic reduction, Fourier analysis and the* FACR *algorithm for the discrete solution of Poisson's equation on a rectangle*, SIAM Rev., 19 (1977), pp. 490–501.

[29] R. VERFÜRTH, *A combined conjugate gradient-multigrid algorithm for the numerical solution of the Stokes problem*, IMA J. Numer. Anal., 4 (1984), pp. 441–455.

[30] A. WEISER AND M. F. WHEELER, *On convergence of block-centered finite differences for elliptic problems*, SIAM J. Numer. Anal., 25 (1988), pp. 351–375.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 905–925, July 1992

013

PRECONDITIONING FOR BOUNDARY INTEGRAL EQUATIONS*

STEPHEN A. VAVASIS†

Abstract. New classes of preconditioners are proposed for the linear systems arising from a boundary integral equation method. The problem under consideration is Laplace's equation in three dimensions. The system arising in this context is dense and unsymmetric. These preconditioners, which are based on solving small linear systems at each node, reduce the number of iterations in some cases by a factor of 8. Three iterative methods are considered: conjugate gradient on the normal equations, CGS of Sonneveld, and GMRES of Saad and Schultz. For a simple model problem, the exact relationship between the preconditioners and the resulting condition number of the system is investigated. This analysis proves that the condition number of the preconditioned system is decreased by a factor asymptotically greater than any constant.

Key words. integral equations, boundary element methods, preconditioners, iterative methods

AMS(MOS) subject classifications. 65F10, 65R20, 65N38, 65F35

1. Boundary integral equations.

The *boundary integral equation method* (BIEM) is a technique for solving certain kinds of boundary value problems including Laplace's equation. The specific problem under consideration is: Given a compact region $\Omega \in \mathbb{R}^3$ whose boundary is homeomorphic to a sphere, find a solution to $\nabla^2\phi = 0$ given Neumann or Dirichlet boundary data. The Neumann and Dirichlet boundary data can be expressed in terms of one another via integral equations. By discretizing these integral equations, we arrive at a dense linear system relating the Neumann and Dirichlet data on the boundary. If Dirichlet data is given, it can be solved for the Neumann data or vice versa. Once both sets of boundary data are known, further integrations yield interior values of ϕ.

Our application for Laplace's equation is a free-surface irrotational incompressible fluid flow in three dimensions (see Liu, Hsu, Lean, and Vavasis [16]). The data determined by BIEM in this case is the normal derivative of the velocity potential on the free surface.

Accordingly, each timestep of the fluid calculation requires a new solution of Laplace's equation, which in turn requires the solution of the dense unsymmetric system of linear equations arising from BIEM.

For the remainder of this section we explain the details of the integral equations and linear systems. The boundary integral equation method is credited to Hess and Smith [12], Jaswon [14], and Symm [24]. For more information about the method, we refer the reader to the series edited by Brebbia [5].

Suppose ϕ satisfies Laplace's equation. Let z be a point on $\partial\Omega$. Let $g_z(x)$ be the function

$$g_z(x) = \frac{1}{2\pi\|x - z\|}.$$

* Received by the editors April 5, 1990; accepted for publication (in revised form) August 26, 1991. This work was supported in part by a gift from Xerox Webster Research Center and by the U.S. Army Research Office through the Mathematical Sciences Institute of Cornell University. Revision work was supported by National Science Foundation Presidential Young Investigator award and by the Applied Mathematical Sciences program, U.S. Department of Energy, under contract DE-AC04-76DP00789, while the author was visiting Sandia National Laboratories.

† Department of Computer Science, Cornell University, Ithaca, New York 14853 (vavasis@cs.cornell.edu).

For now and for the rest of the paper, $\|\cdot\|$ will denote the 2-norm. Lower case letters are used for scalars and for two- or three-dimensional vectors. Boldface is used for vectors of higher dimension. Capital letters are used for matrices. With this definition of g, the following identity always holds:

$$c\phi(z) = \int_{\partial\Omega} \frac{\partial\phi}{\partial n}(x) g_z(x)\, ds - \int_{\partial\Omega} \frac{\partial g_z}{\partial n}(x)\phi(x)\, ds. \tag{1}$$

Here c measures the angle at the point z and would be 1 if the boundary is smooth. There is an analogous equation for two-dimensional Laplace problems with a different choice for the function g (called the *Green's function*). See §5. Note that the integrals in this equation are improper because the function g_z and its derivative are infinite at $x = z$. The integrals still have a well-defined value, but these singularities make numerical methods more complicated.

Suppose we divide the surface of Ω into elements joined by nodes. Let the nodes be $\{z_1, \cdots, z_n\}$. Then an equation like (1) holds for each z_i. Writing down equations for each z_i in this fashion is generally known as *collocation*. We can approximate the solution to the Laplace equation by assuming that the ϕ and $\partial\phi/\partial n$ are piecewise polynomial interior to each element and are therefore determined by their values at nodal points. This means that there are $2n$ variables $\phi(z_1), \cdots, \phi(z_n)$ and $\partial\phi/\partial n(z_1), \cdots, \partial\phi/\partial n(z_n)$. The integrations in (1) are approximated with numerical quadrature; the integral on each element can be expressed as a linear combination of the variables at its vertices. This gives a dense linear system of n equations in the $2n$ variables; the system looks like this:

$$\begin{aligned} c_1\phi(z_1) &= \sum_{j=1}^{n} d_{1j}\partial\phi/\partial n(z_j) - \sum_{j=1}^{n} e_{1j}\phi(z_j), \\ &\vdots \\ c_n\phi(z_n) &= \sum_{j=1}^{n} d_{nj}\partial\phi/\partial n(z_j) - \sum_{j=1}^{n} e_{nj}\phi(z_j). \end{aligned} \tag{2}$$

In this system of equations the coefficients d_{ij} and e_{ij} arise from evaluating the Green's function. Accordingly, all of the coefficients (c's, d's, and e's) are determined entirely by the geometry of $\partial\Omega$. See Atkinson [1] for a description of specific methods for collocation.

In a well-posed Laplace problem, for each point on the boundary either $\partial\phi/\partial n$ or ϕ is specified at the point. Therefore, in the above system of equations, n out of the $2n$ variables will be specified and n will be unknown. Since (2) has n equations, we use these equations to solve for the remaining unknowns. This system will generally be dense and unsymmetric.

It follows that the coefficient matrix of the system to be solved is determined entirely by $\partial\Omega$ and by the specification of which kind of boundary data is provided for each node.

Once the values of ϕ and $\partial\phi/\partial n$ are known, there is an equation similar to (1) to compute any interior value of ϕ.

In some applications of BIEM, interior values are not needed. In our fluid application it is enough to know $\partial\phi/\partial n$ everywhere on the boundary.

The remainder of the paper is organized as follows. In §2 we discuss iterative methods for solving (2). In §3 we introduce our classes of preconditioners for this

problem. In §4 we report on our computational experience with the algorithm. In §5 we give an analysis of one of the preconditioners for a simple model problem. We show that the condition number is reduced by a factor of $\sqrt{\ln n}$ for the model. Finally, in §6 we discuss computational complexity issues connected with the BIEM solution.

The emphasis of this paper is on the use of the preconditioners for Laplace problems in three dimensions with collocation and mixed boundary conditions. There appears to be very little known about such matrices; they lack many of the properties (such as symmetry and positive definiteness) that are instrumental for analysis of finite-element methods.

Accordingly, we do not attempt an analysis of the preconditioners for the problems actually of interest to us. The analysis given in §5 is for a simple two-dimensional model problem and appears to be pessimistic compared to our computational experiments.

2. Iterative methods for linear systems. An *iterative* method for solving a system of linear equations $A\mathbf{x} = \mathbf{b}$ is a method that generates a sequence of vectors converging to the true solution. Applied to a dense $n \times n$ system, one iteration of an iterative method ordinarily requires $O(n^2)$ steps. Solving the system with elimination requires $O(n^3)$ steps. Accordingly, the hope is that a good iterative method will require only a few iterations and be faster than elimination.

Canning [6] has developed a technique to reduce the number of nonzero elements in a BIEM matrix from n^2 to $O(n)$ in many cases. Iterative methods would become even more attractive, because the time per iteration is $O(n)$. We have not tried Canning's method for our problem.

The three iterative methods under consideration in this paper are GMRES, CGS, and conjugate gradient applied to the normal equations. The GMRES algorithm is due to Saad and Schultz [21] and operates directly on the original system. CGS, due to Sonneveld [23], also operates on the original system. Conjugate gradient (CG), due to Hestenes and Stiefel [13], only works for symmetric positive definite systems. Accordingly, CG is applied to $A^TA\mathbf{x} = A^T\mathbf{b}$ instead of $A\mathbf{x} = \mathbf{b}$. Note that the product A^TA is never actually formed—this would ruin the bound on the running time. Instead, whenever a product $A^TA\mathbf{y}$ is needed by CG, we carry out the product as two matrix-vector multiplications $A^T(A\mathbf{y})$. This remark about matrix products also applies to the case described below in which preconditioners are inserted into the problem. CG applied to the normal equations is known as CGNR.

It is known (see Golub and Van Loan [10]) that the convergence rate of CGNR can be bounded in terms of the condition number of the matrix of coefficients. The convergence rate for GMRES is determined by the positions of the eigenvalues or pseudoeigenvalues (see Nachtigal, Reddy, and Trefethen [18]). There is no simple formula known for the convergence of CGS.

Suppose that we have a nonsingular matrix P such that PA has a lower condition number than A alone. Then the system of equations $PA\mathbf{x} = P\mathbf{b}$ has the same solution as $A\mathbf{x} = \mathbf{b}$. Moreover, an iterative method applied to $PA\mathbf{x} = P\mathbf{b}$ might converge faster because of the lower condition number. Such a matrix P is called a *left preconditioner.* Ideally, we want to have PA close to the identity matrix.

Each iteration of conjugate gradient requires one matrix-vector multiply operation. Since we are working on the normal equations, we need two matrix-vector multiplications. Finally, if we use preconditioning then the system becomes A^TP^TPA so we need four matrix-vector multiplications.

Each iteration of CGS requires two matrix-vector multiplications. Each iteration

of GMRES requires one matrix-vector multiplication. The time required for the kth iteration of GMRES is proportional to $O(kn + n^2)$, in contrast to CGNR and CGS, which require only $O(n^2)$ steps per iteration.

3. Preconditioners for BIEM. In this section we introduce three classes of preconditioners tailored to the BIEM problem described in the introduction. All three of our preconditioners are based on the same idea, which is as follows. The rows of preconditioner P are generated independently and can be done in parallel. Let the ith column of P^T be denoted as $\mathbf{p}_i$. As mentioned in the last section, we would ideally like to have $A^T\mathbf{p}_i = \mathbf{e}_i$, where $\mathbf{e}_i$ is the ith column of the identity matrix.

For our preconditioners we take the following approach. From the matrix A and other data we determine some small list L of indices drawn from $\{1, \cdots, n\}$ such that the variables and constraints in L have the most impact on variable i. Then we solve the small system of equations $\bar{A}^T\bar{\mathbf{p}}_i = \bar{\mathbf{e}}_i$ where the bars over the variables indicate that we are deleting all the rows and columns except for those in L. Once this solution is known, we expand it back to the entries of P. This is done for all rows of P; the rows can be generated in parallel.

The brief description in the last paragraph completely explains how to compute our classes of preconditioners. All that remains is the explanation of how L is chosen and how to expand the solution of the small system back to the row of P. In all cases the number of flops to compute the preconditioner is bounded by $O(k^3n)$ where k is the maximum size of the small system solved for each node.

Preconditioner MN. Preconditioner MN stands for "mesh neighbor." Recall that each variable of the linear system arising in a BIEM problem corresponds to a node on the surface of the region. We say that two nodes are *neighbors* if they border on a common element. The choice of L for node i is as follows: We let L be i together with all the indices of neighbors of node i. After the small system involving these rows and columns is solved, then we scatter the entries of the solution vector $\bar{\mathbf{p}}_i$ back to their original coordinate positions in $\mathbf{p}_i$, and we fill in the remaining positions of $\mathbf{p}_i$ with zeros.

The motivation for preconditioner MN is as follows. The matrix coefficient relating node i and node j comes up roughly from a formula like $1/\|x_i - x_j\|$. Therefore, the further apart two nodes are, the less impact we would expect a change at one node to have on the other. Since neighboring nodes are the most interrelated, we put them in our preconditioner.

Note that the preconditioner generated in this manner will be sparse, and in particular, its sparsity pattern will mirror the connectivity of the mesh. In general, sparsity is considered a very desirable feature of a preconditioner. See, for example, Manteuffel [17] or Kolotilina and Yeremin [15].

For our application, sparsity is an added benefit but not a major goal. The reason is that A is already dense, so we have to carry out a dense $n \times n$ matrix-vector multiply operation in each iteration. Therefore, the additional multiplication involving P is not too expensive, relatively speaking, even if P is dense.

Preconditioner ME. Preconditioner ME stands for "matrix entries." The way to choose L for a variable i for this preconditioner is as follows. If a_{ij}, a_{ji} satisfy $|a_{ij}a_{ji}| \geq t|a_{ii}a_{jj}|$ then include j in L. Here, t denotes some user-specified tolerance (our choice was $t = 0.1$). Note that this preconditioner does not depend on any BIEM structure and hence could be applied to an arbitrary matrix.

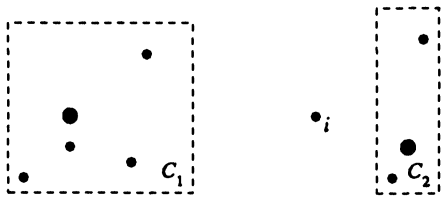


FIG. 1. *C_1 is acceptable; C_2 is not acceptable.*

After the small system is solved, the solution vector is scattered back into P as in preconditioner MN.

This preconditioner is similar to the local matrix idea of Benson and Frederickson [2].

Preconditioner HC. Preconditioner HC stands for "hierarchical clustering." Hierarchical clustering, an idea due to Rokhlin [20], classifies nodes based on how far away they lie from node i. This is the most complicated of the three preconditioners and requires a more extensive explanation.

A *cluster* refers to a set of nodes that are intended to be near one another. The first step in constructing HC is to make a hierarchy of clusters. The top cluster C_0 of the hierarchy contains all the nodes. The next level contains two clusters, each with about half the nodes. These clusters are then recursively subdivided, so that the kth level of the hierarchy consists of 2^k mutually disjoint clusters of about equal size whose union is the entire set of nodes.

There are many algorithms for clustering geometric objects (see, for example, Feder and Greene [9] and Greengard [11]). We have chosen the following simple approach, which seems to work well in practice. The top level cluster C_0 is divided in two by splitting the points according to the median x_1 coordinate. The clusters at the next level are split according to their median x_2 coordinates. The next level is split according to median x_3 coordinates. The further levels cycle through the three coordinates.

Once the cluster hierarchy is built, we next identify a *center* for each cluster. This is the node in the cluster such that the maximum distance to other nodes in the cluster is minimized. In our current implementation we find the exact center of each cluster in the hierarchy; this fairly expensive computation ($O(n^2)$ steps total) could be replaced by a heuristic. The *radius* of a cluster is defined to be the maximum distance from its center to other nodes in the cluster.

Once the clusters, centers, and radii are computed, we next address the actual preconditioner. For each node i, we carry out the following steps to produce the list L of nodes related to i. We say that a cluster C is *acceptable* to node i if the distance from node i to the center of C is at least tr, where r is the radius of C and $t > 1$ is a user-specified parameter. In Fig. 1 we show a sample node i and an acceptable and unacceptable cluster, with $t = 2$. The centers of the two clusters are enlarged.

The algorithm for constructing list L for node i is the following recursive procedure, whose initial arguments are (i, C_0).

function **make-list**(i, C)
    if C is acceptable to i then
        make-list := {center of C}.
    else
        Let C_1, C_2 be the children of C in
          the hierarchy.
        make-list := **make-list**$(i, C_1) \cup$ **make-list**(i, C_2).
    end

Note that this recursive procedure is guaranteed to terminate, since the bottom level of the hierarchy consists of clusters made up of individual nodes. These clusters have radius zero and hence are acceptable to all nodes.

The motivation is as follows. Suppose C is acceptable to i. The coupling in matrix A between the nodes of cluster C and node i will not depend very much on which node of C is selected. Therefore, we could summarize all the nodes of C with a single node, namely, the center. This motivation follows Rokhlin [20] and Greengard [11], although their algorithms for determining the representation of a cluster are considerably more sophisticated since they are interested in getting strong bounds on the difference between the "cluster" approximation and the true effect of all nodes of the cluster.

The list L is generated in this manner, and then the small system for i is solved. In preconditioners MN and ME the elements of the small system solution were scattered throughout $\mathbf{p}_i$ and the remaining elements were filled with zeros. Preconditioner HC will be dense; the entry at position m of $\mathbf{p}_i$ is taken to be equal to the entry in $\bar{\mathbf{p}}_i$ corresponding to the representative of m's cluster, scaled by the cluster size.

For the parameter t in the definition of "acceptable," we used $t = 1.5$ in our experiments. This seemed to give reasonable preconditioning results. It should be noted that larger values of t give more reliable clusters because the "clustering" effect is expected to be more apparent, but they also result in significantly more expensive preconditioners since the lists L become larger. Rokhlin [20] settles on a choice for t based on careful analysis of the clustering effect, which we have not been able to carry out in the setting of preconditioners.

It is also not clear whether it makes sense to force the cluster boundaries to respect nondifferentiable edges and vertices of the region or boundaries between regions with different types of boundary conditions (we did not). The theory for boundary element methods with nonsmooth boundaries and mixed boundary conditions is not understood well enough to make definite statements in this regard.

An important question to ask about our preconditioners is whether they can be stably computed in all cases. The methods break down if one of the small linear systems is singular. We have the following theorem in this regard.

THEOREM 3.1. *If A is either symmetric positive definite or strictly diagonally dominant, then all three preconditioners can be computed stably.*

This theorem is immediately obvious from the fact that the small systems that are solved in all cases are principal submatrices of A. Therefore, these submatrices are symmetric positive definite if A is, and they are strictly diagonally dominant if A is.

Of course, the matrices of interest to us are neither symmetric positive definite nor strictly diagonally dominant. We have not encountered any instabilities in our test runs (such instabilities would most likely manifest themselves as unexpectedly large elements of the preconditioner). There is no theory at present to predict whether this

could happen.

The primary value of the above theorem is to show that our preconditioning ideas may be applicable to a broad variety of dense or even sparse problems. By way of comparison, the well-known incomplete Cholesky preconditioners (see Manteuffel [17] and Elman [8]) can be stably computed for only a subset of symmetric positive definite matrices.

4. Computational experience. We tried all three preconditioners out on three different test problems. For each problem we tried CGS, GMRES, and CGNR. We also noted the condition numbers of the system with and without preconditioners. All experiments were carried out with MATLAB on an Ardent Titan. MATLAB, an interactive language for numerical computation, is a trademark of The Mathworks, Inc. The results are summarized in Tables 1–3. The numbers in the six middle columns are number of iterations and number of millions of floating point operations for the iterative methods as computed by MATLAB. The last column is the 2-condition number. An illustration of the domains of the test problems are in Figs. 2–4.

In all of the domains we used mixed boundary conditions. The surfaces are divided into triangular elements, and the functions are assumed to be piecewise linear on the triangular elements. The collocation points are identical to the nodal points of the discretization.

Dense matrix-vector multiplication was used to compute $PA\mathbf{x}$ in the iterative routines, even when P was sparse (including the case $P = I$). Accordingly, the floating point operation counts are higher in many cases than would be expected from the number of iterations.

The floating point count does not include the time to compute the preconditioner. In the cases of MN and ME, the operations to compute the preconditioner were negligible. In the case of HC, the number of operations to compute the preconditioner was substantial—on the same order as the number of operations of the iterative method. In the domain for Table 1, 0.69 million operations were required to compute HC. The numbers for Tables 2–3 are 2.42 million and 2.20 million, respectively.

Our convergence criterion was as follows. In all three routines we stopped when the residual computed by the routine dropped below $\epsilon|\hat{\mathbf{b}}|$, where ϵ denotes the unit roundoff (around $2.2 \cdot 10^{-16}$ on our computer) and $\hat{\mathbf{b}}$ denotes the right-hand side of the actual system being solved. In GMRES we stopped also when the residual norm did not improve by a factor smaller than 0.9 over two consecutive iterations. This was necessary in GMRES because we noticed its tendency to stop converging when the residual was slightly larger than $\epsilon|\hat{\mathbf{b}}|$. Such a test was not possible in CGS or CGNR because of oscillation in the computed residual.

In most cases the residual $\|\mathbf{b} - A\mathbf{x}\|$ computed a posteriori was on the order of 10^{-14} or 10^{-15}; in a few cases it was as large as 10^{-12}. These anomalous residuals always occurred in trials with CGS, where we discovered a divergence between the a posteriori residual and the computed residual $\|\mathbf{r}\|$ in the algorithm.

In each trial the right-hand side was generated randomly (each component was selected uniformly at random between -0.5 and 0.5). The same right-hand side was used for all trials in a table.

The most striking feature of these tables is the drastic reduction in the number of iterations when comparing preconditioned systems to the case with no preconditioner. In some cases the reduction is over a factor of 8.

The three preconditioners themselves seemed to have similar properties. We notice that preconditioner ME generally did slightly worse than the other two. Precon-

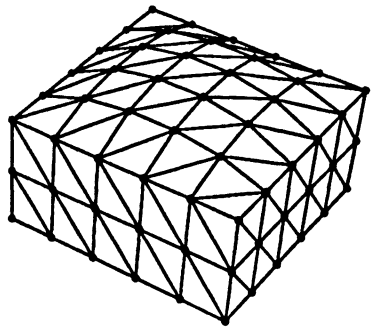


FIG. 2. *Domain D_1 with Dirichlet conditions on the top surface, Neumann conditions on the other surfaces.*

TABLE 1
Domain D_1 computational results ($n = 92$).

Precond.	CGS		GMRES		CGNR		κ_2
	iter.	m.flop.	iter.	m.flop.	iter.	m.flop.	
I	106	8.25	92	4.85	189	13.16	45.7
MN	12	0.88	20	0.82	33	2.28	3.2
HC	13	0.96	22	0.91	41	2.84	4.6
ME	13	0.96	22	0.91	38	2.63	4.3

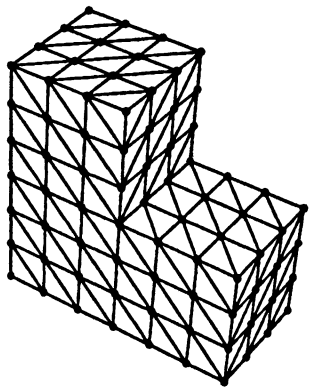


FIG. 3. *Domain D_2 with Dirichlet conditions on the two small end surfaces, Neumann conditions elsewhere.*

TABLE 2
Domain D_2 computational results ($n = 128$).

Precond.	CGS		GMRES		CGNR		κ_2
	iter.	m.flop.	iter.	m.flop.	iter.	m.flop.	
I	98	14.65	96	8.91	243	32.50	70.6
MN	13	1.84	21	1.62	46	6.12	5.3
HC	13	1.84	22	1.69	50	6.66	5.6
ME	14	2.00	23	1.77	55	7.33	6.6

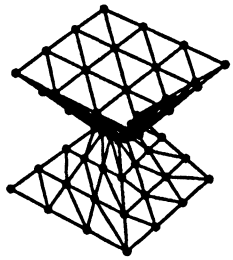


FIG. 4. *Domain D_3 with Dirichlet conditions on all surfaces except the bottom.*

TABLE 3
Domain D_3 computational results ($n = 92$).

Precond.	CGS		GMRES		CGNR		κ_2
	iter.	m.flop.	iter.	m.flop.	iter.	m.flop.	
I	118	9.20	64	3.05	198	13.79	75.1
MN	14	1.04	24	1.00	46	3.19	6.0
HC	14	1.04	24	1.00	38	2.63	3.9
ME	17	1.27	26	1.08	52	3.61	8.4

ditioner MN did slightly better than the other two. Algorithm GMRES marginally outperformed CGS in all trials when comparing the measured number of floating point operations; CGNR was a distant third.

In order to understand in practice how the preconditioners affect the convergence rates, we have plotted a sample of the pseudoeigenvalues of A and PA in Fig. 5, where P denotes preconditioner MN. These tests were done for an L-shaped region similar to domain D_2 but with fewer nodes. A sample of the pseudoeigenvalues of X (where $X = A$ or $X = PA$) was computed by plotting the eigenvalues of 20 complex perturbations of X. The entries of the perturbation were chosen uniformly at random in the disk in $\mathbb{C}$ of radius δ, where δ was taken to be 0.01, 0.005, and 0.0025. A rough rule is that convergence of GMRES is better when the pseudoeigenvalues are clustered away from zero; see Nachtigal, Reddy, and Trefethen [18]. We notice that the clustering seems to be greatly improved in the case of the preconditioned matrix.

To be precise, [18] has the following bound, based on earlier work by Greenbaum, Trefethen [25], and others:

$$\|\mathbf{b} - A\mathbf{x}^{(k)}\| \leq \|\mathbf{b} - A\mathbf{x}^{(0)}\| \cdot \inf\{\sup\{|p(z)| : z \in \Lambda_\epsilon\} : p \in \Pi_k\} \cdot L/(2\pi\epsilon).$$

In this formula, $\mathbf{x}^{(k)}$ denotes the kth iterate of GMRES; Λ_ϵ is the set of ϵ-pseudoeigenvalues of A (a subset of $\mathbb{C}$); L is the length of the boundary of Λ_ϵ; and Π_k is the set of degree-k complex polynomials $p(z)$ such that $p(0) = 1$. Thus, if the pseudospectrum is clustered away from the origin, it is easier to find a low-degree polynomial that nearly vanishes on the pseudospectrum and that is 1 at the origin, indicating that the inf factor in the preceding bound will be small. Conversely, a bad case for GMRES is when the eigenvalues or pseudoeigenvalues are clustered around the origin.

In the plots of Fig. 5, the pseudoeigenvalues of A are shown side by side with the pseudoeigenvalues of PA, for $\delta = 0.01, 0.005, 0.0025, 0$. Note that when $\delta = 0$, we are plotting the exact eigenvalues. Another way to bound the convergence of GMRES is the following formula in terms of exact eigenvalues:

$$\|\mathbf{b} - A\mathbf{x}^{(k)}\| \leq \|\mathbf{b} - A\mathbf{x}^{(0)}\| \cdot \kappa_2(V) \cdot \inf\{\sup\{|p(z)| : z \in \Lambda_0\} : p \in \Pi_k\}.$$

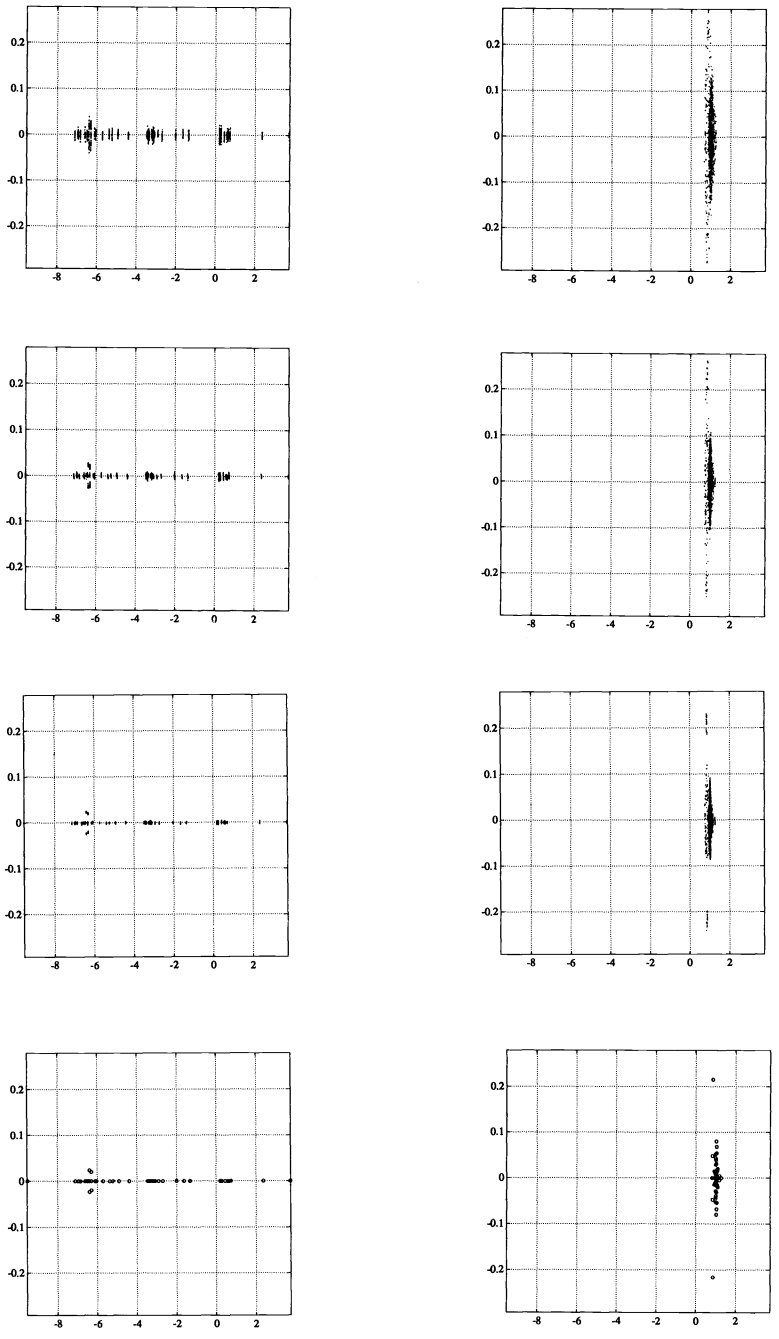


FIG. 5. *A sample of the pseudoeigenvalues of A (left) and PA (right) for L-shaped domain with* $n = 58$. *First row is* $\delta = 0.01$, *second row is* $\delta = 0.005$, *third row is* $\delta = 0.0025$, *fourth row is* $\delta = 0$. *The x and y axes (scaled differently) are the real and imaginary parts, respectively.*

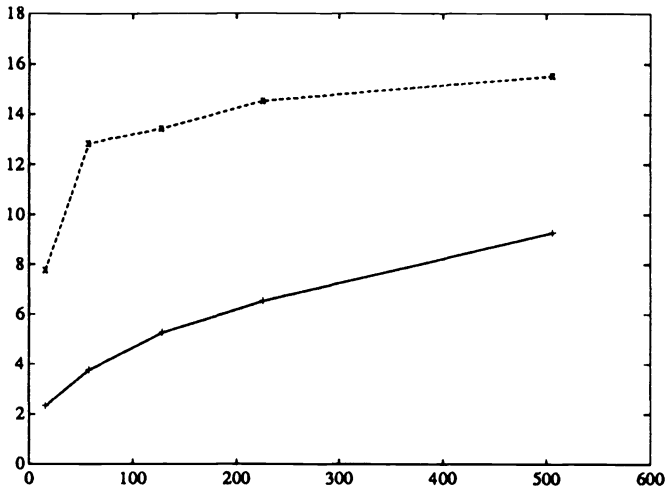


FIG. 6. *Solid line indicates* $\kappa_2(PA)$ *plotted against* n*; dashed line indicates* $\kappa_2(A)/\kappa_2(PA)$ *plotted against* n.

In this formula, Λ_0 denotes the spectrum of A, V denotes the matrix of eigenvectors of A, and $\kappa_2(V)$ denotes the condition number of V. Thus, this formula predicts better convergence when the exact eigenvalues are clustered away from the origin. As illustrated in the figure, the clustering is improved for PA. On the other hand, the factor $\kappa_2(V)$ is worse for PA: for this example, the condition numbers of the eigenvectors for A and PA are 6.19 and 328.37, respectively. Thus, the pseudoeigenvalue bound for the convergence of GMRES seems to be more predictive than the eigenvalue bound for this example.

A final experiment carried out was to compute $\kappa_2(PA)$ and $\kappa_2(A)/\kappa_2(PA)$ for the same shape domain (domain D_2) but varying numbers of nodes. Here, P is preconditioner MN. The results are illustrated in Fig. 6. We notice that the factor improvement in condition number increases as n increases, but so does the condition number of $\kappa_2(PA)$.

5. Analysis of a simple model problem. Let $\Omega \subset \mathbb{R}^2$ be the disk of radius ρ. We assume $\rho \neq 1$ (when $\rho = 1$, the method breaks down). We will try to solve Laplace's equation with Dirichlet boundary data on this region. We provide an asymptotic analysis of the condition number of the dense matrix and of preconditioner MN. We will argue that there is an asymptotic reduction of $\sqrt{\ln n}$ in the condition number. This reduction appears to be pessimistic compared to our three-dimensional computational experiments reported in the last section.

We define three operators on the space of real- or complex-valued functions on $\partial\Omega$, namely, **A**, **B**, and **I**. These operators can be defined for certain normed vector spaces, but the details of the particular function spaces are not important for the analysis in this section.

The three operators are as follows. Let the $\partial\Omega$ be parameterized by

$$\gamma(\theta) = \rho(\cos\theta, \sin\theta).$$

Let f be a complex-valued function on $\partial\Omega$. The function $\mathbf{A}(f)$ evaluated at point $\gamma(t)$ is defined to be

$$\mathbf{A}(f)(\gamma(t)) = -\frac{\rho}{4\pi}\int_0^{2\pi} \ln(\|\gamma(t)-\gamma(s)\|^2)\cdot f(\gamma(s))\,ds. \tag{3}$$

The kernel of this integral operator is the Green's function for a two-dimensional Laplace equation.

Let $\mathbf{B}$ be the following integral operator:

$$\mathbf{B}(f)(\gamma(t)) = \frac{\rho}{2\pi}\int_0^{2\pi} \frac{\langle \gamma(t)-\gamma(s), (\cos s, \sin s)\rangle}{\|\gamma(t)-\gamma(s)\|^2}\cdot f(\gamma(s))\,ds.$$

The kernel here is the normal derivative of Green's function. The notation $\langle\cdot,\cdot\rangle$ denote the inner product on $\mathbb{R}^2$.

Finally, $\mathbf{I}$ denotes the identity operator. Then the following result holds if f is the restriction to $\partial\Omega$ of a solution to Laplace's equation on Ω:

$$(\mathbf{I}+\mathbf{B})(f) = \mathbf{A}\left(\frac{\partial f}{\partial n}\right). \tag{4}$$

Here, $\partial f/\partial n$ denotes the derivative of the function extended to Ω. This identity is the basis for the boundary integral equation method in two dimensions.

In the version of the the boundary element equation method described earlier, functions on $\partial\Omega$ are approximated as a piecewise linear function with n equally spaced breakpoints around $\partial\Omega$. We assume the breakpoints are $v_0,\cdots,v_{n-1}$, where

$$v_j = \gamma(2\pi j/n).$$

A matrix A is created as a discretized representation of $\mathbf{A}$. The discretization procedure is as follows. Operator $\mathbf{A}$ is applied to the basis of piecewise linear "hat" functions. Denote this basis $b_0,\cdots,b_{n-1}$. These basis functions have the property that $b_k(\gamma(\theta))$ is a piecewise linear function of θ. In addition, the b_k's are continuous and satisfy

$$b_k(v_j) = \delta_{jk},$$

where the right-hand side is the Kronecker δ notation. The (j,k) entry of matrix A is then defined to be the function $\mathbf{A}(b_k)$ evaluated at v_j. In general, the entries of A are obtained by numerical integration and so would not be exactly equal to $\mathbf{A}(b_k)(v_j)$, but we assume that a sufficiently high-order quadrature method is used that the truncation error can be ignored.

We can conclude that A will be circulant and symmetric. It is circulant because $\mathbf{A}(b_k)(v_j)$ is the same as $\mathbf{A}(b_{k+1})(v_{j+1})$ (where b_n, v_n are identified with b_0, v_0); this follows from (3). The symmetry of A follows because $\mathbf{A}(b_k)(v_j) = \mathbf{A}(b_j)(v_k)$.

In the boundary element method for Ω given Dirichlet data, we have to solve a system involving matrix A. Accordingly, the goal of this section is to determine how the preconditioner affects the condition number of A.

Since A is circulant, the eigenvectors of A are the discrete Fourier vectors, that is, vectors $\mathbf{w}_0,\cdots,\mathbf{w}_{n-1}$, where the kth component of $\mathbf{w}_j$ is $\exp(2\pi ijk/n)$. We number the components of $\mathbf{w}_j$ with subscripts $0,\cdots,n-1$.

We next attempt to determine the eigenvalue of A corresponding to $\mathbf{w}_j$. Let ψ_j be the piecewise linear function on $\partial\Omega$ corresponding to $\mathbf{w}_j$, that is,

$$\psi_j = \sum_{k=0}^{n-1} w_{jk} b_k.$$

The function $\psi_j(\gamma(\theta))$ is a piecewise linear approximation to the Fourier function $\exp(ij\theta)$ defined on $\partial\Omega$ and has been studied in the literature. In particular, Chandler and Sloan [7] have provided a Fourier series expansion for ψ_j for $j \neq 0$:

$$\psi_j(\gamma(\theta)) = a_j \sum_{m \equiv j} \frac{\sin(\pi m/n)^2 \exp(im\theta)}{(\pi m/n)^2}.$$

Here, the sum is over integers m congruent to j mod n. The number a_j is a scaling constant. Their results also cover higher-order splines.

Note that the eigenvalue for $\mathbf{w}_j$ is equal to the ratio of $\mathbf{A}(\psi_j)$ evaluated at a collocation point divided by ψ_j evaluated at the same collocation point (provided ψ_j is not zero at the collocation point). Since $\mathbf{A}$ is linear, we have:

$$\mathbf{A}(\psi_j) = a_j \sum_{m \equiv j} \frac{\sin(\pi m/n)^2 \mathbf{A}(e_m)}{(\pi m/n)^2},$$

where e_m is the Fourier function $\exp(im\theta)$ defined on the circle of radius ρ.

Thus, the next task is to evaluate $\mathbf{A}(e_m)$. We first treat the case in which $m \neq 0$. We notice that

$$\mathbf{A}(e_m)(\gamma(t)) = -\frac{\rho}{4\pi} \int_0^{2\pi} \ln(\|\gamma(t) - \gamma(s)\|^2) \cdot \exp(ims)\, ds. \tag{5}$$

This integral appears to be difficult to analyze directly, so we instead approach it via identity (4). Focus on the case in which m is a positive integer. We notice that $\exp(ims)$ on the circle of radius ρ can be extended to all of $\mathbb{R}^2$ as the complex analytic function $f(z) = z^m/\rho^m$. Here, we are identifying $\mathbb{R}^2$ with $\mathbb{C}$. Notice also that f, since it is analytic, is a Laplace solution. The normal derivative of f is given by $\partial f/\partial n = me_m/\rho$ for points on the circle of radius ρ. Thus, from (4), we have

$$\mathbf{A}(e_m) = (\rho/m)(\mathbf{I} + \mathbf{B})(e_m).$$

Let us now evaluate the right-hand side:

$$\begin{aligned}
\mathbf{B}(e_m)(\gamma(t)) &= \frac{\rho}{2\pi} \int_0^{2\pi} \frac{\rho(\cos t - \cos s)\cos s + \rho(\sin t - \sin s)\sin s}{\rho^2(\cos t - \cos s)^2 + \rho^2(\sin t - \sin s)^2} \cdot e^{ims}\, ds \\
&= \frac{1}{2\pi} \int_0^{2\pi} \frac{\cos(s-t) - 1}{2 - 2\cos(s-t)} \cdot e^{ims}\, ds \\
&= \frac{1}{2\pi} \int_0^{2\pi} \left(-\tfrac{1}{2}\right) e^{ims}\, ds \\
&= 0
\end{aligned}$$

as long as $m \neq 0$. This is because the real and imaginary parts of e^{ims} go through an integer number of periods over the range of integration. Thus, for positive integers m, we have $\mathbf{A}(e_m) = (\rho/m)\mathbf{I}(e_m)$, i.e.,

$$\mathbf{A}(e_m) = (\rho/m)e_m. \tag{6}$$

The preceding analysis fails for negative integers because z^m is not analytic at zero if $m < 0$. The eigenvalue of e_m for $m < 0$ is seen to be $|m|$; this follows by taking the complex conjugate of (6).

Thus, we conclude for $m \neq 0$ that $\mathbf{A}(e_m) = (\rho/|m|)e_m$. This fact appears also in [7] and is well known in the literature.

The last case to analyze is $m = 0$:

$$\begin{aligned}\mathbf{A}(e_0)(\gamma(t)) &= -\frac{\rho}{4\pi}\int_0^{2\pi} \ln(\rho^2(2-2\cos(s-t)))\,ds \\ &= -\frac{\rho}{4\pi}\int_0^{2\pi} \ln(\rho^2)\,ds - \frac{\rho}{4\pi}\int_0^{2\pi} \ln(2-2\cos s)\,ds \\ &= -\rho\ln(\rho^2)/2.\end{aligned}$$

The second integral on the right-hand side of the second line vanishes, as can be verified with Fourier expansion. Notice that the answer is a constant (independent of t).

Now, we return to the analysis of matrix A. We see now that

$$\mathbf{A}(\psi_j)(\gamma(t)) = a_j \sum_{m\equiv j} \frac{\sin(\pi m/n)^2 e^{imt}\rho}{|m|\cdot(\pi m/n)^2}$$

for $j = 1, \cdots, n-1$. We return to the case when $j = 0$ below.

Recall that the eigenvalue of A corresponding to $\mathbf{w}_j$ is equal to the ratio

$$\mathbf{A}(\psi_j)(v_0)/\psi_j(v_0)$$

provided that $\psi_j(v_0) \neq 0$. The numerator $\mathbf{A}(\psi_j)(v_0)$ is found using the formula in the last paragraph:

$$a_j \sum_{m\equiv j} \frac{\sin(\pi m/n)^2\rho}{|m|\cdot(\pi m/n)^2},$$

which simplifies to

$$\frac{a_j \sin(\pi j/n)^2 \rho n^2}{\pi^2} \sum_{m\equiv j}(1/|m|^3). \tag{7}$$

The preceding summation can be analyzed by splitting it into positive and negative indices, and dividing each summation by its leading term. The result is

$$\sum_{m\equiv j} \frac{1}{|m|^3} = \tau_j\left(\frac{1}{j^3} + \frac{1}{(n-j)^3}\right),$$

where τ_j lies between 1 and 1.3.

The formula for $\psi_j(v_0)$ is

$$a_j \sum_{m\equiv j} \frac{\sin(\pi m/n)^2}{(\pi m/n)^2}.$$

This simplifies to

$$\frac{a_j \sin(\pi j/n)^2 n^2}{\pi^2} \sum_{m\equiv j}(1/m^2). \tag{8}$$

TABLE 4
Condition numbers of A for a disk with $\rho = 2$.

n	$\kappa_2(A)$
10	8.89
20	17.69
40	35.31
80	70.54

The summation in the previous expression can be written

$$\tau_j' \left(\frac{1}{j^2} + \frac{1}{(n-j)^2} \right),$$

where τ_j' lies between 1 and 1.7.

Thus, dividing (7) by (8) gives the eigenvalue of $\mathbf{w}_j$. Simplifying gives

$$A\mathbf{w}_j = \frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1 - 3z + 3z^2}{z(1-z)(1-2z+2z^2)} \mathbf{w}_j, \tag{9}$$

where $z = j/n$.

We next determine the eigenvalue of A for $\mathbf{w}_0$. Notice that $\mathbf{w}_0$ is the vector of all 1s. The value of $A\mathbf{w}_0$ is the collocated values of $\mathbf{A}(\psi_0)$, which is the same as $\mathbf{A}(e_0)$. We showed earlier that $\mathbf{A}(e_0)$ is the constant function with value $-\rho \ln(\rho^2)/2$. Thus we see that the eigenvalue of A for $\mathbf{w}_0$ is independent of n and equal to this constant.

We now have enough information to determine the asymptotic condition number of A. Since A is symmetric, its condition number is the ratio of the extreme magnitudes of its eigenvalues. The second fraction in (9) may be written as

$$\frac{1 - 3y}{y(1-2y)}, \tag{10}$$

where $y = z(1-z)$. Formula (10) is seen to be positive and decreasing for $0 < y < 0.25$ (since z is between zero and 1, y must be between zero and 0.25). Thus (10) is maximized when y is small, i.e., when z is close to zero or 1, i.e., when j is 1 or $n-1$. In this case the second fraction of (9) behaves asymptotically like n for large n. The first fraction behaves like $1/n$. Thus, the largest eigenvalue is for $\mathbf{w}_1$ or $\mathbf{w}_{n-1}$ and has value that is a constant. The smallest eigenvalue occurs when y in (10) is largest, i.e., when z is close to $\frac{1}{2}$, i.e., when j is close to $n/2$. Assume n is even so that $n/2$ is an integer. Then we see that the second fraction of (9) is 2, so the eigenvalue is proportional to $1/n$.

Thus, we see that the largest eigenvalue of A is at $\mathbf{w}_0$ or $\mathbf{w}_1$ and is asymptotically bounded above and below by constants independent of n. The smallest eigenvalue of A behaves like $1/n$. Thus, the condition number of A is expected to be $O(n)$ asymptotically. In fact, this condition number result was established more generally by Richter [19] in the context of Galerkin-type discretizations.

These assertions are borne out by our computational experience; Table 4 indicates the condition number of A in the case $\rho = 2$ for varying values of n.

Next, we turn to preconditioner MN. Let us call the preconditioning matrix P. In the problem on the disk, each node has two nearest neighbors; therefore, each row of P will have three nonzero entries. Each row is determined by solving a 3×3 linear system.

This linear system will be the same for each node $v_0, \cdots, v_{n-1}$ by symmetry, so P will also be circulant with three nonzero bands clustered around the diagonal. In addition, the 3×3 system is symmetric since A is symmetric.

Let the 3×3 system be denoted by $\bar{A}$; it is the principal submatrix of A formed by taking three consecutive rows and columns of A. Thus, it has the form

$$\bar{A} = \begin{pmatrix} \mu & \nu & \xi \\ \nu & \mu & \nu \\ \xi & \nu & \mu \end{pmatrix}.$$

The next task is to determine the values of μ, ν, and ξ. This is equivalent to determining formulas for the entries of A in positions $(1,1)$, $(1,2)$, and $(1,3)$. Let $f(z) = \mathbf{A}(b_0)(z)$; then our problem is to determine $f(v_0), f(v_1), f(v_2)$ (recall that b_0 denotes the piecewise linear hat function nonzero at v_0, and v_0, v_1, v_2 are breakpoints).

We have the following calculation:

$$\begin{aligned} \mathbf{A}(b_0)(v_p) &= -\frac{\rho}{4\pi} \int_{-\pi}^{\pi} \ln(\|v_p - \gamma(s)\|^2) \cdot b_0(\gamma(s))\, ds \\ &= -\frac{\rho}{4\pi} \int_{-2\pi/n}^{2\pi/n} \ln\left(\rho \,\|(\cos(2\pi p/n), \sin(2\pi p/n)) - (\cos s, \sin s)\|\right)^2 \\ &\quad \times b_0(\gamma(s))\, ds. \end{aligned}$$

In this formula, $p = 0, 1, 2$. The second line follows from the fact that $b_0(\gamma(s))$ is nonzero only for $s \in [-2\pi/n, 2\pi/n]$.

Next, notice that

$$\|(\cos(2\pi p/n), \sin(2\pi p/n)) - (\cos s, \sin s)\| = |2\pi p/n - s| \cdot \chi_p(s),$$

where $\chi_p(s) \in [1 - O(1/n^3), 1 + O(1/n^3)]$; this follows because $(\cos(2\pi p/n), \sin(2\pi p/n))$ and $(\cos s, \sin s)$ are two nearby points on the unit circle (recall $p = 0, 1, 2$). Thus we have

$$\begin{aligned} \mathbf{A}(b_0)(v_p) &= -\frac{\rho}{4\pi} \int_{-2\pi/n}^{2\pi/n} \ln\left(\rho \chi_p(s) \cdot |2\pi p/n - s|\right)^2 \cdot b_0(\gamma(s))\, ds \\ &= -\frac{\rho}{2\pi} \int_{-2\pi/n}^{2\pi/n} (\ln \rho + \ln \chi_p(s) + \ln |2\pi p/n - s|) \cdot b_0(\gamma(s))\, ds \\ &= -\frac{\rho}{2\pi}(i_1 + i_2). \end{aligned}$$

Here,

$$\begin{aligned} i_1 &= \int_{-2\pi/n}^{2\pi/n} (\ln \rho + \ln \chi_p(s)) b_0(\gamma(s))\, ds \\ &= (\ln \rho + O(1/n^3)) \frac{2\pi}{n}. \end{aligned}$$

The other term is

$$i_2 = \int_{-2\pi/n}^{2\pi/n} \ln |2\pi p/n - s| \cdot b_0(\gamma(s))\, ds.$$

The integral in the previous equation is the product of a logarithm and a piecewise linear function, which can be integrated analytically. The result is that $i_2 = \delta \ln \delta + c_p \delta$, where $\delta = 2\pi/n$ and $c_0 = -1.5$, $c_1 \approx -0.1137$, and $c_2 \approx 0.6712$.

Thus

$$\mathbf{A}(b_0)(v_p) = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_p + O(1/n^3)).$$

This gives us a formula for the entries of $\bar{A}$. In particular,

$$\mu = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_0 + O(1/n^3)),$$
$$\nu = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_1 + O(1/n^3)),$$
$$\xi = -\frac{\rho\delta}{2\pi}(\ln\rho + \ln\delta + c_2 + O(1/n^3)).$$

The linear system we solve to recover the entries of a row of P is $\bar{A}\mathbf{x} = \mathbf{e}_2$, where $\mathbf{e}_2 = (0,1,0)^T$. We solve this with Cramer's rule. Let d be equal to $1/\det(\bar{A})$; then the solution to $\bar{A}\mathbf{x} = \mathbf{e}_2$ is $(x_1, x_2, x_3)^T$, where

$$x_1 = d(\xi\nu - \mu\nu),$$
$$x_2 = d(\mu^2 - \xi^2),$$
$$x_3 = d(\xi\nu - \mu\nu).$$

We can simplify these expressions; let $d' = (\rho\delta/(2\pi))^2$ and $d'' = \ln\rho + \ln\delta$; then

$$x_1 = x_3 = dd'(d''(c_2 - c_0) + c_1(c_2 - c_0) + d''O(1/n^3))$$

and

$$x_2 = dd'(2d''(c_0 - c_2) + c_0^2 - c_2^2 + d''O(1/n^3)).$$

This gives the entries of P. Notice that

$$x_2/x_1 = -2 + \frac{-c_0 - c_2 + 2c_1 + d''O(1/n^3)}{d'' + c_1 + d''O(1/n^3)}.$$

Recall that $d'' = \ln(\delta) + O(1) = -\ln(n) + O(1)$. Notice also that $-c_0 - c_2 + 2c_1$ is a positive constant approximately 0.6014; call this constant $d^{(3)}$. Thus,

$$x_2/x_1 = -2 - \frac{d^{(3)} + O(\ln(n)/n^3)}{\ln(n) + O(1)}$$
$$= -2 - \frac{d^{(3)}}{\ln n} + O((\ln n)^{-2}).$$

Thus, P/x_1 is a circulant symmetric matrix with $-2 - d^{(3)}/(\ln n) + O((\ln n)^{-2})$ on the diagonal and 1 on the positions adjacent to the diagonal (and in the $(n,1)$ and $(1,n)$ positions).

The next task is to compute the condition number of PA. This is the same as the condition number of PA/x_1, so we work with PA/x_1 instead. Since P is circulant and symmetric, it has $\mathbf{w}_0, \cdots, \mathbf{w}_{n-1}$ as eigenvectors. Therefore, PA/x_1 is also circulant and symmetric and also has $\mathbf{w}_0, \cdots, \mathbf{w}_{n-1}$ as eigenvectors. Accordingly, we now attempt to compute the eigenvalue of $\mathbf{w}_j$ in PA/x_1. This is the same as the product of the eigenvalue of $\mathbf{w}_j$ in P/x_1 multiplied by the eigenvalue in A.

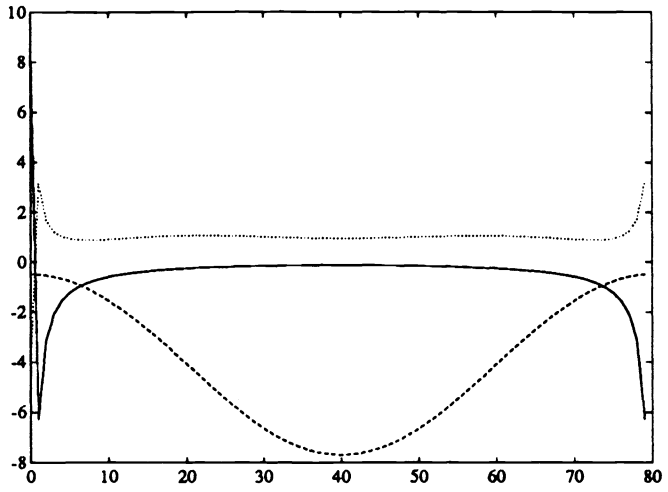


FIG. 7. *The eigenvalues of A, P, and PA plotted against frequency.*

The kth component of $P\mathbf{w}_j/x_1$ (counting from zero) is equal to

$$\begin{aligned}(P\mathbf{w}_j/x_1)_k &= e^{2\pi ij(k-1)/n} + (-2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n} + e^{2\pi ij(k+1)/n} \\ &= (e^{2\pi ij/n} + e^{-2\pi ij/n} - 2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n} \\ &= (2\cos(2\pi j/n) - 2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n} \\ &= (-4\sin^2(\pi j/n) - d^{(3)}/(\ln n) + O((\ln n)^{-2}))e^{2\pi ijk/n}.\end{aligned}$$

Thus, the eigenvalue of P/x_1 for $\mathbf{w}_j$ is

$$-4\sin^2(\pi j/n) - d^{(3)}/(\ln n) + O((\ln n)^{-2}).$$

We can rewrite the first term; in particular, there exists a number τ_j'' lying between π and 4 such that

$$\sin(\pi j/n) = \tau_j''(j/n)(1 - j/n)$$

for $j = 0, \cdots, n-1$. Thus the eigenvalue can be written

$$-4\tau_j''^2 z^2(1-z)^2 - d^{(3)}/(\ln n) + O((\ln n)^{-2}), \tag{11}$$

where $z = j/n$.

Figure 7 shows a plot of the magnitude of the eigenvalue versus j produced by MATLAB. There are three sets of eigenvalues in the plot. The solid line is the eigenvalue in A; the dashed line is the eigenvalue in P; and the dotted line is the eigenvalue in PA. Notice how the eigenvalues of P are of large magnitude roughly in the places where the eigenvalues of A are small. This is the desired feature of the preconditioner. Note that these plots include some scaling factors not described above.

Continuing our analysis, the eigenvalue of $\mathbf{w}_j$ for PA/x_1 for $1 \le j \le n-1$ is given by the product of the eigenvalues in (9) and (11). This is

$$\lambda_j = \frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1 - 3z + 3z^2}{z(1-z)(1-2z+2z^2)} \cdot \left(-4\tau_j''^2 z^2(1-z)^2 - \frac{d^{(3)}}{\ln n} + O((\ln n)^{-2})\right),$$

where $z = j/n$. As above, we can substitute $y = z(1-z)$. Also, let $q_n = d^{(3)}/(\ln n) + O((\ln n)^{-2})$. We obtain

$$\lambda_j = \frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1-3y}{y(1-2y)} \cdot \left(-4\tau_j''^2 y^2 - q_n\right). \tag{12}$$

For n large enough, q_n is positive so the whole expression is negative. Therefore, it suffices to estimate the largest and smallest eigenvalues in order to estimate the condition number. We break (12) into two terms; the first term is

$$\frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1-3y}{1-2y} \cdot (-4\tau_j''^2 y).$$

This number is seen to lie between $\sigma_1 y/n$ and $\sigma_2 y/n$, where σ_1, σ_2 are two negative constants, provided y is between zero and 0.25 (so that $(1-3y)/(1-2y)$ is between $\frac{1}{2}$ and 1).

The second term is

$$\frac{\rho\tau_j}{n\tau_j'} \cdot \frac{1-3y}{y(1-2y)} \cdot (-q_n).$$

Reasoning as in the previous paragraph, we conclude that this lies between $\sigma_3 q_n/(ny)$ and $\sigma_4 q_n/(ny)$ where σ_3, σ_4 are negative constants.

Thus, it suffices to establish upper and lower bounds on functions of the form

$$\sigma' y/n + \sigma'' q_n/(ny), \tag{13}$$

where σ', σ'' can lie between fixed negative bounds. Assume a fixed choice of σ', σ''. Formula (13) is a concave function of y. This means its lower bound is achieved at one of the endpoints. The first endpoint occurs when y is as small as possible, i.e., $j = 1$ or $j = (n-1)$, giving $z = 1/n$, and $z = (n-1)/n$, giving $y = (n-1)/n^2$, which is asymptotically like $1/n$. Plugging $1/n$ into (13) gives an eigenvalue asymptotically like $\sigma'' q_n$ for large n. The other endpoint occurs when y is as large as possible, i.e., $j = n/2$, in which case $z = \frac{1}{2}$ and $y = \frac{1}{4}$. Plugging this in gives an eigenvalue asymptotically like $\sigma'/(4n)$, which is larger than the eigenvalue at 1. (Recall σ', σ'' are negative and $q_n = O(1/(\ln n))$.) Thus, the smallest eigenvalue has the form $\sigma'' q_n$.

The largest value of (13) occurs where the derivative with respect to y is zero. Thus, the largest eigenvalue comes at

$$y_0 = \sqrt{\frac{\sigma'' q_n}{\sigma'}}.$$

We remark that for n large enough, y_0 will lie between zero and 0.25 and hence is achieved asymptotically for some value of j. At y_0, formula (13) has the value $-2\sqrt{\sigma'\sigma'' q_n}/n$. Therefore, this is the largest eigenvalue of PA/x_1 up to the choice of σ', σ''.

This means that the eigenvalues run from $O(1/(\ln n))$ down to $O(1/(n\sqrt{\ln n}))$ in absolute value. We have not yet addressed the eigenvalue for $\mathbf{w}_0$. This turns out to be $O(1/(\ln n))$ also.

Thus we obtain the condition number of PA/x_1 by dividing the largest eigenvalue in absolute value by the smallest. This yields a condition number of $O(n/\sqrt{\ln n})$.

Since the condition number of A was $O(n)$, this concludes our proof that for a simple model problem, the condition number of A is reduced by $O(\sqrt{\ln n})$, a factor growing faster than any constant.

6. Complexity issues. Suppose we could establish a constant upper bound on the number of iterations required by a preconditioned iterative method for BIEM. This would imply a running time of $O(n^2)$ to solve the dense system of equations, clearly an improvement over the running time $O(n^3)$ of Gaussian elimination.

In fact, there are several complexity results in this direction. Rokhlin gets an $O(n)$ running time for solving the linear system using multipole expansion. Brandt and Lubrecht [4] achieve an $O(n \log n)$ running time with multigrid. Earlier, Schippers [22] had achieved an $O(n^2)$ running time, also with multigrid. Bramble, Pasciak, and Xu [3] have proposed a multilevel preconditioner for boundary element methods.

It seems to us, however, that none of these methods apply to (1) with mixed boundary conditions, because solving (1) with mixed boundary conditions yields a system of equations that is a mixture of first- and second-kind integral equations. All of the above-mentioned methods assume that the integral equation is homogeneous.

Accordingly, it remains an interesting open problem to establish asymptotic complexity results on how long it takes to solve the BIEM system of linear equations arising from (1).

Acknowledgments. The author thanks Professor Kendall Atkinson for providing some of the background material for this paper and Professor L. N. Trefethen for explaining some properties of unsymmetric iterations.

REFERENCES

[1] K. E. Atkinson, *Piecewise polynomial collocation for integral equations on surfaces in three dimensions*, J. Integral Equations, 9 (1985), pp. 25–48.

[2] M. W. Benson and P. O. Frederickson, *Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems*, Utilitas Math., 22 (1982), pp. 127–140.

[3] J. H. Bramble, J. E. Pasciak, and J. Xu, private communication, 1991. See also *Parallel Multilevel Computation*, Math. Comp., 55 (1990), pp. 1–22, by the same authors.

[4] A. Brandt and A. A. Lubrecht, *Multilevel matrix multiplication and fast solution of integral equations*, J. Comput. Phys., 90 (1990), pp. 348–370.

[5] C. A. Brebbia (ed.), *Topics in Boundary Element Research*, Springer-Verlag, Berlin, 1987.

[6] F. X. Canning, *Sparse approximation for solving integral equations with oscillatory kernels*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 71–87.

[7] G. A. Chandler and I. H. Sloan, *Spline qualocation methods for boundary integral equations*, Numer. Math., 58 (1990), pp. 537–567.

[8] H. C. Elman, *A stability analysis of incomplete LU factorizations*, Math. Comp., 47 (1986), pp. 191–217.

[9] T. Feder and D. H. Greene, *Optimal algorithms for approximate clustering*, Proc. 20th Annual ACM Symposium Theory of Computing, Chicago, IL, 1988, pp. 434–444.

[10] G. H. Golub and C. F. Van Loan, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[11] L. Greengard, *The Rapid Evaluation of Potential Fields in Particle Systems*, MIT Press, Cambridge, MA, 1988.

[12] J. L. Hess and A. M. O. Smith, *Calculation of potential flow about arbitrary bodies*, in Progress in Aeronautical Sciences, Vol. 8, D. Küchemann, ed., Pergamon Press, London, 1967, pp. 1–138.

[13] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand., 49 (1952), pp. 409–436.

[14] M. A. Jaswon, *Integral equation methods in potential theory*, I, Proc. Royal Soc. Ser. A, 275 (1963), pp. 23–32.

[15] L. Yu. Kolotilina and A. Yu. Yeremin, *On a family of two-level preconditionings of the incomplete block factorization type*, Sov. J. Numer. Anal. Math. Model., 4 (1986), pp. 293–320.

[16] P. L.-F. LIU, H.-W. HSU, M. H. LEAN, AND S. A. VAVASIS, *A boundary element method for three-dimensional free surface flows*, Report X9100283, Webster Research Center, Xerox Corporation, North Tarrytown, NY, 1991.

[17] T. A. MANTEUFFEL, *Shifted incomplete Cholesky factorization,* in Sparse Matrix Proceedings, 1978, I. S. Duff and G. W. Stewart, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979, pp. 41–61.

[18] N. M. NACHTIGAL, S. C. REDDY, AND L. N. TREFETHEN, *How fast are nonsymmetric matrix iterations?* SIAM J. Matrix Anal. Appl., this issue, pp. 778–795.

[19] G. R. RICHTER, *Numerical solution of integral equations of the first kind with unsmooth kernels*, SIAM J. Numer. Anal., 15 (1978), pp. 511–522.

[20] V. ROKHLIN, *Rapid solution of integral equations of classical potential theory*, J. Comput. Phys., 60 (1985), pp. 187–207.

[21] Y. SAAD AND M. H. SCHULTZ, GMRES: *A generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.

[22] H. SCHIPPERS, *Theoretical and practical aspects of multigrid methods in boundary element calculations*, in Topics in Boundary Element Research, Vol. 3, Computational Aspects, C. A. Brebbia, ed., Springer-Verlag, Berlin, 1987.

[23] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.

[24] G. T. SYMM, *Integral equation methods potential theory*, II, Proc. Royal. Soc. Ser. A, 275 (1963), pp. 33–46.

[25] L. N. TREFETHEN, *Pseudospectra of matrices*, Report 91/10, Oxford University Computing Laboratory, Numerical Analysis Group, Oxford, England, August 1991.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 926–943, July 1992

014

LANCZOS METHODS FOR THE SOLUTION OF NONSYMMETRIC SYSTEMS OF LINEAR EQUATIONS*

WAYNE JOUBERT†

Abstract. The Lanczos or biconjugate gradient method is often an effective means for solving nonsymmetric systems of linear equations. However, the method sometimes experiences breakdown, a near division by zero which may hinder or preclude convergence. In this paper we present some theoretical results on the nature and likelihood of the phenomenon of breakdown. We also define several new algorithms that substantially mitigate the problem of breakdown. Numerical comparisons of the new algorithms and the standard algorithms are given.

Key words. linear systems, iterative methods, nonsymmetric, Lanczos

AMS(MOS) subject classifications. 65F10, 65F15

1. Introduction.

In this paper we consider methods for solving the linear system of equations

$$Au = b, \tag{1}$$

where $A \in \mathbb{C}^{N \times N}$ is a given nonsingular matrix.

When A is large and sparse, iterative methods in many cases are effective means for solving (1). In particular, when A is Hermitian and positive definite (HPD), the conjugate gradient (CG) method [21] is an effective solution technique for (1).

However, the case when A is nonsymmetric is substantially more difficult to solve efficiently by means of iterative methods. For example, the important CG method cannot be generalized to the nonsymmetric case without a serious loss of some of its more useful properties (see [9], [10], and [27]). This difficulty has led to the development of a wide variety of generalized CG methods having varying degrees of success (for an overview, see, for example, [42], [2], or [26]; see also [25]).

The biconjugate gradient or Lanczos method [29], [11], [41] is an important example of a generalized CG method. In many cases, Lanczos algorithms give some of the fastest solution times among all generalized CG methods (see, e.g., [30] and [1]).

However, the Lanczos method is known to break down in some cases. In practice, the occurrence of a breakdown or near-breakdown of the method can cause failure to converge to the solution of (1). Furthermore, the size of the iterates generated by the Lanczos method may become arbitrarily large during the iteration process, which can introduce numerical error into the approximate solution.

Comparatively little is known about the theoretical properties of the Lanczos method (see, e.g., [7]). The fact that Lanczos algorithms perform very well in some cases but fail in others heightens the need for further insight into the theoretical properties of the Lanczos method.

In this paper we present theoretical results on the Lanczos method as well as new algorithms that are better able to deal with the problem of breakdown of the Lanczos

*Received by the editors October 8, 1990; accepted for publication March 7, 1991. This work was supported in part by National Science Foundation grant DCR-8518722, by Department of Energy grant DE-FG05-87ER25048, and by Cray Research Inc. grant LTRDTD 1/18/90, with the University of Texas at Austin.

†The University of Texas at Austin, Austin, Texas 78713 (joubert@emx.utexas.edu).

method. This will be organized as follows. In §2 we define the Lanczos method and its algorithms, and in §3 we examine the conditions that lead to breakdown of the algorithms. Then in §§4 and 5 we give results on the likelihood of breakdown of the Lanczos algorithms, and in §6 we analyze the important categories of curable and incurable breakdown. After this, in §7 we define modified Lanczos algorithms, and the results of numerical experiments with these algorithms are presented in §8.

2. The Lanczos method. The Lanczos method is a particular instance of an iterative method, which is defined as a procedure that for a given initial guess $u^{(0)}$ may be used to compute subsequent iterates $\{u^{(i)}\}_{i\geq 1}$, which approximate the true solution $u = A^{-1}b$. We denote the corresponding residuals by $r^{(n)} = b - Au^{(n)}$, and the error vector is denoted by $e^{(n)} = u^{(n)} - u = -A^{-1}r^{(n)}$.

Specifically, the Lanczos method is defined by the following two properties:

$$u^{(n)} - u^{(0)} \in \mathbf{K}_n(r^{(0)}, A), \qquad r^{(n)} \perp \mathbf{K}_n(\tilde{r}^{(0)}, A^*). \tag{2}$$

Here, the Krylov space is defined as $\mathbf{K}_n(v, A) = \operatorname{span}\{A^i v\}_{i=0}^{n-1}$. The vector $\tilde{r}^{(0)}$ is an auxiliary vector supplied to the algorithm, typically defined by $\tilde{r}^{(0)} = \tilde{Z}^* r^{(0)}$ for some matrix $\tilde{Z}$ which is commonly set to $\tilde{Z} = I$. Here we use the notation X^* to denote the complex conjugate of X when the quantity X is a scalar and the conjugate transpose when X is a vector or matrix. We also define the standard inner product $(u, v) = u^* v$.

The first condition of (2) indicates that the method is a *polynomial method* in the matrix A. The second condition of (2), which is the orthogonality or *Petrov–Galerkin* condition, categorizes the method as an example of a *projection method* (see [26]). Unfortunately, in general there is no guarantee that the two conditions of (2) necessarily define a unique iterate $u^{(n)}$.

For certain choices of A and $\tilde{Z}$, the method (2) reduces to a standard conjugate gradient method (see [23] and [26]). For example, if A is HPD and $\tilde{Z} = I$, then (2) reduces to the conjugate gradient method of [21]; if A is HPD and $\tilde{Z} = A$, then (2) gives the standard conjugate residual method (see, e.g., [11] and [2]).

The abstract method (2) may be implemented by various algorithms. Three examples of such Lanczos algorithms are the Lanczos/Orthodir, Lanczos/Orthomin, and Lanczos/Orthores algorithms [23]. Of these, the Lanczos/Orthomin version is most commonly used and is also referred to as the biconjugate gradient (BCG) algorithm.

LANCZOS/ORTHODIR ALGORITHM

$$q^{(0)} = r^{(0)}; \qquad q^{(n)} = Aq^{(n-1)} - a_n q^{(n-1)} - b_n q^{(n-2)}, \quad n > 0,$$

$$\tilde{q}^{(0)} = \tilde{r}^{(0)}; \qquad \tilde{q}^{(n)} = A^*\tilde{q}^{(n-1)} - a_n^*\tilde{q}^{(n-1)} - b_n^*\tilde{q}^{(n-2)}, \quad n > 0,$$

$$a_n = \frac{(A^*\tilde{q}^{(n-1)}, Aq^{(n-1)})}{(\tilde{q}^{(n-1)}, Aq^{(n-1)})}, \qquad b_n = \frac{(\tilde{q}^{(n-1)}, Aq^{(n-1)})}{(\tilde{q}^{(n-2)}, Aq^{(n-2)})} \quad (b_1 = 0),$$

$$u^{(n+1)} = u^{(n)} + \hat{\lambda}_n q^{(n)}, \qquad \hat{\lambda}_n = \frac{(\tilde{q}^{(n)}, r^{(n)})}{(\tilde{q}^{(n)}, Aq^{(n)})},$$

$$r^{(n+1)} = r^{(n)} - \hat{\lambda}_n Aq^{(n)}, \qquad \tilde{r}^{(n+1)} = \tilde{r}^{(n)} - \hat{\lambda}_n^* A^*\tilde{q}^{(n)}.$$

LANCZOS/ORTHOMIN ALGORITHM

$$p^{(0)} = r^{(0)}; \qquad p^{(n)} = r^{(n)} + \alpha_n p^{(n-1)}, \quad n > 0,$$

$$\tilde{p}^{(0)} = \tilde{r}^{(0)}; \qquad \tilde{p}^{(n)} = \tilde{r}^{(n)} + \alpha_n^* \tilde{p}^{(n-1)}, \quad n > 0,$$

$$\alpha_n = \frac{(\tilde{r}^{(n)}, r^{(n)})}{(\tilde{r}^{(n-1)}, r^{(n-1)})},$$

$$u^{(n+1)} = u^{(n)} + \lambda_n p^{(n)}, \qquad \lambda_n = \frac{(\tilde{r}^{(n)}, r^{(n)})}{(\tilde{p}^{(n)}, Ap^{(n)})},$$

$$r^{(n+1)} = r^{(n)} - \lambda_n A p^{(n)}, \qquad \tilde{r}^{(n+1)} = \tilde{r}^{(n)} - \lambda_n^* A^* \tilde{p}^{(n)}.$$

LANCZOS/ORTHORES ALGORITHM

$$u^{(n+1)} = \lambda_n (r^{(n)} + \sigma_{n+1,n} u^{(n)} + \sigma_{n+1,n-1} u^{(n-1)}),$$

$$r^{(n+1)} = -\lambda_n (A r^{(n)} - \sigma_{n+1,n} r^{(n)} - \sigma_{n+1,n-1} r^{(n-1)}),$$

$$\tilde{r}^{(n+1)} = -\lambda_n^* (A^* \tilde{r}^{(n)} - \sigma_{n+1,n}^* \tilde{r}^{(n)} - \sigma_{n+1,n-1}^* \tilde{r}^{(n-1)}),$$

$$\lambda_n = [\sigma_{n+1,n} + \sigma_{n+1,n-1}]^{-1},$$

$$\sigma_{n+1,n} = \frac{(\tilde{r}^{(n)}, A r^{(n)})}{(\tilde{r}^{(n)}, r^{(n)})}, \qquad \sigma_{n+1,n-1} = -\frac{1}{\lambda_{n-1}} \frac{(\tilde{r}^{(n)}, r^{(n)})}{(\tilde{r}^{(n-1)}, r^{(n-1)})}.$$

We will say that an algorithm *breaks down* at a step n if $u^{(n-1)} \neq u$ has been successfully computed by the algorithm but $u^{(n)}$ cannot be computed by the algorithm due to some condition such as division by zero.

In the absence of breakdown of the given algorithm, the above three algorithms are guaranteed to yield the iterates defined by (2), and furthermore, if the algorithm does not break down we have exact convergence $u^{(n)} = u$ if and only if $n = d(r^{(0)}, A)$. Here we define the *degree of a vector* v by

$$d(v, A) = \max\{d : \{A^i v\}_{i=0}^{d-1} \text{ linearly independent}\}.$$

When A is diagonalizable, the quantity $d(v, A)$ is the number of eigenvectors of A represented in v.

As noted above, for certain choices of A and $\tilde{Z}$ the method (2) reduces to a standard CG method. In such cases the above three algorithms above reduce to standard CG algorithms. In these cases, breakdown is known to be impossible (see [2]). However, in more general situations, the above three algorithms may indeed break down; for examples of this, see [24].

3. Breakdown of Lanczos algorithms. In this section we give conditions that characterize the situations in which each of the above three Lanczos algorithms experience breakdown.

For theoretical reasons, it is desirable to find characterizations of the conditions of breakdown of the algorithms that are based on the key *spaces* $\mathbf{K}_n(r^{(0)}, A)$ and $\mathbf{K}_n(\tilde{r}^{(0)}, A^*)$ rather than the *formulas* for the algorithms. In particular, we will characterize breakdown of the three Lanczos algorithms in terms of the *moment matrices*

$K_n(\tilde{r}^{(0)}, A^*)^* K_n(r^{(0)}, A)$ and $K_n(\tilde{r}^{(0)}, A^*)^* A K_n(r^{(0)}, A)$. Here we define the matrix $K_n(v, A) = [v \quad Av \quad \cdots \quad A^{n-1}v]$, a matrix whose columns span the Krylov space $\mathbf{K}_n(v, A)$.

The following three theorems give exact conditions for breakdown of the above algorithms. Detailed proofs may be found in [24]. A result similar to Theorem 2 is found in [11]; see also [41].

THEOREM 1 (Lanczos/Orthodir Breakdown). *Suppose Lanczos/Orthodir has successfully generated $u^{(n-1)} \neq u$. Then the following are equivalent:*

- *The algorithm does not break down at step n.*
- *The matrix $K_n(\tilde{r}^{(0)}, A^*)^* A K_n(r^{(0)}, A)$ is nonsingular.*
- *There exists a unique iterate $u^{(n)}$ satisfying (2).*

THEOREM 2 (Lanczos/Orthomin Breakdown). *Suppose Lanczos/Orthomin has successfully generated $u^{(n-1)} \neq u$. Then the following are equivalent:*

- *The algorithm breaks down at step n.*
- *Either $K_{n-1}(\tilde{r}^{(0)}, A^*)^* K_{n-1}(r^{(0)}, A)$ or $K_n(\tilde{r}^{(0)}, A^*)^* A K_n(r^{(0)}, A)$ is singular.*

THEOREM 3 (Lanczos/Orthores Breakdown). *Suppose Lanczos/Orthores has successfully generated $u^{(n-1)} \neq u$. Then the following are equivalent:*

- *The algorithm breaks down at step n.*
- *Either $K_n(\tilde{r}^{(0)}, A^*)^* K_n(r^{(0)}, A)$ or $K_n(\tilde{r}^{(0)}, A^*)^* A K_n(r^{(0)}, A)$ is singular.*

Thus we see that the Orthodir variant of Lanczos breaks down at a step precisely when (2) fails to define a unique iterate, or equivalently when the moment matrix $K_n(\tilde{r}^{(0)}, A^*)^* A K_n(r^{(0)}, A)$ is singular for these choices of n. On the other hand, the Orthomin and Orthores variants break down under further circumstances involving the moment matrix $K_n(\tilde{r}^{(0)}, A^*)^* K_n(r^{(0)}, A)$. Thus the Orthomin variant breaks down if and only if the Orthores variant breaks down, and if neither algorithm breaks down then the Orthodir variant does not break down.

These observations lead us to make the following definitions. We say that a (*hard*) *breakdown* of the Lanczos method (2) occurs at a step $n \leq d(r^{(0)}, A)$ if the moment matrix $K_n(\tilde{r}^{(0)}, A^*)^* A K_n(r^{(0)}, A)$ is singular. In this case, $u^{(n)}$ defined by (2) does not exist uniquely, so that *no* algorithm can be used to compute this iterate. On the other hand, we say that *soft breakdown* of the Lanczos method occurs at step n if the moment matrix $K_n(\tilde{r}^{(0)}, A^*)^* K_n(r^{(0)}, A)$ is singular. This condition causes failure of the Orthomin and Orthores variants but not the Orthodir variant.

Importantly, the conditions of hard and soft breakdown are conditions associated with the method (2), irrespective of the particular algorithms used to implement the method. Hard breakdown is a serious problem, a failure of the method defined by (2). On the other hand, soft breakdown is a condition that poses a problem only for certain algorithms but is not an intrinsic problem for the method (2), since some algorithms (e.g., Lanczos/Orthodir) may still be used to compute the iterates.

The Orthomin variant of Lanczos is generally preferable to the other variants, due to its economy and relative numerical stability. On the other hand, its vulnerability to the problem of soft breakdown may be remedied in theory by a temporary switch to the Orthodir variant in the event of a soft breakdown. This will be discussed further in §7.

In what follows we will consider some of the theoretical aspects of hard and soft breakdown of the Lanczos method.

4. Basic results on the likelihood of breakdown. One fundamental question to ask about the Lanczos algorithms is: how likely is an occurrence of breakdown for a given choice of A and $\tilde{Z}$? We recall that the matrix $\tilde{Z}$ defines the relationship

$\tilde{r}^{(0)} = \tilde{Z}^* r^{(0)}$.

To answer this question, we will use results from measure theory (see, e.g., [39] and [38]). Specifically, we define a field $\mathbb{K}$ to be either the reals $\mathbb{R}$ or the complex numbers $\mathbb{C}$, and for $A, \tilde{Z} \in \mathbb{K}^{N \times N}$ we ask: what is the measure of the set of vectors $r^{(0)} \in \mathbb{K}^N$ that cause hard or soft breakdown?

The following sequence of results begins to provide an answer to the question of the likelihood of breakdown. We show that in many cases, the set of initial residuals $r^{(0)}$ causing breakdown is only of zero measure, while in a few cases of A and $\tilde{Z}$ breakdown occurs for almost every vector $r^{(0)}$. It is desirable that the set of $r^{(0)}$ causing breakdown be measure zero, since this indicates that an initial guess vector $u^{(0)}$ chosen randomly has zero probability of causing breakdown in exact arithmetic.

To prove these measure-zero results, we begin with the following result on the measure of the set of zeros of a polynomial in several variables.

PROPOSITION 4 (Zero Sets of Polynomials). *Let $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$. If P is a complex nonzero polynomial in the variables $x_1, x_2, \cdots, x_N \in \mathbb{K}$, then $P(x) \neq 0$ for almost every $x = (x_1, x_2, \cdots, x_N) \in \mathbb{K}^N$.*

Proof. If $\mathbb{K} = \mathbb{R}$ and P is nonzero, then either $\operatorname{Re} P(z)$ or $\operatorname{Im} P(z)$ is a nonzero (real) polynomial; if $\mathbb{K} = \mathbb{C}$, we may decompose each x_i into real and imaginary parts, giving $2N$ variables, and consider the real polynomial $P(x)^* P(x)$. In any case, we may assume without loss of generality that P is a nonzero real polynomial of real variables.

We know that for any point x, the polynomial P is the zero polynomial if and only if the polynomial and all its derivatives are zero at x. Let V_0 denote the set of zeros of P in $\mathbb{R}^N$. Suppose the set V_0 has nonzero measure. We know from integration theory (see, for example, [44, pp. 128f]) that almost every point of V_0 is a point of density in each of the N coordinate directions. We recall that $x \in \mathbb{R}$ is a point of density of a measurable subset $S \subseteq \mathbb{R}$ if for any sequence of intervals I_n such that $x \in I_n$ with measure $m(I_n) \to 0$ we have $m(S \cap I_n)/m(I_n) \to 1$.

It is easily seen that at such points in V_0, the first partial derivatives of P must necessarily be zero. Let V_1 be the points of V_0 where all first derivatives are also zero. We have just shown that V_0 and V_1 both have the same nonzero measure. The argument may be repeated for V_1 to show all second partial derivatives of f are zero at almost every point of V_0, and so forth, resulting in the fact that P and all its derivatives are zero on a set which has nonzero measure. The proof is completed by selecting any one of these points. □

This result may be immediately applied to Lanczos moment matrices, by using the fact that the determinant of a matrix is a polynomial in the elements of the matrix.

COROLLARY 5 (Lanczos Moment Matrices). *Let $\mathbb{K} = \mathbb{R}$ or $\mathbb{C}$ and $\tilde{Z}$, $A \in \mathbb{C}^{N \times N}$. For some integer m let $f(r) = \det[K_n(\tilde{Z}^* r, A^*)^* A^m K_n(r, A)]$. Then either $f(r)$ is zero for all $r \in \mathbb{K}^N$ or it is zero only on a measure-zero set of vectors r in $\mathbb{K}^N$.*

Proof. If $\mathbb{K} = \mathbb{R}$, then $f(r)$ is a polynomial in the N variables $r_i \equiv e_i^* r$, where e_i represents the standard unit basis vector. If $\mathbb{K} = \mathbb{C}$, then $f(r)$ is a polynomial in the $2N$ variables $\operatorname{Re} r_i$ and $\operatorname{Im} r_i$. □

From this we conclude that the relevant Lanczos moment matrices are singular either for every $r^{(0)}$ or only for a measure-zero set of vectors $r^{(0)}$.

In order to proceed, we must show some results concerning the degrees of vectors with respect to the matrix A. We define the *degree of a matrix* $d(A) = \min\{\deg(P) : P \text{ monic}, P(A) = 0\}$. We note that for every v we have $0 \leq d(v, A) \leq d(A) \leq N$. We show here that almost every vector v satisfies $d(v, A) = d(A)$.

PROPOSITION 6 (Degrees of Vectors). *Let* $\mathbb{K} = \mathbb{R}$ *or* $\mathbb{C}$ *and* $A \in \mathbb{K}^{N\times N}$. *Then almost every* $v \in \mathbb{K}^N$ *satisfies* $d(v, A) = d(A)$.

Proof. The result follows from the fact that almost every vector necessarily contains a nonzero component of each of the generalized eigenvectors of A. The details are found in [24]. □

We noted earlier that breakdown is equivalent to the singularity of an appropriate moment matrix *only* when the iteration number n satisfies $n \leq d(r^{(0)}, A)$. Due to this technical point, in order to prove the desired results we define notation for the set of vectors $r^{(0)}$ for which this condition is satisfied. We define the set $\mathcal{T}_n(A) = \{v \in \mathbb{C}^N : n \leq d(v, A)\}$. The set $\mathcal{T}_n(A)$ is the set of initial residuals $r^{(0)}$ for which the singularity/nonsingularity of the moment matrices is relevant to the question of breakdown at step n.

We established in Proposition 6 that $\mathcal{T}_{d(A)}(A) \cap \mathbb{K}^N$ contains almost every vector in $\mathbb{K}^N$. It will be noted that $\mathcal{T}_n(A)$ is monotone, in the sense that $m \leq n$ implies $\mathcal{T}_m(A) \supseteq \mathcal{T}_n(A)$. Thus for any $n \leq d(A)$, $\mathcal{T}_n(A) \cap \mathbb{K}^N$ necessarily contains almost every vector in $\mathbb{K}^N$.

We now present the major theorem, which gives the three basic possibilities for breakdown of Lanczos algorithms. The upshot of this result is that for a given iteration number $n \leq d(A)$, for the set of vectors $r^{(0)}$ for which $d(r^{(0)}, A) \geq n$ (which amounts to almost every vector), either breakdown is impossible, breakdown always occurs, or breakdown occurs only for a nonempty measure-zero set of vectors.

THEOREM 7 (Lanczos Breakdown, Iterate n). *Let* $\mathbb{K} = \mathbb{R}$ *or* $\mathbb{C}$, $A, \tilde{Z} \in \mathbb{K}^{N\times N}$, *and* $n \leq d(A)$. *Then exactly one of the following three conditions holds for the Lanczos method with* $\tilde{r}^{(0)} = \tilde{Z}^* r^{(0)}$.

(i) *Hard breakdown at step* n *occurs for every vector* $r^{(0)} \in \mathcal{T}_n(A) \cap \mathbb{K}^N$ (*and thus at least for almost every* $r^{(0)} \in \mathbb{K}^N$).

(ii) *Hard breakdown at step* n *occurs for a nonempty measure-zero set of vectors* $r^{(0)} \in \mathcal{T}_n(A) \cap \mathbb{K}^N$ (*and thus a nonempty measure-zero set of vectors in* $\mathbb{K}^N$).

(iii) *Hard breakdown at step* n *occurs for no vectors* $r^{(0)} \in \mathcal{T}_n(A) \cap \mathbb{K}^N$ (*and thus for at most a measure-zero set of vectors in* $\mathbb{K}^N$).

Furthermore, the same result holds if "hard breakdown" is replaced by "soft breakdown" in the statement of this theorem.

Proof. For vectors $r^{(0)} \in \mathcal{T}_n(A) \cap \mathbb{K}^N$, breakdown is equivalent to singularity of an appropriate moment matrix. The set $\mathcal{T}_n(A) \cap \mathbb{K}^N$ amounts to almost every vector in $\mathbb{K}^N$. Now, by Corollary 5, the set S_n of vectors in $\mathbb{K}^N$ for which the moment matrix of dimension n is singular is either the set of all vectors or a subset of measure zero. If the moment matrix is singular for every vector (i.e., $S_n = \mathbb{K}^N$), then it is singular for every vector in $\mathcal{T}_n(A) \cap \mathbb{K}^N$, giving case (i) above. Otherwise the set S_n is measure zero in $\mathbb{K}^N$. Thus $\mathcal{B}_n \equiv S_n \cap (\mathcal{T}_n(A) \cap \mathbb{K}^N)$ is of measure zero and is either empty or nonempty. □

5. Measure-zero results. When the Lanczos method may be reduced to a standard conjugate gradient method, it is known that breakdown at any step is impossible. In the general case, we would like to show at least that breakdown occurs for no more than a measure-zero set of vectors. We will show below that this is true in many but not all cases.

We begin by considering the simple case of $n = N = d(A)$. We may then show a more general result by projecting the problem to this simpler case. Similar results are shown in [40] and [50].

THEOREM 8 (Measure-Zero Breakdown, Case $n = N = d(A)$). *Let* $\mathbb{K} = \mathbb{R}$ *or*

$\mathbb{C}$, *and let* $A, \tilde{Z} \in \mathbb{K}^{N\times N}$ *with* A *and* $\tilde{Z}$ *nonsingular and* $d(A) = N$, *and let* m *be some integer. Then for* $n = N$ *and almost every* $r \in \mathbb{K}^N$, $K_n(\tilde{Z}^*r, A^*)^*A^mK_n(r, A)$ *is nonsingular.*

Proof. Let $S = \{r \in \mathbb{K}^N : d(r, A) = N\}$ and $S' = \{\tilde{r} \in \mathbb{K}^N : d(\tilde{r}, A^*) = N\}$. By Proposition 6, S and S' each contain almost every vector in $\mathbb{K}^N$. Furthermore, for every $r \in S$, $K_n(r, A)$ is a *square* full-rank matrix, and thus nonsingular. Also, for every $\tilde{Z}^*r \in S'$ (i.e., $r \in \tilde{Z}^{-*}S'$), $K_n(\tilde{Z}^*r, A^*)$ is a square nonsingular matrix. The proof is completed by noting that $S \cap \tilde{Z}^{-*}S'$ contains almost every vector in $\mathbb{K}^N$ and that the product of two square nonsingular matrices is nonsingular. □

We now prove the result for a more general situation including nearly all iteration numbers $n \leq d(A)$. The following result applies to a wide range of choices of $\tilde{Z}$, including, for example, the common choice $\tilde{Z} = I$. The idea of the proof is to reduce the general problem back to the case of $n = N = d(A)$ by projecting onto an appropriate A-invariant Krylov subspace. The range of possible values of n covered by this theorem will be discussed below.

We define here a *definite* matrix to be a matrix B satisfying $v^*Bv \neq 0$ for every nonzero v. A real matrix B is definite if and only if either its Hermitian part $H(B) \equiv (B + B^*)/2$ or its negative $-H(B)$ is HPD (see [26]).

THEOREM 9 (Measure-Zero Breakdown, Iteration n). *For* $\mathbb{K} = \mathbb{R}$ *or* $\mathbb{C}$, *let* A, $\tilde{Z} \in \mathbb{K}^{N\times N}$, *and let* $\tilde{Z}$ *satisfy the condition that for some polynomials* F *and* G *over* $\mathbb{K}$ *and for some definite matrix* $H \in \mathbb{K}^{N\times N}$, $\tilde{Z} = F(A)^*HG(A)$ *is nonsingular. Let* m *be arbitrary and let* $n \leq d(A)$ *be such that there exists* $v \in \mathbb{K}^N$ *satisfying* $d(v, A) = n$. *Then there exist* $r \in \mathbb{K}^N$ *and* $\tilde{r} = \tilde{Z}^*r$ *such that* $K_n(\tilde{r}, A^*)^*A^mK_n(r, A)$ *is nonsingular.*

Proof. We have $\tilde{Z} = [F(A)]^*HG(A)$ nonsingular with $H \in \mathbb{K}^{N\times N}$ definite and polynomials F and G over $\mathbb{K}$. We seek to find $r \in \mathbb{K}^N$ such that

$$K_n([G(A)]^*H^*F(A)r, A^*)^*A^mK_n(r, A)$$

is nonsingular.

By hypothesis, there exists v such that $d(v, A) = n$. Let Q be the ℓ^2-orthogonal projector onto the A-invariant subspace $\mathbf{K}_n(v, A)$ of $\mathbb{K}^N$. We note that $Q \in \mathbb{K}^{N\times N}$ is Hermitian and $AQ = QAQ$. Then it is sufficient to find $r \in \operatorname{Range} Q$ such that

$$K_n([G(\check{A})]^*\check{H}^*F(\check{A})r, \check{A}^*)^*\check{A}^mK_n(r, \check{A})$$

is nonsingular, where $\check{A} = QAQ$ and $\check{H} = QHQ$.

Without loss of generality we may assume $r \in \mathbb{K}^n$ and $\check{A}$, $\check{H} \in \mathbb{K}^{n\times n}$, by restriction of the operators to $\operatorname{Range} Q$. Importantly, since H is definite, the restricted matrix $\check{H}$ is definite, thus nonsingular. Furthermore, since the spectrum of $\check{A}$ is contained in the spectrum of A, we have that $F(\check{A})$ and $G(\check{A})$ are nonsingular. Therefore, by applying Theorem 8 to this restricted problem, we have the desired result. □

It remains to be determined which iteration numbers n the above result may be applied to. In the case of complex spaces, an A-invariant (complex) Krylov subspace of *any* size $n \leq d(A)$ exists. On the other hand, if A is real and subspaces over the reals are sought, then some values of n may be excluded. In particular, if A has no real eigenvalues, then for every real r, the value of $d(r, A)$ must be even. This is true because such r must necessarily contain matched pairs of complex conjugate generalized eigenvectors in order to be a real vector.

These observations are made precise by the following proposition.

PROPOSITION 10 (Vectors of a Given Degree). *Let* $\mathbb{K} = \mathbb{R}$ *or* $\mathbb{C}$, $A \in \mathbb{K}^{N\times N}$, *and* $0 \le n \le d(A)$. *Then there does not exist a vector* $r \in \mathbb{K}^N$ *satisfying* $d(r, A) = n$, *if and only if all of the following conditions hold:* $\mathbb{K} = \mathbb{R}$, n *is odd, and* A *has no real eigenvalues.*

Proof. For a detailed proof, see [24]. □

To summarize, we see that for the common choice of $\tilde{Z} = I$, breakdown cannot occur for any of the three Lanczos algorithms except for a set of vectors $r^{(0)}$ which has zero measure in $\mathbb{C}^N$. On the other hand, when A is real, in order to guarantee the same result for vectors in $\mathbb{R}^N$, it is sufficient that A have at least one real eigenvalue.

Though this condition may not be necessary, nonetheless some restriction is necessary in order to limit breakdown to a measure-zero set of real vectors, as the following example shows. Let

$$A = \begin{bmatrix} 1 & -1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 3 & -1 \\ 0 & 0 & 1 & 3 \end{bmatrix}$$

and let $\tilde{r} = r$. We note that A is a normal matrix and $d(A) = 4$. After some algebra it may be shown that for $n = 3$ and for *every* real r, $K_n(\tilde{r}, A^*)^* K_n(r, A)$ is singular. As a practical consequence, the standard BCG algorithm (Lanczos/Orthomin, $\tilde{Z} = I$) necessarily breaks down at step 4 for every real $r^{(0)}$ satisfying $d(r^{(0)}, A) \ge 3$, that is, for almost every real vector $r^{(0)}$.

This result contradicts a result of [50, p. 278]. It also sheds light on the comments of [22, p. 21], [49, p. 389], and [4, p. 328] on selecting an alternate pair of initial vectors r, $\tilde{r}$ in the case of breakdown. In particular, this counterexample shows that for some cases of A and $\tilde{Z}$, there is no practical way to apply the standard Lanczos algorithms in real arithmetic if the initial vectors r and $\tilde{r}$ are forced to satisfy the relationship $\tilde{r} = \tilde{Z}^* r$. Thus, in some cases it is necessary to choose r and $\tilde{r}$ as arbitrary *independent* vectors, or else they must be chosen complex. This is also in agreement with the comments of [7], where it is suggested that complex vectors be used, since in general the invariant subspaces for A and A^* are complex.

We conclude that for a certain small class of matrix problems, the standard BCG algorithm cannot be used successfully. On the other hand, since the problem of hard or soft breakdown for almost every real $r^{(0)}$ can only occur for *odd* values of n, a look-ahead procedure could be used to skip over these steps. This will be described further in §7.

6. Curable and incurable breakdown. We now analyze hard and soft breakdown of the Lanczos method from an alternate viewpoint. In the previous sections we considered the behavior of the Lanczos method at a given step n as the initial residual $r^{(0)}$ was allowed to range over all possible values in $\mathbb{C}^N$ or $\mathbb{R}^N$. In this section we will instead consider the behavior of Lanczos for a *fixed* value of $r^{(0)}$, for various values of n.

If hard (soft) breakdown of the Lanczos method occurs at step n, then we say that it is an *incurable hard* (*soft*) *breakdown* if for every step m satisfying $n < m \le d(r^{(0)}, A)$, hard (soft) breakdown also occurs at step m. A hard (soft) breakdown that is not incurable is said to be *curable.* These concepts are an adaptation of the definitions given in [46] and [37] regarding the nonsymmetric Lanczos method for solving the eigenvalue problem.

As mentioned earlier, an occurrence of soft breakdown, whether curable or incurable, could be remedied by a temporary switch to the Orthodir variant of Lanczos. On

the other hand, an instance of curable hard breakdown could in theory be remedied by an algorithm that skipped over the steps for which hard breakdown occurred. This idea of a look-ahead Lanczos algorithm is developed for the eigenvalue problem in [37]. Look-ahead techniques for Lanczos and other methods constitute a current area of research; see, for example, [18]–[20], [3], [35], [13]–[15], and [28].

Unfortunately, a curable hard breakdown may occur for as many as $N-1$ steps, where N is the size of A. For example, we let

$$A = e_N e_1^* + \sum_{i=1}^{N-1} e_i e_{i+1}^*,$$

a permutation matrix, and we let $r^{(0)} = \tilde{r}^{(0)} = e_N$. Here, e_i represents the standard unit basis vector. Then

$$K_N(\tilde{r}^{(0)}, A^*)^* A K_N(r^{(0)}, A) = \sum_{i=1}^{N} e_i e_{N-i+1}^*.$$

Thus the Lanczos method with $\tilde{Z} = I$ experiences hard breakdown at every step $n < N$ but not at step $n = N$. In this case, a look-ahead algorithm would have to perform a look-ahead over a prohibitively large number of steps.

We now consider *incurable* hard and soft breakdown. The phenomena of incurable hard and soft breakdown have straightforward characterizations. The following result is similar to the Mismatch Theorem of [46].

THEOREM 11 (Mismatch Theorem). *Suppose that for a particular choice of initial vectors r and $\tilde{r}$ and for a given m and for a given value of n satisfying $n < d(r, A)$, we have that either $n = 0$ or $K_n(\tilde{r}, A^*)^* A^m K_n(r, A)$ is nonsingular. Then the matrix $K_{n+k}(\tilde{r}, A^*)^* A^m K_{n+k}(r, A)$ is singular for every k satisfying $n < n + k \leq d(r, A)$ if and only if there exist vectors p and $\tilde{p}$ and integers $d > 0$ and $d' \geq 0$ such that $d(r, A) = n + d$, $d(\tilde{r}, A^*) = n + d'$, and*

$$\mathbf{K}_{n+d}(r, A) = \text{Range}[K_n(r, A) \quad K_d(p, A)],$$

$$\mathbf{K}_{n+d'}(\tilde{r}, A^*) = \text{Range}[K_n(\tilde{r}, A^*) \quad K_{d'}(\tilde{p}, A^*)],$$

$$K_{d'}(\tilde{p}, A^*)^* A^m K_{n+d}(r, A) = 0, \qquad K_{n+d'}(\tilde{r}, A^*)^* A^m K_d(p, A) = 0,$$

where $[K_n(r, A) \quad K_d(p, A)]$ and $[K_n(\tilde{r}, A^*) \quad K_{d'}(\tilde{p}, A^*)]$ represent block matrices.

Proof. See [24]. □

From this result we have the following consequences.

COROLLARY 12 (Breakdown Equivalence). *For a particular run of Lanczos, incurable hard breakdown occurs at step n if and only if incurable soft breakdown occurs at step n.*

Proof. First, we suppose incurable hard breakdown occurs at step n. Let $m + 1 \leq n$ indicate the first step at which the incurable hard breakdown has occurred. Using the characterization from Theorem 11, since $\mathbf{K}_{m+d}(r, A) = A\mathbf{K}_{m+d}(r, A)$ and $\mathbf{K}_{m+d'}(\tilde{r}, A^*) = A^*\mathbf{K}_{m+d'}(\tilde{r}, A^*)$, we see that incurable soft breakdown also occurs at step $m + 1$, and thus at step n. The opposite implication is proved analogously. □

COROLLARY 13 (Restarting of Lanczos). *Suppose that for a run of the Lanczos method breakdown does not occur at step $n \geq 0$ but incurable breakdown occurs at step $n + 1$. Then $d(r^{(n)}, A) = d(r^{(0)}, A) - n$. Thus the Lanczos method applied to initial*

residual $r^{(0)\prime} = r^{(n)}$ *with any auxiliary residual vector* $\tilde{r}^{(0)\prime}$ *is guaranteed to converge (if breakdown does not recur) within* $d(r^{(0)}, A) - n$ *steps.*

Proof. We first show that $d(r^{(n)}, A) \leq d(r^{(0)}, A) - n$. Using (2) and the characterization of Theorem 11, we have $r^{(n)} - r^{(0)} \in A\mathbf{K}_n(r^{(0)}, A)$ so that $r^{(n)} \in \mathbf{K}_{n+d}(r^{(0)}, A)$, i.e., $r^{(n)} = [K_n(r^{(0)}, A)\ K_d(p, A)]\beta$ for some β. Furthermore, $r^{(n)} \perp \mathbf{K}_n(\tilde{r}^{(0)}, A^*)$, so that $r^{(n)} \in \mathbf{K}_d(p, A)$, since $K_n(\tilde{r}^{(0)}, A^*)^* K_n(r^{(0)}, A)$ is nonsingular and $K_n(\tilde{r}^{(0)}, A^*)^* K_d(p, A)$ is zero. We also note that $\mathbf{K}_d(p, A)$ is A-invariant: $v \in \mathbf{K}_d(p, A)$ implies $Av \in \mathbf{K}_{n+d}(r^{(0)}, A)$. But $(w, Av) = (A^*w, v) = 0$ for all $w \in \mathbf{K}_{n+d'}(\tilde{r}^{(0)}, A^*)$. Thus $Av \perp \mathbf{K}_{n+d'}(\tilde{r}^{(0)}, A^*)$, implying $Av \in \mathbf{K}_d(r^{(0)}, A)$. Thus it is necessarily true that $d(r^{(n)}, A) \leq d$.

We now show that $d(r^{(n)}, A) \geq d(r^{(0)}, A) - n$. We have that $r^{(n)} = P(A)r^{(0)}$ where $P(0) = 1$ and $\deg P \leq n$. Now, let $\tilde{P}$ be the minimal polynomial for $r^{(n)}$. Then $0 = \tilde{P}(A)r^{(n)} = \tilde{P}(A)P(A)r^{(0)}$, from which we obtain $\deg \tilde{P} + n \geq \deg \tilde{P} + \deg P = \deg(\tilde{P} \cdot P) \geq d(r^{(0)}, A)$, so that $\deg P = n$ and $d(r^{(n)}, A) = d(r^{(0)}, A) - n$. □

This result suggests that the restarting of the Lanczos method is a possible remedy for the problem of incurable breakdown. It is not clear, however, how restarting affects the number of iterations required to satisfy some convergence criterion such as $||r^{(n)}||/||r^{(0)}|| < \zeta$. Restarting of the Lanczos method is a technique that was investigated experimentally in [31] and also mentioned in [47]. This technique will be considered in more detail below.

7. Modified Lanczos algorithms. The theoretical results given above for the Lanczos method may be applied to the development of modified Lanczos algorithms that are better able to deal with the problem of breakdown or near-breakdown. In [24] the algorithm BCGNB is defined. It embodies three particular strategies for remedying the breakdown problem of the standard BCG algorithm, which uses $\tilde{Z} = I$:

1. In the case of a near-soft-breakdown, a switch to the Orthodir variant is made for the step for which soft breakdown is a problem. A similar idea was used in [5] to develop a hybrid Orthodir/Orthomin conjugate residual algorithm for the case of A symmetric indefinite. Such a hybrid algorithm may be easily developed for the Lanczos method, based on the observation that the vector $p^{(n)}$ (alternatively, $\tilde{p}^{(n)}$) of the Orthomin variant of Lanczos is a scalar multiple of the vector $q^{(n)}$ (alternatively, $\tilde{q}^{(n)}$) of the Orthodir variant, whenever these quantities are well defined.

2. In the case of near-hard-breakdown for up to $s - 1$ steps, where s is a parameter supplied to the algorithm, a look-ahead procedure similar to that of [37] is used to skip over those steps.

3. For near-hard-breakdown for more than $s - 1$ steps, a restart of the algorithm is performed, based on the last iterate for which near-hard-breakdown was not indicated. This is done in the hope that this is a case of an incurable breakdown.

Derivations of the formulas for the BCGNB algorithm are given in [24]. The implementation of this algorithm requires the development of adequate tests for near-hard-breakdown and near-soft-breakdown that account for the difficulties of finite precision arithmetic. Tests are defined and experimental results for the choices of tolerances for the tests are given in [24].

Experimental evidence indicates that the most effective of the three remedies listed above is the technique of restarting. Due to its simplicity, we will state here the criterion used to determine whether to restart. In particular, using the notation for the Lanczos/Orthomin algorithm given in §2, we say that if the criterion

$$\frac{|(\tilde{p}^{(n)}, Ap^{(n)})|}{||\tilde{p}^{(n)}|| \cdot ||Ap^{(n)}||} < \epsilon_3$$

is satisfied, then a restart of the algorithm will be performed, using the new initial residual $r^{(0)\prime} = r^{(n)}$ and the new auxiliary initial residual $\tilde{r}^{(0)\prime} = \tilde{Z}^* r^{(0)\prime}$ with $\tilde{Z} = I$. A simple restarted BCG algorithm may be implemented based solely on this technique. The particular choice of $\epsilon_3 = \epsilon_M^{1/2}$ has proven to be useful for a number of problems where ϵ_M is unit roundoff error, the smallest floating point number such that $1+\epsilon_M > 1$. This particular criterion for testing for near-hard-breakdown has the advantage of limiting the size of the growth of $||r^{(n)}||$ over the course of the run.

Another algorithm that suffers from the same breakdown problems as the Lanczos algorithm and can be remedied by similar techniques is the conjugate gradient squared (CGS) algorithm of [45]. The CGS algorithm is defined as follows:

CGS ALGORITHM

$$p_2^{(0)} = f^{(0)} = r^{(0)},$$

$$f^{(n)} = r^{(n)} + \alpha_n h^{(n)}, \qquad p_2^{(n)} = f^{(n)} + \alpha_n (h^{(n)} + \alpha_n p_2^{(n-1)}),$$

$$h^{(n+1)} = f^{(n)} - \lambda_n A p_2^{(n)},$$

$$u^{(n+1)} = u^{(n)} + \lambda_n (f^{(n)} + h^{(n+1)}), \qquad r^{(n+1)} = r^{(n)} - \lambda_n A (f^{(n)} + h^{(n+1)}),$$

$$\lambda_n = \frac{(\tilde{r}^{(0)}, r^{(n)})}{(\tilde{r}^{(0)}, A p_2^{(n)})}, \qquad \alpha_{n+1} = \frac{(\tilde{r}^{(0)}, r^{(n+1)})}{(\tilde{r}^{(0)}, r^{(n)})}.$$

This algorithm experiences breakdown in precisely the same instances as the BCG algorithm, in exact arithmetic. The CGS algorithm commonly requires roughly the same computational work per iteration as BCG, and half the number of iterations; however, the numerical effects of near-hard-breakdown are frequently more severe (see, e.g., [48]).

A restarted CGS algorithm, CGSNB, is defined in [24]. The criterion used to test for near-breakdown is

$$\frac{|(\tilde{r}^{(0)}, A p_2^{(n)})|}{||\tilde{r}^{(0)}|| \cdot ||A p_2^{(n)}||} \leq \epsilon_h.$$

In the runs presented below, we use the setting $\epsilon_h = 10\epsilon_M^{1/2}$.

We now describe one further technique for mitigating the problem of breakdown. The Mismatch Theorem [46], [24] indicates that incurable breakdown is caused by irregular left- and right-eigenvector distributions in $r^{(0)}$ and $\tilde{r}^{(0)}$. To remedy this problem, a randomized vector may be used for $r^{(0)}$. This may easily be done by setting the initial guess $u^{(0)}$ to be a vector of random entries of an appropriate size. To do this, we let v be a vector whose elements are random numbers uniformly distributed on $[-1, 1]$, and we set $u^{(0)}$ to be a multiple of v scaled so that $||Au^{(0)}|| = ||b||$. Experiments using this technique will be given below. Similarly, if a choice of $u^{(0)}$ is already known that is near the true solution, then the given vector may be perturbed by a small random vector in order to give a new randomized choice of $u^{(0)}$. Experimental results from using random vectors with Lanczos algorithms are also reported in [12].

8. Numerical results. In this section we present numerical results with the algorithms described in this paper. We are primarily concerned with the algorithms BCG, BCGNB, CGS, and CGSNB. For comparison, we will also consider the full

TABLE 1
Model problem, $h^{-1} = 128$, ITMAX=3000. *Number of iterations.*

Method \ Dh:	0	2^{-3}	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}	2^{4}	2^{5}
GMRES(∞)	290	269	245	220	200	189	186	189	207	249
BCG	308	341	299	1518	–	–	–	–	533	–
BCG, random $u^{(0)}$	309	354	300	310	313	301	299	302	290	293
BCGNB	308	353	284	338	253	240	243	240	302	962
CGS	272	254	222	–	–	–	–	–	–	–
CGS, random $u^{(0)}$	193	189	200	192	193	175	225	212	216	197
CGSNB	272	284	212	196	151	162	158	173	156	256

GMRES algorithm GMRES(∞) [43], the restarted algorithm GMRES(k), and the normal equations algorithm LSQR [33].

The GMRES(∞) algorithm is of particular interest, since it gives iteration counts that are minimal among all polynomial methods, in the sense that $||r^{(n)}||$ is minimized with respect to all possible polynomial methods in A. However, the method is generally too expensive to be practical.

Unless otherwise stated, we make the following assumptions. For the test runs we utilize the initial guess vector of $u^{(0)} = 0$. For the sake of simplicity, we use the simple stopping test

$$\frac{||r^{(n)}||}{||b||} < \zeta = 10^{-6}.$$

Also, we use the basic choice $\tilde{r}^{(0)} = r^{(0)}$, i.e., $\tilde{Z} = I$, for the Lanczos-type algorithms.

The University of Texas System Cray X-MP/24 vector computer was used to perform the runs presented here. Single precision real arithmetic was used, with unit roundoff given by $\epsilon_M = (7.1 \times 10^{-15})$.

The first set of test matrix problems to be considered arises from the finite difference discretization of the boundary value problem

$$-u_{xx}(x,y) - u_{yy}(x,y) + Du_x(x,y) = G(x,y) \quad \text{on } \Omega = [0,1]^2,$$
$$u(x,y) = 1 + xy \quad \text{on } \partial\Omega.$$

We utilize central differencing to discretize this problem, with uniform mesh spacing h in either direction. This yields a matrix of size $N = (n_h - 1)^2$ (where $h = 1/n_h$), after boundary points have been eliminated. The right-hand-side function $G(x,y)$ is defined so that the true solution is $u(x,y) = 1 + xy$ on Ω. By varying the constant D, the amount of nonsymmetry of the matrix may be varied. Specifically, for a given h there exist a symmetric matrix A_S and a skew-symmetric matrix A_N, independent of D, such that $A = A_S + D \cdot A_N$.

In Tables 1–3 we consider the unpreconditioned problem and also the (left) ILU- and MILU-preconditioned problem (see [17] and [32]). Runs for which convergence was not possible in ITMAX iterations are labeled by (–).

We make the following observations about these runs.

- For the unpreconditioned problem, the standard BCG and CGS algorithms break down in a number of cases, but the use of random $u^{(0)}$ or the use of BCGNB or CGSNB resulted in convergence. Furthermore, the iteration counts for the algorithms BCG and BCGNB are in general comparatively close to those of the "best" method, GMRES(∞), while these algorithms have short economical recurrences, unlike GMRES(∞). This underscores the importance of the Lanczos algorithms as economical solution techniques.

TABLE 2
Model problem, $h^{-1} = 128$, ILU-preconditioning, ITMAX=500. *Number of iterations.*

Method \ Dh:	0	2^{-3}	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}	2^{4}	2^{5}
GMRES(∞)	92	83	74	64	52	41	32	26	19	14
BCG	94	102	90	82	178	53	35	28	22	17
BCG, random $u^{(0)}$	94	102	92	87	55	45	34	28	22	17
BCGNB	94	102	88	77	128	67	35	27	21	17
CGS	74	68	64	90	–	97	26	18	12	9
CGS, random $u^{(0)}$	63	59	58	55	48	33	25	18	13	9
CGSNB	74	68	61	73	48	39	26	18	12	9

TABLE 3
Model problem, $h^{-1} = 128$, MILU-preconditioning, ITMAX=500. *Number of iterations.*

Method \ Dh:	0	2^{-3}	2^{-2}	2^{-1}	2^{0}	2^{1}	2^{2}	2^{3}	2^{4}	2^{5}
GMRES(∞)	27	25	24	26	28	28	25	19	14	10
GMRES(∞), random $u^{(0)}$	33	29	28	29	31	31	29	24	19	14
BCG	31	27	29	33	30	37	30	23	15	10
BCG, random $u^{(0)}$	38	34	33	37	44	40	38	29	23	18
BCGNB	28	27	29	30	34	35	30	23	15	10
CGS	21	18	17	20	22	22	19	15	9	6
CGS, random $u^{(0)}$	24	18	20	22	22	23	21	16	12	9
CGSNB	21	18	17	20	22	27	20	15	9	6

- For the ILU-preconditioned problems, in most cases all methods worked well. For the case of $Dh = 1$, BCG gave an excessive number of iterations, but this was remedied significantly by BCGNB and much more so by the use of random $u^{(0)}$. Similarly, CGS could not converge, but CGSNB and CGS with random $u^{(0)}$ both converged.
- For all of the MILU-preconditioned problems, all of the Lanczos-type algorithms performed quite well. In particular, the BCG algorithm gave approximately the same number of iterations as GMRES(∞).

Figures 1 and 2 give representative plots of the convergence behavior of the algorithms for the case of $h^{-1} = 128$, $Dh = 4$, and no preconditioning. These results show that the new algorithms keep the residual size better behaved than the standard BCG and CGS algorithms over the course of the run.

We now consider a more difficult class of finite difference problems, namely, central finite differencing applied to the Dirichlet problem

$$-u_{xx}(x,y) - u_{yy}(x,y) + D[(y - \tfrac{1}{2})u_x(x,y) + (x - \tfrac{1}{3})(x - \tfrac{2}{3})u_y(x,y)],$$

$$-43\pi^2 u(x,y) = G(x,y) \quad \text{on } \Omega = [0,1]^2,$$

$$u(x,y) = 1 + xy \quad \text{on } \partial\Omega,$$

with $G(x,y)$ chosen as before so that the true solution is $u(x,y) = 1 + xy$. Again, we let h denote the mesh size in each direction. For $D = 0$ and h small, the matrix generated by this problem is a symmetric indefinite matrix with 16 distinct negative eigenvalues and the rest of the spectrum positive.

The standard conjugate residual algorithm applied to this problem with $h^{-1} = 128$ and $D = 0$ requires 766 iterations to converge to $||r^{(n)}||/||b|| < \zeta = 10^{-6}$. In any case, this is a difficult problem to solve.

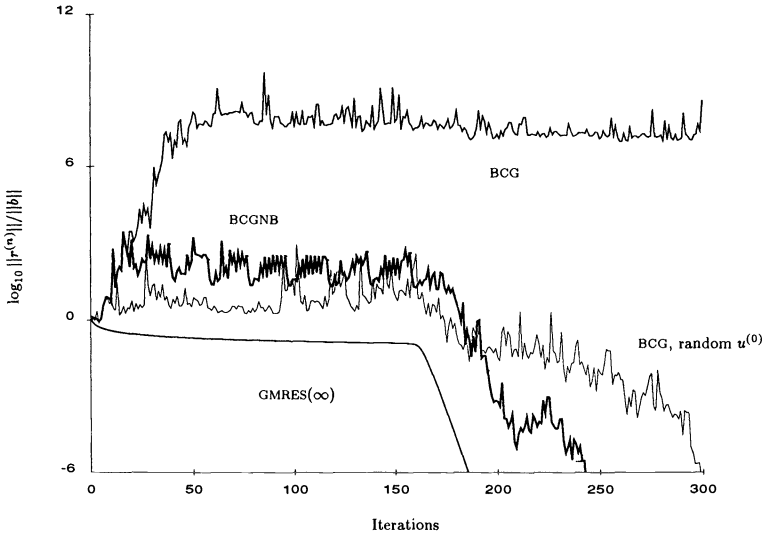


FIG. 1. *Residual behavior:* $h^{-1} = 128$, $Dh = 4$.

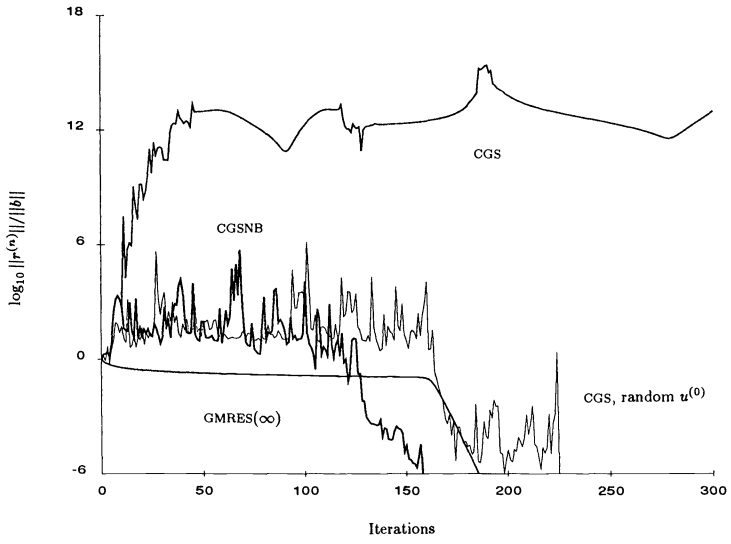


FIG. 2. *Residual behavior:* $h^{-1} = 128$, $Dh = 4$.

In Table 4 we give numerical results for various algorithms applied to this problem. The BCG algorithm applied to the $Dh = .5$ case with a second random vector $u^{(0)}$ failed to converge. Also, for the CGS algorithm applied to $Dh = .5$ with random $u^{(0)}$, convergence was indicated, but the final *true* value of $||r^{(n)}||/||b||$ had a (degraded) value of $.12 \times 10^{-3}$.

We now comment on these runs.

- The BCG algorithm applied to this problem gave good results. Similar results

TABLE 4
Indefinite problem, $h^{-1} = 128$, ITMAX=8000. *Iterations.*

Method \ Dh:	0	2^{-3}	2^{-2}	2^{-1}
BCG	820	1803	2209	3384
BCG, random $u^{(0)}$	839	1767	2707	4079
BCGNB	835	1814	2530	–
BCGNB, $\epsilon_3 = \epsilon_M^{1/2}/100$	835	1814	2397	4794
CGS	939	–	–	–
CGS, random $u^{(0)}$	789	2307	3268	6329
CGSNB	–	–	–	–
CGSNB, $\epsilon_h = \epsilon_M^{1/2}/100$	939	–	–	–
GMRES(5)	–	–	–	–
GMRES(10)	–	–	–	–
GMRES(20)	–	–	–	–
GMRES(40)	–	–	–	–
LSQR	–	–	–	–

were given by using random $u^{(0)}$, but for some random choices of $u^{(0)}$ the algorithm did not converge.

- The BCGNB algorithm converged for all cases except $Dh = .5$. It was found that a smaller value of ϵ_3, causing a more stringent requirement for restarting, was able to give convergence in this case.
- The CGS algorithm was not able to converge for the nonsymmetric cases. This was effectively remedied by random $u^{(0)}$ except for the $Dh = .5$ case. The CGSNB algorithm was not able to converge for the nonsymmetric cases. This might suggest that the use of random $u^{(0)}$ is a safer strategy than restarting, in general.
- For comparison purposes, we give run results for the GMRES(k) algorithm, which is the GMRES(∞) algorithm restarted every k iterations. This is known to be an effective algorithm for solving linear systems for which A is a definite matrix. We see that for a wide range of choices of k, the restarted GMRES algorithm was not able to converge for any of the problems in the given number of iterations, even when k was rather large. Furthermore, the normal equations algorithm LSQR was not able to solve these problems either in the given number of iterations.

A major conclusion to be drawn from this example is that the class of Lanczos-type methods is an important alternative for solving difficult matrix problems such as this problem. In fact, for this problem the Lanczos-type methods were the only methods that converged among all the methods tried.

9. Conclusions. In this paper we have examined some theoretical aspects of the Lanczos method and have defined and tested modified Lanczos algorithms which are of significant use in remedying the convergence problems of the Lanczos method. The results given here indicate that the class of Lanczos-type methods is an important alternative for solving nonsymmetric systems of equations. However, further research is necessary to better understand the behavior of the Lanczos algorithms, particularly in finite precision arithmetic. A theoretical understanding of the phenomenon of loss of orthogonality of the Lanczos vectors for the nonsymmetric case is necessary (see [16] and [34]). Despite these difficulties, positive experimental results with the Lanczos method continue to make it a significant method for solving nonsymmetric systems of equations.

REFERENCES

[1] R. O. ABBASIAN, *Lanczos algorithms for the acceleration of nonsymmetrizable iterative methods*, Master's thesis, Report CNA-193, Center for Numerical Analysis, University of Texas at Austin, Austin, TX, May 1984.

[2] S. F. ASHBY, T. A. MANTEUFFEL, AND P. E. SAYLOR, *A taxonomy for conjugate gradient methods*, SIAM J. Numer. Anal., 27 (1990), pp. 1542–1568.

[3] D. BOLEY AND G. GOLUB, *The nonsymmetric Lanczos algorithm and controllability*, Numerical Analysis Project Manuscript NA-90-06, Computer Science Department, Stanford University, Stanford, CA, May 1990.

[4] R. L. CAUSEY AND R. T. GREGORY, *On Lanczos' algorithm for tridiagonalizing matrices*, SIAM Rev., 3 (1961), pp. 322–328.

[5] R. CHANDRA, S. C. EISENSTAT, AND M. H. SCHULTZ, *The modified conjugate residual method for partial differential equations*, in Advances in Computer Methods for Partial Differential Equations II, R. Vichnevetsky, ed., IMACS, Rutgers University, New Brunswick, NJ, 1977, pp. 13–19.

[6] J. CULLUM, W. KERNER, AND R. WILLOUGHBY, *A generalized nonsymmetric Lanczos procedure*, Comput. Phys. Comm., 53 (1989), pp. 19–48.

[7] J. CULLUM AND R. A. WILLOUGHBY, *A practical procedure for computing eigenvalues of large sparse nonsymmetric matrices*, in Large Scale Eigenvalue Problems: Proceedings of the IBM Europe Institute Workshop on Large Scale Eigenvalue Problems, Oberlech, Austria, July 8–12, 1985, J. Cullum and R. A. Willoughby, eds., North-Holland, Amsterdam, 1986, pp. 193–240.

[8] ———, *Computing eigenvalues of large matrices, some Lanczos algorithms and a shift and invert strategy*, Research Report RC 15835(#67507) 11/13/89, IBM T. J. Watson Research Center, Yorktown Heights, NY; in Proc. 5th Instituto de Investigaciones en Mathematicas Aplicadas y en Sistemas, Universidad Nacional Autonomada Mexico, 5th International Workshop on Numerical Analysis, Merida, Mexico, January 3–6, 1989, S. Gomez and J. P. Hennart, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, March 1991.

[9] V. FABER AND T. MANTEUFFEL, *Necessary and sufficient conditions for the existence of a conjugate gradient method*, SIAM J. Numer. Anal., 21 (1984), pp. 352–362.

[10] ———, *Orthogonal error methods*, SIAM J. Numer. Anal., 24 (1987), pp. 170–187.

[11] R. FLETCHER, *Conjugate gradient methods for indefinite systems*, in Numerical Analysis Dundee 1975, G. A. Watson, ed., Lecture Notes in Mathematics 506, Springer-Verlag, New York, 1976, pp. 73–89.

[12] R. FREUND, *Conjugate gradient type methods for linear systems with complex symmetric coefficient matrices*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 425–448.

[13] R. W. FREUND, M. H. GUTKNECHT, AND N. M. NACHTIGAL, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices, Part* I, RIACS Tech. Report 90.45, NASA Ames Research Center, Moffett Field, CA, November 1990.

[14] R. W. FREUND AND N. NACHTIGAL, *An implementation of the look-ahead Lanczos algorithm for non-Hermitian matrices, Part* II, RIACS Tech. Report 90.46, NASA Ames Research Center, Moffett Field, CA, November 1990.

[15] ———, QMR: *A quasi-minimal residual method for non-Hermitian linear systems*, Numer. Math., 60 (1991), pp. 315–339.

[16] A. GREENBAUM, *Behavior of slightly perturbed Lanczos and conjugate-gradient recurrences*, Linear Algebra Appl., 113 (1989), pp. 7–63.

[17] I. GUSTAFSSON, *Stability and rate of convergence of modified incomplete Cholesky factorization methods*, Ph.D. thesis, Chalmers University of Technology and the University of Goteborg, Goteborg, Sweden, April 1979.

[18] M. H. GUTKNECHT, *The unsymmetric Lanczos algorithms and their relations to Padé approximation, continued fractions, and the QD algorithm*, in Proc. Copper Mountain Conference on Iterative Methods, April 1–5, 1990.

[19] ———, *A completed theory of the unsymmetric Lanczos process and related algorithms, Part* I, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 594–639.

[20] ———, *A completed theory of the unsymmetric Lanczos process and related numerical algorithms, Part* II, IPS Research Report 90-16, ETH-Zentrum, Zurich, Switzerland, Sept. 1990; SIAM J. Matrix Anal. Appl., submitted.

[21] M. R. HESTENES AND E. L. STIEFEL, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.
[22] A. S. HOUSEHOLDER, *The Theory of Matrices in Numerical Analysis*, Dover, New York, 1964.
[23] K. C. JEA AND D. M. YOUNG, *On the simplification of generalized conjugate-gradient methods for nonsymmetrizable linear systems*, Linear Algebra Appl., 52/53 (1983), pp. 399–417.
[24] W. D. JOUBERT, *Generalized conjugate gradient and Lanczos methods for the solution of nonsymmetric systems of linear equations*, Ph.D. thesis and Report CNA-238, Center for Numerical Analysis, University of Texas, Austin, TX, January 1990.
[25] ———, *Iterative methods for the solution of nonsymmetric systems of linear equations*, Report CNA-242, Center for Numerical Analysis, University of Texas at Austin, Austin, TX, February 1990.
[26] W. D. JOUBERT AND T. A. MANTEUFFEL, *Iterative methods for nonsymmetric linear systems*, in Iterative Methods for Large Linear Systems, D. R. Kincaid and L. J. Hayes, eds., Academic Press, Boston, 1990, pp. 149–171.
[27] W. D. JOUBERT AND D. M. YOUNG, *Necessary and sufficient conditions for the simplification of generalized conjugate gradient algorithms*, Linear Algebra Appl., 88/89 (1987), pp. 449–485.
[28] S. K. KIM AND A. T. CHRONOPOULOS, *An efficient nonsymmetric Lanczos method on parallel vector computers*, Tech. Report TR 90-38, Computer Science Department, University of Minnesota, Minneapolis, MN, July 1990.
[29] C. LANCZOS, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Res. Nat. Bur. Standards, 45 (1950), pp. 255–282.
[30] H. P. LANGTANGEN AND A. TVEITO, *A numerical comparison of conjugate gradient-like methods*, Comm. Appl. Numer. Methods, 4 (1988), pp. 793–798.
[31] R. T. MCLAY, *Finite element simulation of coupled fluid flow, heat transfer and magnetic fields with applications to welding*, Ph.D. thesis, Department of Aerospace Engineering and Engineering Mechanics, University of Texas at Austin, Austin, TX, August 1988.
[32] J. A. MEIJERINK AND H. A. VAN DER VORST, *An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix*, Math. Comp., 31 (1977), pp. 148–162.
[33] C. C. PAIGE AND M. A. SAUNDERS, LSQR: *an algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71.
[34] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice–Hall, Englewood Cliffs, NJ, 1980.
[35] ———, *Reduction to tridiagonal form and minimal realizations*, SIAM J. Matrix Anal. Appl., 13 (1992), pp. 567–593.
[36] B. N. PARLETT AND D. TAYLOR, *A look ahead Lanczos algorithm for unsymmetric matrices*, Report PAM-43, Center for Pure and Applied Mathematics, University of California, Berkeley, CA, 1981.
[37] B. N. PARLETT, D. R. TAYLOR, AND Z. A. LIU, *A look-ahead Lanczos algorithm for unsymmetric matrices*, Math. Comp., 44 (1985), pp. 105–124.
[38] H. L. ROYDEN, *Real Analysis*, Second Edition, MacMillan, New York, 1968.
[39] W. RUDIN, *Real and Complex Analysis*, McGraw-Hill, New York, 1966.
[40] H. RUTISHAUSER, *Beiträge zur Kenntnis des Biorthogonalisierungs-Algorithmus von Lanczos*, Z. Angew. Math. Phys., 4 (1953), pp. 35–56. (In German.)
[41] Y. SAAD, *The Lanczos biorthogonalization algorithm and other oblique projection methods for solving large unsymmetric systems*, SIAM J. Numer. Anal., 19 (1982), pp. 485–506.
[42] Y. SAAD AND M. H. SCHULTZ, *Conjugate gradient-like algorithms for solving nonsymmetric linear systems*, Math. Comp., 44 (1985), pp. 417–424.
[43] ———, GMRES: *a generalized minimal residual algorithm for solving nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 856–869.
[44] S. SAKS, *The Theory of the Integral*, G. E. Stechert, New York, 1937.
[45] P. SONNEVELD, CGS, *a fast Lanczos-type solver for nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 36–52.
[46] D. R. TAYLOR, *Analysis of the look ahead Lanczos algorithm*, Ph.D. thesis, Department of Mathematics, University of California, Berkeley, CA, November 1982; Report PAM-108, Center for Pure and Applied Mathematics, University of California, Berkeley, CA.
[47] H. A. VAN DER VORST, *Conjugate gradient methods*, in Proc. Modern Numerical Algorithms for Supercomputers: A Tutorial Seminar, October 9–13 1989, University of Texas System Center for High Performance Computing, Austin, TX, pp. 268–324.

[48] H. A. VAN DER VORST, Bi-CGSTAB: *a fast and smoothly converging variant of* Bi-CG *for the solution of nonsymmetric linear systems*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 631–644.

[49] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.

[50] T. YAMAMOTO, *On Lanczos' algorithm for tri-diagonalization*, J. Sci. Hiroshima Univ. Ser. A-I, 32 (1968), pp. 259–284.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 944–961, July 1992

015

ORDERING METHODS FOR PRECONDITIONED CONJUGATE GRADIENT METHODS APPLIED TO UNSTRUCTURED GRID PROBLEMS*

E. F. D'AZEVEDO†, P. A. FORSYTH‡, AND WEI-PAI TANG‡

Abstract. It is well known that the ordering of the unknowns can have a significant effect on the convergence of preconditioned conjugate gradient (PCG) methods. There has been considerable experimental work on the effects of ordering for finite difference problems. In many cases, good results have been obtained with preconditioners based on diagonal, spiral, red/black reduced system orderings, or some others. The reduced system approach generally gives rapid convergence. There has been comparatively less work on the effect of ordering for finite element problems on unstructured meshes. In this paper, an ordering technique for unstructured grid problems is developed. At any stage of the partial elimination, the next pivot node is selected so as to minimize the norm of the discarded fill matrix. Numerical results are given for model problems and for problems arising in groundwater contamination. Computations are reported for two-dimensional triangular grids, and for three-dimensional tetrahedral grids. The examples show that ordering is important even if a reduced system (based on a generalized red/black ordering) method is used.

Key words. ordering method, preconditioned conjugate gradient method

AMS(MOS) subject classifications. 65F10, 76S05

1. Introduction.

It is well known that the ordering of the unknowns can affect the convergence behavior of preconditioned conjugate gradient methods. There have been many studies of the use of various ordering techniques coupled with incomplete LU (ILU) factorization preconditioners [3]–[5], [7] [11], [12], [13]–[16], [17], [25], [29], [30], [34], [35], [38], [39].

Most of these studies have been restricted to the analysis of partial differential equation problems arising from five- or seven-point finite difference discretizations in two or three dimensions. For the most part, these ordering methods are based on the graph of the matrix, and do not use actual values of the matrix entries.

In general, the results can be summarized as follows:

1. Random orderings are poor.
2. "Natural" row orderings perform quite well.
3. Fill-reducing orderings, such as minimum degree and nested dissection, are poor.
4. Reduced systems are very effective (red/black ordering of a bipartite graph, and exact elimination of the red nodes).

Incomplete factorizations can be classified by the allowed level of fill [17], [29], [38], [39]. A rigorous definition of the level is given in §2.2. It is generally agreed that level-1 or level-2 ILU factorizations are usually the best in terms of total work required for

* Received by the editors February 12, 1990; accepted for publication (in revised form) March 16, 1991. This work was supported by the Natural Sciences and Engineering Research Council of Canada, by the Information Technology Research Centre, which is funded by the Province of Ontario, and by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U.S. Department of Energy, under contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, Inc., through an appointment to the U.S. Department of Energy Postgraduate Research Program administered by Oak Ridge Associated Universities.

† Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1. Present address, Mathematical Sciences Section, Oak Ridge National Laboratory, Oak Ridge, Tennessee 37831.

‡ Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada N2L 3G1.

convergence [3], [4], [17], [25], [29], [38]. Consequently, at least for five- or seven-point molecules, a reduced system level-1 or level-2 ILU is a popular choice.

For finite element–type discretizations on unstructured grids, it is possible to define a *generalized red/black ordering*, and use a reduced system preconditioner, even in the finite element case. Red nodes are defined as being connected only to black nodes, while black nodes have at least one red neighbor [4], [20]. The red nodes are eliminated exactly and the remaining matrix constitutes the reduced system. However, if the average node connectivity is large, then the number of red nodes can be small, and this approach may not be very advantageous. It is also not clear how to order the remaining black nodes in the reduced system.

An ordering based solely on the graph of the matrix cannot detect anisotropies. For example, consider the equation

$$(KU_x)_x + U_{yy} = f(x, y) \tag{1.1}$$

with $K \gg 1$. If this equation is discretized using the usual five-point molecule, then, as will be shown in the numerical results (see §4), the effect of ordering on the convergence of PCG is very large.

Note that numerical anisotropy is very common in practical situations. Even if the equation coefficients are not anisotropic, it is often the case that the grids are very anisotropic. This is especially common in geophysical applications (reservoir simulation, groundwater contamination) where the vertical distance is often one or two orders of magnitude smaller than the horizontal distances.

The idea of developing an ILU factorization based on a drop tolerance has been suggested by Munksgaard [28], Zlatev [40], and Tuff and Jennings [37]. In this case, the sparsity pattern of the ILU was determined by a drop tolerance. However, if the initial ordering is poor then the fill may not decay very rapidly, leading to a dense ILU factorization, and hence an inefficient PCG method.

The objective of this paper is to develop an *automatic* method for producing an ordering that reduces the discarded fill in an ILU factorization. We are particularly interested in solving time-dependent problems, which are typical of groundwater contamination modeling. In this case, the order of magnitude of the matrix coefficients is determined by time-invariant physical parameters. Consequently, an ordering can be determined at the start of a simulation and used for many Newton iterations. The cost of the ordering can then be amortized over many solves [6], [10].

The ordering method used in this work assumes that the level of fill is given, and then the ordering is selected so as to minimize discarded fill. Comprehensively varying the level of fill as well as the ordering is also a possibility, but is beyond the scope of this work.

Note that if a very high level of fill is allowed in the ILU factorization, then a fill-reducing ordering such as reverse Cuthill–McKee (RCM) [8], [27] may become efficient. This is because a higher level of fill can be retained for a given number of nonzeros in the ILU factorization, compared to orderings that do not try to minimize the number of nonzeros in the incomplete factors. For a five-point molecule on a square grid, RCM has fewer nonzeros in the factors for level-3 factorizations (and higher) compared to natural row orderings [3]. Of course, in the extreme case that the allowed level of fill becomes infinite, then fill-reducing orderings are clearly more efficient than other orderings (since the method is now a direct technique). Consequently, we shall concern ourselves with low levels of fill in the following, since this is usual in practice.

Natural or row orderings applied to structured finite difference grids have the

property that nodes ordered consecutively are (graph) neighbors of previously ordered nodes. An obvious generalization of this idea to unstructured grids is an RCM ordering.

Test results will be presented for some matrices generated by two- and three-dimensional groundwater contamination simulations (triangular and tetrahedral elements). These problems have large jump discontinuities in absolute permeability [24], and therefore constitute a severe test of the ordering algorithm. Some results are also given for some standard two-dimensional model problems [15], [36].

The results are compared using natural (row) ordering, RCM, and the minimum discarded fill (MDF) technique developed in this work. Factorization levels are varied from levels 0–3, and both full and reduced system methods are used.

2. Minimum discarded fill ordering.

2.1. Motivation by matrix formulation. The Cholesky factorization of an $n \times n$ symmetric positive definite matrix A can be described by the following equations:

$$A = A_0 = \begin{bmatrix} d_1 & \gamma_1{}^t \\ \gamma_1 & B_1 \end{bmatrix} = L_1 \begin{bmatrix} 1 & 0 \\ 0 & A_1 \end{bmatrix} L_1{}^t, \tag{2.1}$$

where

$$L_1 = \begin{bmatrix} \sqrt{d_1} & 0 \\ \gamma_1/\sqrt{d_1} & I_{n-1} \end{bmatrix}, \qquad A_1 = B_1 - \gamma_1 \gamma_1{}^t/d_1. \tag{2.2}$$

At the kth step,

$$A_{k-1} = \begin{bmatrix} d_k & \gamma_k{}^t \\ \gamma_k & B_k \end{bmatrix} = L_k \begin{bmatrix} 1 & 0 \\ 0 & A_k \end{bmatrix} L_k{}^t, \tag{2.3}$$

where

$$L_k = \begin{bmatrix} \sqrt{d_k} & 0 \\ \gamma_k/\sqrt{d_k} & I_{n-k} \end{bmatrix}, \qquad A_k = B_k - \gamma_k \gamma_k{}^t/d_k. \tag{2.4}$$

Here I_k denotes a $k \times k$ identity matrix, d_k a scalar, and γ_k is a column vector of length $n-k$. The matrix A_k is the $(n-k) \times (n-k)$ submatrix that remains to be factored after the first k steps of the factorization.

In the incomplete factorization of matrix A, some of the entries in the factor are discarded to prevent excessive fill and computation. Let matrix F_k contain the discarded values. Then the incomplete factorization proceeds with the perturbed matrix

$$\tilde{A}_k = A_k - F_k = B_k - \gamma_k \gamma_k{}^t/d_k - F_k. \tag{2.5}$$

The minimum discarded fill ordering is motivated by the observation that a small discarded fill matrix F_k would produce a more "authentic" factorization for matrix A. We define the *discarded fill* for eliminating the kth node as the Frobenius norm of the discarded fill matrix F_k,

$$\|F_k\|_F = \left(\sum_{i \geq 1} \sum_{j \geq 1} |f_{ij}^{(k)}|^2 \right)^{1/2}. \tag{2.6}$$

The discarded fill at the kth step for an arbitrary node is similarly defined by performing a symmetric permutation that exchanges this node with the kth node. To determine the sparsity pattern for matrix F_k that will yield a high-quality preconditioner is still a very interesting research subject. A current popular choice is to discard the fills that have a "higher fill level" during the incomplete factorization [23]. The simplest strategy is ILU(0) where all new fill is discarded and ILU(1) where only level-1 fills produced by eliminating original nonzeros are retained but higher-level fill produced in the elimination of level-1 fill is discarded. The notion of "fill level" will be defined more precisely through the graph model presented in §2.2.

The basic idea of the minimum discarded fill-ordering scheme is to eliminate the node with the minimum discarded fill at each stage of the incomplete factorization. This scheme can be considered as the numerical analogue of the minimum deficiency ordering strategy [14] for minimizing the amount of fill. The most computationally intensive calculations are in the updating of new discard values after each stage of the factorization process.

2.2. Graph model. In this section we present a graph model [31], [33] for describing the factorization process as a series of node eliminations. The graph model provides invaluable insight into the minimum discarded fill ordering.

To simplify notation, we present a symmetric case and assume the elimination sequence is $v_1, v_2, \cdots, v_n$. Let graph $G_k = (\mathcal{V}_k, \mathcal{E}_k)$, $k = 0, 1, \cdots, n-1$ be the graph corresponding to matrix $A_k = \left[a_{ij}^{(k)}\right]$ of (2.4). The vertex set and edge set are defined as

$$\mathcal{V}_k = \{v_{k+1}, v_{k+2}, \cdots, v_n\}, \qquad \mathcal{E}_k = \left\{(v_i, v_j) \mid a_{ij}^{(k)} \neq 0\right\}. \tag{2.7}$$

We assume each vertex has a self-loop edge (v_i, v_i) and each edge (v_i, v_j) has a value of $a_{ij}^{(k)}$.

The notion of "fill level" can be defined through reachable sets [22] in the graph G_0. Let $\mathcal{S}$ be a subset of the node set, $\mathcal{S} \subset \mathcal{V}_0$, and nodes $u, v \notin \mathcal{S}$. Node u is said to be reachable from a vertex v through $\mathcal{S}$ if there exists a path $(v, u_1, \cdots, u_m, u)$ in graph G_0, such that each $u_i \in \mathcal{S}$, $1 \leq i \leq m$. Note that m can be zero, so that any adjacent pair of nodes $u, v \notin \mathcal{S}$, is reachable through $\mathcal{S}$. The reachable set of v through $\mathcal{S}$ is denoted by

$$Reach(v, \mathcal{S}) = \{u \mid u \text{ is reachable from } v \text{ through } \mathcal{S}\}. \tag{2.8}$$

Let $\mathcal{S}$ be the set of eliminated nodes so far, $\{v_1, \cdots, v_k\}$, and let $v_j \in Reach(v_i, \mathcal{S})$ with the *shortest* path $(v_i, u_1, \cdots, u_m, v_j)$, and nodes u's in $\mathcal{S}$ are eliminated nodes. We define the fill level for entry $a_{ij}^{(k)}$ to be the length of the shortest path from v_i to v_j minus one, i.e., $Level(a_{ij}^{(k)}) = m$. We initially set

$$Level(a_{ij}^{(0)}) = \begin{cases} 0 & \text{if } a_{ij} \neq 0, \\ \infty & \text{otherwise.} \end{cases} \tag{2.9}$$

Since $Level(a_{ij}^{(k)})$ is defined by reachable sets through $\{v_1, \cdots, v_k\}$, as more nodes are eliminated, there may be a shorter path between v_i and v_j. Thus as the elimination proceeds, the fill levels are modified by

$$Level(a_{ij}^{(k)}) := \min\left(Level(a_{ik}^{(k-1)}) + Level(a_{kj}^{(k-1)}) + 1,\ Level(a_{ij}^{(k-1)})\right). \tag{2.10}$$

It is possible to define a fill level independent of k, if the order of the unknowns is predetermined. In this application, however, the order of the unknowns is dynamically changing during the incomplete factorization. A predetermined level, therefore, is not practical.

The elimination of v_k to form A_k can be modeled as a graph transformation [33]

$$a_{ij}^{(k)} = \begin{cases} a_{ij}^{(k-1)} - \dfrac{a_{ik}^{(k-1)} a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}} & \text{if } (v_i, v_k) \text{ and } (v_k, v_j) \in \mathcal{E}_{k-1}, \\ a_{ij}^{(k-1)} & \text{otherwise.} \end{cases}$$

Note that if $a_{ij}^{(k-1)}$ is a zero entry, $(v_i, v_j) \notin \mathcal{E}_{k-1}$, then the elimination of their common neighbor v_k would create at position $a_{ij}^{(k)}$, a new fill of value

$$0 - \frac{a_{ik}^{(k-1)} a_{kj}^{(k-1)}}{a_{kk}^{(k-1)}}.$$

With the minimum discarded fill reordering that corresponds to an ILU(0) factorization, only entries with fill level zero are kept, i.e., all new fill-ins must be discarded. If node v_m were eliminated after the kth stage of the incomplete factorization (2.5), the *discarded fill* value for node v_m would be

$$discard(v_m) = \left[\sum_{(v_i, v_j) \in \mathcal{F}} \left(\frac{a_{im}^{(k-1)} a_{mj}^{(k-1)}}{a_{mm}^{(k-1)}} \right)^2 \right]^{1/2}, \tag{2.11}$$

where

$$\mathcal{F} = \{(v_i, v_j) \mid (v_i, v_j) \notin \mathcal{E}_{k-1},\ (v_i, v_m) \in \mathcal{E}_{k-1},\ (v_m, v_j) \in \mathcal{E}_{k-1}\}, \tag{2.12}$$

and $G_{k-1} = (\mathcal{V}_{k-1}, \mathcal{E}_{k-1})$ is the graph corresponding to matrix A_{k-1}. The minimum discarded fill strategy can be generalized to correspond to ILU(ℓ) factorization by accounting for only new fill-ins with fill level greater than ℓ in the computing of *discarded fill* value. Then set $\mathcal{F}$ in (2.11) is taken to be

$$\begin{aligned} \mathcal{F} = \{(v_i, v_j) \mid (v_i, v_j) \notin \mathcal{E}_{k-1},\ (v_i, v_m) \in \mathcal{E}_{k-1}, \\ (v_m, v_j) \in \mathcal{E}_{k-1} \text{ and } Level(a_{ij}^{(k)}) > \ell\}. \end{aligned} \tag{2.13}$$

In the following discussion, we shall denote MDF(ℓ) as the minimum discarded fill ordering corresponding to an ILU(ℓ) factorization.

Observation 1. For the MDF(ℓ) algorithm, discard values for all nodes can be initially precomputed. At each elimination step, if v_k is chosen to be eliminated, only discard values of the neighbors of v_k need to be updated.

Observation 2. The MDF(0) algorithm overwrites the original matrix A with the corresponding ILU(0) incomplete factorization.

2.3. MDF(ℓ) algorithm. The MDF(ℓ)-ordering algorithm can be described as follows.

Initialization:

$A = A_0$

for each $a_{ij} \neq 0$

$Level(a_{ij}) := 0$

end

for each node v_i

Compute the discarded fill value $discard(v_i)$ from (2.11) and (2.13).

end

for $\mathbf{k} = 1 \cdots n-1$

Choose a node v_m that has the minimum $discard(v_m)$ as the next pivot node (see §2.4 on tie breaking).

Update the decomposition,

$$\tilde{A}_k = B_k - \gamma_k \gamma_k^t / d_k - F_k, \quad \text{where } P_k A_{k-1} P_k^t = \begin{bmatrix} d_k & \gamma_k \\ \gamma_k^t & B_k \end{bmatrix}.$$

P_k is permutation matrix to exchange v_k with v_m and F_k is the matrix of discarded fill-in entries,

$$F_{ij}^{(k)} := \begin{cases} \dfrac{a_{im}^{(k-1)} a_{mj}^{(k-1)}}{a_{mm}^{(k-1)}} & \text{if } Level(a_{ij}^{(k)}) > \ell \text{ and } a_{ij}^{(k-1)} = 0, \\ 0 & \text{otherwise.} \end{cases}$$

Update the discarded values of v_m's neighbors.

Update the fill level of entries in $\tilde{A}_k$ by (2.10).

end

2.4. Tie breaking. There are often cases where many nodes will have the same (typically zero) discarded fill. Several possible tie-breaking strategies are investigated in the following.

Ties can be broken by selecting the nodes that have the smallest degree in the incomplete factorization (smallest number of nonzeros in the row). Another possibility is to use the node with the smallest deficiency (smallest number of new nonzero fill elements introduced if this node is used as a pivot) [14]. If there are still ties remaining, then the node with the smallest discarded fill from a previous stage of the incomplete factorization is selected first. If further ties exist, the unordered node with the smallest original number is selected. The minimum deficiency and minimum degree strategies attempt to minimize the number of fill elements in the case of ties.

Tests were run using minimum deficiency, minimum degree, and random tie breaking for all our test problems. On average, the minimum degree strategy required 2 percent more solution time than minimum deficiency, while random tie breaking required 13 percent more solution time than minimum deficiency. Consequently, all test results will be reported using minimum deficiency tie breaking. Our tests also show that these tie-breaking algorithms have little effect on the cost of the MDF ordering. Therefore, no timing comparison is given.

2.5. An example of MDF(0) ordering. In this section we consider an example of an MDF(0) ordering on the model Laplace problem. The Laplace problem with Dirichlet boundary conditions is discretized by the five-point molecule on a regular 4×4 grid. Figure 2.1 displays the grid with initial natural row ordering.

The initial discard values for the four corner nodes $\{v_1, v_4, v_{13}, v_{16}\}$ are equal to

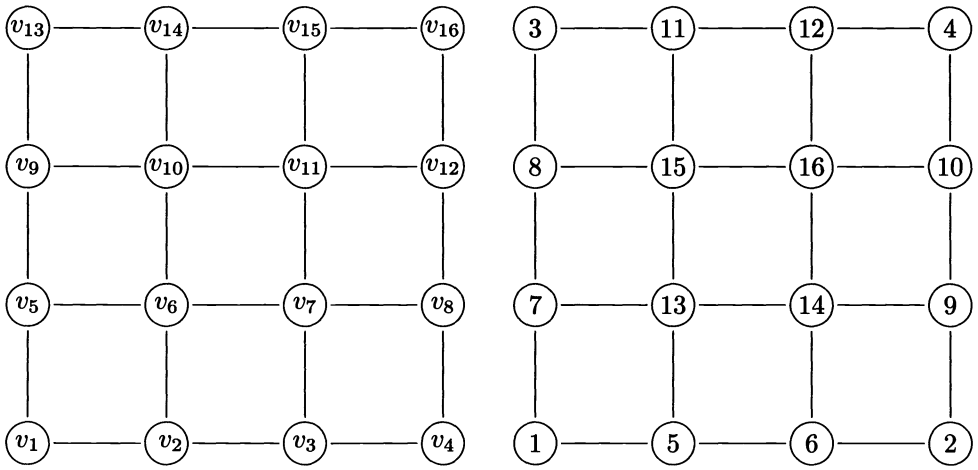


FIG. 2.1. *Natural row ordering and final* MDF(0) *ordering.*

$discard(v_1) = \sqrt{0.125} \approx 0.354$. The discard values for the boundary nodes $\{v_2, v_3, v_5, v_8, v_9, v_{12}, v_{14}, v_{15}\}$ are equal to $discard(v_2) = \sqrt{0.375} \approx 0.612$. Similarly, discard values of interior nodes $\{v_6, v_7, v_{10}, v_{11}\}$ are equal to $discard(v_6) = \sqrt{0.75} \approx 0.866$. The corner nodes have the smallest discard values and should be eliminated first. Note that these corner nodes are not connected and their discard values are unaffected by the elimination of other corner nodes. After the corner node v_1 is eliminated, its boundary neighbor node v_2 has new discard value given as $discard(v_2) = \sqrt{32}/15 \approx 0.377$. After the four corner nodes are eliminated, the discard values for the interior nodes are unchanged. For example, $discard(v_2) = \sqrt{32}/15 \approx 0.377$, and discard values for interior nodes are unchanged at $discard(v_6) = \sqrt{0.75} \approx 0.866$. The boundary nodes have the smallest discard values. Boundary node v_2 is chosen by the tie-breaking strategy, and its neighbor v_3 with $discard(v_3) = 0$, could be eliminated next with no fill. Similarly, nodes v_5 and v_9, v_8 and v_{12}, and v_{14} and v_{15} will be eliminated in sequence. Node v_{15} will be eliminated next since $discard(v_{15}) = 0$. Further computation would show that an MDF(0) ordering for this example is given by Fig. 2.1.

2.6. MDF(1) ordering. In §2.5 we looked at an example of minimum discarded fill ordering that corresponds to an ILU(0) incomplete factorization. This ordering is abbreviated as the MDF(0) ordering. The MDF(1) ordering is the extension of this basic strategy to correspond to an ILU(1) incomplete factorization.

Although an MDF(0) ordering overwrites the original matrix A with its ILU(0) factorization, MDF(1) does not *exactly* reproduce the ILU(1) factorization. There are some subtleties in the computing of level-2 contributions that happen to fall upon nonzero entries. Consider the scenario in Fig. 2.2, where v_1 and v_2 have been eliminated causing fill contribution to edges (v_3, v_4) and (v_3, v_5). Suppose we wish to eliminate v_3 next. This would cause a level-2 fill contribution to edge (v_4, v_5). The subtle problem is in deciding whether this (v_4, v_5) level-2 fill should be discarded. Note that if we knew in advance that v_6 would be eliminated before v_4 and v_5, then this elimination of v_6 would cause a level-1 fill contribution to edge (v_4, v_5). Thus this level-2 contribution of (v_4, v_5) would fall on a nonzero entry and may be accepted. The MDF(1) algorithm always discards (pessimistically) the level-2 fill con-

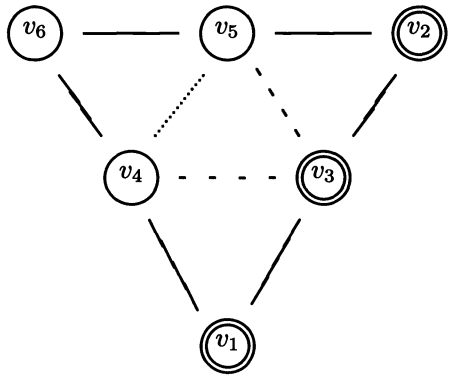


FIG. 2.2. *Level-2 fill at* (v_4, v_5).

tribution (v_4, v_5).

Axelsson and Gustafsson [1] have observed that reduced system preconditioners are very effective (red/black partitioning of nodes and exact elimination of the red nodes). It is interesting to note that a generalized red/black partitioning of the nodes is an MDF(1) ordering of the red nodes. A generalized red/black partitioning of a graph has the property that each red node has *only* black neighbors and each black node has at least one red neighbor. In the special case where each black node has *only* red neighbors, the graph is bipartite, or the corresponding matrix is two-cyclic. Note that in this special case the red nodes form an independent set so that the discard value for each red node is unaffected by elimination of other red nodes; therefore, the elimination order of the red nodes is immaterial.

Remark 2.1. A generalized red/black partitioning of the nodes is an MDF(1) ordering of the red nodes.

In an ILU(1) incomplete factorization, all level-1 fill is accepted. Hence if a vertex has no eliminated neighbors, its discard value is zero and would be a candidate for selection by the MDF(1) criterion. A generalized red/black partition of the nodes orders red nodes first, and these red nodes have (by definition) zero discarded fill.

Remark 2.2. If a matrix is symmetric and two-cyclic (its graph is bipartite), then an MDF(0) ordering on the reduced matrix formed with the bipartite red/black partition is an MDF(1) reordering on the original matrix.

The reduced matrix is obtained by exact elimination of the red nodes. Since the graph is bipartite, there are no black-to-black connections in the original graph. Therefore, all black-to-black connections in the reduced matrix are level-1 fill. Level-3 fill from the original graph is exactly the new level-1 fill generated from the reduced matrix. Thus an MDF(0) ordering on the reduced matrix is an MDF(1) reordering on the original matrix.

While a generalized red/black partition is an MDF(1) ordering of the red nodes, an MDF(1) ordering may or may not produce a generalized red/black partitioning. Consider a tridiagonal matrix. All nodes initially have no level-1 discarded fill. Consequently, the ordering depends crucially on the tie-breaking strategy. For example, either a red/black partition or the perfect elimination order (no fill) would be consistent with the MDF strategy, in this case.

Although the description of MDF(0) and MDF(1) orderings has been given for

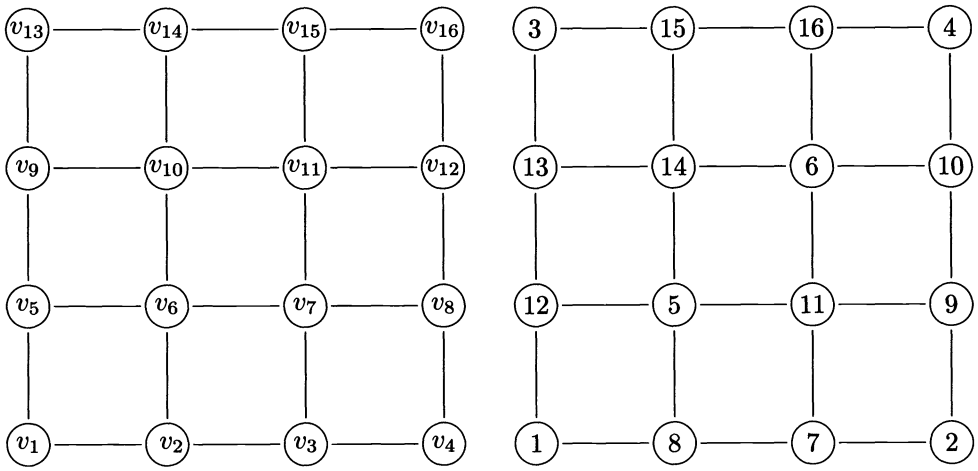


FIG. 2.3. *Natural row ordering and final* MDF(1) *ordering.*

symmetric matrices, it is clearly trivial to generalize to the case of a nonsymmetric matrix having a symmetric incidence matrix. This is, in fact, how we have implemented the MDF(ℓ)-ordering algorithms. Of course, the minimum discarded fill algorithm can also be applied to matrices with nonsymmetric nonzero structure, and our findings will be reported in a forthcoming paper [9]

2.7. An example of MDF(1) ordering. We consider an example of an MDF(1) ordering on the model Laplace problem used in §2.5. The minimum deficiency criterion is used for tie breaking.

Since level-1 fill entries are accepted, initially all nodes have zero discard values. The four corner nodes $\{v_1, v_4, v_{13}, v_{16}\}$ are chosen based on deficiency. Nodes v_6 and v_{11} are chosen next since these have zero discard values. Nodes $\{v_3, v_8, v_9, v_{14}\}$ have the same discard value $discard(v_3) \approx 0.094$, nodes $\{v_2, v_5, v_{12}, v_{15}\}$ have discard value $discard(v_2) \approx 0.226$, nodes $\{v_7, v_{10}\}$ have discard value $discard(v_7) \approx 0.381$. By the tie-breaking criterion, v_3 will be the next eliminated node. Then v_2 followed by v_8 and v_{12} are eliminated with no new fill. Among the uneliminated nodes, v_7 has the smallest discard value of $discard(v_7) \approx 0.056$. After v_7 is eliminated, there is a perfect elimination sequence of v_5, v_9, v_{10}, v_{14}, and v_{15}. The final ordering is shown in Fig. 2.3.

3. Test problems. The minimum discarded fill orderings were tested on a variety of problems. For Problems 1, 2, and 4 below, the matrices are only positive semidefinite. The solution is determined only to within a constant. These matrices can be made definite by fixing the solution at a single node. However, the conjugate gradient method still converges even if this is not done. In fact, if the solution is fixed at a node, the algorithm actually converges more slowly [2], [21]. For Problems 1, 2, and 4, the matrices are left as semidefinite.

3.1. Problem 1 (STRONGX). The first problem solves the equation

$$\frac{\partial}{\partial x}\left(K_x \frac{\partial P}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial P}{\partial y}\right) = -q \tag{3.1}$$

on the region $x \in [0,1]$, $y \in [0,2]$, using a five-point cell-centered finite difference discretization [32] with Neumann boundary conditions, where $K_x = 1000$ and $K_y = 1$.

Let $h = 1/30$ be the cell spacing; let $n_x = 30$, $n_y = 60$ be the number of cells in the x and y direction, respectively. The source term is

$$q(x,y) = \begin{cases} -1/h^2 & \text{if } (x,y) = (h/2, h/2), \\ 1/h^2 & \text{if } (x,y) = (1 - h/2, 2 - h/2), \\ 0 & \text{elsewhere.} \end{cases}$$

3.2. Problem 2 (STRONGY). This problem is identical to Problem 1, except that the anisotropic property is reversed: $K_x = 1$, $K_y = 1000$.

3.3. Problem 3 (LAPD5). This is Laplace's equation on the unit square with Dirichlet boundary conditions, as used in [15]. The usual five-point finite difference discretization was used on a regular 30×30 grid.

3.4. Problem 4 (STONE). This problem is Stone's third problem [36]. The equation

$$\frac{\partial}{\partial x}\left(K_x \frac{\partial P}{\partial x}\right) + \frac{\partial}{\partial y}\left(K_y \frac{\partial P}{\partial y}\right) = -q \tag{3.2}$$

was discretized on the unit square using a vertex-centered finite difference technique [32], with Neumann boundary conditions. If the node spacing is $h = 1/30$, then

$$x_i = ih, \quad y_j = jh, \quad 0 \le i, j \le 30.$$

We will refer to the location of the source and sink terms by

$$q(x_i, y_j) = q(i,j)$$

in the following and in Fig. 3.1, which shows the problem domain.

The values of K_x, K_y, and q were

$$(K_x, K_y) = \begin{cases} (1,100) & \text{if } (x_i,y_j) \in B, \quad 14 \le i \le 30, \ 0 \le j \le 16, \\ (100,1) & \text{if } (x_i,y_j) \in C, \quad 5 \le i \le 12, \ 5 \le j \le 12, \\ (0,0) & \text{if } (x_i,y_j) \in D, \quad 12 \le i \le 19, \ 21 \le j \le 28, \\ (1,1) & \text{if } (x_i,y_j) \in A, \end{cases} \tag{3.3}$$

$$\begin{aligned} &q_1(3,3) = 1.0, \quad q_2(3,27) = 0.5, \quad q_3(23,4) = 0.6, \\ &q_4(14,15) = -1.83, \quad q_5(27,27) = -0.27. \end{aligned} \tag{3.4}$$

A 31×31 grid was used, and an harmonic average was used to define K_x and K_y [3] at cell boundaries.

Test problems 5–7 are derived from two- and three-dimensional pressure equations arising in groundwater contamination simulations [18], [24]. The pressure equation is essentially equation (3.2). Since the actual values of K_x, K_y, q, and the boundary conditions are quite complicated, only a brief description of these problems will be given. The choice of boundary conditions (fixed pressure) resulted in a sparse right-hand vector.

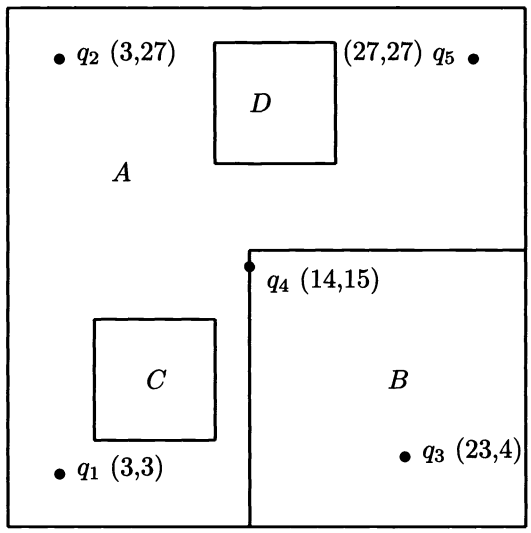


FIG. 3.1. *Stone's third problem.*

3.5. Problem 5 (REFINE2D). A finite element method using linear triangular basis functions was used to discretize this problem. In this example, K_x and K_y were constant. The triangulation is such that the resulting equation is an M-matrix [18]. The grid was constructed by first defining a very coarse triangulation, and then repeatedly defining finer grids by subdividing a triangle into four smaller triangles with new nodes determined by the nodes of the original triangle, and the midpoints of the original triangle edges. This problem had 1161 nodes, and is described in more detail in [18]. The nodes are originally ordered using an RCM ordering.

3.6. Problem 6 (FE2D). A finite element method using linear triangular basis functions was also used on this problem. However, in this example, K_x and K_y (3.2) varied by four orders of magnitude. The grid, which had 1521 nodes, was defined by constructing a distorted quadrilateral grid, and then triangulating in the obvious manner. A Delaunay-type edge swap was used to produce an M-matrix. The original ordering for this problem used a natural or lexicographic ordering based on the distorted quadrilaterals. This problem is described in more detail in [18].

3.7. Problem 7 (FE3D). This problem is a three-dimensional version of equation (3.2). A finite element discretization was used, with linear basis functions defined on tetrahedra. The absolute permeabilities (K_x,K_y,K_z) varied by eight orders of magnitude (this model was derived from actual field data). The nodes were defined on a $25 \times 13 \times 10$ grid (3250 nodes) of distorted hexahedra, which were then divided into tetrahedra. The resulting matrix was *not* an M-matrix, and the average node connectivity was 15. In general, it is not possible for a given node placement to obtain an M-matrix in three dimensions if linear tetrahedral elements are used [26]. The original ordering for this problem used a natural ordering based on distorted hexahedra. This problem is described in more detail in [19].

4. Results. The computations to solve the test problems 1–5 were done on a Sun SPARC SLC workstation in double precision and using

$$\|\mathbf{r}^k\|_2 \leq \varepsilon \|\mathbf{r}^0\|_2, \qquad \varepsilon = 10^{-6} \tag{4.1}$$

TABLE 4.1
Summary for test problem STRONGX.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
ORG(0)	3510	n/a	0.25	4.49(33)
ORG(1)	5221	n/a	0.27	4.64(32)
ORG(2)	6903	n/a	0.31	4.65(31)
ORG(3)	10238	n/a	0.42	5.05(30)
RCM(0)	3510	0.06	0.22	4.46(33)
RCM(1)	5221	0.07	0.27	4.65(32)
RCM(2)	6903	0.06	0.30	1.98(13)
RCM(3)	8527	0.06	0.36	2.11(13)
MDF(0)	3510	1.65	0.23	4.46(33)
MDF(1)	6867	3.38	0.31	2.01(13)
MDF(2)	7074	3.50	0.32	1.72(11)
MDF(3)	10210	7.83	0.46	1.56(9)
Reduced system (black nodes= 900, red nodes= 900)				
ORG(1)	3421	n/a	0.31	3.97(40)
ORG(3)	5060	n/a	0.36	4.10(38)
RCM(1)	3421	0.07	0.31	3.97(40)
RCM(3)	5060	0.07	0.36	1.99(18)
MDF(1)	3421	2.14	0.31	1.17(11)
MDF(3)	6584	6.54	0.44	1.00(8)

as the stopping criterion, where $\mathbf{r}^k$ is the residual vector after the kth iteration in the conjugate gradient acceleration and the zero vector is the initial guess.

Some tests were carried out using a random initial guess (random numbers between (-1,+1)), and the results were qualitatively similar. The tests were also repeated using a stopping criteria of $\varepsilon = 10^{-12}$, and the trends were similar to the results obtained with $\varepsilon = 10^{-6}$, and hence will not be shown.

The reduced system factorizations were constructed by first using a generalized red/black partitioning of the nodes. The initial red node was selected as the initial node in the given ordering. The initial ordering (ORG) was x–y natural for Problems 1–4, and RCM ordering for Problems 5–7.

The levels of fill will be defined so that all original entries in the full system have level 0. This means that the lowest level reduced system factorization will be level 1. If the original matrix has a bipartite graph, then the next level of fill in the reduced system is level 3. Note that in the finite element case, the next level of fill in the reduced system is level 2. For this reason, our definition of levels for reduced systems differs from that used previously [38]. For all reduced system methods, the ordering was determined using the reduced system. For example, RCM on the reduced system refers to the following sequence of steps: the full system is red/black ordered, the red nodes are eliminated exactly, and the reduced system is reordered using an RCM algorithm.

Table 4.1 shows the results for Problem STRONGX. This problem has a strong coupling in the x-direction. As discussed in [9], ORG ordering (x–y natural) is very poor for this example, since the entries in LU factorization decay very slowly with this ordering. This is reflected in the results for ORG ordering, full system, all levels of fill. The full system computations for RCM are poor for levels 0 and 1, but become competitive with MDF as the level of the ILU increases. MDF(0) is poor (level 0

TABLE 4.2
Summary for test problem STRONGY.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
ORG(0)	3510	n/a	0.22	8.06(60)
ORG(1)	5221	n/a	0.26	2.92(20)
ORG(2)	6903	n/a	0.30	3.02(20)
ORG(3)	10238	n/a	0.41	1.72(10)
RCM(0)	3510	0.06	0.22	8.05(60)
RCM(1)	5221	0.06	0.27	2.93(20)
RCM(2)	6903	0.07	0.30	2.87(19)
RCM(3)	8527	0.06	0.35	1.63(10)
MDF(0)	3510	1.63	0.24	8.25(60)
MDF(1)	6873	3.35	0.31	2.16(14)
MDF(2)	7064	3.48	0.31	2.32(15)
MDF(3)	10150	7.69	0.45	1.24(7)
Reduced system (black nodes = 900, red nodes = 900)				
ORG(1)	3421	n/a	0.31	1.93(19)
ORG(3)	5060	n/a	0.36	1.36(12)
RCM(1)	3421	0.06	0.31	1.94(19)
RCM(3)	5060	0.07	0.37	1.14(10)
MDF(1)	3421	2.11	0.32	1.45(14)
MDF(3)	6572	6.53	0.45	0.77(6)

factorizations cannot detect anisotropies [9]), but is much improved for level 1. Note that the ordering time for RCM varies slightly for different runs. This is due to the inaccuracy in the system timing calls.

For the reduced system, MDF(1) is much faster than either RCM(1) or ORG(1). In all cases, the amount of fill in the ILU(1) factorization is identical. This demonstrates that the orderings for reduced system factorizations can be very important. As the level of factorization on the reduced system is increased, the ORG ordering actually becomes slower, while RCM(3) shows a large improvement. However, MDF(1) is still superior to RCM(3) with less fills. If the levels are increased to very high levels, we would expect RCM to eventually become more efficient than MDF, due to the fill-reducing property of RCM as discussed in the introduction.

Note also that the cost of the MDF ordering is quite high. However, as discussed previously, we expect to carry out the ordering only once for many matrix solves, for time-dependent, nonlinear problems [6], [10].

Table 4.2 lists the results for Problem STRONGY. In this case, the ORG orderings result in rapid decay in the size of the fill entries, and hence the ORG orderings (for level greater than 0) are quite efficient. The reduced system factorizations are generally more efficient than the full system factorizations.

For the reduced system, MDF(1) is superior to RCM(1) and ORG(1). All methods have the same fill. MDF(3) is also faster than either ORG(3) or RCM(3), but at the cost of greater fill.

Problem LAPD5 (Table 4.3) has constant coefficients for all interior nodes, and as expected, all the orderings behave very similarly.

Table 4.4 shows the results for Problem STONE. The reduced system factorizations are the most efficient for this problem. Again, the reduced system MDF(1) is faster than RCM(1) or ORG(1). However, reduced systems MDF(3) and RCM(3) are quite close, especially if the factorization time is included.

TABLE 4.3
Summary for test problem LAPD5.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
ORG(0)	1740	n/a	0.11	1.73(26)
ORG(1)	2581	n/a	0.13	1.24(17)
ORG(2)	3393	n/a	0.15	1.05(14)
ORG(3)	4988	n/a	0.20	0.92(11)
RCM(0)	1740	0.04	0.11	1.74(26)
RCM(1)	2581	0.03	0.13	1.24(17)
RCM(2)	3393	0.04	0.15	0.83(11)
RCM(3)	4177	0.03	0.17	0.80(10)
MDF(0)	1740	0.84	0.11	1.67(25)
MDF(1)	3391	1.67	0.16	0.90(12)
MDF(2)	3571	1.79	0.16	0.92(12)
MDF(3)	5041	3.80	0.22	0.60(7)
Reduced system (black nodes = 450, red nodes = 450)				
ORG(1)	1681	n/a	0.15	0.70(14)
ORG(3)	2465	n/a	0.18	0.56(10)
RCM(1)	1681	0.04	0.16	0.70(14)
RCM(3)	2465	0.03	0.18	0.50(9)
MDF(1)	1681	1.09	0.16	0.65(13)
MDF(3)	3261	3.29	0.22	0.48(8)

TABLE 4.4
Summary for test problem STONE.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
ORG(0)	1860	n/a	0.12	3.31(47)
ORG(1)	2760	n/a	0.14	2.08(27)
ORG(2)	3630	n/a	0.16	1.82(23)
ORG(3)	5340	n/a	0.22	1.43(16)
RCM(0)	1860	0.04	0.12	3.32(47)
RCM(1)	2760	0.03	0.14	2.07(27)
RCM(2)	3630	0.03	0.16	1.35(17)
RCM(3)	4471	0.04	0.19	1.18(14)
MDF(0)	1860	0.92	0.12	3.25(46)
MDF(1)	3658	1.81	0.17	1.36(17)
MDF(2)	3819	1.90	0.18	1.38(17)
MDF(3)	5361	3.98	0.24	0.99(11)
Reduced system (black nodes = 480, red nodes = 481)				
ORG(1)	1798	n/a	0.17	1.16(22)
ORG(3)	2638	n/a	0.20	0.87(15)
RCM(1)	1798	0.03	0.17	1.16(22)
RCM(3)	2638	0.03	0.20	0.76(13)
MDF(1)	1798	1.16	0.17	0.91(17)
MDF(3)	3487	3.42	0.24	0.69(11)

The first finite element problem REFINE2D tests are listed in Table 4.5. The reduced system factorization is effective for this problem since almost half the nodes are exactly eliminated. The reduced system MDF(1) has the smallest solution cost.

Table 4.6 shows the results for Problem FE2D. Even though the generalized red/black partitioning has only 430 (out of 1521) red nodes, the reduced system factorizations are superior to the full system factorizations. For a given level of fill,

TABLE 4.5
Summary for test problem REFINE2D.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
RCM(0)	2480	0.05	0.15	4.38(49)
RCM(1)	3571	0.05	0.17	2.86(30)
RCM(2)	4758	0.05	0.21	2.59(26)
MDF(0)	2480	1.19	0.15	4.29(48)
MDF(1)	4645	2.35	0.21	2.10(21)
MDF(2)	5292	3.04	0.24	2.16(21)
Reduced system (black nodes= 648, red nodes= 513)				
RCM(1)	2629	0.04	0.21	1.96(28)
RCM(2)	2952	0.05	0.23	1.80(25)
MDF(1)	2753	1.72	0.23	1.51(21)
MDF(2)	3390	2.52	0.25	1.59(21)

TABLE 4.6
Summary for test problem FE2D.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
RCM(0)	4373	0.06	0.25	6.40(49)
RCM(1)	5846	0.07	0.29	4.12(30)
RCM(2)	8325	0.07	0.37	3.31(22)
MDF(0)	4373	2.09	0.25	3.99(30)
MDF(1)	7942	4.66	0.36	3.43(23)
MDF(2)	10620	8.39	0.49	2.64(16)
Reduced system (black nodes = 1091, red nodes = 430)				
RCM(1)	5286	0.08	0.38	4.00(34)
RCM(2)	6888	0.08	0.43	2.93(23)
MDF(1)	5782	3.86	0.41	2.84(23)
MDF(2)	8565	7.90	0.54	2.33(17)

reduced system MDF has a smaller solution cost than reduced system RCM.

For the three-dimensional problem FE3D, high levels of fill are not very effective because of the large amount of fill in the ILU factorization. If factorization cost is included, the best method is MDF(0) (Table 4.7).

To summarize, for Problems STRONGX, STRONGY, and STONE, the reduced system MDF(1) ordering outperforms RCM(1) and ORG(1). All orderings have the identical amount of fill for level-1 reduced systems. Reduced system MDF(3) is either the best or tied with RCM(3), although at the expense of greater fill. Note that for STRONGX, reduced system RCM(1) is almost four times slower than reduced system MDF(1).

For Problems REFINE2D and FE2D, reduced system MDF(1) again outperforms RCM(1). Reduced system RCM becomes more competitive with reduced system MDF as the level increases. It is interesting to note that for the three-dimensional problem FE3D, MDF(0) is superior to RCM(0).

5. Conclusions. In agreement with previous work, we have found that the ordering of the unknowns has a major effect on the convergence of the ILU preconditioned PCG iterative methods.

In some cases, it is possible to select an a priori ordering that results in rapid

TABLE 4.7
Summary for test problem FE3D.

Ordering & level ℓ	Nonzeros in L	Ordering time	Fact. time	Solution time & iterations
Full system				
RCM(0)	19725	0.23	1.02	16.13(41)
RCM(1)	39066	0.24	2.01	11.70(24)
RCM(2)	72038	0.23	5.27	10.29(16)
MDF(0)	19725	16.49	1.11	10.08(25)
MDF(1)	49599	150.2	3.45	9.88(18)
MDF(2)	92181	1002	10.6	10.69(14)
Reduced system (black nodes = 2593, red nodes = 657)				
RCM(1)	38822	0.23	3.00	10.66(23)
RCM(2)	66933	0.23	6.32	10.32(17)
MDF(1)	43141	143	3.36	8.96(18)
MDF(2)	87472	1067	9.34	9.97(14)

convergence. However, for partial differential equation problems that have rapidly varying coefficients and are discretized on unstructured grids, a good ordering is far from obvious.

As demonstrated in the anisotropic examples, RCM orderings can be quite poor for level-1 fill, for both full systems and reduced systems, compared to MDF orderings. Reduced system methods were superior to full system iteration for all the two-dimensional problems. Reduced system MDF orderings with lower fill level outperformed reduced system natural and RCM orderings.

Because of the large factorization cost and the relatively small number of red nodes exactly eliminated, the reduced system approach was not very effective for the three-dimensional problem. MDF(0) was the best choice.

In all our tests, the MDF ordering method always resulted in good convergence behavior, even for anisotropic and inhomogeneous (rapidly varying equation coefficient) problems. Of course, the ability of MDF ordering to perform well for anisotropic, inhomogenous problems comes at a price. The time taken for determining the MDF ordering is much larger than the ordering cost for RCM. Consequently, we believe that the major application of MDF ordering will be in the solution of time-dependent or nonlinear problems. In these situations, a sequence of matrix problems must be solved, where the matrix elements are only slightly changed from one timestep to the next. An ordering determined from one of these matrices can be used for the sequence. The ordering cost can then be amortized over the cost of many solves. Applications of this idea to reservoir simulation and Navier–Stokes equations are discussed in [6] and [10].

If a single solution is required for a two-dimensional problem that is isotropic, then a reduced system RCM method would be a good choice. On the other hand, if several similar anisotropic problems are being solved, then it is worthwhile to use a reduced-system MDF ordering. For three-dimensional problems, MDF(0) would appear to be a good choice.

We are currently developing approximate MDF ordering methods that are less expensive to compute, and hence can be applied to problems with a large node connectivity, which is typical of discretized systems of partial differential equations.

REFERENCES

[1] O. AXELSSON AND I. GUSTAFSSON, *On the use of preconditioned conjugate gradient methods for red-black ordered five-point difference schemes*, J. Comput. Phys., 35 (1980), pp. 284–299.

[2] A. BEHIE, *Comparison of nested factorization, constrained pressure residual and incomplete factorization preconditionings*, in Proc. 1985 Reservoir Simulation Symposium, Dallas, TX, 1985, pp. 387–396. Paper SPE 13531.

[3] A. BEHIE AND P. A. FORSYTH, *Comparison of fast iterative methods for symmetric systems*, IMA J. Numer. Anal., 3 (1983), pp. 41–63.

[4] ———, *Incomplete factorization methods for fully implicit simulation of enhanced oil recovery*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 543–561.

[5] G. BRUSSION AND V. SONNAD, *A comparison of direct and preconditioned iterative techniques for sparse, unsymmetric systems of linear equations*, Internat. J. Numer. Methods. Engrg., 28 (1989), pp. 801–815.

[6] P. CHIN, E. F. D'AZEVEDO, P. A. FORSYTH, AND W.-P. TANG, *Preconditioned conjugate gradient methods for incompressible Navier–Stokes equations*, Tech. Report CS-91-04, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 1991; Internat. J. Numer. Methods Fluids, to appear.

[7] P. CONCUS, G. H. GOLUB, AND G. MEURANT, *Block preconditioning for the conjugate gradient method*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 543–561.

[8] E. CUTHILL AND J. MCKEE, *Reducing the bandwidth of sparse symmetric matrices*, in Proc. 24th National Conference of the Association for Computing Machinery, 1969, Brandon Press, Princeton, NJ, pp. 157–172.

[9] E. F. D'AZEVEDO, P. A. FORSYTH, AND W.-P. TANG, *Towards a cost-effective high order ILU preconditioner*, Research Report CS-90-50, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 1990.

[10] ———, *An automatic ordering method for incomplete factorization iterative solvers*, in Proc. 1991 Reservoir Simulation Symposium, Anaheim, CA, 1991 (Paper SPE 21226).

[11] S. DOI, *A Gustafsson-type modification for parallel ordered incomplete LU factorization*, in Advances in Numerical Methods for Large Sparse Sets of Linear Systems, T. Nodera, ed., Keio University Press, Tokyo, Japan, 1991, pp. 36–42.

[12] ———, *On parallelism and convergence of incomplete LU factorizations*, Appl. Numer. Math., 7 (1991), pp. 417–436.

[13] S. DOI AND A. LICHNEWSKY, *A graph-theory approach for analyzing the effects of ordering on ILU preconditioning*, Tech. Report No. 1452, INRIA, Le Chesnay, France, June 1991.

[14] I. S. DUFF, A. M. ERISMAN, AND J. K. REID, *Direct methods for sparse matrices*, Oxford University Press, London, 1986.

[15] I. S. DUFF AND G. A. MEURANT, *The effect of ordering on preconditioned conjugate gradients*, BIT, 29 (1989), pp. 635–657.

[16] V. EIJKHOUT, *Vectorizable and parallelizable preconditioners for the conjugate gradient method*, Ph.D. thesis, University of Nijmegen, Nijmegen, the Netherlands, 1990.

[17] S. C. EISENSTAT, H. C. ELMAN, AND M. H. SCHULTZ, *Block-preconditioned conjugate-gradient-like methods for numerical reservoir simulation*, SPE J. Res. Engrg., 3 (1988), pp. 307–312.

[18] P. A. FORSYTH, *A control volume finite element approach to* NAPL *groundwater contamination*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 1029–1057.

[19] P. A. FORSYTH AND F. W. LETNIOWSKI, *A control volume finite element method for three-dimensional* NAPL *groundwater contamination*, Internat. J. Numer. Methods Fluids, 13 (1991), pp. 955–970.

[20] P. A. FORSYTH AND P. H. SAMMON, *Local mesh refinement and modelling of faults and pinchouts*, SPE J. Form. Eval., 1 (1986), pp. 275–285.

[21] G. FORSYTHE AND W. WASOW, *Finite difference methods for partial differential equations*, John Wiley, New York, 1960.

[22] A. GEORGE AND J. W. H. LIU, *Computer solution of large sparse positive-definite systems*, Prentice-Hall, Englewood Cliffs, NJ, 1981.

[23] I. GUSTAFSSON, *A class of first order factorization methods*, BIT, 18 (1978), pp. 142–156.

[24] P. S. HUYAKORN AND G. F. PINDER, *Computational Methods in Subsurface Flow*, Academic Press, New York, 1983.

[25] H. P. LANGTANGEN, *Conjugate gradient methods and ILU preconditioning of non-symmetric matrix systems with arbitrary sparsity patterns*, Internat. J. Numer. Methods Fluids, 9 (1989), pp. 213–233.

[26] F. W. LETNIOWSKI, *Three-dimensional Delaunay triangulations for finite element approximations to a second-order diffusion operator*, SIAM J. Sci. Statist. Comput., 13 (1992), pp. 765–770.

[27] J. W.-H. LIU AND A. H. SHERMAN, *Comparative analysis of the Cuthill–McKee and the reverse Cuthill–McKee ordering algorithms for sparse matrices*, SIAM J. Numer. Anal., 13 (1975), pp. 198–213.

[28] N. MUNKSGAARD, *Solving sparse symmetric sets of linear equations by preconditioned conjugate gradients*, ACM Trans. Math. Software, 6 (1980), pp. 206–219.

[29] E. J. NORTHROP AND P. T. WOO, *Application of preconditioned conjugate gradient methods in reservoir simulation*, SPE J. Res. Engrg., 3 (1988), pp. 295–301.

[30] D. P. O'LEARY, *Solving sparse matrix problems on parallel computers*, Tech. Report 1234, Computer Science Center, University of Maryland, College Park, MD, 1982.

[31] S. V. PARTER, *The use of linear graphs in Gaussian elimination*, SIAM Rev., 3 (1961), pp. 364–369.

[32] D. W. PEACEMAN, *Fundamentals of numerical reservoir simulation*, Elsevier, New York, 1977.

[33] D. ROSE, *A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations*, in Graph Theory and Computing, R. C. Read, ed., Academic Press, New York, 1972, pp. 183–217.

[34] R. SCHREIBER AND W.-P. TANG, *Vectorizing the conjugate gradient method*, in Proc. Symposium on CYBER 205 Applications, Ft. Collins, CO, 1982.

[35] H. D. SIMON, *Incomplete LU preconditioners for conjugate-gradient-type iterative methods*, SPE J. Res. Engrg., 3 (1988), pp. 302–306.

[36] H. L. STONE, *Iterative solution of implicit approximations of multidimensional partial differential equations*, SIAM J. Numer. Anal., 5 (1968), pp. 530–558.

[37] A. D. TUFF AND A. JENNINGS, *An iterative method for large systems of linear structural equations*, Internat. J. Numer. Methods Engrg., 7 (1973), pp. 175–183.

[38] J. R. WALLIS, *Incomplete Gaussian elimination as a preconditioning for generalized conjugate gradient acceleration*, in Proc. 1983 SPE Symposium on Reservoir Simulation, San Francisco, 1983 (Paper SPE 12265).

[39] J. W. WATTS, *A conjugate gradient-truncated direct method for the iterative solution of the reservoir pressure equation*, Soc. Pet. Engrg. J., 21 (1981), pp. 345–353.

[40] Z. ZLATEV, *Use of iterative refinement in the solution of sparse linear systems*, SIAM J. Numer. Anal., 19 (1982), pp. 381–399.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 962–978, July 1992

016

IMPLICIT NULLSPACE ITERATIVE METHODS FOR CONSTRAINED LEAST SQUARES PROBLEMS*

DOUGLAS JAMES†

Abstract. A class of iterative algorithms is proposed for solving equality constrained least squares problems, generalizing an order-reducing algorithm first analyzed by Barlow, Nichols, and Plemmons (algorithm BNP). The new algorithms, called implicit null space methods, are based on the classical nullspace method, except that the basis for the nullspace of the constraint matrix is not explicitly formed. The implicit basis acts as a preconditioner for a set of normal equations in factored form. Implicit nullspace methods allow great flexibility in the choice of preconditioner, and can be used to solve certain problems for which algorithm BNP is not well suited. In addition, they offer the opportunity for parallel implementation on substructured problems. Some numerical results based on both structural engineering applications and Stokes flow are included.

Key words. constrained least squares, force method, nullspace method, Stokes flow, structural analysis, substructuring

AMS(MOS) subject classifications. 65F10, 73K25

1. Introduction.

Our interest is the solution of the *equality constrained least squares problem*, or *problem* LSE: given an $m_1 \times n$ matrix E, an $m_2 \times n$ matrix G, an $m_1 \times 1$ vector b, and an $m_2 \times 1$ vector c,

$$\text{minimize } \|Gy - c\|_2 \quad \text{such that } Ey = b. \tag{1}$$

We will assume that E has full row rank, so b is in the range of E and the problem has at least one solution. We will also require that the nullspaces of E and G intersect trivially (or, equivalently, that $\left[\begin{smallmatrix} E \\ G \end{smallmatrix}\right]$ has full column rank), guaranteeing that the solution is unique. Finally, we presume that the matrices are large and sparse, so that it is plausible to consider iterative algorithms. All three assumptions are realistic for a wide range of important applications.

A number of essentially equivalent formulations of LSE will prove helpful:

Constrained minimization problem. Let $F = G^T G$ and $s = -G^T c$. Note that $\|Gy - c\|_2^2 = y^T F y + 2y^T s + c^T c$, and that the term $c^T c$ has no effect on the value of y at which the quantity is minimized. Hence, LSE is equivalent to the problem

$$\text{minimize } (y^T F y + 2y^T s) \quad \text{such that} \quad Ey = b. \tag{2}$$

Kuhn–Tucker formulation. Introduce the Lagrange multiplier λ and the residual $r = c - Gy$. Then solving LSE amounts to finding y, r, and λ satisfying

$$\begin{bmatrix} E & 0 & 0 \\ G & I & 0 \\ 0 & G^T & E^T \end{bmatrix} \begin{bmatrix} y \\ r \\ \lambda \end{bmatrix} = \begin{bmatrix} b \\ c \\ 0 \end{bmatrix}. \tag{3}$$

Saddle-point formulation. Introduce the Lagrange multiplier μ to the constrained minimization formulation. Then we can find y satisfying (2) by solving

$$\begin{bmatrix} -F & E^T \\ E & 0 \end{bmatrix} \begin{bmatrix} y \\ \mu \end{bmatrix} = \begin{bmatrix} s \\ b \end{bmatrix}. \tag{4}$$

* Received by the editors April 5, 1990; accepted for publication (in revised form) March 5, 1991.
† Department of Mathematical Sciences, U.S. Air Force Academy, Colorado, 80840. Graduate studies were sponsored by the Department of Mathematical Sciences, U.S. Air Force Academy, Colorado, and funded by the Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio. Research was directed by Robert J. Plemmons under Air Force grant AFOSR-88-0285.

The multiplier μ is related to λ in the Kuhn–Tucker equations by $\mu = -\lambda$.

The motivating example is the static analysis of engineering structures: given a large structure subjected to an external load, find the internal forces at equilibrium. When the problem is modeled using the *force method* (see, for example, [7]), the constraint $Ey = b$ captures the fact that the forces sum to zero at any node in the structure (the *equilibrium condition*). The vector y represents the unknown internal forces, b is the vector of external loads, and E (commonly called the *equilibrium matrix*) is determined by the shape and connectivity of the structure. The vector $-\lambda$ represents the nodal displacements. We seek the particular solution to the constraint that minimizes *complementary energy*; the matrix F, which helps define the energy functional, is determined by the material properties of the elements in the structure. In a rigid bar truss, F is a diagonal matrix with positive diagonal entries determined by Hooke's Law (see Strang [16]). In more general structures, F is block diagonal, with one small block for every element in the structure. The blocks in F are always symmetric nonnegative definite, and are symmetric positive definite when all elements behave elastically (see, for example, Przemieniecki [14]). Unless the elements are prestressed, the vectors c and s are zero in this application.

Static analysis of engineering structures is but one example of a more general physical principle at work: minimizing an energy functional subject to an equilibrium constraint is a central idea throughout the physical sciences (see Strang [15], [16]). Stokes flow provides a second interesting application. For simplicity, assume the density of the fluid is constant and unchanging, so the continuous velocity vector $\underline{v}$ satisfies the conservation of mass equation $\nabla \cdot \underline{v} = f$, where $f = 0$ in the absence of sources and sinks. This continuous equation becomes $Ey = b$ after discretizing, where E approximates the divergence, y represents the discrete velocity components, and b captures boundary and source information. The velocity $\underline{v}$ and pressure p are related by the conservation of momentum equation $\nabla^2 \underline{v} - \nabla p = g$, where g reflects boundary and forcing terms. Discretizing this equation produces the rest of the saddle point formulation (4). The symmetric positive definite matrix F (typically block tridiagonal) represents the negative of the Laplacian applied to each component of $\underline{v}$, the matrix $-E^T$ approximates the gradient, and s contains boundary and forcing terms (see Strang [15], [16] for a more complete discussion of the Stokes model, and Hall [6] for a treatment of the more general Navier–Stokes equations). The matrix G, if needed, could be taken to be the Cholesky factor of F, but a more natural approach is to recognize the Laplacian as the divergence of the gradient. Then G can be defined to be some suitable discretization of the gradient operator acting componentwise on $\underline{v}$.

Our approach to solving problem LSE and its equivalent forms is an extension of an algorithm first proposed and analyzed by Barlow, Nichols, and Plemmons [1]. Their algorithm, which we refer to as algorithm BNP, starts with a repartitioned form of the Kuhn–Tucker equations which has square, nonsingular, and easily invertible diagonal blocks (equation (7) below). Block Gauss elimination produces a symmetric positive definite subproblem (equation (9) below) with the $n - m_1$ leading components of the residual $r = c - Gy$ as the unknowns. The authors then apply a variation of the conjugate gradient algorithm to this subproblem, generating at each iteration an approximation to the original unknown y. The method is *order-* or *dimension-reducing* in the sense that the subproblem has fewer unknowns than the original problem in y.

In theory, BNP requires assumptions on problem LSE no stronger than the ones listed above. In practice, however, implementation is difficult without the stronger

assumption that G itself has full column rank. This assumption holds for many but not all applications (it fails to hold, for example, if any of the elements deform plastically in the structures problem). The algorithm may also be difficult to apply to the saddle-point formulation (4) (e.g., Stokes flow), where F and s rather than G and c are available. Our primary goal is to overcome these limitations.

The key to the extension is recognizing a connection between algorithm BNP and the classical *nullspace method*. The latter begins with some convenient solution y_p to the constraint, and a matrix N whose columns form a basis for the nullspace of E. We then seek a coordinate vector x such that $y = y_p + Nx$ is the solution to the constrained problem. We show that BNP may be viewed as a variation of the nullspace method, in which distinguished choices of y_p and N are used but never formed (we will call such a method an *implicit nullspace method*, or INM). The basis matrix N is seen to be acting as a preconditioner for a set of normal equations in factored form. We generalize algorithm BNP by producing implicit nullspace methods for other choices of N. The extension preserves the spirit of BNP (in particular, the algorithms are order-reducing), but is somewhat more flexible: it becomes relatively easy to construct preconditioners for problems in which G lacks full column rank, as well as problems in saddle-point form. Both BNP and the more general implicit nullspace algorithms can be implemented in parallel when the matrices reflect a substructuring (domain decomposition) of the physical model.

Section 2 introduces algorithm BNP as it is derived in [1] and summarizes some of its properties. We briefly mention recent results comparing BNP to both p-cyclic successive over-relaxation (SOR) and a two-parameter generalization of SOR known as block accelerated over-relaxation (AOR). We also discuss a parallel implementation for substructured problems (first reported in [10]) that proves helpful in understanding the extension. In §3, we establish the relationship between BNP and the nullspace method, and exploit the connection to derive the more general implicit nullspace methods. We describe in §4 some numerical experiments on both structures and Stokes problems, and offer some concluding remarks.

2. Algorithm BNP. It is difficult to apply traditional iterative methods to the Kuhn–Tucker equations as written in (3): the diagonal blocks are not even square, let alone nonsingular. We therefore start by repartitioning these equations. Since we are assuming E has full row rank, and that $\left[\begin{smallmatrix} E \\ G \end{smallmatrix}\right]$ has full column rank, we can reorder the rows of G and c, and repartition

$$G = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix}$$

so that

$$A_1 = \begin{bmatrix} E \\ G_1 \end{bmatrix} \tag{5}$$

is square and nonsingular. Defining $A_2 = G_2$ for convenience, we now have

$$\begin{bmatrix} E \\ G \end{bmatrix} = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix}. \tag{6}$$

Make these substitutions in (3) and reorder the column blocks to obtain the *modified Kuhn–Tucker equations*

$$\begin{bmatrix} A_1 & 0 & K \\ A_2 & I & 0 \\ 0 & A_2^T & A_1^T \end{bmatrix} \begin{bmatrix} y \\ r_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ c_2 \\ 0 \end{bmatrix}, \tag{7}$$

where

$$c = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix}, \quad r = \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}, \quad b_1 = \begin{bmatrix} b \\ c_1 \end{bmatrix}, \quad z_3 = \begin{bmatrix} \lambda \\ r_1 \end{bmatrix}, \quad \text{and} \quad K = \begin{bmatrix} 0 & 0 \\ 0 & I \end{bmatrix}.$$

Note that the diagonal blocks are now square and nonsingular.

The derivation of BNP in [1] begins with this repartitioned system. The authors first apply block elimination to reduce the system to block upper triangular form. The result of this reduction is the system

$$\begin{bmatrix} A_1 & 0 & K \\ & I & -A_2A_1^{-1}K \\ & & (I+BK) \end{bmatrix} \begin{bmatrix} y \\ r_2 \\ z_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ c_2 - A_2A_1^{-1}b_1 \\ A_1^{-T}A_2^T(A_2A_1^{-1}b_1 - c_2) \end{bmatrix}, \tag{8}$$

where $B = A_1^{-T}A_2^TA_2A_1^{-1}$. The linear system associated with the third (lowest) diagonal block in (8) is particularly important. A careful look at the matrix $I + BK$ (see [1] and [8]) reveals that it too is block upper triangular. There are two diagonal blocks; the second (lowest) of these diagonal blocks is the coefficient matrix of the linear system

$$(I + Y^TY)r_1 = h, \quad \text{where } Y = A_2A_1^{-1}\bar{K}, \quad \bar{K} = \begin{bmatrix} 0 \\ I \end{bmatrix}. \tag{9}$$

The right-hand side vector is given by $h = Y^T(A_2A_1^{-1}b_1 - c_2)$. The unknown r_1 consists of the $n - m_1$ leading components of the residual $c - Gy$.

In principle, one can solve this reduced system for r_1, then use back substitution on the transformed Kuhn–Tucker system to recover the remaining unknowns. In practice, this is not necessary: the analysis in [1] shows that a variant of conjugate gradients applied to the reduced system produces as by-products all that is needed to recover the full solution y. It is this order-reducing conjugate gradient algorithm that we call BNP.

It is worth noting that the symmetric positive definite matrix $I + Y^TY$ is likely to have a number of desirable properties. In particular, the scalar $\lambda = 1$ is often a multiple eigenvalue due to column rank deficiencies in Y. This, of course, limits the maximum number of conjugate gradient iterations required for convergence.

The key to implementing BNP is producing an A_1 that is easily invertible, and the most convenient form to seek is an upper triangular matrix. If G has full column rank, one way to accomplish this, depicted in Fig. 1, is as follows:

1. Use Gauss elimination (or orthogonal reduction) with column pivoting to replace E with the upper trapezoidal matrix

$$E_t = \begin{bmatrix} E_L & E_R \end{bmatrix}, \tag{10}$$

where E_L is upper triangular and nonsingular. Here the subscript t indicates "trapezoidal," while L and R indicate left and right, respectively.

2. Apply the same column interchanges to G, and partition compatibly with E_t. Use orthogonal rotations on G and c to replace G with the nonsingular upper triangular matrix

$$G = \begin{bmatrix} G_{21} & G_{22} \\ & G_{12} \end{bmatrix}. \tag{11}$$

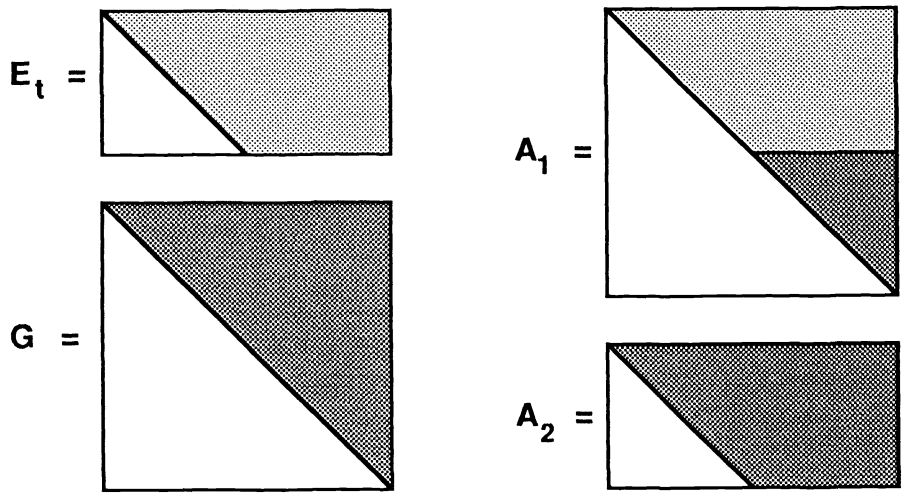


FIG. 1. *Forming A_1 from trapezoidal E_t.*

Here the leading subscripts indicate whether the blocks will become part of G_1 or G_2. We emphasize that the original matrix G is not necessarily square; because it has full column rank, however, it can be reduced to nonsingular triangular form using orthogonal rotations. In some applications it is helpful and easy to achieve this form (in the structural engineering problem, for example, both F and G are block diagonal with relatively small blocks). In other problems (most notably the Stokes problem) it would be inappropriate (and unnecessary) to attempt to reduce G to this form (see §4 for an alternative). In either case, however, the algorithm is easier to picture when presented using a triangular G; without loss of generality, we proceed under this assumption.

3. Define $G_1 = \begin{bmatrix} 0 & G_{12} \end{bmatrix}$ and $G_2 = \begin{bmatrix} G_{21} & G_{22} \end{bmatrix}$, obtaining

$$A_1 = \begin{bmatrix} E_L & E_R \\ & G_{12} \end{bmatrix}. \tag{12}$$

With these substitutions, the matrix Y in the BNP system (9) can be written in a way that will prove useful in the next section:

$$Y = A_2 \begin{bmatrix} -E_L^{-1}E_R \\ I \end{bmatrix} G_{12}^{-1}. \tag{13}$$

While this method of constructing A_1 is useful for analysis, it may not be efficient: column interchanges may destroy exploitable structure in the matrices E and G. Another approach, first reported in [10], is to reduce E without column pivoting to produce a "stairstep" form E_s (Fig. 2). Now form A_1 by interlacing rows of G with rows of E_s. A permutation matrix $P = \begin{bmatrix} P_L & P_R \end{bmatrix}$ defines the interlacing

$$A_1 = P \begin{bmatrix} E_s \\ G_1 \end{bmatrix} = P_L E_s + P_R G_1. \tag{14}$$

Notice that the same permutation matrix (not its transpose) relates the stairstep and trapezoidal forms:

$$E_s P = \begin{bmatrix} E_s P_L & E_s P_R \end{bmatrix} = \begin{bmatrix} E_L & E_R \end{bmatrix} = E_t. \tag{15}$$

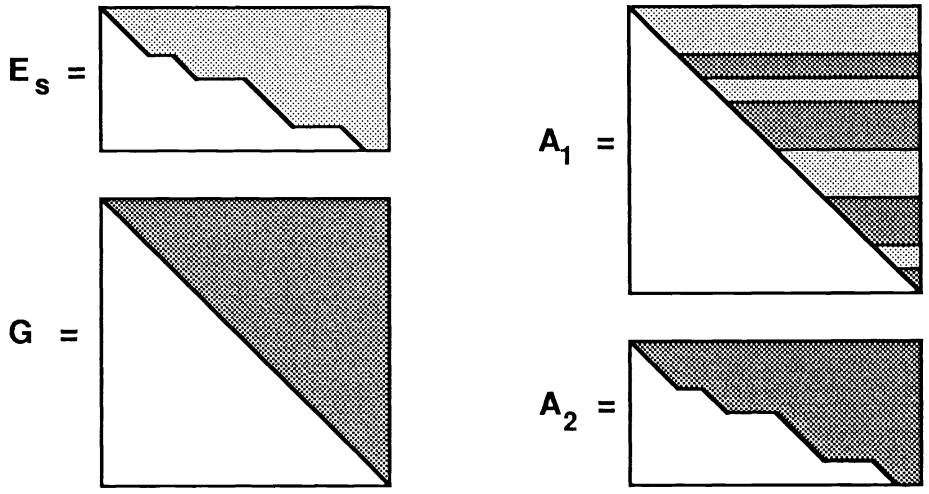


FIG. 2. *Forming A_1 from stairstep E_s.*

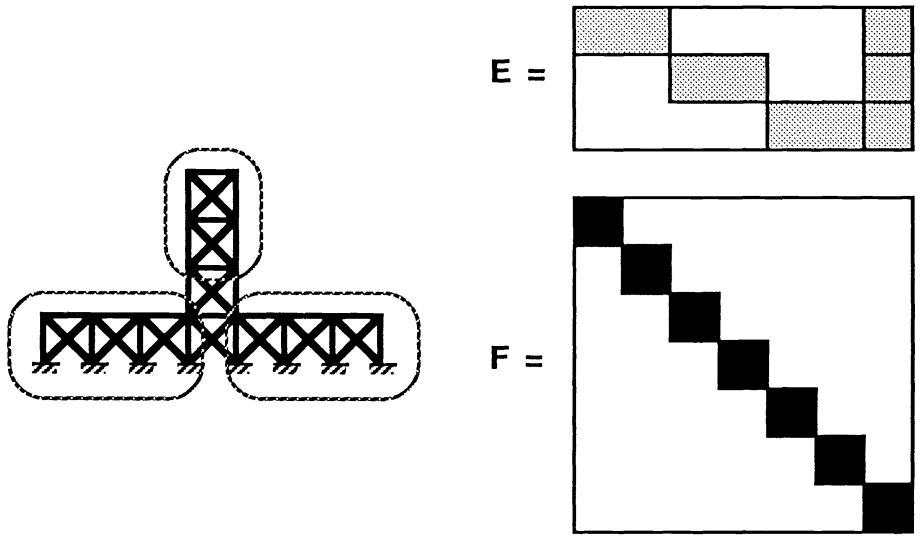


FIG. 3. *Substructuring of a flexible truss.*

Interlacing without column permutations is especially helpful when the matrices E and G reflect a substructuring (domain decomposition) of the physical domain. Substructuring of the flexible truss in Fig. 3, for example, results in matrices with the depicted block structure (the matrices in the figure have been modified for clarity: there should be one small block in F for each bar in the truss, and the subblocks in E are not necessarily the same size). The domain decomposition of the Stokes problem is somewhat more subtle, but the ideas are similar (see [8]).

Given such a substructuring, Fig. 4(a) depicts the result of reduction of E and factorization of F. Both E_s and G can be produced in parallel by assigning to a

single processor the rows associated with the block being reduced. Additionally, notice that solves involving A_1 and A_1^T provide opportunities for block-based parallel computation: when A_1 is the coefficient matrix of a linear system, we can first solve for the unknowns associated with the last (lowest) diagonal block, then solve for all remaining blocks of unknowns in parallel (assign each block of unknowns to its own processor). A similar process is possible for solves involving A_1^T. The matrices in our extension of algorithm BNP will allow similar opportunities for parallel computation.

Ideally, we would like to reduce E in a stable fashion, yet still preserve the structure of both E and G. Of course, these are in some ways competing goals: column pivoting improves the likelihood of a stable reduction, but such pivoting can destroy exploitable structure in the matrices. Fortunately, there is a compromise approach: if column pivoting is needed or desired when reducing E, we can preserve the structure of E by restricting eligible pivot columns to those associated with the current substructure. The price of such pivoting includes additional but predictable fill during the factorization of F. In either case, we can now complete the construction of A_1 by interlacing (Fig. 4(b)).

We should also observe that one can apply other more traditional iterative algorithms to the modified Kuhn–Tucker equations (7), including in particular block successive over-relaxation (SOR). Partitioned into either two or three diagonal blocks, the coefficient matrix in (7) is a so-called *block cyclic matrix*. The associated block SOR methods, called *p-cyclic* SOR *methods*, enjoy an elegant theory due largely to Young [19] and Varga [17], [18]. Barlow, Nichols, and Plemmons [1], extending work by Freund [4] and Markham, Neumann, and Plemmons [11], prove that algorithm BNP applied to LSE is superior to both two- and three-cyclic SOR in exact arithmetic. Another approach to solving the modified Kuhn–Tucker equations is a two-parameter variant of block SOR known as accelerated over-relaxation or AOR (see, for example, [12]). We have recently extended the work in [1] and [4] to show that BNP is superior to two- and three-block AOR as well [8]. Numerical experiments (see [9] as well as §4 below) suggest that BNP is in fact faster than both p-cyclic SOR and block AOR by a wide margin.

3. Implicit nullspace algorithms. Algorithm BNP as outlined in §2 has several limitations. Suppose, for example, that G lacks full column rank. Given a specified column of G, there may be no row of G with leading nonzero in that column, even after orthogonal reduction. If we need such a row to augment E, we will be unable to produce an upper triangular A_1. A second limitation applies to problems expressed in saddle-point form (4). When F and s are given instead of G and c, it may be impractical to recover G and c, both of which are needed. We seek to extend the algorithm to overcome these limitations.

We begin by considering the classical *nullspace method* (see, for example, [2] and [13]). Suppose we are given a particular solution y_p to the constraint $Ey = b$. Let N be a matrix whose columns form a basis for the nullspace of E (for convenience, we will call N a *basis matrix*). Then any vector satisfying the constraint can be written as $y = y_p + Nx$ for some choice of x, and minimizing $\|Gy - c\|_2$ subject to the constraint amounts to minimizing $\|G(y_p + Nx) - c\|_2$ over all possible x. The latter minimization is an unconstrained problem of order $n - m_1$. Forming the associated normal equations confirms that the required x solves the symmetric positive definite system

$$N^T G^T G N x = N^T G^T (c - G y_p). \tag{16}$$

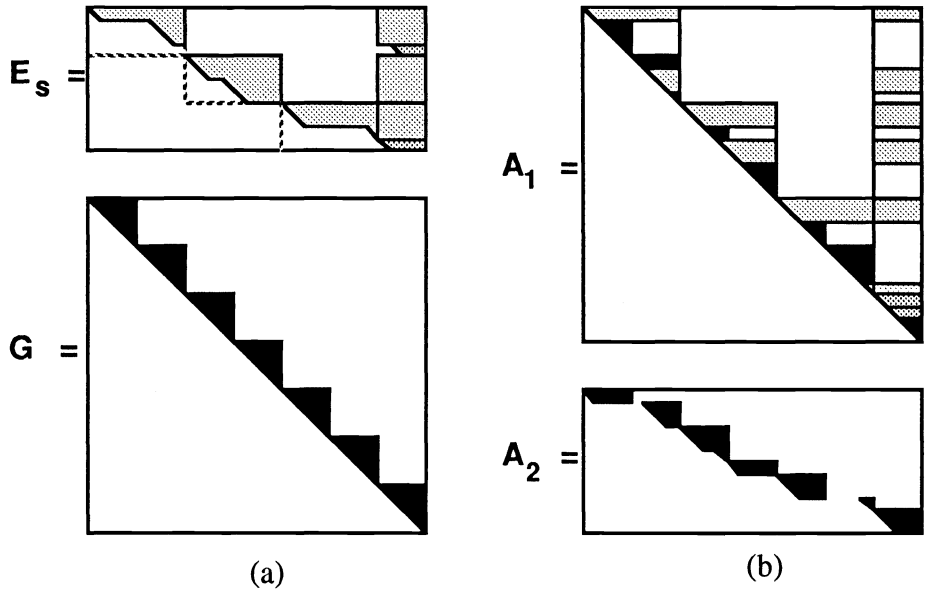


FIG. 4. *Parallel implementation of interlacing:* (a) *Matrices after reduction;* (b) *Results of interlacing.*

Now consider the upper trapezoidal matrix $E_t = \begin{bmatrix} E_L & E_R \end{bmatrix}$ given in (10). Note that

$$y_p = \begin{bmatrix} E_L^{-1} b \\ 0 \end{bmatrix} \tag{17}$$

is a particular solution of the constraint $E_t y = b$, and the columns of

$$N_I = \begin{bmatrix} -E_L^{-1} E_R \\ I \end{bmatrix} \tag{18}$$

form a basis for the nullspace of E_t. Consider the nullspace normal equations (16) with these choices of the basis matrix and particular solution. Precondition in the standard way with G_{12}, where G_{12} is the lower right-hand corner of G defined in (11):

$$G_{12}^{-T} N_I^T G^T G N_I G_{12}^{-1} w = G_{12}^{-T} N_I^T G^T (c - G y_p), \quad \text{where } w = G_{12} x. \tag{19}$$

The formidable-looking coefficient matrix simplifies nicely. First, note that

$$N_I G_{12}^{-1} = \begin{bmatrix} -E_L^{-1} E_R G_{12}^{-1} \\ G_{12}^{-1} \end{bmatrix} = A_1^{-1} \bar{K}, \tag{20}$$

where A_1 and $\bar{K}$ are as in (9). Now observe that

$$\begin{bmatrix} E \\ G_1 \end{bmatrix} A_1^{-1} \bar{K} = A_1 A_1^{-1} \bar{K} = \bar{K}, \quad \text{so } G_1 A_1^{-1} \bar{K} = I. \tag{21}$$

This gives us

$$GN_I G_{12}^{-1} = \begin{bmatrix} G_1 \\ G_2 \end{bmatrix} A_1^{-1}\bar{K} = \begin{bmatrix} I \\ A_2 A_1^{-1}\bar{K} \end{bmatrix} = \begin{bmatrix} I \\ Y \end{bmatrix}, \tag{22}$$

where Y is as in the BNP system (9). Moreover, we can relate the BNP unknown r_1 to the preconditioned nullspace unknown w. Begin by partitioning the LSE unknown y into

$$\begin{bmatrix} y_L \\ y_R \end{bmatrix},$$

and use the fact that $y = y_p + N_I x$ to find that $x = y_R$. Now, since $r_1 = c_1 - G_1 y$, which is $c_1 - G_{12} y_R$, we find that $w = c_1 - r_1$. All of this allows us to rewrite (19) as

$$(I + Y^T Y)(c_1 - r_1) = G_{12}^{-T} N_I^T G^T (c - G y_p). \tag{23}$$

Finally, move c_1 to the right-hand side and simplify; the result is precisely the BNP system (9). Thus BNP is essentially a preconditioned form of the nullspace method, where a distinguished choice of the nullspace basis appears in the equations but does not need to be formed explicitly. In fact, we can say more: if N_I is a basis matrix, so is $N = N_I C$ for any nonsingular C. This means $N = N_I G_{12}^{-1}$ is a basis for the nullspace of E_t, and (19) is a set of normal equations associated with the nullspace method for this choice of basis. So BNP is itself an implicit nullspace method, with $N = N_I G_{12}^{-1}$ as the (unformed) basis matrix.

The basic idea behind the extension of BNP is now clear: instead of using G_{12} as a preconditioner for the nullspace normal equations, choose any convenient nonsingular $(n - m_1) \times (n - m_1)$ matrix M, and use $N = N_I M$ as the basis matrix. But so far we have only considered the trapezoidal matrix E_t obtained by column pivoting. To make this approach practical, we would like to use interlacing to construct a basis matrix and particular solution for the stairstep matrix E_s.

Recalling equation (15), let $P = \begin{bmatrix} P_L & P_R \end{bmatrix}$ be the permutation matrix relating the trapezoidal matrix E_t and the stairstep matrix E_s:

$$E_s P = \begin{bmatrix} E_s P_L & E_s P_R \end{bmatrix} = \begin{bmatrix} E_L & E_R \end{bmatrix} = E_t. \tag{24}$$

Now let M_1 be a matrix of size $(n - m_1) \times n$ such that

$$\hat{B}_1 = \begin{bmatrix} E_s \\ M_1 \end{bmatrix} \tag{25}$$

is nonsingular. It is quite easy to construct such an M_1: use rows with leading nonzeros in the $(n - m_1)$ columns in which the stairstep matrix E_s does not have leading nonzeros. Think of M_1 as generalizing the role that G_1 played in BNP. In the special case $E_s = E_t$, we would have $M_1 = \begin{bmatrix} 0 & M \end{bmatrix}$, where M plays the role of G_{12}.

Finally, by analogy with (14), define

$$B_1 = P\hat{B}_1 = P \begin{bmatrix} E_s \\ M_1 \end{bmatrix}. \tag{26}$$

We are now in a position to define the basis matrix N and the particular solution y_p.

THEOREM 3.1. *The vector* $y_p = B_1^{-1}P\begin{bmatrix} b \\ v \end{bmatrix}$ *satisfies the constraint* $E_s y = b$ *for any choice of* v.

Proof. Note that $B_1^{-1} = \hat{B}_1^{-1}P^T$, where $\hat{B}_1$ is defined in (25). Also, by an argument similar to (21), we find that $E_s\hat{B}_1^{-1} = \begin{bmatrix} I & 0 \end{bmatrix}$. Use these two facts to establish the result:

$$E_s y_p = E_s\hat{B}_1^{-1}P^T P\begin{bmatrix} b \\ v \end{bmatrix} = E_s\hat{B}_1^{-1}\begin{bmatrix} b \\ v \end{bmatrix} = \begin{bmatrix} I & 0 \end{bmatrix}\begin{bmatrix} b \\ v \end{bmatrix},$$

which is b as required. □

The choices $v = 0$ and $v = c_1$ are particularly convenient when using the results of Theorem 3.1 to define y_p; both are present naturally when initializing quantities in advance of forming B_1.

THEOREM 3.2. *The columns of* $N = B_1^{-1}P_R$ *form a basis for the nullspace of the stairstep matrix* E_s.

Proof. N is the proper size and has full column rank, so we need only establish that $EN = 0$. Recall from the proof of Theorem 3.1 that $B_1^{-1} = \hat{B}_1^{-1}P^T$ and $E_s\hat{B}_1^{-1} = \begin{bmatrix} I & 0 \end{bmatrix}$. Also note that $P^T P_R = \begin{bmatrix} 0 \\ I \end{bmatrix}$ by orthogonality. So

$$EN = E_s B_1^{-1}P_R = E_s\hat{B}_1^{-1}P^T P_R = \begin{bmatrix} I & 0 \end{bmatrix}\begin{bmatrix} 0 \\ I \end{bmatrix},$$

which is zero as required. □

We now implement the algorithm by applying the conjugate gradient algorithm to the factored nullspace normal equations (16), with N and y_p as given in the theorems. In the description below, s_k is the conjugate gradient direction vector, and $d_k = N^T G^T(c - Gy_p) - N^T G^T GNx_k$ is the residual associated with the normal equations. The vector q_k stores the product of the coefficient matrix with the direction vector. We use a starting vector of $x_0 = 0$ (so $y_0 = y_p$), but an arbitrary starting vector presents no difficulties.

IMPLICIT NULLSPACE METHOD (INM).

1. Use Gauss Elimination or orthogonal reduction on E and b to replace E with its stairstep form E_s.
2. Choose a convenient augmentation matrix M_1, and store the interlacing information in a permutation vector.
3. Form

$$B_1 = P\begin{bmatrix} E_s \\ M_1 \end{bmatrix} \quad \text{and} \quad b_0 = P\begin{bmatrix} b \\ 0 \end{bmatrix}.$$

4. Initialize:
   $x_0 = 0,$
   $d_0 = P_R^T B_1^{-T} G^T(c - GB_1^{-1}b_0),$
   $s_0 = d_0.$
5. For $k = 0, 1, \cdots,$ until $d_k^T d_k <$ tolerance:
   $q_k = P_R^T B_1^{-T} F B_1^{-1} P_R s_k,$
   $\alpha_k = d_k^T d_k / s_k^T q_k,$
   $\begin{bmatrix} x_{k+1} \\ d_{k+1} \end{bmatrix} = \begin{bmatrix} x_k \\ d_k \end{bmatrix} + \alpha_k \begin{bmatrix} s_k \\ -q_k \end{bmatrix},$
   $\beta_{k+1} = d_{k+1}^T d_{k+1} / d_k^T d_k,$

$s_{k+1} = d_{k+1} + \beta_{k+1} s_k$.

6. Recover $y = B_1^{-1}(P_R x + b_0)$ and exit.

The algorithm is easy to modify for problems in saddle-point form (4): when c and G are not available, simply substitute $s = -G^T c$ in the nullspace normal equations (16) to obtain the new right-hand side $-N^T(s + Fy_p)$.

A number of obvious choices for the augmentation matrix M_1 provide a menu of potential algorithms. Some of these include:

INMI. Form B_1 by interlacing the stairstep matrix E_s with rows of the identity matrix I (this amounts to setting $M_1 = P_R^T$). The result is an interlaced version of the nullspace method with N_I as the basis matrix (18), and may be a reasonable choice when G is unavailable, prohibitively dense, lacks full column rank, or is otherwise unsuitable for interlacing. One would expect little preconditioning effect beyond the order reduction itself. On the other hand, solves involving B_1 can be coded to exploit the presence of rows of the identity.

INMG. Interlace E_s with rows of G. This is essentially BNP in nullspace form: the coefficient matrices are identical, while the unknowns and right-hand sides differ (when $c_1 = 0$, however, they differ only in sign). Depending on the structure of F and G, it may be desirable to use the fact that $G_1 B_1^{-1} P_R = I$ when computing the direction vector s_k. The algorithm may fail if G lacks full column rank, since the rows of G needed to augment E_s may not be available.

INMGI. Interlace E_s with rows of G when suitable rows are available, then complete the construction of B_1 by including rows of the identity matrix I. This is another way of dealing with G when it lacks full column rank, and it proves to be much more effective than INMI (see §4). As with INMI and INMG, there are opportunities to code triangular solves and matrix-vector products efficiently. There is, however, one subtlety: unless the elements of G and I are of roughly the same order, the basis matrix associated with this algorithm will have columns of widely varying magnitude, and the resulting normal equations will be badly conditioned [8]. We must scale either the matrix G or the interlaced rows to overcome this difficulty.

INMF. Use information from the matrix F to construct B_1. Possibilities include an incomplete Cholesky factorization of F, or a Cholesky factorization of a portion of F (for example, the block diagonal portion). M_1 can then be constructed from rows of the resulting matrix. This approach may be particularly appropriate for saddle-point problems.

The next section includes results of experiments with each of the above methods.

4. Numerical experiments. Here we test the algorithms on a series of test problems based on the physical models shown in Figs. 5 and 6. We use two small structural engineering problems (DAM2 and SOLID1) to compare the performance of the conjugate gradient algorithms with the linear stationary methods 2-SOR and 3-AOR. We also test the conjugate gradient methods on larger, more realistic versions of these problems (DAM10 and SOLID2). We then consider problems for which algorithm BNP is not well suited. We test implicit nullspace methods on "damaged" versions of the structures problems (DAM10D and SOLID2D), in which the matrix G lacks full column rank. We also look at the marker-and-cell saddle-point formulation of the Stokes problem inside the unit rectangle (problem FLOW). See [8] for a more complete description of the problems, as well as more extensive experiments.

All experiments were run on a two-processor Alliant FX/40 using sparse data structures and double-precision arithmetic (vectorization on). The experiments conducted on two processors employed the parallel techniques outlined in §2 (see discussion accompanying Fig. 4). To compute errors on the small engineering problems DAM2 and SOLID1, we obtained the "true" solution by solving the original Kuhn–Tucker equations using LINPACK [3]. For the four larger structures problems (DAM10, SOLID2, DAM10D, and SOLID2D), we validated the codes by running them on smaller versions of the problems (comparing the solution to that produced by LINPACK). We then took the true solution to be the result obtained by algorithm INMGI with a stop tolerance of 10^{-10}. The true continuous solution was used for error computations in problem FLOW. The stop tolerances for the algorithms were adjusted so that they each produced a relative error of $2 \cdot 10^{-4}$ for the structures problems, and $3 \cdot 10^{-5}$ for the flow problem. The execution times (in seconds), obtained using the Alliant `etime` intrinsic, include all operations except input/output (for problem FLOW, this includes the cost of generating the matrices). In all problems, however, only the iteration times were significant: preprocessing, including factoring E and forming B_1, typically required only 1–2 percent of the cpu time.

Small full-rank structures problems. The first two test problems describe small elastic engineering structures; we developed the models using techniques described in Przemieniecki [14]. DAM2 (Fig. 5(a)) models a trapezoidal region intended to be a rough approximation of a cross-section of a dam. The blocks in F are 5×5 for the square elements, and 3×3 for triangular elements. The external load simulates a body of water against the left vertical wall. This version of DAM has 104 constraints and 244 unknowns. SOLID1 (Fig. 5(b)) is a rough model of a building subjected to a steady wind approaching one of its vertical edges. This version of SOLID consists of 60 solid tetrahedral elements. The problem involves 360 unknowns and 81 constraints; each of the 60 blocks in F is 6×6. The matrices for both problems reflect two substructures. In practice, of course, both DAM2 and SOLID1 are too small to justify substructuring techniques: the transition zones are far too large (15 percent and 33 percent of the total number of columns, respectively), and the substructures themselves are not well balanced. We developed substructured models primarily to validate the codes.

We used these problems to compare the relative performance of the conjugate gradient algorithms with the linear stationary methods. Since G has full column rank for these problems, algorithms BNP, INMG, and INMGI are equivalent algorithms. Since F is block diagonal with small diagonal blocks, it is easy to reduce G to upper triangular form, and there is no need to consider algorithms that presume it is difficult to do so. Thus, the only two implicit nullspace methods shown are INMI and BNP. The SOR and AOR runs are for the approximate optimal values of the iteration parameters (determined experimentally). The results are summarized in Table 1; both the times and the iteration counts suggest that the linear stationary methods are not competitive with BNP for this class of problems. We observed similar behavior on several other test problems of comparable size, supporting the very conservative theoretical results in [1] and [9]. Because of the poor performance of 2-SOR and 3-AOR on the small test problems, and the difficulty of determining appropriate iteration parameters for these algorithms, we did not feel it necessary to test the linear stationary methods on larger problems.

Larger full-rank structures problems. DAM10 and SOLID2 are larger versions of the models described above. In DAM10 there are 1,220 planar elements,

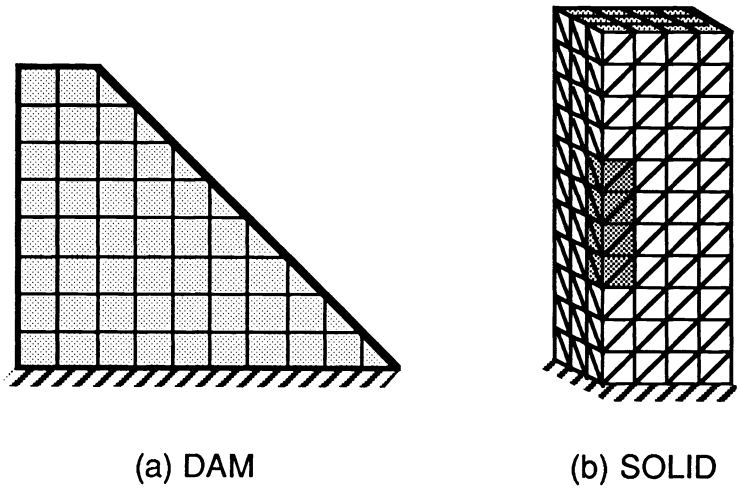


FIG. 5. *Structural engineering test problems.*

TABLE 1
Numerical results: Small full-rank structures problems.

DAM2
(244 unknowns, 104 constraints)

	BNP	INMI	2-SOR	3-AOR
Iterations	52	78	818	5,991
2 proc. time	.595	.791	8.24	60.8
1 proc. time	.899	1.30	12.4	91.7
Speedup	1.51	1.64	1.50	1.51

SOLID1
(360 unknowns, 81 constraints)

	BNP	INMI	2-SOR	3-AOR
Iterations	41	88	208	609
2 proc. time	.735	1.32	3.32	9.56
1 proc. time	1.08	2.06	4.81	14.0
Speedup	1.47	1.56	1.45	1.46

producing a problem with 2,440 constraints and 6,020 unknowns. We consider two versions of the problem: one involving no substructuring of the physical domain, and another with two substructures of fairly equal size (and 178 columns in the transition zone). SOLID2 models a tall rectangular solid; there are 220 free nodes and 660 tetrahedral elements in this model, producing a problem with 660 constraints and 3,960 unknowns. As with DAM10, we consider one version with no substructuring, and a second version with two substructures. Even though the matrices are fairly large for this problem, the geometry of the model does not allow a small transition zone: there are 360 transition columns in SOLID2, which is almost 10 percent of the total number of columns in the equilibrium matrix.

TABLE 2
Numerical results: Large full-rank structures problems.

	DAM10 (E is $2,440 \times 6,020$)		**SOLID2** (E is $660 \times 3,960$)	
	Two Substructures:		Two Substructures:	
	BNP	INMI	BNP	INMI
Iterations	910	1,416	223	689
2 proc. time	227.	355.	63.3	191.
1 proc. time	386.	603.	87.5	277.
Speedup	1.70	1.70	1.38	1.45
	No Substructuring:		No Substructuring:	
	BNP	INMI	BNP	INMI
Iterations	782	1,139	429	1,146
1 proc. time	304.	444.	112.	291.

Once again, G has full column rank for these problems, so BNP, INMG, and INMGI are equivalent algorithms. Moreover, it is easy to produce G in upper triangular (in fact, block diagonal) form, so there is no need to consider algorithms that presume G is difficult to use. Thus, we test only BNP and INMI. The results are summarized in Table 2. Algorithm BNP outperforms INMI by a wide margin on both problems. This supports what intuition might suggest: it is best to use rows of G to form the preconditioner when it is possible to do so.

Comparing the results with and without substructuring is also quite interesting: the problems change character completely when we change the partitioning of the physical domain. For DAM10, convergence is significantly better without substructuring, while for SOLID2, it it significantly better with two substructures. On both problems, however, the results with two substructures on two processors are superior to those obtained with one processor on one substructure. Speedups (comparing substructured results on one versus two processors) were not as good as we would have liked, but clearly reflect the size of the transition.zones: tests on SOLID2, which has a large transition zone, exhibit significantly poorer parallel performance.

Rank-deficient structures problems. To test our ability to solve problem LSE when the matrix G lacks full column rank, we modified the problems DAM10 and SOLID2 to simulate the presence of "damaged" elements. We did this by adjusting the physical parameters so that selected elements are "damaged" in the sense that the associated blocks of F are rank deficient. In problem SOLID2D, we "damaged" a collection of elements along an edge facing the wind (the shaded area in Fig. 5(b)); the result is a problem with 660 constraints, 3,960 unknowns, and 20 rank-deficient blocks in F. The rank of G is 3,940, which is 20 short of the number of columns in the matrix. Similarly, DAM10D is a modified version of DAM10. The "damaged" region consists of 100 elements, or 8 percent of the total area of the model. The rank of G is 5,920, which is 100 short of the total number of columns in G.

Because the matrix G lacks full column rank in each of these problems, algorithms BNP and INMG cannot be used; the same is true of the block SOR and AOR methods. Two examples of more general implicit nullspace methods handle the problems nicely, however (Table 3). Both INMI and INMGI successfully solve the problems, with INMGI producing substantially better results. In fact, INMGI comes very close to

TABLE 3
Numerical results: Rank-deficient structures problems.

	DAM10D (E is $2,440 \times 6,020$)		**SOLID2D** (E is $660 \times 3,960$)	
	Two Substructures:		Two Substructures:	
	INMGI	INMI	INMGI	INMI
Iterations	1,645	2,380	1,266	1,802
2 proc. time	410.	591.	349.	497.
1 proc. time	713.	1,020.	478.	692.
Speedup	1.74	1.73	1.37	1.39

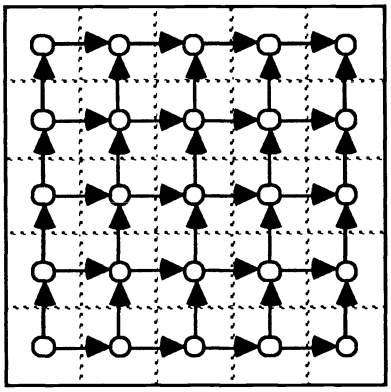


FIG. 6. *Marker and cell grid for Stokes problem.*

achieving a preconditioner based entirely on rows of G. On problem DAM10D, INMGI used only 10 rows of the scaled identity to produce an upper triangular matrix B_1. On problem SOLID2D, INMGI needed only 12 rows of the identity matrix. Again, the results support intuition: apparently, the more rows of G we can use in the preconditioner, the better the convergence.

Saddle-point problems. To test implicit nullspace methods on a saddle-point problem, we consider problem FLOW, a discretization of the Stokes equations subject to Dirichlet boundary conditions on the unit square

$$\nabla^2 \underline{v} - \nabla p = g,$$
$$\nabla \cdot \underline{v} = f,$$
$$\underline{v} = \underline{v}_0 \quad \text{on the boundary.}$$

The variables are defined in §1. We employ the MAC (marker-and-cell) finite differencing described in Hall [6], in which pressure is specified at nodes placed at the center of rectangular cells, and velocity is specified at the centers of edges connecting the pressure nodes (Fig. 6). The reported test problem has 100 cells in each direction, resulting in 19,800 unknown velocity components and 9,999 pressure values (constraints). We test the algorithm on an artificial "flow" that is irrotational but has nonzero divergence:

$$v_1 = 2x \cos y,$$

TABLE 4
Summary of results for problem FLOW.

Two Subdomains:

	INMI	INMF	INMG
Iterations	8,534	8,352	1,132
2 proc. time	2,990.	3,400.	463.
1 proc. time	5,180.	6,090.	835.
Speedup	1.73	1.79	1.80

No Domain Decomposition:

	INMI	INMF	INMG
Iterations	8,721	8,503	385
1 proc. time	5,010.	5,930.	274.

$$v_2 = -x^2 \sin y,$$
$$p = xy^2.$$

Before substructuring, the matrix F is block diagonal with two large blocks, each of which reflects a five-point discretization of the Laplacian. Its Cholesky factor is too dense to be used efficiently as G; instead, we take G in INMG to be two copies of the discretized gradient operator. For algorithm INMF, we compute the Cholesky factor of the block diagonal portion of F (each block in the diagonal of F is a tridiagonal matrix), and use rows of the resulting matrix to construct M_1.

Table 4 shows the results of experiments on FLOW. Here again, algorithm BNP cannot be used: the vector c is not explicitly available. On the other hand, algorithm INMG, the nullspace version of BNP, is well suited to deal with this problem. Here we compare INMG (which uses rows of the discrete divergence G to form the preconditioner), INMI (which uses rows of the identity), and INMF (which uses rows of F, the negative of the Laplacian). We observe that the tridiagonal portion of F produces a poor preconditioner in INMF. It seems clear we cannot afford to ignore the information in the outer bands of F when constructing the preconditioner, especially when we are only using selected rows of the factorization we obtain. Similarly, INMI performs poorly. INMG, on the other hand, performs fairly well; once again, the results suggest that we should use rows of G whenever possible when constructing the preconditioner.

Notice that we obtain more reasonable speedups on the flow problem: the transition zone is quite small (1.5 percent of the dimension of F), and the subdomains are virtually identical in structure. On the other hand, the results for INMG without domain decomposition reflect an extraordinary reduction in the number of iterations. The change in the rate of convergence is so dramatic that the time required to solve the problem on one processor without domain decomposition is almost half that needed for the substructured problem on two processors. Clearly, we would need a more sophisticated approach to constructing M_1 to have any hope of overcoming the penalty associated with this type of domain decomposition applied to the Stokes problem. Incomplete Cholesky decompositions offer one possibility worth exploring.

We believe the results suggest that implicit nullspace methods provide a promising alternative to existing iterative methods for solving constrained minimization problems. The INM approach offers considerable flexibility in the construction of the preconditioner, allowing us to exploit special characteristics of the matrices (e.g.,

substructured form) as well as to overcome difficulties inherent in the problem (e.g., a rank-deficient G). We are convinced that further research on more effective ways to produce the augmentation matrix M_1 will prove fruitful.

Acknowledgments. The author gratefully acknowledges the assistance and encouragement of Jesse L. Barlow of The Pennsylvania State University throughout the course of this research. He also appreciates the thoughtful comments of the anonymous referees.

REFERENCES

[1] J. Barlow, N. Nichols, and R. Plemmons, *Iterative methods for equality constrained least squares problems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 892–906.
[2] A. Bjorck, *Least squares methods*, in Handbook for Numerical Methods, Vol. 1, P. Ciarlet and J. Lions, eds., Elsevier/North-Holland, Amsterdam, 1989.
[3] J. Dongarra, J. Bunch, C. Moler, and G. Stewart, LINPACK *Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.
[4] R. Freund, *A note on two block*-SOR *methods for sparse least squares problems*, Linear Algebra Appl., 88–89 (1987), pp. 211–221.
[5] A. Hadjidimos, *Accelerated overrelaxation method*, Math. Comp., 32 (1978), pp. 149–157.
[6] C. Hall, *Numerical solution of Navier–Stokes problems by the dual variable method*, SIAM J. Algebraic Discrete Methods, 6 (1985), pp. 220–236.
[7] M. Heath, R. Plemmons, and R. Ward, *Sparse orthogonal schemes for structural optimization using the force method*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 514–532.
[8] D. James, *Conjugate gradient methods for constrained least squares problems*, Ph.D. thesis, Department of Mathematics, North Carolina State University, Raleigh, NC, 1990.
[9] ———, *Order-reducing conjugate gradients versus block* AOR *for constrained least squares problems*, Linear Algebra Appl., 154–156 (1991), pp. 23–43.
[10] D. James and R. Plemmons, *An iterative substructuring algorithm for equilibrium equations*, Numer. Math., 57 (1990), pp. 625–633.
[11] T. Markham, M. Neumann, and R. Plemmons, *Convergence of a direct-iterative method for large scale least squares problems*, Linear Algebra Appl., 69 (1985), pp. 155–167.
[12] E. Papadopoulou, Y. Saridakis, and T. Papatheodorou, *Block* AOR *iterative schemes for large-scale least-squares problems*, SIAM J. Numer. Anal., 26 (1989), pp. 637–660.
[13] R. Plemmons and R. White, *Substructuring methods for computing the nullspace of equilibrium matrices*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 1–22.
[14] J. Przemieniecki, *Theory of Matrix Structural Analysis*, Dover, New York, 1985.
[15] G. Strang, *A framework for equilibrium equations*, SIAM Rev., 30 (1988), pp. 283–297.
[16] ———, *Introduction to Applied Mathematics*, Wellesley Cambridge Press, Wellesley, MA, 1986.
[17] R. Varga, *p-cyclic matrices: A generalization of the Young–Frankel successive overrelaxation scheme*, Pacific J. Math., 9 (1959), pp. 617–623.
[18] ———, *Matrix Iterative Analysis*, Prentice–Hall, Englewood Cliffs, NJ, 1963.
[19] D. Young, *Iterative methods for solving partial differential equations of elliptic type*, Trans. Amer. Math. Soc., 76 (1954), pp. 92–111.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 3, pp. 979–991, July 1992

017

ACCELERATION OF RELAXATION METHODS FOR NON-HERMITIAN LINEAR SYSTEMS*

M. EIERMANN†, W. NIETHAMMER†, AND R. S. VARGA‡

Abstract. Let $A = I - B \in \mathbb{C}^{n,n}$, with $\operatorname{diag}(B) = \mathbf{0}$, denote a nonsingular non-Hermitian matrix. To iteratively solve the linear system $A\mathbf{x} = \mathbf{b}$, two splittings of A, together with induced relaxation methods, have been recently investigated in [W. Niethammer and R. S. Varga, *Results in Math.*, 16 (1989), pp. 308–320]. The *Hermitian splitting* of A is defined by $A = M^h - N^h$, where $M^h := (A + A^*)/2$ is the Hermitian part of A. The *skew-Hermitian splitting* of A is similarly defined by $A = M^s - N^s$ with $M^s := I + (A - A^*)/2$.

This paper considers k-step iterative methods to accelerate the relaxation schemes (involving a relaxation factor ω) that are generated by these two splittings. The primary interest is not to determine the optimal relaxation factor ω that minimizes the spectral radius of the associated iteration operator. Rather, a value of ω is sought such that the resulting relaxation method can be most efficiently accelerated by a k-step method. For the Hermitian splitting, the choice $\omega = 1$ (together with a suitable Chebyshev acceleration) turns out to be optimal in this sense. For the skew-Hermitian splitting, a *hybrid* scheme is proposed that is nearly optimal.

As another application of this latter hybrid procedure, the block Jacobi method arising from a model equation for a convection-diffusion problem is analyzed.

Key words. iterative methods for non-Hermitian matrix equations, relaxation methods, Hermitian splittings, skew-Hermitian splittings, Chebyshev acceleration

AMS(MOS) subject classification. 65F10

1. Introduction.

To solve a nonsingular linear system of algebraic equations

$$A\mathbf{x} = \mathbf{x} - B\mathbf{x} = \mathbf{b} \qquad (A, B \in \mathbb{C}^{n,n},\ \operatorname{diag}(B) = \mathbf{0},\ \mathbf{b} \in \mathbb{C}^n) \tag{1.1}$$

whose coefficient matrix A is *non-Hermitian*, Niethammer and Varga [11] recently studied relaxation methods based on either the Hermitian or the skew-Hermitian splitting of $A = I - B$. Letting

$$F := (B + B^*)/2 \quad \text{and} \quad G := (B - B^*)/2 \tag{1.2}$$

denote, respectively, the Hermitian and skew-Hermitian parts of B, then the *Hermitian splitting* of A is defined by

$$A = M^h - N^h \quad \text{with } M^h := I - F \quad \text{and} \quad N^h := G \tag{1.3}$$

(here, we assume that M^h is invertible, which is, for instance, guaranteed if the Hermitian part M^h of A is positive definite). The naturally associated *skew-Hermitian splitting* of A is given by

$$A = M^s - N^s \quad \text{with } M^s := I - G \quad \text{and} \quad N^s := F\,. \tag{1.4}$$

It should be mentioned that Concus and Golub [2] earlier introduced the Hermitian splitting (1.3) of A. Under the assumption that M^h is positive definite, they chose M^h as a preconditioner for an associated conjugate gradient method.

* Received by the editors April 5, 1990; accepted for publication June 20, 1991.

† Institut für Praktische Mathematik, Universität Karlsruhe, D-7500 Karlsruhe 1, Germany.

‡ Institute for Computational Mathematics, Kent State University, Kent, Ohio 44242. The research of this author was supported by the Alexander von Humboldt Foundation.

Each splitting $A = M - N$ of A gives rise to a class of one-parameter relaxation schemes for the solution of (1.1), namely,

$$\{(1-\omega)I + \omega M\}\,\mathbf{x}_m := \{(1-\omega)I + \omega N\}\,\mathbf{x}_{m-1} + \omega\mathbf{b}\,, \tag{1.5}$$

or equivalently,

$$\mathbf{x}_m = \mathcal{T}_\omega \mathbf{x}_{m-1} + \mathbf{c}_\omega \qquad (m = 1, 2, \cdots), \tag{1.6}$$

where $\omega \neq 0$ is an arbitrary complex number for which $(1-\omega)I + \omega M$ is nonsingular and

$$\mathcal{T}_\omega := \{(1-\omega)I + \omega M\}^{-1}\{(1-\omega)I + \omega N\}, \qquad \mathbf{c}_\omega := \omega\{(1-\omega)I + \omega M\}^{-1}\mathbf{b}\,.$$

In this way, the specific splittings defined in (1.3) and (1.4) generate the following two relaxation methods:

$$\mathbf{x}_m := \mathcal{T}_\omega^h \mathbf{x}_{m-1} + \mathbf{c}_\omega^h \qquad (m = 1, 2, \cdots), \tag{1.7}$$

where

$$\mathcal{T}_\omega^h := (I - \omega F)^{-1}\{(1-\omega)I + \omega G\}, \qquad \mathbf{c}_\omega^h := \omega(I - \omega F)^{-1}\mathbf{b},$$

and

$$\mathbf{x}_m := \mathcal{T}_\omega^s \mathbf{x}_{m-1} + \mathbf{c}_\omega^s \qquad (m = 1, 2, \cdots), \tag{1.8}$$

where

$$\mathcal{T}_\omega^s := (I - \omega G)^{-1}\{(1-\omega)I + \omega F\}, \qquad \mathbf{c}_\omega^s := \omega(I - \omega G)^{-1}\mathbf{b},$$

these methods each depending on a single relaxation parameter ω.

Under the assumption (cf. (1.2)) that $I - F$ is Hermitian and positive definite, Niethammer and Varga [11] determined inclusion sets for the eigenvalues of the corresponding relaxation matrices $\mathcal{T}_\omega^h$ and $\mathcal{T}_\omega^s$. To be more precise, they showed that the spectrum $\sigma(\mathcal{T}_\omega^h)$ of $\mathcal{T}_\omega^h$ is contained in a certain rectangle (which degenerates to an interval on the imaginary axis when $\omega = 1$) (cf. [11, Fig. 1]), whereas $\sigma(\mathcal{T}_\omega^s)$ is contained in a *bow-tie region* (cf. [11, Fig. 2]). In this paper, we examine the question of whether these facts can be used to effectively *accelerate* the procedures (1.7) and (1.8) by the application of *k-step iterative methods*, such as the *Chebyshev semi-iterative method* (when $k = 2$).

To go beyond the special schemes (1.7) and (1.8), we need some additional terminology. Let

$$\mathbf{x}_m = T\mathbf{x}_{m-1} + \mathbf{c} \qquad (m = 1, 2, \ldots), \tag{1.9}$$

with $1 \notin \sigma(T)$, be called a *basic iteration* for the solution of (1.1) which results from a splitting of the matrix A. We assume that we have a priori information about the eigenvalues of T of the form

$$\sigma(T) \subseteq \Omega\,, \tag{1.10}$$

where Ω is a compact subset of the complex plane with $1 \notin \Omega$. In addition, with the notation $\overline{\mathbb{C}} := \mathbb{C} \cup \{\infty\}$, we always require that Ω has no isolated points and that

$\overline{\mathbb{C}} \setminus \Omega$ is of finite connectivity. Note that for $\sigma(\mathcal{T}_\omega^h)$ of (1.7), as well as for $\sigma(\mathcal{T}_\omega^s)$ of (1.8), inclusions of this type are available, as previously mentioned. To (1.9), we now apply the *k-step method* (cf. [10])

$$\mathbf{y}_m := \mu_{m,0}(T\,\mathbf{y}_{m-1} + \mathbf{c}) + \mu_{m,1}\mathbf{y}_{m-1} + \mu_{m,2}\mathbf{y}_{m-2} + \cdots + \mu_{m,k}\mathbf{y}_{m-k}\,, \tag{1.11}$$

where $\mu_{m,j} \in \mathbb{C}$ ($\mu_{m,j} := 0$ for $j > m$) and $\sum_{j=0}^{k} \mu_{m,j} = 1$ ($m = 1, 2, \cdots$), in order to accelerate the convergence of the basic iterations (1.9). It is well known that these k-step methods belong to the class of *semi-iterative methods* or *polynomial acceleration methods* applied to (1.9) (cf. Varga [12]). The error vectors $\mathbf{e}_m := (I - T)^{-1}\mathbf{c} - \mathbf{y}_m$, associated with the mth iterate $\mathbf{y}_m$ of (1.11), can be written as $\mathbf{e}_m = p_m(T)\mathbf{e}_0$, where the polynomials $p_m(z) = \sum_{j=0}^{m} \pi_{m,j} z^j$ are recursively defined by $p_m(z) := 0$ ($m < 0$), $p_0(z) := 1$ and

$$p_m(z) := (\mu_{m,0}z + \mu_{m,1})p_{m-1}(z) + \mu_{m,2}p_{m-2}(z) + \cdots + \mu_{m,k}p_{m-k}(z) \quad (m = 1, 2, \cdots).$$

For notational convenience, we collect the Taylor coefficients of each of the polynomials p_m into an infinite lower triangular matrix $P = (\pi_{m,j})_{m \geq j \geq 0}$ which we call the *generating matrix* for the k-step method (1.11). Then it is known (cf. [4]) that, for a given P, the value of

$$\kappa(T, P) := \limsup_{m \to \infty} \sup_{\mathbf{e}_0 \neq \mathbf{0}} \left[\frac{\|\mathbf{e}_m\|}{\|\mathbf{e}_0\|} \right]^{1/m}$$

depends only on the structure of the Jordan canonical form of the matrix T. With regard to (1.10), we therefore define the *asymptotic convergence factor of the k-step method* (1.11), *with respect to the information* $\sigma(T) \subseteq \Omega$, by

$$\kappa(\Omega, P) := \max\{\kappa(T, P) : T \in \mathbb{C}^{n,n}, n \geq 1, \text{ with } \sigma(T) \subseteq \Omega\}\,. \tag{1.12}$$

The best, i.e., *smallest*, convergence factor we can hope to achieve by any k-step method ($k = 1, 2, \cdots$) in this worst-case philosophy is the *asymptotic convergence factor* of Ω, defined by

$$\kappa(\Omega) := \inf\{\kappa(\Omega, P) : \ P \text{ generates a } k\text{-step method }, k = 1, 2, \cdots\}\,. \tag{1.13}$$

The infimum in (1.13) is actually a minimum (cf. [4]), and each k-step method for which this minimum is attained will be called *asymptotically optimal with respect to* Ω. The best-known examples of asymptotically optimal methods are the Chebyshev semi-iterative methods studied by Manteuffel [9]. In [9], Ω is an ellipse with either real foci or complex conjugate foci, with $1 \notin \Omega$, and in these cases there exists a Chebyshev semi-iterative method, i.e., a two-step method, which is asymptotically optimal with respect to Ω. (We note, more generally, that *any* ellipse Ω in $\mathbb{C}$ with $1 \notin \Omega$ admits an asymptotically optimal two-step method (cf. [10]).)

The quantity $\kappa(\Omega)$ of (1.13) has some interesting *capacity-like* properties. For example, it is known that

$$\kappa(\Omega_1) < \kappa(\Omega_2) \tag{1.14}$$

if Ω_1 is a proper subset of Ω_2 (cf. [3, Prop. 3]). To compare the convergence factors of two compact sets Ω_1 and Ω_2 with $\Omega_1 \not\subseteq \Omega_2$ and $\Omega_2 \not\subseteq \Omega_1$, another observation is

helpful. Let Ω be a compact subset of $\mathbb{C}$ with $1 \notin \Omega$, and let $t_m \in \Pi_m$ be a polynomial of the exact degree m satisfying $t_m(1) = 1$ and the condition

$$t_m(z) = 1 \quad \text{implies} \quad z \notin \Omega\,.$$

Then it is known (cf. [3, Lemma 4]) that

$$\kappa(\Omega) \leq [\kappa(t_m(\Omega))]^{1/m}\,, \tag{1.15}$$

where $t_m(\Omega)$ denotes the image of Ω under the polynomial transformation $z \mapsto t_m(z)$. Moreover, equality holds in (1.15) if and only if the implication

$$z \notin \Omega \quad \text{implies} \quad t_m(z) \notin t_m(\Omega) \tag{1.16}$$

is valid for every $z \in \mathbb{C}$ (cf. [3, Thm. 6]).

If $\overline{\mathbb{C}} \setminus \Omega$ is simply connected, and if Φ denotes the Riemann mapping function that maps the complement of Ω conformally onto the exterior of the unit circle such that the points at infinity correspond to each other, then $\kappa(\Omega)$ can be expressed explicitly (cf. [4, Thm. 11]) as

$$\kappa(\Omega) = \frac{1}{|\Phi(1)|} < 1\,, \tag{1.17}$$

the last inequality in (1.17) following from the assumption that $1 \notin \Omega$. In what follows, (1.17) turns out to be useful when one must decide whether it is worthwhile to apply a k-step method of the form (1.11) to a given basic iteration (1.9) whose operator T satisfies $\sigma(T) \subseteq \Omega$.

We briefly describe the contents of this paper. For the relaxation method (1.7) associated with the Hermitian splitting of A, we examine in §2 which choice of ω in (1.7) is best. Here, we are not interested in merely minimizing $\rho(\mathcal{T}_\omega^h)$ as a function of ω; rather, we seek a value of ω such that (1.7) can be most efficiently accelerated by a k-step method. It turns out that the choice $\omega = 1$ is best in this sense. For the relaxation method (1.8) associated with the skew-Hermitian splitting of A, we accelerate (1.8), in §3, by means of a new hybrid scheme, consisting of a polynomial transformation of the given linear system, together with the application of a stationary one-step method to the transformed equations. An optimized Chebyshev acceleration of (1.8), as given in Chin and Manteuffel [1], will be shown here to converge more *slowly* than this hybrid method. In §4, we apply our hybrid procedure to the block Jacobi method for a model equation of a convection-diffusion problem. Again, we make use of results due to Chin and Manteuffel [1], who determined sets in the complex plane containing all eigenvalues of the block Jacobi matrix in this example. Finally, a comparison with the associated block successive overrelaxation (SOR) method is given.

2. The Hermitian splitting. We first consider the *Hermitian* splitting (1.3) of A and its associated relaxation methods (1.7). Let $\{\gamma_j\}_{j=1}^n$ denote the eigenvalues of F (cf. (1.2)) with $\alpha := \gamma_1 \leq \gamma_2 \leq \cdots \leq \gamma_n =: \beta$. We always assume that the Hermitian part of A, namely $I - F$, is positive definite. This assumption, coupled with the hypothesis of (1.1) that the trace of F is zero, implies that

$$\alpha \leq 0 \leq \beta < 1\,. \tag{2.1}$$

For all $\omega \in (0, 1/\beta)$, the matrix $I - \omega F$ is evidently nonsingular and the relaxation matrix $\mathcal{T}_\omega^h$ of (1.7) is thus defined for these values of ω. From [11, (3.13) and Fig. 1],

we further have the inclusion

$$\sigma(\mathcal{T}_\omega^h) \subseteq \Omega_\omega := \begin{cases} \{z \in \mathbb{C} : c \le \operatorname{Re} z \le d,\ |\operatorname{Im} z| \le f\}, & \text{if } 0 < \omega < 1, \\ [-if, if], & \text{if } \omega = 1, \\ \{z \in \mathbb{C} : d \le \operatorname{Re} z \le c,\ |\operatorname{Im} z| \le f\}, & \text{if } 1 < \omega < 1/\beta, \end{cases} \tag{2.2}$$

where

$$c = c(\omega) := \frac{1-\omega}{1-\omega\alpha}, \quad d = d(\omega) := \frac{1-\omega}{1-\omega\beta}, \quad \text{and} \quad f = f(\omega) := \frac{\omega\rho(G)}{1-\omega\beta}.$$

In every case, the rectangle Ω_ω is contained in the open unit disk if and only if $d^2(\omega) + f^2(\omega) < 1$, which is equivalent (cf. [11, Thm. 3.2]) to

$$0 < \omega < \omega_g := \frac{2(1-\beta)}{1+\rho^2(G)-\beta^2}. \tag{2.3}$$

The relaxation method (1.7) is therefore guaranteed to converge for any ω in the interval $(0, \omega_g)$.

A natural question now is which choice of ω is optimal. The classical concept of optimality seeks a minimum of the spectral radius $\rho(\mathcal{T}_\omega^h)$ as a function of ω. With our limited information, $\sigma(\mathcal{T}_\omega^h) \subseteq \Omega_\omega$, i.e.,

$$\rho(\mathcal{T}_\omega^h) \le \hat{\rho}(\omega) := \sqrt{d^2(\omega) + f^2(\omega)},$$

we seek to minimize $\hat{\rho}(\omega)$ for $\omega \in (0, \omega_g)$. An easy calculation shows that $\hat{\rho}$ has exactly one minimum in $(0, \omega_g)$, which is attained at

$$\omega^* = \frac{1-\beta}{1-\beta+\rho^2(G)} \qquad \text{with } \hat{\rho}(\omega^*) = \frac{\rho(G)}{\sqrt{(1-\beta)^2+\rho^2(G)}}. \tag{2.4}$$

The question is whether this definition of optimality really makes sense in our context. Let us consider the following example. For $\rho(G) = 1$ and $\beta = 0.5$, we obtain from (2.4) that $\omega^* = \frac{1}{3}$ and $\rho(\mathcal{T}_{1/3}^h) \le \hat{\rho}(\omega^*) = 2/\sqrt{5} = 0.8944\cdots$. On the other hand, if we choose $\omega = 1$, then Ω_1 reduces to a line segment, i.e., $\Omega_1 = [-2i, 2i]$, and $\mathcal{T}_1^h$ may be divergent. But the Chebyshev semi-iterative method for this interval, or equivalently, the stationary two-step method

$$\mathbf{y}_m := \mu_0(\mathcal{T}_1^h \mathbf{y}_{m-1} + \mathbf{c}_1) + \mu_1 \mathbf{y}_{m-1} + \mu_2 \mathbf{y}_{m-2} \qquad (m = 2, 3, \cdots) \tag{2.5}$$

with

$$\mu_0 := (\sqrt{5}-1)/2, \quad \mu_1 = 0, \quad \text{and} \quad \mu_2 = 1 - \mu_0$$

has an asymptotic convergence factor of $(\sqrt{5}-1)/2 = 0.6180\cdots$ (cf. Niethammer and Varga [10, Ex. 2]). Is $\omega = 1$ therefore a better choice than $\omega = \frac{1}{3}$? At this stage, this could be a hasty conclusion since we can also accelerate $\mathbf{x}_m = \mathcal{T}_{1/3}^h \mathbf{x}_{m-1} + \mathbf{c}_{1/3}^h$ by a Chebyshev procedure or another k-step method, with the goal of constructing an even faster scheme.

After these considerations, we believe that it is more appropriate to determine an ω that minimizes $\kappa(\Omega_\omega)$ (cf. (1.13)), rather than $\hat{\rho}(\omega)$. The assertion of the following theorem is that $\omega = 1$ is optimal in this sense, i.e., the introduction of a relaxation

parameter $\omega \neq 1$ does *not* improve the iterative method based on the Hermitian splitting (1.3) of A.

THEOREM 1. *For each rectangle* Ω_ω (*cf.* (2.2)) *with* $0 < \omega < \omega_g$ (*cf.* (2.3)) *and* $\omega \neq 1$, *there holds*

$$\kappa(\Omega_\omega) > \kappa(\Omega_1) = \frac{\rho(G)}{1-\beta+\sqrt{(1-\beta)^2+\rho^2(G)}}.$$

Proof. Let ω be arbitrary (but fixed) in $(0, \omega_g)$ with $\omega \neq 1$. The interval

$$I_\omega := [d(\omega) - if(\omega),\ d(\omega) + if(\omega)]$$

(cf. (2.2)) is a proper subset of the rectangle Ω_ω. The polynomial (in Π_1) defined by

$$t_1(z) := \frac{1-\omega\beta}{\omega(1-\beta)}(z-1) + 1 \qquad (t_1(1) = 1),$$

induces a bijection of I_ω onto Ω_1. From (1.14) and (1.15), we therefore obtain

$$\kappa(\Omega_\omega) > \kappa(I_\omega) = \kappa(t_1(I_\omega)) = \kappa(\Omega_1)\,.$$

The explicit expression for $\kappa(\Omega_1)$ has been derived in [10, Ex. 2]. □

We conclude this section with a simple example which shows that the bounds of (2.2) may be a considerable *overestimation* of the spectrum of $\mathcal{T}_\omega^h$.

Example 2.1. If we discretize the boundary value problem

$$-u''(t) + \tau u'(t) = f(t) \quad \text{on } (0,1) \text{ with given } u(0), u(1) \in \mathbf{R} \tag{2.6}$$

($\tau > 0$), by using central differences with mesh size $h = 1/(n+1)$, a linear system results whose coefficient matrix A is an $n \times n$ Toeplitz tridiagonal matrix. Normalizing its diagonal entries to be equal to 1, we have

$$A = \text{tridiag}[-(1+R)/2, 1, -(1-R)/2] \in \mathbf{R}^{n,n} \tag{2.7}$$

with the mesh Reynolds number $R := \tau h/2$ (cf. Elman and Golub [5, §2]). Thus, its Hermitian splitting is given (cf. (1.3)) by

$$A = (I - \text{tridiag}[1/2, 0, 1/2]) - \text{tridiag}[R/2, 0, -R/2]. \tag{2.8}$$

Since the eigenvalues of the matrix $\text{tridiag}[a, 0, b] \in \mathbf{C}^{n,n}$ are known (cf. [5, Lemma 2]) to be

$$\lambda_k = 2\sqrt{ab}\cos\left(\frac{\pi k}{n+1}\right) \qquad (k = 1, 2, \cdots, n),$$

we deduce from (2.2) the estimate

$$\rho(\mathcal{T}_1^h) \leq \frac{R\cos(\pi h)}{1-\cos(\pi h)} = \frac{\tau}{\pi^2 h} + \mathcal{O}(h) \qquad (h \to 0) \tag{2.9}$$

(where, because of Theorem 1, we only consider $\omega = 1$). On the other hand, in this simple example, the eigenvalues λ of $\mathcal{T}_1^h = (I-F)^{-1}G$ can be computed explicitly. Let λ be such an eigenvalue. If $\lambda \neq 0$, then

$$(I-F) - \frac{1}{\lambda}G = \text{tridiag}\left[-\frac{1-R/\lambda}{2}, 1, -\frac{1+R/\lambda}{2}\right]$$

must be singular, or equivalently, there must be a $k \in \{1, 2, \cdots, n\}$ such that

$$2\sqrt{\frac{1}{4} - \frac{R^2}{4\lambda^2}}\cos(\pi k h) = 1\,.$$

From this, it is easy to see that the eigenvalues of $\mathcal{T}_1^h = (I - F)^{-1}G$ are given by

$$\lambda = \pm R\cot(\pi k h)i, \qquad \begin{cases} k = 1, 2, \cdots, n/2, & \text{if } n \text{ is even,} \\ k = 1, 2, \cdots, (n-1)/2, & \text{if } n \text{ is odd.} \end{cases}$$

(If n is odd, then $\mathcal{T}_1^h$ has in addition the eigenvalue $\lambda = 0$.) This implies that

$$\rho(\mathcal{T}_1^h) = R\cot(\pi h) = \frac{\tau}{2\pi} + \mathcal{O}(h^2) \qquad (h \to 0),$$

which is an order of magnitude smaller than the estimate (2.9).

3. The skew-Hermitian splitting. We turn now to the investigation of the relaxation methods (1.8) induced by the skew-Hermitian splitting $A = (I - G) - F$ (cf. (1.4)) of A. As in the previous section, we again assume that the Hermitian part $M^h = I - F$ of A is positive definite, i.e., for the eigenvalues $\alpha = \gamma_1 \leq \cdots \leq \gamma_n = \beta$ of F, there holds, as in (2.1),

$$\alpha \leq 0 \leq \beta < 1\,.$$

For the relaxation matrix $\mathcal{T}_\omega^s$ of (1.8) with $\omega > 0$, Niethammer and Varga [11] derived the eigenvalue inclusion

$$\sigma(\mathcal{T}_\omega^s) \subseteq \tilde{\Omega}_\omega := \{z \in \mathbb{C} : |z - c_1(\omega)| \leq |c_1(\omega)| \text{ or } |z - c_2(\omega)| \leq |c_2(\omega)|\}, \tag{3.1}$$

where

$$c_1(\omega) := \frac{1 - \omega + \omega\beta}{2}, \qquad c_2(\omega) := \frac{1 - \omega + \omega\alpha}{2}.$$

They actually proved more, namely, that $\sigma(\mathcal{T}_\omega^s)$ is contained in a bow-tie region (cf. [11, Fig. 2]), which is itself contained in $\tilde{\Omega}_\omega$. These bow-tie regions depend on the spectral radius $\rho(G)$ of the skew-Hermitian part G of A, and fill out $\tilde{\Omega}_\omega$ as $\rho(G)$ tends to infinity. The estimate (3.1) has the advantage of being independent of $\rho(G)$.

Note that under the given assumption on α and β, we have

$$c_2(\omega) \leq c_1(\omega) < \tfrac{1}{2},$$

i.e., $\rho(\mathcal{T}_\omega^s) < 1$ if $c_2(\omega) > -\frac{1}{2}$, which is equivalent to $\rho(\mathcal{T}_\omega^s) < 1$ for $0 < \omega < \omega_g := 2/(1 - \alpha)$ (cf. [11, Thm. 4.1]). The optimal relaxation factor ω_0, with respect to the information (3.1), occurs when the condition $c_2(\omega_0) = -c_1(\omega_0)$ holds, i.e.,

$$\omega_0 = \frac{2}{2 - (\alpha + \beta)} \quad \text{with } \rho(\mathcal{T}_{\omega_o}^s) \leq \frac{\beta - \alpha}{2 - (\alpha + \beta)} \tag{3.2}$$

(cf. [11, Thm. 4.1]). In this latter case, we give the associated bow-tie region in Fig. 1.

We now apply these results to the example already discussed in the last section.

Example 3.1. The skew-Hermitian splitting (1.4) of the matrix A of (2.7) has the form $A = (I - G) - F$, where $F = \text{tridiag}[1/2, 0, 1/2]$ and $G = \text{tridiag}[R/2, 0, -R/2]$ (cf. (2.8)). With $\alpha = -\cos(\pi h)$ and $\beta = \cos(\pi h)$, we have $\omega_0 = 1$ and

$$\rho(\mathcal{T}_1^s) \leq \cos(\pi h) < 1\,.$$

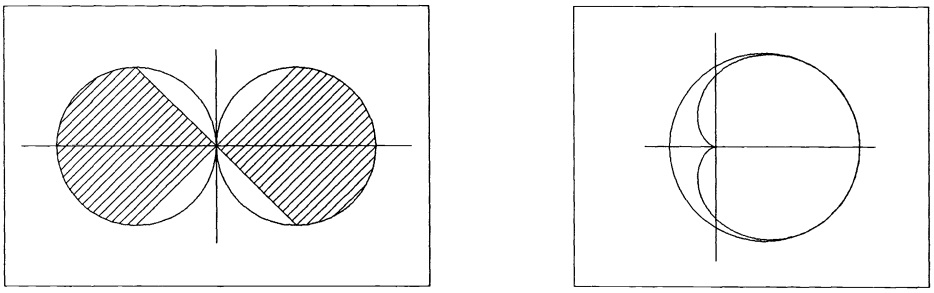


FIG. 1. *Shape of the bow-tie region containing* $\sigma(\mathcal{T}_\omega^s)$ (*cf.* (3.1)) *together with* Λ_c (*cf.* (3.4)), *where* $c_1(\omega) = -c_2(\omega)$ (*left-hand side*). Λ_c^2 (*cf.* (3.9)) *and the enclosing* ("*embracing*") *disk* $D(\zeta(\mu_0^*), r(t_0))$ (*cf. Proposition* 3.2) (*right-hand side*).

Thus, the matrix $\mathcal{T}_1^s$ is convergent for *any* value of the parameter τ (cf. (2.6)) and for *any* mesh size h. As in the case of $\mathcal{T}_1^h$, the eigenvalues of $\mathcal{T}_1^s$ can be calculated explicitly in this simple example. With the technique used in Example 1, we see that the spectrum of $\mathcal{T}_1^s$ consists of the points

$$\lambda = \pm \frac{\cos(\pi k h)}{\sqrt{R^2 \cos^2(\pi k h) + 1}}, \qquad \begin{cases} k = 1, 2, \cdots, \frac{n}{2}, & \text{if } n \text{ is even,} \\ k = 1, 2, \cdots, \frac{n-1}{2}, & \text{if } n \text{ is odd.} \end{cases}$$

(If n is odd, we again note that $\mathcal{T}_1^s$ has the additional eigenvalue $\lambda = 0$.) Note further that $\sigma(\mathcal{T}_1^s)$ possesses only *real* elements. We thus conclude that

$$\rho(\mathcal{T}_1^s) = \frac{\cos(\pi h)}{\sqrt{R^2 \cos^2(\pi h) + 1}}.$$

To accelerate the convergence of the basic iteration (1.8), the Chebyshev semi-iterative method can be applied to the interval $[-\rho(\mathcal{T}_1^s), \rho(\mathcal{T}_1^s)]$.

The problem we wish to consider now is how we can use the information of (3.1) to accelerate the convergence of the basic iterative method $\mathbf{x}_m := \mathcal{T}_\omega^s \mathbf{x}_{m-1} + \mathbf{c}^s$.

To simplify our notation, we consider the basic iteration method

$$\mathbf{x}_m = T\mathbf{x}_{m-1} + \mathbf{c} \qquad (m = 1, 2, \cdots), \tag{3.3}$$

where we assume that $\sigma(T) \subseteq \Lambda_c$, where

$$\Lambda_c := \{z \in \mathbb{C} : |z - c| \le c \quad \text{or} \quad |z + c| \le c\}, \tag{3.4}$$

with $0 < c < \frac{1}{2}$. (Note that $\tilde{\Omega}_\omega$ of (3.1) has this form for $\omega = \omega_0$.)

We first wish to design a stationary two-step method

$$\mathbf{y}_m = \mu_0(T\mathbf{y}_{m-1} + \mathbf{c}) + \mu_1 \mathbf{y}_{m-1} + \mu_2 \mathbf{y}_{m-2} \qquad (m = 2, 3, \cdots) \tag{3.5}$$

(with $\mu_0 + \mu_1 + \mu_2 = 1$), which is compatible with this information for $\sigma(T)$. Recently, Chin and Manteuffel [1] solved an analogous problem. They determined the optimal relaxation parameter ω of the SOR method under the assumption that the corresponding Jacobi matrix T is weakly cyclic of index 2 and satisfies the condition (3.4). Their result is the following proposition.

PROPOSITION 3.1. [1, §3.4]. *For* $0 < c < \frac{1}{2}$, *define the positive numbers* $t \in (\sqrt{2}, 2)$ *and* $\kappa_2 \in (0,1)$ *by*

$$t^2 := \frac{3+\sqrt{5-4c^2}}{2(1+c^2)} \quad \textit{and} \quad \kappa_2 := \sqrt{\frac{t+1}{t-1}}\left(\frac{1-\sqrt{1-c^2t^2}}{ct}\right). \tag{3.6}$$

Then, the asymptotic convergence factor $\kappa(\Lambda_c, P)$ (*cf.* (1.12)) *of any stationary two-step method* (3.5) *satisfies the inequality*

$$\kappa(\Lambda_c, P) \geq \kappa_2\,,$$

with equality holding if and only if the parameters $\{\mu_j\}_{j=0}^2$ *are chosen to be*

$$\mu_0 = 1+\kappa_2^2\,, \quad \mu_1 = 0, \quad \textit{and} \quad \mu_2 = -\kappa_2^2\,.$$

The quantity κ_2 of Proposition 3.1 also represents the *best* asymptotic convergence factor that can be obtained by applying a Chebyshev acceleration to (3.3), since these Chebyshev procedures are asymptotically *stationary* two-step methods (cf. Golub and Varga [7]).

But neither a Chebyshev method nor a stationary two-step method is asymptotically optimal with respect to the information $\sigma(T) \subseteq \Lambda_c$, where Λ_c is defined in (3.4). The asymptotic convergence factor of such a method, i.e., the quantity $\kappa(\Lambda_c)$ of (1.13), is given by

$$\kappa(\Lambda_c) = \frac{1-\cos(\pi c)}{\sin(\pi c)}\,. \tag{3.7}$$

This follows from (1.17), together with the fact that the exterior mapping function Φ of Λ_c is known in closed form (cf. [8, §5.7]). Knowing Φ, we can construct an asymptotically optimal nonstationary one-step method based, for example, on the Fejér nodes of Λ_c (cf. [6]).

Finally, we present a hybrid scheme that is nearly optimal with respect to the information $\sigma(T) \subseteq \Lambda_c$ $(0 < c < \frac{1}{2})$. Instead of $\mathbf{x} = T\mathbf{x}+\mathbf{c}$, we consider the equivalent linear system

$$\mathbf{x} = T^2\mathbf{x} + T\mathbf{c} + \mathbf{c}\,. \tag{3.8}$$

The eigenvalues of T^2 are contained in Λ_c^2, whose boundary is the cardioid

$$\partial\Lambda_c^2 = \{z \in \mathbb{C} : |z| = 2c^2(1+\cos(\arg z))\}\,. \tag{3.9}$$

Since Λ_c^2 can be easily enclosed ("embraced") by a circle, we solve (3.8) by the following stationary one-step method, often also called a *stationary first-order Richardson method*:

$$\mathbf{x}_m = \mu_0(T^2\mathbf{x}_{m-1} + T\mathbf{c} + \mathbf{c}) + (1-\mu_0)\mathbf{x}_{m-1} \qquad (m = 1, 2, \cdots) \tag{3.10}$$

($\mu_0 \in \mathbb{C}$, $\mu_0 \neq 0$). In general, as it certainly is not efficient to compute T^2 explicitly (especially if T is sparse), we divide (3.10) into two half-steps:

$$\begin{aligned} \mathbf{x}_{m-1/2} &= T\mathbf{x}_{m-1} + \mathbf{c}, \\ \mathbf{x}_m &= \mu_0(T\mathbf{x}_{m-1/2} + \mathbf{c}) + (1-\mu_0)\mathbf{x}_{m-1} \qquad (m = 1, 2, \cdots)\,. \end{aligned} \tag{3.11}$$

Before we answer the question as to which choice of μ_0 is optimal (with respect to the given information $\sigma(T^2) \subseteq \Lambda_c^2$), a remark concerning the asymptotic convergence factor of a hybrid method such as (3.10) or (3.11) should be made. If we apply a k-step method, as in (1.11) for instance, we must perform *one* matrix-vector multiplication by T in each iteration step. The hybrid method (3.11), however, requires *two* such matrix-vector multiplications per step. To compare both schemes fairly, we must compare κ_2 of Proposition 3.1 with $[\kappa(\Lambda_c^2, P(\mu_0))]^{1/2}$, where $P(\mu_0)$ denotes the generating matrix (cf. (1.13)) of the stationary first-order Richardson extrapolation with parameter μ_0. Our new result is found in Proposition 3.2.

PROPOSITION 3.2. *For $0 < c < \frac{1}{2}$, the effective convergence factor*

$$\kappa_h(\mu_0) := [\kappa(\Lambda_c^2, P(\mu_0))]^{1/2}$$

of any hybrid scheme of the form (3.11) *satisfies the inequality*

$$\kappa_h(\mu_0) \geq \frac{c}{1-c^2} \left[\frac{27}{4}(1-c^2)\right]^{1/4} =: \kappa_h(\mu_0^*), \tag{3.12}$$

with equality holding if and only if

$$\mu_0^* = \frac{2+c^2}{2-2c^2}.$$

Proof. Since the residual polynomials associated with a stationary first-order Richardson method are $p_m(z) = [\mu_0 z + 1 - \mu_0]^m$ $(m = 0, 1, \cdots)$, the convergence factor $\kappa(\Lambda_c^2, P(\mu_0))$ is given by

$$\kappa(\Lambda_c^2, P(\mu_0)) = \max_{z \in \Lambda_c^2} |\mu_0 z + 1 - \mu_0| = \max_{z \in \Lambda_c^2} \left| \frac{z-\zeta}{1-\zeta} \right|, \tag{3.13}$$

where $\zeta = \zeta(\mu_0) := 1 - 1/\mu_0$. Writing $\zeta = 2c^2 t$, observe that to minimize (3.13) as a function of ζ, we can confine our attention to those t that are contained in $[0, 2]$. For the function $r(t) := \max_{z \in \Lambda_c^2} |z - 2c^2 t|$, there holds

$$r^2(t) = \begin{cases} 4c^4(t-2)^2 & \text{for } 0 \leq t \leq \frac{2}{3}, \\ 8c^4 \frac{t^3}{2t-1} & \text{for } \frac{2}{3} \leq t \leq 2. \end{cases}$$

Furthermore, $\kappa^2(\Lambda_c^2, P(\mu_0)) = r^2(t)/(1-2c^2t)^2$ (with $\mu_0 = 1/(1-2c^2t)$) is monotonically decreasing for $t \in [0, \frac{2}{3}]$ and attains its minimum in $[\frac{2}{3}, 2]$ at $t_0 = 3/(4+2c^2)$. Therefore,

$$\mu_0^* = \frac{1}{1-2c^2 t_0} = \frac{2+c^2}{2-2c^2}$$

is the optimal extrapolation parameter, and substituting this into (3.13) gives the desired result of (3.12). □

The asymptotic convergence factors of the relaxation method (1.8) with ω of (3.2), of the stationary two-step (or Chebyshev) acceleration (3.5) described in Proposition 3.1, of the hybrid procedure (3.11) (cf. Proposition 3.2), and finally, of an asymptotically optimal method with respect to Λ_c (cf. (3.7)), are compared in Table 1.

TABLE 1

c	Convergence factor of equation (1.8)	(3.6)	(3.12)	(3.7)
0.2	0.4000	0.3420	0.3324	0.3249
0.4	0.8000	0.7451	0.7348	0.7265
0.45	0.9000	0.8661	0.8595	0.8541
0.495	0.9900	0.9859	0.9851	0.9844

The entries of the last two columns of the table below are seen to be more nearly equal as c increases to $\frac{1}{2}$. More precisely, on setting

$$c =: \tfrac{1}{2} - \varepsilon \qquad (0 < \varepsilon < \tfrac{1}{2} \text{ with } \varepsilon \text{ small}),$$

it can be verified from (3.12) that, as a function of ε,

$$\kappa_h(\mu_0^*) = 1 - 3\varepsilon + \frac{25}{6}\varepsilon^2 - \frac{437}{54}\varepsilon^3 + \mathcal{O}(\varepsilon^4) \qquad (\varepsilon \to 0),$$

whereas (3.7), as a function of ε, is

$$\kappa(\Lambda_{1/2-\varepsilon}) = 1 - \pi\varepsilon + \frac{\pi^2}{2}\varepsilon^2 - \frac{\pi^3}{3}\varepsilon^3 + \mathcal{O}(\varepsilon^4) \qquad (\varepsilon \to 0).$$

In terms of *rates of convergence*, we have that

$$\lim_{\varepsilon \to 0} \left\{ \frac{-\log \kappa(\Lambda_{1/2-\varepsilon})}{-\log \kappa_h(\mu_0^*)} \right\} = \frac{\pi}{3} = 1.0471\cdots.$$

Thus for ε small, the *loss* in the rate of convergence of the hybrid method (3.11) over that of the best rate of convergence, i.e., $-\log \kappa(\Lambda_{1/2-\varepsilon})$, is *less than* 5 percent. (In fact, from the numerical evaluation of the quantity in braces above, it appears that this loss in the rate of convergence never exceeds 5 percent for *any* c with $0 < c < \frac{1}{2}$.)

4. An example. The constant coefficient convection-diffusion equation

$$-\Delta u + \tau u_x = f \tag{4.1}$$

($\tau \geq 0$) on the unit square $(0,1) \times (0,1)$, with Dirichlet boundary conditions, is often used to construct test problems for iterative methods (cf., e.g., Chin and Manteuffel [1] or Elman and Golub [5]). The standard central difference discretization with mesh size $h = 1/(n+1)$ in both coordinate directions leads to a linear system whose coefficient matrix A has the block tridiagonal form

$$A = \text{tridiag}[-I, K, -I] \in \mathbf{R}^{n^2,n^2} \quad \text{with} \quad K = \text{tridiag}[-(1+R_x), 4, -(1-R_x)] \in \mathbf{R}^{n,n},$$

where $R_x := \tau h/2$ for the rowwise natural ordering of the mesh points. The corresponding block Jacobi matrix T has the eigenvalues

$$\lambda_{k,l} = \frac{\cos(\pi k h)}{2 - \sqrt{1 - R_x^2}\cos(\pi l h)} \qquad (k,l = 1,2,\cdots,n) \tag{4.2}$$

(cf. [5, Thm. 1]). For $R_x \leq 1$, the eigenvalues of T are therefore all real with

$$\lambda_{k,l} \in \left[-\frac{1}{2 - \sqrt{1 - R_x^2}}\,,\ \frac{1}{2 - \sqrt{1 - R_x^2}} \right] \qquad (k,l = 1,2,\cdots,n).$$

If, however, $R_x \geq 1$, then from (3.4),

$$\sigma(T) \subseteq \Lambda_{1/4}\,.$$

This inclusion is not sharp: Chin and Manteuffel showed that $\sigma(T)$ is contained in a certain bow-tie region (cf. [1, Fig. 2.2]) whose size depends on R_x. As R_x becomes larger, however, these bow-tie regions fill out $\Lambda_{1/4}$. (Besides that, we have ignored the factor $\cos(\pi h)$, which is not essential for our analysis.)

For $R_x > 1$, we now apply the hybrid scheme (3.11) to the block Jacobi method, which has the asymptotic convergence factor

$$\kappa_h(\mu_0) = 0.4229\cdots \quad \text{for the optimal } \mu_0 = 1.1 \quad (\text{when } c = \tfrac{1}{4})$$

(cf. Proposition 3.2). For the same problem, Chin and Manteuffel [1, (4.15)] found the spectral radius of the associated block SOR matrix $\mathcal{L}_\omega$ (with optimal relaxation parameter) which is $\rho(\mathcal{L}_\omega) = 0.1885\cdots$ if $R_x \geq 1.7177$. Clearly, the straightforward application of the hybrid procedure (3.11) is not competitive with the block SOR method for this model problem.

However, the block Jacobi matrix T in our example is weakly cyclic of index 2 (cf. [13, p. 39]), i.e., there exists an $n \times n$ permutation matrix Q such that

$$\tilde{T} = QTQ^* = \left[\begin{array}{c|c} 0 & T_1 \\ \hline T_2 & 0 \end{array}\right], \tag{4.3}$$

where the null submatrices on the diagonal are square. The transformed system $\mathbf{x} = \tilde{T}^2\mathbf{x} + \tilde{T}\mathbf{c} + \mathbf{c}$, which was the starting point of the hybrid scheme (3.11), then has the form

$$\left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \end{array}\right] = \left[\begin{array}{c|c} T_1T_2 & 0 \\ \hline 0 & T_2T_1 \end{array}\right] \left[\begin{array}{c} \mathbf{x}_1 \\ \mathbf{x}_2 \end{array}\right] + \left[\begin{array}{c} \tilde{\mathbf{c}}_1 \\ \tilde{\mathbf{c}}_2 \end{array}\right], \tag{4.4}$$

where the vectors $\mathbf{x}$ and $\tilde{\mathbf{c}} := \tilde{T}\mathbf{c} + \mathbf{c}$ are partitioned conformally with respect to the partitioning of (4.3). In view of (4.4), it is sufficient to solve the reduced problem

$$\mathbf{x}_2 = T_2T_1\mathbf{x}_2 + T_2\mathbf{c}_1 + \mathbf{c}_2$$

for the vector $\mathbf{x}_2$. Since $\sigma(T_2T_1) \setminus \{0\} = \sigma(T^2) \setminus \{0\}$, we have $\sigma(T_2T_1) \subseteq \Lambda_{1/4}^2$, and thus the optimal extrapolation parameter μ_0 of the *cyclically reduced hybrid scheme*

$$\begin{aligned} \mathbf{x}_{m-1/2} &= T_1\mathbf{x}_{m-1} + \mathbf{c}_1, \\ \mathbf{x}_m &= \mu_0(T_2\mathbf{x}_{m-1/2} + \mathbf{c}_2) + (1-\mu_0)\mathbf{x}_{m-1} \qquad (m = 1, 2, \cdots) \end{aligned} \tag{4.5}$$

is again given by Proposition 3.2. One step of the iterative method (4.5) requires *one* matrix-vector multiplication by each of the blocks T_1 and T_2. The effective asymptotic convergence factor of (4.5) (with optimal $\mu_0 = 1.1$) is therefore $[\kappa_h(\mu_0)]^2 = 0.1789\cdots$, indicating that (4.5) is marginally *faster* than the block SOR method, since $\rho(\mathcal{L}_\omega) = 0.1885\cdots$ for this latter method when $R_x \geq 1.7177$.

REFERENCES

[1] R. C. Y. Chin and T. A. Manteuffel, *An analysis of block successive overrelaxation for a class of matrices with complex spectra*, SIAM J. Numer. Anal., 25 (1988), pp. 564–585.

[2] P. CONCUS AND G. H. GOLUB, *A generalized conjugate gradient method for nonsymmetric systems of linear equations*, in Lecture Notes in Econom. and Math. Systems 134, R. Glowinski and J. R. Lions, eds., Springer-Verlag, Berlin, 1976, pp. 56–65.

[3] M. EIERMANN, X. LI, AND R. S. VARGA, *On hybrid semi-iterative methods*, SIAM J. Numer. Anal., 26 (1989), pp. 152–168.

[4] M. EIERMANN, W. NIETHAMMER, AND R. S. VARGA, *A study of semiiterative methods for nonsymmetric systems of linear equations*, Numer. Math., 47 (1985), pp. 505–533.

[5] H. C. ELMAN AND G. H. GOLUB, *Iterative methods for cyclically reduced non-self-adjoint linear systems*, Math. Comp., 54 (1990), pp. 671–700.

[6] B. FISCHER AND L. REICHEL, *A stable Richardson iterative method for complex linear systems*, Numer. Math., 54 (1988), pp. 225–242.

[7] G. H. GOLUB AND R. S. VARGA, *Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second order Richardson iterative methods, Part* I *and Part* II, Numer. Math., 3 (1961), pp. 147–168.

[8] H. KOBER, *Dictionary of Conformal Representations*, Dover, New York, 1952.

[9] T. A. MANTEUFFEL, *The Tchebychev iteration for nonsymmetric linear systems*, Numer. Math., 28 (1977), pp. 307–327.

[10] W. NIETHAMMER AND R. S. VARGA, *The analysis of k-step iterative methods for linear systems from summability theory*, Numer. Math., 41 (1983), pp. 177–206.

[11] ———, *Relaxation methods for non-Hermitian linear systems*, Results in Math., 16 (1989), pp. 308–320.

[12] R. S. VARGA, *A comparison of the successive overrelaxation method and semi-iterative methods using Chebyshev polynomials*, J. Soc. Indust. Appl. Math., 5 (1957), pp. 39–46.

[13] ———, *Matrix Iterative Analysis*, Prentice–Hall, Englewood Cliffs, NJ, 1962.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 993–1014, October 1992

GENERALIZATIONS OF THE SINGULAR VALUE AND QR DECOMPOSITIONS*

BART DE MOOR† AND PAUL VAN DOOREN‡

We dedicate this paper to Gene Golub, a true source of inspiration for our work, but also a genuine friend, on the occasion of his 60th birthday

Abstract. This paper discusses multimatrix generalizations of two well-known orthogonal rank factorizations of a matrix: the generalized singular value decomposition and the generalized QR- (or URV-) decomposition. These generalizations can be obtained for any number of matrices of compatible dimensions. This paper discusses in detail the structure of these generalizations and their mutual relations and gives a constructive proof for the generalized QR-decompositions.

Key words. singular value decomposition, QR-factorization, URV-decomposition, complete orthogonal decomposition

AMS(MOS) subject classifications. 15A09, 15A18, 15A21, 15A24, 65F20

1. Introduction.

In this paper, we present *multimatrix generalizations* of some well-known orthogonal rank factorizations. We show how the idea of a QR-decomposition (QRD), a URV-decomposition (URVD), and a singular value decomposition (SVD) for one matrix can be generalized to any number of matrices. While generalizations of the SVD for any number of matrices have been derived in [9], one of the main contributions of this paper is the constructive derivation of a generalization for the QRD (or URVD) for any number of matrices of compatible dimensions. The idea is to reduce the set of matrices $A_1, A_2, \ldots, A_k$ to a simpler form using unitary transformations only. Hereby, we avoid explicit products and inverses of the matrices that are involved. We show that these *generalized* QR-*decompositions* (GQRD) can be considered as a preliminary reduction for any *generalized singular value decomposition* (GSVD). The reason is that there is a certain one-to-one relation between the structure of a GQRD and the "corresponding" GSVD, which is explained in detail below.

This paper is organized as follows. In §2, we provide a summary of *orthogonal rank factorizations* for one matrix. We briefly review the SVD, the QRD, and the URVD as special cases. In §3, we give a survey of existing generalizations of the SVD and QRD for two or three matrices. In §4, we summarize the results on GSVDs for any number of matrices of compatible dimensions. Section 5, which contains the main new contribution of this paper, describes a generalization of the QRD and the URVD for any number of matrices. We derive a constructive, inductive proof which shows that a GQRD can be used as a preliminary reduction for a corresponding GSVD. In §6, we analyze in detail the structure of the GQRDs and GSVDs and show that there is a one-to-one relation between the two generalizations. This relation is elaborated in more detail in §7, where we illustrate how a GQRD can be used as a preliminary step in the derivation of a corresponding GSVD.

* Received by the editors April 16, 1991; accepted for publication (in revised form) October 18, 1991. This research was partially supported by the Belgian Program on Interuniversity Attraction Poles and the European Community Research Program ESPRIT, BRA 3280.

† Department of Electrical Engineering, Katholieke Universiteit Leuven, B-3001 Leuven, Belgium (demoor@esat.kuleuven.ac.be). This author is a Research Associate of the Belgian National Fund for Scientific Research (NFWO).

‡ Coordinated Science Laboratory, University of Illinois, Urbana, Illinois 61801 (vdooren@uicsl.csl.uiuc.edu).

While all results in this paper are stated for complex matrices, they can be specialized to the real case without much difficulty. This can be done in much the same way as with the SVD for complex and real matrices. In particular, it suffices to restate most results using the term *real orthonormal* instead of *unitary* and to replace a superscript "$*$" (which denotes the complex conjugate transpose of a matrix) by a superscript "t" (which is the transpose of a matrix).

2. Orthogonal rank factorizations. Any matrix $A \in C^{m\times n}$ can be factorized as

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \Pi, \tag{1}$$

where $R \in C^{n\times n}$ is upper trapezoidal and Π is a real $n \times n$ permutation matrix that permutes the columns of A so that the first $r_a = \operatorname{rank}(A)$ columns are linearly independent. The matrix $Q \in C^{m\times m}$ is unitary and can be partitioned as

$$\begin{matrix} & r_a & m - r_a \\ Q = & (Q_1 & Q_2 \quad). \end{matrix}$$

If we partition R accordingly as $R = (R_{11} \quad R_{12})$, where $R_{11} \in C^{r_a \times r_a}$ is upper triangular and nonsingular, we obtain

$$A = Q_{11}(R_{11} \quad R_{12})\Pi$$

which is sometimes called *the* QR-*factorization* of A.

If we rewrite (1) as

$$Q^* A = \begin{pmatrix} R \\ 0 \end{pmatrix} \Pi,$$

we see that Q is an orthogonal transformation that compresses the rows of A. Therefore, it is called a *row compression.* A similar construction exists, of course, for a *column compression.* A *complete orthogonal factorization* of an $m \times n$ matrix A is any factorization of the form

$$A = U \begin{pmatrix} T & 0 \\ 0 & 0 \end{pmatrix} V^*, \tag{2}$$

where T is $r_a \times r_a$ square nonsingular and $r_a = \operatorname{rank}(A)$. One particular case is the SVD, which has become an important tool in the analysis and numerical solution of numerous problems, especially since the development of numerically robust algorithms by Golub and his coworkers [15], [16], [17]. The SVD is a complete orthogonal factorization where the matrix T is diagonal with positive diagonal elements:

$$A = U\Sigma V^*.$$

Here $U \in C^{m\times m}$ and $V \in C^{n\times n}$ are unitary and $\Sigma \in \Re^{m\times n}$ is of the form [1]

$$\Sigma = \begin{pmatrix} \sigma_1 & 0 & \cdots & 0 & 0 \\ 0 & \sigma_2 & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & \cdots & \sigma_{r_a} & 0 \\ 0 & 0 & \cdots & 0 & 0 \end{pmatrix}.$$

[1] In this paper, we use the convention that zero blocks may be "empty" matrices, i.e., certain block dimensions may be 0.

The positive numbers $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_{r_a} > 0$ are called the *singular values* of A, while the columns of U and V are the *left* and *right singular vectors.*

In applications where $m \gg n$, it is often a good idea to use the QRD of the matrix as a preliminary step in the computation of its SVD. The SVD of A is obtained via the SVD of its triangular factor as

$$A = QR = Q(U_r \Sigma_r V_r^*) = (QU_r)\Sigma_r V_r^*.$$

This idea of combining the QRD and the SVD of the triangular matrix, in order to compute the SVD of the full matrix, is mentioned in [22, p. 119] and more fully analyzed in [3]. In [18] the method is referred to as R-bidiagonalization. Its flop count is (mn^2+n^3), as compared to $(2mn^2-2/3n^3)$ for a bidiagonalization of the full matrix. Hence, whenever $m \geq 5/3n$, it is more advantageous to use the R-bidiagonalization algorithm.

There exist still other complete orthogonal factorizations of the form (2) where only T is required to be triangular (upper or lower) (see, e.g., [18]). Such a factorization was called a URV-*decomposition* in [27]. Here

$$A = U \begin{pmatrix} R & 0 \\ 0 & 0 \end{pmatrix} V^*,$$

where $U \in C^{m\times m}$, $V \in C^{n\times n}$ are unitary matrices and $R \in C^{r_a \times r_a}$ is square nonsingular upper triangular.

It is well known that the QR-factorization of a singular matrix A and of its transpose A^* can be used for finding the image and kernel of A (URV-decompositions actually give both at once). In this paper, we try to extend these ideas to several matrices. Suppose we have a sequence of matrices $A_i, i = 1, \ldots, k$, and we want to know the kernels (or null spaces) of each partial product $A_1 \cdot A_2 \ldots A_j$. We could form these products and compute QR-decompositions of each of them. That can, in fact, be avoided, as shown below. Let us take the "special" example $A_i = A$, $i = 1, 2, 3$, with

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

It is well known that the null spaces of A^i in fact give the Jordan structure of A, and this structure is already obvious from the form of A. But let us reconstruct it from a sequence of QR-decompositions (in fact we need here RQ-decompositions of A). The first one is, of course, a column compression of A_1, for which we use the permutation of columns 2 and 4 (denoted by the matrix P_{24}):

$$A_1 P_{24} = \left[\begin{array}{cc|ccc} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

The separation line here indicates that the first two columns of P_{24} (i.e., e_1 and e_4) span the kernel of $A = A_1$. For the kernel of $A^2 = A_1 A_2$ we do not form this product,

but apply the inverse of the orthogonal transform P_{24} (which is again P_{24}) to the rows of $A_2 = A$:

$$P_{24}A_2 = \left[\begin{array}{ccccc} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

Since $A_1A_2 = (A_1P_{24})(P_{24}A_2)$, it is clear that the kernel of A_1A_2 is also the kernel of the bottom part of $P_{24}A_2$. The following column compression of $P_{24}A_2$ actually yields the kernel of both A_2 and the product A_1A_2. Perform indeed the orthogonal transformation $P_{24}P_{35}$:

$$P_{24}A_2P_{24}P_{35} = \left[\begin{array}{cc|cc|c} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right].$$

We see that the kernel of A_2 comprises the first two columns of $P_{24}P_{35}$ (i.e., e_1 and e_4 as before) and the kernel of A_1A_2 comprises the first four columns of $P_{24}P_{35}$, i.e., e_1, e_2, e_4, and e_5. An additional step of this procedure finally shows that the kernel of the product $A_1A_2A_3 = (A_1P_{24})(P_{24}A_2P_{24}P_{35})(P_{35}P_{24}A_3)$ is that of the bottom part of the matrix

$$P_{35}P_{24}A_3 = \left[\begin{array}{ccccc} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \end{array}\right],$$

which is a zero matrix. Hence the kernel of A^3 is the whole space, as expected. The interesting part of this simple example is the fact that we have not formed the intermediate products to get their corresponding kernels. The case treated here of equal matrices A_i is a simple one (and could be solved using the results of [19]), but in the next few sections we show how this can also be done for arbitrary sequences of matrices. The key idea is that at each step we do a number of QR-factorizations on the blocks of a partitioned matrix (column blocks in our case). This then induces a new partitioning on the rows of this matrix, on the columns of the next matrix, and so on.

3. Generalizations for two or three matrices. In the last decade or so, several generalizations for the SVD have been derived. The motivation is basically the necessity to avoid the explicit formation of products and matrix quotients in the computation of the SVD of products and quotients of matrices. Let A and B be nonsingular square matrices and assume that we need the SVD of $AB^{-*} = USV^*$.[2] It is well known that the explicit calculation of B^{-1}, followed by the computation of the product, may result in loss of numerical precision (digit cancellation), even before any factorization is attempted! The reason is the finite machine precision of

[2] The notation B^{-*} refers to the complex conjugate transpose of the inverse of the matrix B.

any calculator. Therefore, it seems more appropriate to come up with an implicit combined factorization of A and B separately, such as

$$\begin{aligned} A &= UD_1X^{-1}, \\ B &= X^{-*}D_2V^*, \end{aligned} \tag{3}$$

where U and V are unitary and X nonsingular. The matrices D_1 and D_2 are real but "sparse" (*quasi-diagonal*), and the product $D_1D_2^{-t}$ is diagonal with positive diagonal elements. Then we find

$$AB^{-*} = UD_1X^{-1}XD_2^{-t}V^* = U(D_1D_2^{-t})V^*.$$

A factorization as in (3) is always possible for two square nonsingular matrices. In fact, it is always possible for two matrices $A \in C^{m\times n}$ and $B \in C^{n\times p}$ (as long as the number of columns of A is the same as the number of rows of B, which we refer to as a *compatibility condition*). In general, the matrices A and B may even be rank deficient. The combined factorization (3) is called the *quotient singular value decomposition* (QSVD) and was first suggested in [32] and refined in [23] (it was originally called the generalized SVD, but we have suggested a standardized nomenclature in [6]).

A similar idea might be exploited for the SVD of the product of two matrices $AB = USV^*$, via the so-called *product singular value decomposition* (PSVD)

$$\begin{aligned} A &= UD_1X^{-1}, \\ B &= XD_2V^*, \end{aligned} \tag{4}$$

so that $AB = U(D_1D_2)V^*$, which is an SVD of AB. The combined factorization (4) was proposed in [13] as a formalization of ideas in [21]. In the general case, for two compatible matrices A and B (which may be rank deficient), the PSVD of (4) always exists and provides the SVD of AB without the explicit construction of the product. Similarly, if A and B are compatible, the QSVD always exists. However, it does not always deliver the SVD of $AB^\dagger$ when B is rank deficient ($B^\dagger$ is the pseudoinverse of B).

Another generalization, this time for three matrices, is the *restricted singular value decomposition* (RSVD). It was proposed in [35], and numerous applications were reviewed in [7]. It was soon found that all of these generalized SVDs for two or three matrices are special cases of a general theorem, presented in [9]. The main result is that there exist GSVDs for any number of matrices $A_1, A_2, \ldots, A_k$ of compatible dimensions. The general structure of these GSVDs was further analyzed in [10]. The dimensions of the blocks that occur in any GSVD can be expressed as ranks of the matrices involved and as certain products and concatenations of these. We present a summary of the results below.

As for generalizations of the QRD, it is mainly Paige [25] who pointed out the importance of generalized QRDs for two matrices as a basic conceptual and mathematical tool. The motivation is that in some applications, we need the QRD of a product of two matrices AB where $A \in \Re^{m\times n}$ and $B \in \Re^{n\times p}$. For general matrices A and B such a computation avoids forming the product explicitly, and transforms A and B separately to obtain the desired results. Paige [25] refers to such a factorization as a *product* QR *factorization.* Similarly, in some applications we need the QR-factorization of AB^{-1} where B is square and nonsingular. A general numerically robust algorithm would not compute the inverse of B nor the product explicitly, but would transform A and B separately. Paige [25] proposed calling such a combined

decomposition of two matrices a *generalized* QR *factorization*, following [20]. We propose here to reserve the name *generalized* QRD for the *complete* set of generalizations of the QR-decompositions, which are developed in this paper. We also propose a novel nomenclature, as we did for the generalizations of the SVD in [6].

Stoer [28] appears to be the first to have given a reliable computation of this type of generalized QR-factorization for two matrices (see [14]). Computational methods for producing the two types of generalized QR factorizations for two matrices, as described above, have appeared regularly in the literature as (intermediate) steps in the solution of some problems. In this paper, we derive a constructive proof of generalizations of the QRD for any number of matrices. As we see below, our generalized QRDs can also be considered the appropriate generalization of the URVD of a matrix.

4. Generalized singular value decompositions. In this section, we present a general theorem that can be considered the appropriate generalization for any number of matrices of the SVD of one matrix. It contains the existing generalizations of the SVD for two matrices (i.e., the PSVD and the QSVD) and three matrices (i.e., the RSVD) as special cases. A constructive proof can be found in [9].

THEOREM 4.1 (generalized singular value decompositions for k matrices). *Consider a set of k matrices with compatible dimensions: A_1 $(n_0 \times n_1), A_2$ $(n_1 \times n_2), \ldots,$ A_{k-1} $(n_{k-2} \times n_{k-1}), A_k$ $(n_{k-1} \times n_k)$. Then there exist*

—Unitary matrices U_1 $(n_0 \times n_0)$ and V_k $(n_k \times n_k)$.

—Matrices $D_j, j = 1, 2, \ldots, (k-1)$ of the form

$$
(5)\qquad \underset{n_{j-1} \times n_j}{D_j} = \begin{array}{l} \\ r_j^1 \\ r_{j-1}^1 - r_j^1 \\ r_j^2 \\ r_{j-1}^2 - r_j^2 \\ r_j^3 \\ \cdots \\ \cdots \\ r_j^j \\ n_{j-1} - r_{j-1} - r_j^j \end{array} \begin{array}{c} \begin{array}{ccccccc} r_j^1 & r_j^2 & r_j^3 & \cdots & \cdots & r_j^j & n_j - r_j \end{array} \\ \begin{pmatrix} I & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & I & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & I & \cdots & \cdots & 0 & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & \cdots & \cdots & I & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 \end{pmatrix} \end{array},
$$

where

$$
(6)\qquad r_0 = 0, \qquad r_j = \sum_{i=1}^{j} r_j^i = \operatorname{rank}(A_j).
$$

—A matrix S_k of the form

$$
(7)\qquad \underset{n_{k-1} \times n_k}{S_k} = \begin{array}{l} \\ r_k^1 \\ r_{k-1}^1 - r_k^1 \\ r_k^2 \\ r_{k-1}^2 - r_k^2 \\ r_k^3 \\ \cdots \\ \cdots \\ r_k^k \\ n_{k-1} - r_{k-1} - r_k^k \end{array} \begin{array}{c} \begin{array}{ccccccc} r_k^1 & r_k^2 & r_k^3 & \cdots & \cdots & r_k^k & n_k - r_k \end{array} \\ \begin{pmatrix} S_k^1 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & S_k^2 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & 0 & \cdots & \cdots & 0 & 0 \\ 0 & 0 & S_k^3 & \cdots & \cdots & 0 & 0 \\ 0 & \cdots & \cdots & \cdots & \cdots & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & \cdots & \cdots & \cdots & \cdots & S_k^k & 0 \\ 0 & 0 & \cdots & \cdots & \cdots & 0 & 0 \end{pmatrix} \end{array},
$$

where

$$r_k = \sum_{i=1}^{k} r_k^i = \operatorname{rank}(A_k) \tag{8}$$

and the $r_k^i \times r_k^i$ matrices S_k^i are diagonal with positive diagonal elements. Expressions for the integers r_j^i are given in §6.[3]

—*Nonsingular matrices X_j $(n_j \times n_j)$ and Z_j, $j = 1, 2, \ldots, (k-1)$ where Z_j is either $Z_j = X_j^{-*}$ or $Z_j = X_j$ (i.e., both choices are always possible), such that the given matrices can be factorized as*

$$\begin{aligned}
A_1 &= U_1 D_1 X_1^{-1}, \\
A_2 &= Z_1 D_2 X_2^{-1}, \\
A_3 &= Z_2 D_3 X_3^{-1}, \\
&\cdots = \cdots, \\
A_i &= Z_{i-1} D_i X_i^{-1}, \\
&\cdots = \cdots, \\
A_k &= Z_{k-1} S_k V_k^*.
\end{aligned}$$

Observe that the matrices D_j in (5) and S_k in (7) are generally not diagonal. Their only nonzero blocks, however, are diagonal block matrices. We propose to label them as *quasi-diagonal* matrices. The matrices $D_j, j = 1, \ldots, k-1$ are quasi-diagonal, their only nonzero blocks being identity matrices. The matrix S_k is quasi-diagonal and its nonzero blocks are diagonal matrices with positive diagonal elements. Observe that we always take the last factor in every factorization as the inverse of a nonsingular matrix, which is only a matter of convention (another convention would result in a modified definition of the matrices Z_i). As for the name of a certain GSVD, we propose to adopt the following convention (see also [9]).

DEFINITION 4.2 (the nomenclature for GSVDs). If $k = 1$ in Theorem 4.1, then the corresponding factorization of the matrix A_1 will be called the (ordinary) singular value decomposition. If for a matrix pair $A_i, A_{i+1}, 1 \le i \le k-1$ in Theorem 4.1, we have $Z_i = X_i$, then the factorization of the pair is said to be of P type. If, on the other hand, for a matrix pair $A_i, A_{i+1}, 1 \le i \le k-1$ in Theorem 4.1, we have $Z_i = X_i^{-*}$, then the factorization of the pair is said to be of Q type. The name of a GSVD of the matrices A_i, $i = 1, 2, \ldots k > 1$ as in Theorem 4.1, is then obtained by simply enumerating the different factorization types.

Let us give some examples.

Example. Consider two matrices A_1 $(n_0 \times n_1)$ and A_2 $(n_1 \times n_2)$. Then, we have two possible GSVDs:

	P type	Q type
A_1	$U_1 D_1 X_1^{-1}$	$U_1 D_1 X_1^{-1}$
A_2	$X_1 S_2 V_2^*$	$X_1^{-*} S_2 V_2^*$

The P-type factorization is called the PSVD (see [8] and references therein), while the Q-type factorization is called the QSVD.

[3] In [9], these block dimensions follow from the constructive proof.

Example. Let us write a PQQP-SVD for five matrices:

$$\begin{aligned} A_1 &= U_1 D_1 X_1^{-1}, \\ A_2 &= X_1 D_2 X_2^{-1}, \\ A_3 &= X_2^{-*} D_3 X_3^{-1}, \\ A_4 &= X_3^{-*} D_4 X_4^{-1}, \\ A_5 &= X_4 S_5 V_5^*. \end{aligned}$$

We also introduce a notation using powers that symbolize a certain repetition of a letter or of a sequence of letters:

- $\mathrm{P}^3\mathrm{Q}^2$-SVD = PPPQQ-SVD,
- $(\mathrm{PQ})^2\mathrm{Q}^3(\mathrm{PPQ})^2$-SVD = PQPQQQQPPQPPQ-SVD.

Despite the fact that there are 2^{k-1} different sequences of letters P and Q at level $k > 1$, not all of these sequences correspond to different GSVDs. The reason for this is that, for instance, the QP-SVD of (A^1, A^2, A^3) can be obtained from the PQ-SVD of $((A^3)^*, (A^2)^*, (A^1)^*)$. Similarly, the $\mathrm{P}^2(\mathrm{QP})^3$-SVD of $(A^1, \ldots, A^9)$ is essentially the same as the $(\mathrm{PQ})^3\mathrm{P}^2$-SVD of $((A^9)^*, \ldots, (A^1)^*)$. The number of different factorizations for k matrices is, in fact, $\frac{1}{2}(2^{k-1} + 2^{k/2})$ for k even and $\frac{1}{2}(2^{k-1} + 2^{(k-1)/2})$ for k odd.

A possible way to visualize Theorem 4.1 is to build a tree with all different factorizations for 1, 2, 3, etc $\cdots$ matrices as follows:

1					**O**			
2				**P**		**Q**		
3			**P²**		**PQ**		**Q²**	
4	**P³**	**P²Q**		**PQP**		**PQ²**	**QPQ**	**Q³**
...	...	...	...	...	...	...	...	...

5. Generalized URVDs. In this section, we derive a generalization for several matrices, of the URVD of one matrix. We proceed in several stages. First, we show how k matrices can be reduced to block triangular matrices using unitary transformations only. Next, we show how the block triangular factors can be triangularized further to triangular factors.

THEOREM 5.1. *Given k complex matrices A_1 $(n_0 \times n_1)$, A_2 $(n_1 \times n_2)$, $\ldots$, A_k $(n_{k-1} \times n_k)$, there always exist unitary matrices $Q_0, Q_1, \ldots, Q_k$ such that*

$$T_i = Q_{i-1}^* A_i Q_i,$$

where T_i is a block lower triangular or block upper triangular matrix (both cases are always possible) with the following structures:

—Lower block triangular (denoted by a superscript l) :

$$T_i^l = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{cccccc} r_i^1 & r_i^2 & \ldots & r_i^{i-1} & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} T_{i,1} & 0 & \ldots & 0 & 0 & 0 \\ * & T_{i,2} & \ldots & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & \ldots & * & T_{i,i} & 0 \end{pmatrix} \end{array}. \tag{9}$$

—*Upper block triangular* (*denoted by a superscript* u):

$$T_i^u = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{cccccc} r_i^1 & r_i^2 & \dots & r_i^{i-1} & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} T_{i,1} & * & \dots & * & * & 0 \\ 0 & T_{i,2} & \dots & * & * & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \dots & 0 & T_{i,i} & 0 \end{pmatrix} \end{array}, \tag{10}$$

where $T_{i,j}$, $j = 1, \dots, i$ *are full column rank matrices and each* $*$ *represents a nonzero block. The block dimensions coincide with those of Theorem* 4.1. *In particular,*

$$\begin{aligned} r_0^1 &= n_0, \\ r_i^{i+1} &= \text{nullity}(A_i) = n_i - r_i, \end{aligned}$$

and

$$\begin{aligned} \sum_{j=1}^{i} r_i^j &= \text{rank}(A_i) = r_i, \\ \sum_{j=1}^{i} r_{i-1}^j &= n_{i-1}. \end{aligned}$$

Our proof of Theorem 5.1 is inductive: We obtain the required factorization of A_i from that of A_{i-1}.

Proof. The induction is initialized for $i = 1$ as follows. First, take the case where T_1 is to be lower block triangular. Use a unitary column compression matrix Q_1 to reduce the matrix A_1 to

$$T_1^l = A_1 Q_1 = \begin{array}{c} \\ r_0^1 \end{array} \begin{array}{c} \begin{array}{cc} r_1^1 & r_1^2 \end{array} \\ \begin{pmatrix} T_{1,1} & 0 \end{pmatrix} \end{array},$$

where

$$\begin{aligned} r_1^1 &= \text{rank}(T_{1,1}) = \text{rank}(A_1), \\ r_1^2 &= \text{nullity}(A_1) = n_1 - r_1, \end{aligned}$$

and

$$r_0^1 = n_0.$$

The case where T_1 is required to be upper block triangular is similar:

$$T_1^u = A_1 Q_1 = \begin{array}{c} \\ r_0^1 \end{array} \begin{array}{c} \begin{array}{cc} r_1^1 & r_1^2 \end{array} \\ \begin{pmatrix} T_{1,1} & 0 \end{pmatrix} \end{array}.$$

Observe that we have taken $Q_0 = I_{n_0}$.

Now, we can start our induction. Assume that we have the required factorization for the first $i - 1$ matrices

$$\begin{aligned} T_1 &= Q_0^* A_1 Q_1, \\ &\vdots = \vdots, \\ T_{i-1} &= Q_{i-2}^* A_{i-1} Q_{i-1}, \end{aligned}$$

where the matrices $T_j, j = 1, \ldots, i-1$ have the block structure as in Theorem 5.1. We now want to find a unitary matrix Q_i such that $T_i = Q_{i-1}^* A_i Q_i$ is either lower or upper block triangular. First, consider the case where T_i is to be lower block triangular. The matrix $Q_{i-1}^* A_i$ can be partitioned according to the dimensions of the block columns of T_{i-1} as

$$Q_{i-1}^* A_i = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} n_i \\ \begin{pmatrix} * \\ * \\ \vdots \\ * \end{pmatrix} \end{array}. \tag{11}$$

It is always possible to construct a unitary matrix Q_i to compress the columns of each of the block rows to the left as

$$T_i^l = Q_{i-1}^* A_i Q_i = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{ccccc} r_i^1 & r_i^2 & \ldots & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} T_{i,1} & 0 & \ldots & 0 & 0 \\ * & T_{i,2} & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & \ldots & T_{i,i} & 0 \end{pmatrix} \end{array}, \tag{12}$$

where the subblocks $T_{i,j}$ are of full column rank, denoted by r_i^l, $l = 1, \ldots, i$ and $r_i^{i+1} = \text{nullity}(A_i)$. Hereto, we first compress the first block row of (11) to the left with unitary column transformations applied to the full matrix. Then we proceed with the second block row in the deflated matrix (i.e., without modifying the previous block column). By repeating this procedure i times, we find the required form (12).

Obviously,

$$r_i^l \leq r_{i-1}^l, \qquad l = 1, \ldots, i. \tag{13}$$

The construction of T_i when it is required to be upper block triangular is similar. Construct a unitary matrix Q_i that compresses the columns of the block rows of $Q_{i-1}^* A_i$ to the right. The only difference is that we now start from the bottom to find that

$$T_i^u = Q_{i-1}^* A_i Q_i = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{ccccc} r_i^{i+1} & r_i^1 & r_i^2 & \ldots & r_i^i \end{array} \\ \begin{pmatrix} 0 & T_{i,1} & * & \ldots & * \\ 0 & 0 & T_{i,2} & \ldots & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & \ldots & 0 & T_{i,i} \end{pmatrix} \end{array}. \tag{14}$$

We can now apply an additional (block) column permutation to the right of the matrix T_i^u so as to find the matrix of (10). This completes the proof. □

We now demonstrate that the matrices $T_{i,j}$ can always be further reduced to triangular form using unitary transformations into

$$\begin{pmatrix} 0 \\ R_{i,j}^l \end{pmatrix}$$

in the case when T_i is lower block triangular. Here, $R_{i,j}^l$ is a lower triangular matrix. Similarly, we can always reduce $T_{i,j}$ to

$$\begin{pmatrix} R_{i,j}^u \\ 0 \end{pmatrix}$$

in the case where T_i is upper block triangular. Here $R_{i,j}^u$ is an upper triangular matrix. In order to demonstrate this, we need the following result.

LEMMA 5.2. *Let $P_1, \ldots, P_k$ be k given complex matrices where P_i has dimensions $p_{i-1} \times p_i$, $p_{i-1} \geq p_i$ and* $\operatorname{rank}(P_i) = p_i$. *Then there always exist unitary matrices $Q_0, Q_1, \ldots, Q_k$ such that*

$$R_i = Q_{i-1}^* P_i Q_i,$$

where R_i is either of the form

$$R_i = \begin{matrix} \\ p_{i-1} - p_i \\ p_i \end{matrix} \begin{matrix} p_i \\ \begin{pmatrix} 0 \\ R_i^l \end{pmatrix} \end{matrix} \tag{15}$$

with R_i^l a lower triangular matrix, or

$$R_i = \begin{matrix} \\ p_i \\ p_{i-1} - p_i \end{matrix} \begin{matrix} p_i \\ \begin{pmatrix} R_i^u \\ 0 \end{pmatrix} \end{matrix} \tag{16}$$

with R_i^u upper triangular. For every $i = 1, \ldots, k$, both choices, (15) *and* (16), *are always possible.*

Proof. Again, the proof is by induction, but now for *decreasing* index i. For the initialization, start with $i = k$ and obtain a QR-decomposition of P_k with either an upper or a lower triangular factor as required. This defines the unitary matrix Q_{k-1}. We take $Q_k = I_{p_k}$. Hence, we find

—Lower triangular:

$$P_k = Q_{k-1} R_k = Q_{k-1} \begin{pmatrix} 0 \\ R_k^l \end{pmatrix};$$

—Upper triangular:

$$P_k = Q_{k-1} R_k = Q_{k-1} \begin{pmatrix} R_k^u \\ 0 \end{pmatrix}.$$

We can now start the induction for $i = k-1, k-2, \ldots, 1$. Therefore, assume that we have the required factorizations for the matrices $P_k, P_{k-1}, \ldots, P_{i+1}$:

$$\begin{aligned} R_k &= Q_{k-1}^* P_k Q_k, \\ R_{k-1} &= Q_{k-2}^* P_{k-1} Q_{k-1}, \\ &\vdots = \vdots, \\ R_{i+1} &= Q_i^* P_{i+1} Q_{i+1}. \end{aligned}$$

Then, if R_i is to be lower triangular, obtain a QR-decomposition of the product P_iQ_i as

$$P_iQ_i = Q_{i-1}R_i = Q_{i-1}\begin{pmatrix} 0 \\ R_i^l \end{pmatrix},$$

so that

$$R_i = \begin{pmatrix} 0 \\ R_i^l \end{pmatrix} = Q_{i-1}^* P_i Q_i.$$

If R_i is required to be upper triangular, obtain a QR-decomposition as

$$P_iQ_i = Q_{i-1}R_i = Q_{i-1}\begin{pmatrix} R_i^u \\ 0 \end{pmatrix},$$

so that

$$R_i = \begin{pmatrix} R_i^u \\ 0 \end{pmatrix} = Q_{i-1}^* P_i Q_i.$$

This completes the construction. □

We now repeatedly apply Lemma 5.2 on the full rank blocks in the matrices T_i in (9) and (10). First, we apply Lemma 5.2 to the sequence of k subblocks

$$T_{1,1} \;\; T_{2,1} \;\; \ldots \;\; T_{k,1}.$$

Next, we apply it to the sequence of the $k-1$ subblocks

$$T_{2,2} \;\; T_{3,2} \;\; \ldots \;\; T_{k,2}.$$

In general, we apply Lemma 5.2 k times to the k sequences of subblocks

$$T_{j,j}, T_{j+1,j}, \ldots, T_{k,j} \quad \text{for } j = 1, \ldots, k.$$

In applying Lemma 5.2 to the jth of these sequences, we can find a sequence of unitary matrices $Q_0^{[j]}, Q_1^{[j]}, \ldots, Q_{k-j+1}^{[j]}$ and matrices $R_{i,j}$ such that

$$T_{i,j} = Q_{i-j}^{[j]} R_{i,j} Q_{i-j+1}^{[j]*}, \qquad i = j, \ldots, k,$$

where

$$R_{i,j} = \begin{pmatrix} 0 \\ R_{i,j}^l \end{pmatrix}$$

or

$$R_{i,j} = \begin{pmatrix} R_{i,j}^u \\ 0 \end{pmatrix}.$$

We now define the *unitary* matrices $\tilde{Q}_i$ for $i = 0, \ldots, k$, which are block diagonal with blocks

$$\tilde{Q}_i = \operatorname{diag}(Q_i^{[1]}, Q_{i-1}^{[2]}, \ldots, Q_1^{[i]}, Q_0^{[i+1]}), \qquad i = 0, \ldots, k,$$

with

$$Q_0^{[k+1]} = I.$$

Next we define

$$\tilde{T}_i = \tilde{Q}_{i-1}^* T_i \tilde{Q}_i, \qquad i = 0, \ldots, k.$$

Then, it can be verified that for the lower triangular case we obtain

$$\tilde{T}_i = \tilde{T}_i^l = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{ccccc} r_i^1 & r_i^2 & \ldots & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} R_{i,1} & 0 & \ldots & 0 & 0 \\ * & R_{i,2} & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & \ldots & R_{i,i} & 0 \end{pmatrix} \end{array}, \tag{17}$$

and for the upper triangular case we find that

$$\tilde{T}_i = \tilde{T}_i^u = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{ccccc} r_i^1 & \ldots & r_i^{i-1} & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} R_{i,1} & * & \ldots & * & 0 \\ 0 & R_{i,2} & \ldots & * & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \ldots & 0 & R_{i,i} & 0 \end{pmatrix} \end{array}. \tag{18}$$

If we now combine (9)–(17) and (10)–(18), we obtain a combined factorization of the form

$$\tilde{T}_i = (Q_{i-1}\tilde{Q}_{i-1})^* A_i (Q_i \tilde{Q}_i).$$

Hence, we have proved the following theorem.

THEOREM 5.3 (generalized URVDs). *Given k complex matrices A_1 $(n_0 \times n_1)$, A_2 $(n_1 \times n_2)$, $\ldots$, A_k $(n_{k-1} \times n_k)$, there always exist unitary matrices $Q_0, Q_1, \ldots, Q_k$ such that*

$$\tilde{T}_i = Q_{i-1}^* A_i Q_i,$$

where $\tilde{T}_i$ is a lower triangular or upper triangular matrix (both cases are always possible) with the following structures:

—Lower triangular (denoted by a superscript l):

$$\tilde{T}_i^l = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{ccccc} r_i^1 & r_i^2 & \ldots & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} R_{i,1} & 0 & \ldots & 0 & 0 \\ * & R_{i,2} & \ldots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & \ldots & R_{i,i} & 0 \end{pmatrix} \end{array},$$

where

$$R_{i,j} = \begin{pmatrix} 0 \\ R_{i,j}^l \end{pmatrix},$$

and $R^l_{i,j}$ is a square nonsingular lower triangular matrix.

—Upper triangular (denoted by a superscript u):

$$\tilde{T}^u_i = \begin{array}{c} \\ r^1_{i-1} \\ r^2_{i-1} \\ \vdots \\ r^i_{i-1} \end{array} \begin{array}{c} \begin{array}{ccccc} r^1_i & \dots & r^{i-1}_i & r^i_i & r^{i+1}_i \end{array} \\ \begin{pmatrix} R_{i,1} & * & \dots & * & 0 \\ 0 & R_{i,2} & \dots & * & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 0 & R_{i,i} & 0 \end{pmatrix} \end{array},$$

where

$$R_{i,j} = \begin{pmatrix} 0 \\ R^u_{i,j} \end{pmatrix},$$

and $R^u_{i,j}$ is a square nonsingular upper triangular matrix. The block dimensions coincide with those of Theorem 4.1.

As for the nomenclature of these generalized URVDs, we propose the following definition.

DEFINITION 5.4 (nomenclature for generalized URV). The name of a generalized URVD of k matrices of compatible dimensions is generated by enumerating the letters L (for lower) and U (for upper), according to the lower or upper triangularity of the matrices $T_i, i = 1, \dots, k$ in the decomposition of Theorem 5.3.

For k matrices, there are 2^k different sequences with two letters. For instance, for $k = 3$, there are eight generalized URVD (LLL, LLU, LUL, LUU, ULL, ULU, UUL, UUU).

Remarks. The decompositions in Theorems 5.1 and 5.3 both use column and row compressions of a matrix as a cornerstone for the rank determination of the individual blocks. As already pointed out in §2, the rank determination can be done via an ordinary SVD (OSVD), but a more economical method uses the QRD as initial step, since typically the matrices involved here have many more columns than rows or vice versa. A further alternative would be to replace the OSVD of the triangular matrix resulting from the initial QRD by a rank-revealing QRD. Since the time of the initial paper drawing attention to this [5], much progress has been made in this area, and we only want to stress here that such alternatives can only benefit our decomposition.

The overall complexity of this GQRD is easily seen to be comparable to that of performing two QRDs of each matrix A_i involved. For each A_i we indeed apply the left transformation Q^*_{i-1} derived from the previous matrix and then apply a "special" compression Q_i of the resulting matrix while respecting its block structure. Both steps have a complexity comparable to a QRD of a matrix of the same dimensions. For parallel machines we can check that the "block" algorithms [18] for one-sided orthogonal transformations such as the QRD can also be applied to the present decomposition, and that they will yield satisfactory speedups. The main reason for this is that the two-sided orthogonal transforms applied to each A_i are done separately, and hence they can essentially be considered one-sided for parallellization purposes.

6. On the structure of the GSVD and the GQRD. In this section, we first point out how for each GSVD there are two generalized URVDs, and we clarify the correspondence between the two types of generalized decompositions. Next, we give a summary of expressions for the block dimensions r^j_i in Theorems 4.1 and 5.1,

in terms of the ranks of the matrices $A_1, \ldots, A_k$ and concatenations and products thereof. These expressions were derived in [10].

Recall the nomenclature for the generalized URVDs (Definition 5.4) and the GSVDs (Definition 4.2). The relationship between these two definitions is as follows. A pair of identical letters, i.e., L-L or U-U, that occurs in the factorization of A_i, A_{i+1} corresponds to a P-type factorization of the pair. A pair of alternating letters, i.e., L-U or U-L, that occurs in the factorization of A_i, A_{i+1} corresponds to a Q-type factorization of the pair. As an example, for a PQP-SVD of four matrices, there are two possible corresponding generalized URVDs, namely an LLUL-decomposition and a UULU-decomposition. As with the GSVD, we can also introduce the convention to use powers of (a sequence of) letters. For instance, for a P^3Q^2-SVD, there are two GURVs, namely, an L^4UL-URV and a U^4LU-URV.

We now derive expressions for the block dimensions r_i^j.[4] Let us first consider the case of a GSVD that consists only of P-type factorizations. Denote the rank of the product of the matrices $A_i, A_{i+1}, \ldots, A_j$ with $i \leq j$ by

$$r_{i(i+1)\cdots(j-1)j} = \operatorname{rank}(A_i A_{i+1} \ldots A_{j-1} A_j).$$

THEOREM 6.1 (on the structure of P^{k-1}-SVD, L^k-URV, and U^k-URV). *Consider any of the factorizations above for the matrices $A_1, A_2, \ldots, A_k$. Then, the block dimensions r_j^i that appear in Theorems* 4.1, 5.1, *and* 5.3 *are given by:*

$$r_j^1 = r_{(1)(2)\cdots(j)}, \tag{19}$$

$$r_j^i = r_{i(i+1)\cdots(j)} - r_{(i-1)(i)\cdots(j)}, \tag{20}$$

with $r_j^i = r_i$ if $i = j$.

Next, consider the case of a GSVD that only consists of Q-type factorizations. Denote the rank of the block bidiagonal matrix

$$\begin{pmatrix} A_i & 0 & 0 & \ldots & 0 & 0 & 0 \\ A_{i+1}^* & A_{i+2} & 0 & \ldots & 0 & 0 & 0 \\ 0 & A_{i+3}^* & A_{i+4} & \ldots & 0 & 0 & 0 \\ 0 & \ldots & \ldots & \ldots & \ldots & \ldots & 0 \\ 0 & \ldots & \ldots & \ldots & A_{j-3}^* & A_{j-2} & 0 \\ 0 & \ldots & \ldots & \ldots & 0 & A_{j-1}^* & A_j \end{pmatrix} \tag{21}$$

(by $r_{i|i+1|\ldots|j-1|j}$).

THEOREM 6.2 (on the structure of Q^{k-1}-SVD, $(LU)^{k/2}$-URV (k even), $(UL)^{k/2}$-URV (k even), $(UL)^{(k-1)/2}U$-URV (k odd), and $(LU)^{(k-1)/2}L$-URV (k odd)). *Consider any of the above factorizations for the matrices $A_1, A_2, \ldots, A_k$. Then,*

- *If $j - i$ is even,*

$$r_{i|\cdots|j} = r_{i|\cdots|j-1} + (r_j^1 + r_j^2 + \cdots + r_j^i) + r_j^{i+2} + r_j^{i+4} + \cdots + r_j^{j-2} + r_j^j;$$

- *If $j - i$ is odd,*

$$r_{i|\cdots|j} = r_{i|\cdots|j-1} + (r_j^{i+1} + r_j^{i+3} + \cdots + r_j^{j-2} + r_j^j).$$

[4] Recall that the subscript i refers to the ith matrix, while the superscript j refers to the jth block in that matrix.

For the general case, we need a mixture of the two preceding notations for block bidiagonal matrices, the blocks of which can be products of matrices, such as

$$\begin{pmatrix} A_{i_0}A_{i_0+1}\dots A_{i_1-1} & 0 & 0 & \dots & 0 \\ (A_{i_1}\dots A_{i_2-1})^* & A_{i_2}\dots A_{i_3-1} & 0 & \dots & 0 \\ 0 & (A_{i_3}\dots A_{i_4-1})^* & A_{i_4}\dots A_{i_5-1} & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & \dots & \dots & \dots & A_{i_l}\dots A_j \end{pmatrix},$$

where $1 \le i_0 < i_1 < i_2 < i_3 < \cdots < i_l \le j \le k$. Their rank is denoted by

$$r_{(i_0)\cdots(i_1-1)|i_1\cdots(i_2-1)|\cdots|i_l\cdots(j)}.$$

For instance, the rank of the matrix

$$\begin{pmatrix} A_2A_3 & 0 & 0 \\ A_4^* & A_5A_6A_7 & 0 \\ 0 & (A_8A_9)^* & A_{10} \end{pmatrix}$$

is represented by $r_{(2)(3)|4|(5)(6)(7)|(8)(9)|(10)}$.

THEOREM 6.3 (on the structure of a GSVD and a GURV). *The rank* $r_{(i_0)(i_0+1)\dots(i_1-1)|i_1\dots(i_2-1)|\dots|i_l\dots j}$ *can be derived as follows:*

1. *Calculate the following* $l+1$ *integers* s_j^i, $i = 1, 2, \dots, l+1$:

$$\begin{aligned} s_j^1 &= r_j^1 + r_j^2 + \cdots + r_j^{i_0}, \\ s_j^2 &= r_j^{i_0+1} + r_j^{i_0+2} + \cdots + r_j^{i_1}, \\ \cdots &= \cdots, \\ s_j^{l+1} &= r_j^{i_{l-1}+1} + r_j^{i_{l-1}+2} + \cdots + r_j^{i_l}. \end{aligned}$$

2. *Depending on* l *even or odd there are two cases:*

—l *even*:

$$r_{i_0\cdots i_1-1|i_1\cdots i_2-1|\cdots|i_l\cdots j} = r_{i_0\cdots i_1-1|i_1\cdots i_2-1|\cdots|i_{l-1}\cdots i_l-1} + s_j^1 + s_j^3 + \cdots + s_j^{l+1};$$

—l *odd*:

$$r_{i_0\cdots i_1-1|i_1\cdots i_2-1|\cdots|i_l\cdots j} = r_{i_0\cdots i_1-1|i_1\cdots i_2-1|\cdots|i_{l-1}\cdots i_l-1} + s_j^2 + s_j^4 + \cdots + s_j^{l+1}.$$

Observe that Theorems 6.1 and 6.2 are special cases of Theorem 6.3. While Theorem 6.1 provides a direct expression of the dimensions r_j^i in terms of differences of ranks of products, Theorems 6.2 and 6.3 do so only implicitly. This is illustrated in the following examples.

Example. Let us determine the block dimensions of the quasi-diagonal matrix S_4 in a QPP-SVD of the matrices A_1, A_2, A_3, A_4 (which are also the block dimensions of an LUUU- or a ULLL-decomposition). From Theorem 6.2 we find that

$$\begin{aligned} r_4^4 &= r_4 - r_{34}, \\ r_3^4 &= r_{34} - r_{234}. \end{aligned}$$

From Theorem 6.3, we find that

$$s_4^1 = r_4^1, \qquad s_4^2 = r_4^2,$$

and

$$r_{(1)|(2)(3)(4)} = r_1 + s_4^2,$$

so that

$$r_4^2 = r_{1|(2)(3)(4)} - r_1.$$

Finally, since $r_4 = r_4^1 + r_4^2 + r_4^3 + r_4^4$, we find that

$$r_4^1 = r_1 + r_{(2)(3)(4)} - r_{1|(2)(3)(4)}.$$

Observe that this last relation can be interpreted geometrically as the dimension of the intersection between the row spaces of A_1 and $A_2A_3A_4$:

$$r_4^1 = \dim \operatorname{span}_{\text{row}}(A_1) + \dim \operatorname{span}_{\text{row}}(A_2A_3A_4) - \dim \operatorname{span}_{\text{row}} \begin{pmatrix} A_1 \\ A_2A_3A_3 \end{pmatrix}.$$

Example. Consider the determination of $r_5^1, r_5^2, r_5^3, r_5^4, r_5^5$ in a PQ^3-SVD of five matrices A_1, A_2, A_3, A_4, A_5 with Theorem 6.3, which coincides with the structure of a UULUL-URV or an LLULU-URV (see Table 1).

TABLE 1

	s_5^i
$r_{4\|5}$	$s_5^1 = r_5^1 + r_5^2 + r_5^3 + r_5^4$ $s_5^2 = r_5^5$
$r_{3\|4\|5}$	$s_5^1 = r_5^1 + r_5^2 + r_5^3$ $s_5^2 = r_5^4$ $s_5^3 = r_5^5$
$r_{2\|3\|4\|5}$	$s_5^1 = r_5^1 + r_5^2$ $s_5^2 = r_5^3$ $s_5^3 = r_5^4$ $s_5^4 = r_5^5$
$r_{(1)(2)\|3\|4\|5}$	$s_5^1 = r_5^1$ $s_5^2 = r_5^2 + r_5^3$ $s_5^3 = r_5^4$ $s_5^4 = r_5^5$

These relations can be used to set up a set of equations for the unknowns r_5^1, r_5^2, r_5^3, r_5^4, r_5^5, using Theorem 6.3 as

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} r_5^1 \\ r_5^2 \\ r_5^3 \\ r_5^4 \\ r_5^5 \end{pmatrix} = \begin{pmatrix} r_5 \\ r_{4|5} - r_4 \\ r_{3|4|5} - r_{3|4} \\ r_{2|3|4|5} - r_{2|3|4} \\ r_{(1)(2)|3|4|5} - r_{(1)(2)|3|4} \end{pmatrix},$$

the solution of which is

$$r_5^1 = r_{3|4|5} - r_{3|4} + r_{(1)(2)|3|4} - r_{(1)(2)|3|4|5},$$

$$r_5^2 = r_{(1)(2)|3|4|5} - r_{(1)(2)|3|4} - r_{2|3|4|5} + r_{2|3|4},$$
$$r_5^3 = r_{2|3|4|5} - r_{2|3|4} - r_{4|5} + r_4,$$
$$r_5^4 = r_5 - r_{3|4|5} + r_{3|4},$$
$$r_5^5 = r_{4|5} - r_4.$$

7. A further block diagonalization of the GQRD. In this section, we note that a further block diagonalization of a GQRD can be interpreted as a preliminary step towards the corresponding GSVD. We proceed in two stages. First, we observe that each upper or lower triangular matrix in the generalized URVD of Theorem 5.3 can be block diagonalized. Next, we show how these block diagonalizations can be propagated backward through the GQRD. The first step is the factorization of the upper and lower triangular matrices $\tilde{T}_i$ of Theorem 5.3 into an upper or lower triangular matrix and a block diagonal matrix. For lower triangular matrices $\tilde{T}_i = \tilde{T}_i^l$, we can obtain a factorization of the form

$$\tilde{T}_i^l = L_i \tilde{D}_i^l,$$

where

$$L_i = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{cccccc} r_{i-1}^1 & r_{i-1}^2 & \dots & r_{i-1}^{i-1} & r_{i-1}^i \end{array} \\ \begin{pmatrix} I & 0 & \dots & 0 & 0 \\ * & I & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ * & * & \dots & * & I \end{pmatrix} \end{array},$$

$$\tilde{D}_i^l = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{cccccc} r_i^1 & r_i^2 & \dots & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} R_{i,1} & 0 & \dots & 0 & 0 \\ 0 & R_{i,2} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & R_{i,i} & 0 \end{pmatrix} \end{array}.$$

Since the diagonal blocks $R_{i,j}$ are of full column rank, such a factorization is always possible. In a similar way, for upper triangular matrices $\tilde{T}_i = \tilde{T}_i^u$, we find a factorization of the form

$$\tilde{T}_i^u = U_i \tilde{D}_i^u,$$

with U_i an upper triangular block matrix with identity matrices on the block diagonal:

$$U_i = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{cccccc} r_{i-1}^1 & r_{i-1}^2 & \dots & r_{i-1}^{i-1} & r_{i-1}^i \end{array} \\ \begin{pmatrix} I & * & \dots & * & * \\ 0 & I & \dots & * & * \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & I \end{pmatrix} \end{array},$$

$$\tilde{D}_i^u = \begin{array}{c} \\ r_{i-1}^1 \\ r_{i-1}^2 \\ \vdots \\ r_{i-1}^i \end{array} \begin{array}{c} \begin{array}{cccccc} r_i^1 & r_i^2 & \dots & r_i^i & r_i^{i+1} \end{array} \\ \begin{pmatrix} R_{i,1} & 0 & \dots & 0 & 0 \\ 0 & R_{i,2} & \dots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & R_{i,i} & 0 \end{pmatrix} \end{array}.$$

Now suppose that we have done this for all matrices $\tilde{T}_i, i = 1, \ldots, k$ in a GQRD of Theorem 5.3. We show how we can propagate a further block diagonalization backward through the GQRD, in a way that is completely consistent with the corresponding GSVD of Theorem 4.1. To simplify the notation, we simply replace $\tilde{T}_i$ by T_i and $\tilde{D}_i$ by D_i in the following.

First, assume that T_k is lower block triangular. It then follows from the previous section that we can factorize T_k as

$$T_k^l = L_k D_k.$$

Depending on whether T_{k-1} is upper or lower triangular, we have two cases:

—$T_{k-1} = T_{k-1}^l$ lower triangular. In this case, the product $T_{k-1}^l L_k$ is lower triangular as well, and we can obtain a similar decomposition

$$T_{k-1}^l L_k = L_{k-1} D_{k-1},$$

where L_{k-1} is again lower triangular and D_{k-1} has the same diagonal blocks $R_{i,j}$ as T_{k-1}^l.

—$T_{k-1} = T_{k-1}^u$ upper triangular. In this case, the product $T_{k-1}^u L_k^{-*}$ is upper triangular, and we can obtain a factorization

$$T_{k-1}^u L_k^{-*} = U_{k-1} D_{k-1},$$

where U_{k-1} is upper triangular and D_{k-1} has the same diagonal blocks $R_{i,j}$ as T_{k-1}^u.

It is easily verified that when T_k is upper triangular, similar conclusions can be obtained.

In general, let T_i be lower triangular and assume that it is factorized as

$$T_i^l = L_i D_i Z_i.$$

Assume that T_{i-1} is lower triangular. Then T_{i-1} can be factored as

$$T_{i-1}^l = L_{i-1} D_{i-1} L_i^{-1}.$$

If T_{i-1} is upper triangular, it can be factored as

$$T_{i-1}^u = U_{i-1} D_{i-1} L_i^*,$$

where U_{i-1} is upper triangular. The cases with T_i upper triangular are similar. Table 2 summarizes all possibilities.

TABLE 2

T_i Lower triangular $T_i = L_i D_i Z_i$	T_{i-1} Lower triangular T_{i-1} Upper triangular	$T_{i-1} = L_{i-1} D_{i-1} L_i^{-1}$ $T_{i-1} = U_{i-1} D_{i-1} L_i^*$
T_i Upper triangular $T_i = U_i D_i Z_i$	T_{i-1} Lower triangular T_{i-1} Upper triangular	$T_{i-1} = L_{i-1} D_{i-1} U_i^*$ $T_{i-1} = U_{i-1} D_{i-1} U_i^{-1}$

Example. Let us apply this result to a sequence of four matrices A_1, A_2, A_3, A_4 with compatible dimensions. If the required sequence is ULUU, then

$$\begin{aligned}
A_1 &= Q_0 T_1^u Q_1^* = Q_0 (U_1 D_1 L_2^*) Q_1^* = (Q_0 U_1) D_1 (Q_1 L_2)^*, \\
A_2 &= Q_1 T_2^l Q_2^* = Q_1 (L_2 D_2 U_3^*) Q_2^* = (Q_1 L_2) D_2 (Q_2 U_3)^*, \\
A_3 &= Q_2 T_3^u Q_3^* = Q_2 (U_3 D_3 U_4^{-1}) Q_3^* = (Q_2 U_3) D_3 (Q_3 U_4)^{-1}, \\
A_4 &= Q_3 T_4^u Q_4^* = Q_3 (U_4 D_4) Q_4^* = (Q_3 U_4) D_4 Q_4^*.
\end{aligned}$$

Note that $U_1 = I_{n_0}$. This follows immediately from the block structure of U_i for $i = 1$. Observe that the relationships between the common factors in the left-hand side of these expressions conform with the requirements for a QQP-SVD. Only the middle factors $D_i, i = 1, 2, 3, 4$ are not quasi-diagonal.

8. Conclusions. In this paper, a constructive proof was given of a multimatrix generalization of the concept of rank factorization. The connection of this new decomposition with the analogous GSVD was also shown. The block structure of both generalizations and the ranks of the individual diagonal blocks in both decompositions were indeed shown to be identical. As is shown in a forthcoming paper, the spaces spanned by certain block columns of the orthogonal transformation matrices Q_i are, in fact, identical to those of the GSVD. The difference lies only in a particular choice of basis vectors for these spaces. The consequences of these connections are still under investigation. We mention the following results here:

- Updating the above decomposition to yield the GSVD requires nonorthogonal transformation. These updating transformations can be chosen block triangular with diagonal block sizes compatible with the index sets derived in Theorem 4.1.
- A modified orthogonal decomposition can be defined where the compound matrix is *not* triangularized but diagonalized. This new factorization is a variant of the above decomposition where now a special coordinate system is chosen for each of the individual orthogonal transformations Q_i. The result is an orthogonal decomposition of the type of Theorem 5.3 where now the generalized singular values can be extracted from the diagonal elements of some triangular blocks. The orthogonal updating needed to obtain this new decomposition can be done with techniques described in [2].
- A geometric interpretation can be given of the bases obtained from the transformation matrices Q_i in Theorem 5.1. As particular examples of these spaces we retrieve the following well-known concepts.

(a) For the case $A_i = (A - \alpha I)$ the GQRD in fact reconstructs the nested null spaces of the matrices $(A - \alpha I)^i$, which reveal the Jordan structure of the matrix A at the eigenvalue α (see also the example in §2).

(b) For the cases $A_{2i} = (A - \alpha B)$ and $A_{2i+1} = B$ the decomposition reconstructs the nested null spaces of the sequences $[B^{-1}(A - \alpha B)]^i$ and $[(A - \alpha B)B^{-1}]^i$, which reveal the Kronecker structure of the pencil $\lambda B - A$ at the generalized eigenvalue α (see [30] and [31]).

(c) For the cases $A_1 = D$ and $A_i = C \cdot A^{i-1} \cdot B, i = 1, \ldots$, the decomposition reconstructs the invertibility subspaces of the discrete time system

$$x_{k+1} = Ax_k + Bu_k,$$
$$y_k = Cx_k + Du_k.$$

These are in fact also the spaces constructed by the structure algorithm of Silverman [29], and they play a role in several key problems of geometrical systems theory [34].

Other applications of GSVDs have been described in [7], [8], [11], [13], and [35], while applications of the generalized QR-decompositions are described in [25] and [36].

Acknowledgment. Gene Golub's hospitality in the summer of 1989 lies at the origin of this joint paper. We also would like to thank Steven Van Leemput for

generating the MATLAB code for the GSVD and the GQRD based on the constructive proofs of the GSVD in [9] and of the GQRD of Theorem 5.3 of this paper.

REFERENCES

[1] Å. BJÖRCK, *Solving linear least squares problems by Gram–Schmidt orthogonalization*, BIT, 7 (1967), pp. 1–21.

[2] A. BOJANCZYK, M. EWERBRING, F. LUK, AND P. VAN DOOREN, *An accurate product* SVD *algorithm*, in Proc. Second Internat. Sympos. on SVD and Signal Processing, Kingston, RI, pp. 217–228, June 1990; Signal Proc., to appear.

[3] T. CHAN, *An improved algorithm for computing the singular value decomposition*, ACM Trans. Math. Software, 8 (1982), pp. 72–83.

[4] ———, *Algorithm* 581: *An improved algorithm for computing the singular value decomposition*, ACM Trans. Math. Software, 8 (1982), pp. 84–88.

[5] ———, *Rank revealing* QR *factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82.

[6] B. DE MOOR AND G. H. GOLUB, *Generalized singular value decompositions: A proposal for a standardized nomenclature*, ESAT-SISTA Report 1989-10, Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium, April 1989.

[7] ———, *The restricted singular value decomposition: Properties and applications*, SIAM J. Matrix Anal. Appl., 12 (1991), pp. 401–425.

[8] B. DE MOOR, *On the structure and geometry of the product singular value decomposition*, Linear Algebra Appl., 168 (1992), pp. 95–136.

[9] B. DE MOOR AND H. ZHA, *A tree of generalizations of the singular value decomposition*, ESAT-SISTA Report 1990-11, Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium, June 1990; Linear Algebra Appl., to appear.

[10] B. DE MOOR, *On the structure of generalized singular value decompositions*, ESAT-SISTA Report 1990-12, Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium, June 1990.

[11] ———, *Generalizations of the* OSVD: *Structure, properties, applications*, in Proc. Second Internat. Workshop on SVD and Signal Processing, University of Rhode Island, Kingston, RI, June 1990, pp. 209–216; Signal Proc., to appear.

[12] ———, *A history of the singular value decomposition*, ESAT-SISTA Report, Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium, February 1991.

[13] K. V. FERNANDO AND S. J. HAMMARLING, *A product induced singular value decomposition for two matrices and balanced realisation*, in Linear Algebra in Signal Systems and Control, B. N. Datta, C. R. Johnson, M. A. Kaashoek, R. Plemmons, and E. Sontag, eds., Society for Industrial and Applied Mathematics, Philadelphia, PA, 1988, pp. 128–140.

[14] P. E. GILL, W. MURRAY, AND M. H. WRIGHT, *Practical Optimization*, Academic Press, London, 1981.

[15] G. H. GOLUB AND W. KAHAN, *Calculating the singular values and pseudo-inverse of a matrix*, SIAM J. Numer. Anal., 2 (1965), pp. 205–224.

[16] P. A. BUSINGER AND G. H. GOLUB, *Algorithm* 358: *Singular value decomposition of a complex matrix*, Comm. Assoc. Comp. Mach., 12 (1969), pp. 564–565.

[17] G. H. GOLUB AND C. REINSCH, *Singular value decomposition and least squares solutions*, Numer. Math., 14 (1970), pp. 403–420.

[18] G. H. GOLUB AND C. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[19] G. GOLUB AND J. WILKINSON, *Ill-conditioned eigensystems and the computation of the Jordan canonical form*, SIAM Rev., 18 (1976), pp. 578–619.

[20] S. HAMMARLING, *The numerical solution of the general Gauss–Markov linear model*, NAG Tech. Report TR2/85, Numerical Algorithms Group Limited, Oxford, 1985.

[21] M. T. HEATH, A. J. LAUB, C. C. PAIGE, AND R. C. WARD, *Computing the singular value decomposition of a product of two matrices*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1147–1159.

[22] R. J. HANSON AND C. L. LAWSON, *Extensions and applications of the Householder algorithms for solving linear least squares problems*, Math. Comp., 23 (1969), pp. 787–812.

[23] C. C. PAIGE AND M. A. SAUNDERS, *Towards a generalized singular value decomposition*, SIAM J. Numer. Anal., 18 (1981), pp. 398–405.

[24] C. C. PAIGE, *Computing the generalized singular value decomposition*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1126–1146.

[25] ———, *Some aspects of generalized* QR *factorizations*, in Reliable Numerical Computation, M.

Cox and S. Hammarling, eds., Oxford University Press, Oxford, U.K., 1990, pp. 73–91.

[26] G. W. STEWART, *A method for computing the generalized singular value decomposition*, in Proc. Matrix Pencils, Pite Havsbad, 1982, B. Kagström and A. Ruhe, eds., Lecture Notes in Mathematics 973, Springer-Verlag, Berlin, New York, 1983.

[27] ———, *An updating algorithm for subspace tracking*, UMIACS-TR-90-86, CS-TR 2494, Computer Science Tech. Report Series, University of Maryland, College Park, MD, July 1990.

[28] J. STOER, *On the numerical solution of constrained least-squares problems*, SIAM J. Numer. Anal., 8 (1971), pp. 382–411.

[29] L. M. SILVERMAN, *Discrete Riccati Equations: Alternative Algorithms, Asymptotic Properties and System Theory Interpretations*, in Control and Dynamical Systems, Academic Press, New York, 1976.

[30] P. VAN DOOREN, *The computation of Kronecker's canonical form of a singular pencil*, Linear Algebra Appl., 27 (1979), pp. 103–140.

[31] ———, *Reducing subspaces: Definitions, properties and algorithms*, in Proc. Matrix Pencils, Lecture Notes in Mathematics 973, Springer-Verlag, Berlin, New York, 1983, pp. 58–73.

[32] C. F. VAN LOAN, *Generalizing the singular value decomposition*, SIAM J. Numer. Anal., 13 (1976), pp. 76–83.

[33] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, The Clarendon Press, Oxford, U.K., 1965.

[34] M. WONHAM, *Linear Multivariable Control, a Geometric Approach*, Springer-Verlag, New York, 1974.

[35] H. ZHA, *The restricted* SVD *for matrix triplets and rank determination of matrices*, Scientific Report 89-2, Konrad–Zuse Zentrum für Informationstechnik, Berlin, Germany, 1989; SIAM J. Matrix Anal. Appl., 12 (1991), pp. 172–194.

[36] ———, *The implicit* QR *decomposition and its applications*, ESAT-SISTA Report 1989-25, Department of Electrical Engineering, Katholieke Universiteit Leuven, Leuven, Belgium.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1015–1038, October 1992

002

A SINGULAR VALUE DECOMPOSITION UPDATING ALGORITHM FOR SUBSPACE TRACKING*

MARC MOONEN†§, PAUL VAN DOOREN‡, AND JOOS VANDEWALLE†

Abstract. In this paper, the well-known QR updating scheme is extended to a similar but more versatile and generally applicable scheme for updating the singular value decomposition (SVD). This is done by supplementing the QR updating with a Jacobi-type SVD procedure, where apparently only a few SVD steps after each QR update suffice in order to restore an acceptable approximation for the SVD. This then results in a reduced computational cost, comparable to the cost for merely QR updating.

The usefulness of such an approximate updating scheme when applied to subspace tracking is examined. It is shown how an $\mathcal{O}(n^2)$ SVD updating algorithm can restore an acceptable approximation at every stage, with a fairly small tracking error of approximately the time variation in $\mathcal{O}(n)$ time steps.

Finally, an error analysis is performed, proving that the algorithm is stable, when supplemented with a Jacobi-type reorthogonali zation procedure, which can easily be incorporated into the updating scheme.

Key words. singular value decomposition, recursive algorithms

AMS(MOS) subject classifications. 65F15, 65F25

C.R. classification. G.1.3

1. Introduction.

In many signal processing applications, it is necessary to continuously update matrix decompositions as new measurement vectors are appended as additional rows. Such problems frequently occur in beam forming, direction finding, spectral analysis, etc. [21]. Efficient updating techniques have long been known for the QR decomposition [8], while the more difficult problem of updating the (ordinary) singular value decomposition (SVD) has only recently been addressed [1], [2], [4], [9], [20].

Previously described techniques for row updating of the SVD mostly reduce to computing the rank-one modification of the corresponding symmetric eigenvalue problem [1], [2], [9]. A major drawback is the necessary knowledge of the exact eigenstructure of the original matrix in order to compute the updated eigenstructure. For real-time applications, where in each time step an exact updating is thus to be performed, this results in an unacceptably heavy computational load, viz., $\mathcal{O}(n^3)$ per update. Moreover, round-off errors due to the use of finite precision arithmetic are likely to accumulate unboundedly.

In this paper, we derive a fast SVD updating technique as a combination of QR updating on the one hand and a Jacobi-type SVD procedure on the other hand. The updating scheme provides only an approximate decomposition after each update. It has a low computational complexity—$\mathcal{O}(n^2)$ per update—and it is particularly suited to parallel implementation [18].

* Received by the editors January 24, 1990; accepted for publication (in revised form) December 26, 1990. This work was sponsored in part by the BRA3280 project of the European Community.

† Department of Electrical Engineering (ESAT), Katholieke Universiteit Leuven, K. Mercierlaan 94, 3001 Heverlee, Belgium (moonen@esat.kuleuven.ac.be and vandewalle@esat.kuleuven.ac.be).

‡ Department of Electrical and Computer Engineering, University of Illinois, 1101 West Springfield Avenue, Urbana, Illinois 61801 (vdooren@uicsl.csl.uiuc.edu).

§ This author is a senior research assistant with the N.F.W.O. (Belgian National Fund for Scientific Research).

When combined with exponential weighting, such an algorithm is seen to be highly applicable to subspace tracking problems. SVD methods are known to be extremely reliable in this respect, but are considered "too expensive" when it comes to real-time applications (cf. the $\mathcal{O}(n^3)$ complexity). Cheaper alternatives usually suffer from poor numerical properties (see [4] for a survey). We will show how an $\mathcal{O}(n^2)$ approximate SVD updating algorithm can restore an acceptable approximation at every stage, with a fairly small tracking error approximately equal to the time variation in $\mathcal{O}(n)$ time steps.

Finally, an error analysis is performed, proving that the algorithm is stable, when supplemented with a certain Jacobi-type reorthogonalization procedure, which can easily be incorporated into the updating scheme.

The SVD updating procedure is developed in §2. In §3 a performance analysis for subspace tracking (in infinite precision arithmetic) is sketched. Finally, error build-up issues (finite precision arithmetic) are addressed in §4.

2. An SVD updating algorithm. The SVD of a real matrix $A_{m\times n}$ $(m \geq n)$ is a factorization of A into a product of three matrices [10]

$$\underbrace{A}_{m\times n} = \underbrace{U}_{m\times n} \cdot \underbrace{\Sigma}_{n\times n} \cdot \underbrace{V^T}_{n\times n}$$

where

$$U^T \cdot U = I_{n\times n},$$
$$V^T \cdot V = I_{n\times n},$$

and Σ is a diagonal matrix, with the singular values along the diagonal

$$\Sigma = \operatorname{diag}\{\sigma_1, \sigma_2, \cdots, \sigma_n\},$$

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0.$$

Updating the SVD after appending a new row consists of computing the SVD of the modified matrix

$$A_+ = \begin{bmatrix} A \\ a^T \end{bmatrix} = \underbrace{U_+}_{(m+1)\times n} \cdot \underbrace{\Sigma_+}_{n\times n} \cdot \underbrace{V_+^T}_{n\times n}$$

by making use of the original SVD of A. In on-line applications, a new updating often has to be performed after each sampling, and the data matrix at time step k is defined in a recursive manner $(k \geq n)$

$$A_{[k]} = \begin{bmatrix} \lambda_{[k]} \cdot A_{[k-1]} \\ a_{[k]}^T \end{bmatrix} = \underbrace{U_{[k]}^\circ}_{k\times n} \cdot \underbrace{\Sigma_{[k]}^\circ}_{n\times n} \cdot \underbrace{V_{[k]}^{\circ T}}_{n\times n}.$$

Factor $\lambda_{[k]}$ is a weighting factor, and $a_{[k]}$ is the measurement vector at time instant k. For the sake of brevity, we consider only the case where $\lambda_{[k]}$ is a constant λ, although everything can easily be recast for the case where it is time-varying. Since we will consider approximate decompositions below, we use an additional superscript $^\circ$ here to denote exact decompositions. Finally, in most applications the $U_{[k]}^\circ$-matrix, with growing matrix dimensions, need not be computed, and only $V_{[k]}^\circ$ and $\Sigma_{[k]}^\circ$ are explicitly updated.

2.1. SVD updating, first version. An SVD updating algorithm (for infinite precision arithmetic) is readily constructed by combining QR updating with a Jacobi-type SVD diagonalization procedure (Kogbetliantz's algorithm, modified for triangular matrices [14], [15]).

Suppose that at a certain time step $k-1$, the $A_{[k-1]}$-matrix is reduced to $R_{[k-1]}$—upper triangular and almost diagonal—with corresponding matrices $U_{[k-1]}$ and $V_{[k-1]}$:

$$A_{[k-1]} = U_{[k-1]} \cdot R_{[k-1]} \cdot V_{[k-1]}^T.$$

After appending a new row $a_{[k]}^T$, we have a decomposition of the type

$$\begin{aligned} A_{[k]} &= \begin{bmatrix} \lambda \cdot A_{[k-1]} \\ a_{[k]}^T \end{bmatrix} \\ &= \begin{bmatrix} U_{[k-1]} & \\ & 1 \end{bmatrix} \cdot \begin{bmatrix} \lambda \cdot R_{[k-1]} \\ a_{[k]}^T \cdot V_{[k-1]} \end{bmatrix} \cdot V_{[k-1]}^T. \end{aligned}$$

The updating can then be carried out in the following three steps.

1. *Matrix-vector multiplication and exponential weighting.* The triangular factor is multiplied by the weighting factor, and the input vector $a_{[k]}$ is transformed to $\tilde{a}_{[k]}$ by making use of the current $V_{[k-1]}$-matrix:

$$\begin{aligned} \tilde{R}_{[k-1]} &= \lambda \cdot R_{[k-1]}, \\ \tilde{a}_{[k]}^T &= a_{[k]}^T \cdot V_{[k-1]}. \end{aligned}$$

2. QR *updating* with $\tilde{a}_{[k]}$ in order to restore the triangular structure:

$$\begin{aligned} A_{[k]} &= \begin{bmatrix} U_{[k-1]} & \\ & 1 \end{bmatrix} \cdot \begin{bmatrix} \tilde{R}_{[k-1]} \\ \tilde{a}_{[k]}^T \end{bmatrix} \cdot V_{[k-1]}^T \\ &= \begin{bmatrix} U_{[k-1]} & \\ & 1 \end{bmatrix} \cdot Q_{[k]} \cdot \begin{bmatrix} \hat{R}_{[k]} \\ 0 \end{bmatrix} \cdot V_{[k-1]}^T \\ &= \underbrace{\begin{bmatrix} U_{[k-1]} & \\ & 1 \end{bmatrix} \cdot Q_{[k]} \cdot \begin{bmatrix} I_{n \times n} \\ 0 \end{bmatrix}}_{\hat{U}_{[k]}} \cdot \hat{R}_{[k]} \cdot V_{[k-1]}^T. \end{aligned}$$

The QR updating is done by applying a sequence of Givens rotations (see, e.g., [8] for details). Note that the QR updating does not alter the V-matrix. The U-matrix does change, but it does not have to be stored anyway, as we are only interested in R and V.

3. SVD *steps* in order to obtain a diagonal matrix. This diagonalization procedure consists in applying a sequence of plane rotations as follows (see [14] and [15] for details):

$$\begin{aligned} R_{[k]} &\Leftarrow \hat{R}_{[k]} \\ V_{[k]} &\Leftarrow V_{[k-1]} \end{aligned}$$

for $j = 1, \cdots, r$
  for $i = 1, \cdots, n-1$

$$R_{[k]} \Leftarrow \Theta_{[i,j,k]}^T \cdot R_{[k]} \cdot \Phi_{[i,j,k]}$$
$$V_{[k]} \Leftarrow V_{[k]} \cdot \Phi_{[i,j,k]}$$

end

end

The parameter i is called the *pivot index*. The matrices $\Theta_{[i,j,k]}$ and $\Phi_{[i,j,k]}$ represent *rotations in the* $(i, i+1)$*-plane*:

$$\Theta_{[i,j,k]} = \begin{bmatrix} I_{i-1} & & & \\ & \cos\theta_{[i,j,k]} & \sin\theta_{[i,j,k]} & \\ & -\sin\theta_{[i,j,k]} & \cos\theta_{[i,j,k]} & \\ & & & I_{n-i-1} \end{bmatrix},$$

$$\Phi_{[i,j,k]} = \begin{bmatrix} I_{i-1} & & & \\ & \cos\phi_{[i,j,k]} & \sin\phi_{[i,j,k]} & \\ & -\sin\phi_{[i,j,k]} & \cos\phi_{[i,j,k]} & \\ & & & I_{n-i-1} \end{bmatrix},$$

where I_l is an $l \times l$ identity matrix. The rotation angles $\theta_{[i,j,k]}$ and $\phi_{[i,j,k]}$ should be chosen so as to annihilate the $(i, i+1)$ element in $R_{[k]}$, while preserving $R_{[k]}$ in upper triangular form. Each iteration thus essentially reduces to applying a 2×2 SVD on the main diagonal. The SVD procedure then consists in performing r sequences of $n-1$ such plane rotations, where the pivot index repeatedly takes up all the values $i = 1, \cdots, n-1$. It is well known that if use is made of *outer rotations* (see [23] and [16]), exactly n such sequences constitute a double sweep for a cyclic ordering (pipelined forward and backward sweep). Each rotation reduces the off-norm in $R_{[k]}$, and $R_{[k]}$ eventually converges to a diagonal matrix [6], so that finally we will have

$$R_{[k]} = \Sigma_{[k]}^{\circ},$$
$$V_{[k]} = V_{[k]}^{\circ}.$$

With the above procedure, the diagonal structure of R can be restored after each update. With $r = \mathcal{O}(n)$ on the average, the operation count is $\mathcal{O}(n^3)$ per update. In practice, however, it generally suffices to keep $R_{[k]}$ "close" to a (block) diagonal matrix, instead of *completely* reducing it to a diagonal matrix in each time step. In some sense (see §3) $V_{[k]}$ is then "close" to $V_{[k]}^{\circ}$ as well, which, for subspace tracking applications, for instance (ESPRIT, systems identification, recursive total least squares), is the only thing that matters. The number of rotations $r(n-1)$ in a certain time step can then be fixed, turning an iterative algorithm into a seemingly noniterative one. Of course, the crux is then to show how only a few SVD steps after each QR update can restore an acceptable approximation at every stage.

From now on, we make a fairly arbitrary choice and set r equal to 1, so that after each QR update, the pivot index takes up all values $i = 1, \cdots, n-1$ only once. In this case, both the QR updating and the rotations following the update take $\mathcal{O}(n^2)$ operations, which, e.g., results in an elegant parallel implementation [18]. It should be stressed, however, that all of our further results can straightforwardly be recast for other choices for r. In §3, the problem is addressed of "how closely" the obtained estimates then approximate the exact decomposition—in particular for the subspace tracking problem—and for this particular choice for r.

For the time being, the updating procedure is thus summarized as follows (with $n-1$ rotations after each update).

Initialization

$$V_{[0]} \Leftarrow I_{n\times n},$$
$$R_{[0]} \Leftarrow O_{n\times n}$$

Loop

for $k = 1, \cdots, \infty$

1. input new measurement vector $a_{[k]}$

$$\tilde{a}^T_{[k]} \Leftarrow a^T_{[k]} \cdot V_{[k-1]},$$
$$\tilde{R}_{[k-1]} \Leftarrow \lambda \cdot R_{[k-1]}$$

2. QR updating

$$\begin{bmatrix} \hat{R}_{[k]} \\ 0 \end{bmatrix} \Leftarrow Q^T_{[k]} \cdot \begin{bmatrix} \tilde{R}_{[k-1]} \\ \tilde{a}^T_{[k]} \end{bmatrix},$$

$$R_{[k]} \Leftarrow \hat{R}_{[k]},$$
$$V_{[k]} \Leftarrow V_{[k-1]}$$

3. SVD steps

for $i = 1, \cdots, n-1$

$$R_{[k]} \Leftarrow \Theta^T_{[i,k]} \cdot R_{[k]} \cdot \Phi_{[i,k]},$$
$$V_{[k]} \Leftarrow V_{[k]} \cdot \Phi_{[i,k]}$$

end

end

2.2. Version 2, including reorthogonalizations. In view of round-off accumulation (finite precision arithmetic), the above updating algorithm has one shortcoming. The stored matrix V is iteratively updated by orthogonal column transformations according to

$$V_{[k]} \Leftarrow V_{[k]} \cdot \Phi_{[i,k]}.$$

While $V_{[0]}$ is orthogonal through the initialization, $V_{[k]}$ ($k \gg 1$) is probably not, due to round-off. The deviation from orthogonality apparently grows linearly with k, as can be verified experimentally. Keeping $V_{[k]}$ close to orthogonal is crucial, however, for the overall error propagation stability (see §4). Including some kind of reorthogonalization procedure is therefore indispensable. An efficient procedure that elegantly combines with the updating scheme (e.g., on a systolic array [18]) can be constructed as follows.

Suppose two vectors x_p and x_q are *almost orthonormal*, in the sense that

$$\|x_p\|_2 = 1 + \mathcal{O}(\epsilon),$$
$$\|x_q\|_2 = 1 + \mathcal{O}(\epsilon),$$
$$x^T_p \cdot x_q = \mathcal{O}(\epsilon),$$

TABLE 1

Number of sweeps	$\|X^T \cdot X - I\|_F$
	1.7335e−01
1	2.0767e−03
2	1.5657e−06
3	0.6442e−13
4	—

where ϵ is a small number. We can easily verify that a (symmetrized) Gram–Schmidt-like transformation

$$\begin{bmatrix} x_p^\star & x_q^\star \end{bmatrix} = \begin{bmatrix} x_p & x_q \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\|x_p\|} & -\frac{x_p^T \cdot x_q}{2} \\ -\frac{x_p^T \cdot x_q}{2} & \frac{1}{\|x_q\|} \end{bmatrix}$$

yields two new vectors, which satisfy

$$\begin{aligned} \|x_p^\star\|_2 &= 1 + \mathcal{O}(\epsilon^2), \\ \|x_q^\star\|_2 &= 1 + \mathcal{O}(\epsilon^2), \\ x_p^{\star T} \cdot x_q^\star &= \mathcal{O}(\epsilon^2). \end{aligned}$$

Note that an exact Gram–Schmidt orthogonalization is computationally somewhat more involved, while on the other hand it yields only marginally better results.

For an $n \times n$ close-to-orthogonal matrix

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix},$$

we can straightforwardly apply the 2×2 transformations in a cyclic manner. One (forward) sweep consists in computing transformations as follows.

Loop
  for $p = 1, \cdots, n-1$
    for $q = p+1, \cdots, n$

$$\begin{bmatrix} x_p & x_q \end{bmatrix} \Leftarrow \begin{bmatrix} x_p & x_q \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\|x_p\|} & -\frac{x_p^T \cdot x_q}{2} \\ -\frac{x_p^T \cdot x_q}{2} & \frac{1}{\|x_q\|} \end{bmatrix}$$

    end
  end

The above algorithm is seen to be a one-sided Jacobi-type procedure with nonunitary transformations, or two-sided if we consider the effect on $X^T \cdot X$. Furthermore, if $\|X^T \cdot X - I\|_F = \mathcal{O}(\epsilon)$ for some small ϵ, the 2×2 transformations are ϵ close to $I_{2\times 2}$. By making use of this, we can easily prove that the procedure converges quadratically, in other words, that $\|X^T \cdot X - I\|_F = \mathcal{O}(\epsilon^2)$ after one sweep. It suffices to copy Wilkinson's proof [24] with appropriate substitutions. In Table 1 we show the effect of the procedure on a *random* 10×10 X-matrix (i.e., *not even close* to an orthogonal one).

Let us now return to the SVD updating algorithm. If we choose

$$X = V$$

and interlace the above reorthogonalization with the updating procedure, the former no longer converges quadratically, due to the updating transformations $X \Leftarrow X \cdot \Phi_{[i,k]}$ that clearly change $X^T \cdot X$ as well and thus interfere with the reorthogonalization. However, if we choose

$$X = V^T,$$

in other words, if we apply the reorthogonalization scheme onto the rows of V, the updating transformations do not change $X^T \cdot X$, at least not for (local) infinite precision arithmetic ($X^T \cdot \Phi_{[i,k]} \cdot \Phi_{[i,k]}^T \cdot X = X^T \cdot X$), so that *apparently* both processes do not interfere. In finite precision, these processes of course do interfere. Both the updating transformations and the reorthogonalization steps introduce new round-off errors, which have to be annihilated by the reorthogonalization itself. On the other hand, the reorthogonalization (slightly) changes the $V_{[k]}$ matrix in an abitrary manner with respect to the data. These issues will be analyzed in detail in §4.

The updating and the reorthogonalization can now be interlaced. For instance, we could alternately perform one updating rotation, one reorthogonalization step, etc. The body of the inner "for"-loop in the updating algorithm then becomes

$$\vdots$$

3. SVD steps and reorthogonalization
   for $i = 1, \cdots, n-1$

$$R_{[k]} \Leftarrow \Theta_{[i,k]}^T \cdot R_{[k]} \cdot \Phi_{[i,k]}$$
$$V_{[k]} \Leftarrow T_{[i,k]} \cdot V_{[k]} \cdot \Phi_{[i,k]}$$

   end

$$\vdots$$

where $T_{[i,k]}$ is a reorthogonalization in the $(p_{[i,k]}, q_{[i,k]})$-plane. Here the number of pairwise reorthogonalizations per update equals the number of updating transformations. Again, this is an arbitrary choice, which in some cases might be overly conservative. Further results on the obtained accuracy can however easily be recast for other choices. Furthermore, e.g., in a systolic array implementation [18] with a separate V-array and R-array, the above additional reorthogonalizations do not introduce any computational overhead, but rather a load balancing between the two arrays, so that there is no point in reducing the number of reorthogonalizations.

Finally, in view of efficient (parallel) implementation (see also [18]), the cyclic reorthogonalization can of course be reordered, by making use of additional permutations (outer transformations) and a pipelining of the forward and backward sweep (so that $p_{[i,k]} = i$ and $q_{[i,k]} = i + 1$). For the time being, however, we do not pursue this, as it would considerably complicate our notation in the subsequent analysis.

3. Subspace tracking. In this section, we analyze the performance of the SVD updating scheme, when applied to subspace tracking. We assume that all computations are performed with infinite precision, so that the reorthogonalization is superfluous. Round-off errors introduce second-order effects, which for the time being are left out for the sake of simplicity.

First of all, a data model is put forward, which applies to popular signal processing applications, such as direction finding (ESPRIT, MUSIC) and system identification [17]. For such applications, an SVD step is used to separate a signal subspace from a noise subspace (see below), corresponding to certain submatrices of $V^{\circ}_{[k]}$. Further information (e.g., the angles of arrival for ESPRIT) can then be computed from the knowledge of these submatrices only. As for an approximate SVD with an approximately (block) diagonal triangular factor $R_{[k]}$, the corresponding subspace separation error is shown to be related to the norm of some off-diagonal block of cross terms in the triangular factor. The effect of the SVD procedure on this norm is investigated and finally, all these are applied to the updating problem.

3.1. Data model. The data model we consider only assumes that at each time step k the data matrix $A_{[k]}$ as defined in §2 has a fixed number d of large singular values

$$\sigma_{i_{[k]}}, \quad i = 1, \cdots, d$$

and a remaining number of *small* singular values

$$\sigma_{i_{[k]}}, \quad i = d+1, \cdots, n.$$

The ratio

$$SN_{[k]} = \frac{\sigma_{d_{[k]}}}{\sigma_{d+1_{[k]}}}$$

can be interpreted as a *signal-to-noise ratio*, and is assumed to be large, e.g., at least 10 or 100, and lower bounded by a constant SN

$$SN_{[k]} \geq SN.$$

The corresponding submatrices $Vs^{\circ}_{[k]}$ and $Vn^{\circ}_{[k]}$ in $V^{\circ}_{[k]}$ define the *signal subspace* $\mathcal{R}(Vs^{\circ}_{[k]})$ and the *noise subspace* $\mathcal{R}(Vn^{\circ}_{[k]})$, orthogonal to $\mathcal{R}(Vs^{\circ}_{[k]})$.

As we consider time-varying systems, we need to define a measure of time variation one way or another. In view of the applications at hand, it is indicated to make use of the canonical angles θ_i between $\mathcal{R}(Vs^{\circ}_{[k-1]})$ and $\mathcal{R}(Vs^{\circ}_{[k]})$, the cosines of which are the singular values of a corresponding matrix product [10]

$$\cos\theta_i = \sigma_i\{Vs^{\circ T}_{[k-1]} \cdot Vs^{\circ}_{[k]}\}.$$

We then define the *time variation* from time step $k-1$ to time step k (for a prespecified choice for d) as the distance between the corresponding signal subspaces, which in turn can be defined in terms of the above canonical angles as follows:

$$TV_{[k-1]\to[k]} \stackrel{\text{def}}{=} \text{dist}\{\mathcal{R}(Vs^{\circ}_{[k-1]}), \mathcal{R}(Vs^{\circ}_{[k]})\}$$
$$\stackrel{\text{def}}{=} \sqrt{\sum_{i=1}^{d} \tan^2\theta_i}.$$

The reason for this will become clear in what follows.

3.2. Tracking error. The adaptive SVD algorithm of §2 at each time step stores a triangular factor $R_{[k]}$ of the data matrix $A_{[k]}$, as well as an approximation $V_{[k]}$ for the matrix of right singular vectors. Suppose that at a certain time step k, $R_{[k]}$ and $V_{[k]}$ can be split up as follows:

$$R_{[k]} = \begin{bmatrix} Rs_{[k]} & Rsn_{[k]} \\ 0 & Rn_{[k]} \end{bmatrix},$$
$$V_{[k]} = \begin{bmatrix} Vs_{[k]} & Vn_{[k]} \end{bmatrix},$$

where $\|Rn_{[k]}\|$ and $\|Rsn_{[k]}\|$ are "small." For the time being, we thus assume that at time instant k, the large diagonal elements in $R_{[k]}$ occupy adjacent positions, as this considerably simplifies our further analysis. We return to this in Remark 2, below.

It is now clear that $Vs_{[k]}$ provides an approximate basis for the signal subspace. We can then define the *tracking error* at time step k as follows:

$$TE_{[k]} \stackrel{\text{def}}{=} \text{dist}\{\mathcal{R}(Vs_{[k]}), \mathcal{R}(Vs^{\circ}_{[k]})\}.$$

Our aim is ιo derive a useful estimate for the tracking error TE in terms of the time variation TV. In order to obtain this, we can make use of a well-known property, relating the tracking error to the distance of $R_{[k]}$ from a (block) diagonal matrix, as follows. If ε denotes the Frobenius norm of the matrix with cross terms

$$\varepsilon = \|Rsn_{[k]}\|_F$$

and δ is the gap between the singular values of $Rs_{[k]}$ and $Rn_{[k]}$,

$$\delta = \sigma_{\min}\{Rs_{[k]}\} - \sigma_{\max}\{Rn_{[k]}\},$$

and furthermore if

$$2\varepsilon < \delta,$$

then it is has been shown [3], [22] that

$$TE_{[k]} < 2\frac{\varepsilon}{\delta}.$$

In other words, the tracking error is proportional to the norm of the block of cross terms in $R_{[k]}$.

3.3. Kogbetliantz's algorithm. Suppose we would now perform a few sweeps of Kogbetliantz's SVD algorithm (without any QR updates!) and then again check the norm of the cross terms. Classical convergence results for Kogbetliantz's algorithm turn out to be useless in this respect. Linear convergence bounds are extremely conservative [5], [6], [11], [13], while ultimate quadratic convergence results do not apply to the initial convergence, where the off-diagonal elements in $Rs_{[k]}$ can be very large [3], [19], [24]. Jacobi-type algorithms are considered extremely fast, but for the general case no estimates are available whatsoever for the speed of the initial convergence. For our specific subspace separation problem, however, it is possible to derive a useful estimate for the reduction of the cross terms in each sweep. In Appendix A, the following rule of thumb is obtained.

If the cross terms are small compared to the gap and the large diagonal elements are grouped, each double sweep in Kogbetliantz's SVD algorithm reduces the cross terms by a factor $1/SN^2$.

TABLE 2

Number of sweeps	ε
	2.3145e−02
2	0.7311e−06
4	0.6931e−10
6	0.7499e−14

Our derivation relies on a few simplifying assumptions, in order to avoid tedious mathematics and overly conservative results. Our results, however, are practical, and easy to verify by experiments.

Example 1. For a triangular factor $R =$

3.7047	0.4920	0.4312	1.1988	0.8095	−0.0051	−0.0032	0.0009	−0.0019	−0.0014
	3.0436	1.0955	1.1852	2.5027	0.0099	−0.0018	0.0038	−0.0093	−0.0061
		2.8701	1.3763	0.6623	0.0050	0.0026	−0.0024	0.0013	−0.0020
			1.4314	1.1373	0.0080	0.0027	0.0058	0.0055	0.0008
				2.5905	0.0017	−0.0006	−0.0021	0.0070	0.0042
					0.0133	0.0028	0.0031	0.0003	0.0072
						0.0130	0.0011	0.0027	0.0004
							0.0057	0.0031	0.0035
								0.0089	0.0047
									0.0056

with singular values

$$\sigma = 5.4467, 3.5381, 2.7569, 1.9772, 1.1424,$$
$$0.0173, 0.0126, 0.0104, 0.0051, 0.0043,$$

SN is approximately equal to 100. If the SVD procedure were carried out as such, the cross terms would be reduced much faster than could be predicted with the above rule of thumb. As the convergence soon turns into (much faster) quadratic convergence, the $1/SN^2$ reduction of the cross terms will not be visible. Therefore, it is more instructive to see what happens if in each sweep *only* the cross terms are being annihilated, while all other rotations are skipped (corresponding to Part (b) in Appendix A). Now the $1/SN^2$ reduction is much more clearly displayed (Table 2). Note, however, that in our updating algorithm, all the rotations *are* performed, such that the cross-term reduction is indeed much faster. The point is that no (sharper) bound is available for this faster convergence. Also, as far as our algorithmic description is concerned, the sizes of the subblocks thus need not be identified whatsoever (see also Remark 2).

From the above rule of thumb, we can straightforwardly infer an estimate for the reduction of the subspace separation error. As this latter is bounded by the norm of the cross terms $\|Rsn_{[k]}\|_F$, we can conclude that each double sweep in Kogbetliantz's SVD algorithm reduces the subspace separation error $\text{dist}\{\mathcal{R}(Vs_{[k]}), \mathcal{R}(Vs^{\circ}_{[k]})\}$ by a factor $(1/SN^2)$.

Example 2. Similar to the experiment in Example 1, we checked the reduction of the subspace separation error per double sweep (for three successive double sweeps), for different triangular matrices. The matrix dimension n ranges from 10 through 50, while the singular value spectra were chosen to be

$$\sigma = \frac{n}{2}, \frac{n}{2} - 1, \cdots, 2, 1,$$
$$\frac{1}{SN}, \frac{\frac{n}{2} - 1}{\frac{n}{2} \cdot SN}, \cdots, \frac{1}{\frac{n}{2} \cdot SN}$$

for $SN = 10$ and $SN = 100$. For all cases, the reduction factor is seen to be approximately equal to $1/SN^2$. Apparently, the matrix dimension has little influence on this.

3.4. Subspace tracking. Let us now return to the SVD updating algorithm, applied to subspace tracking. In the adaptive algorithm, a pipelined double sweep ($n \times n - 1$ rotations) is interlaced with n QR updates, one after each series of $n - 1$ rotations. If all these QR updates were performed *after* the double sweep, we would end up with the following inequality:

$$\begin{aligned}&\operatorname{dist}\{\mathcal{R}(Vs_{[k+n]}), \mathcal{R}(Vs^{\circ}_{[k+n]})\}\\ &\leq \left(\frac{1}{SN^2}\right) \cdot \operatorname{dist}\{\mathcal{R}(Vs_{[k]}), \mathcal{R}(Vs^{\circ}_{[k]})\} + \operatorname{dist}\{\mathcal{R}(Vs^{\circ}_{[k]}), \mathcal{R}(Vs^{\circ}_{[k+n]})\}.\end{aligned}$$

The "$\leq$" sign is due to the fact that the different terms in the right-hand side can partially eliminate each other. The first term corresponds to the reduction of the subspace separation error in a double sweep (n time steps). The second term corresponds to the time variation from time instant k to time instant $k + n$.

If the QR updates are interlaced in the double sweep, the subspace separation error is expected to be even smaller, as time variations can then immediately be taken into account and corrected correspondingly. Although we can easily think of set-ups where the above statement does not even hold, in general it is confirmed by simulations (see below for an example). The above inequality can therefore be assumed to provide a reasonable estimate for the SVD updating scheme as well.

Furthermore, as we assumed that SN is fairly large (e.g., 100), it follows that

$$\underbrace{\operatorname{dist}\{\mathcal{R}(Vs_{[k+n]}), \mathcal{R}(Vs^{\circ}_{[k+n]})\}}_{TE_{[k+n]}} \leq \underbrace{\operatorname{dist}\{\mathcal{R}(Vs^{\circ}_{[k]}), \mathcal{R}(Vs^{\circ}_{[k+n]})\}}_{TV_{[k]\to[k+n]}}.$$

In other words, we can conclude that the tracking error is bounded by the time variation in n time steps.

A few remarks on the above derivations and results are as follows.

Remark 1. The above results were derived for the case where only one sequence of $n - 1$ SVD rotations is performed after each QR update ($r = 1$ in §2). For other choices for r, we would obviously end up with

$$TE_{[k+\frac{n}{r}]} \leq TV_{[k]\to[k+\frac{n}{r}]},$$

as one double sweep is then performed in $\frac{n}{r}$ time steps. In other words, the tracking error is inversely proportional to r.

Remark 2. Throughout the computations (in Appendix A), we have assumed that the small diagonal elements in the triangular factor occupy (circularly) adjacent positions (cf. the configurations in Appendix A after each sequence of rotations). Note that a similar assumption had to be made for the proof of the quadratic convergence for matrices with pathologically close or repeated singular values [3], [24]. In an adaptive scheme, obtaining such a set-up is sometimes merely a matter of careful initialization (once the small elements are grouped, they do not change their "affiliation," at least not for slowly time-varying systems). Furthermore, it is observed (by performing simulations) that even when the large and small diagonal elements are *not* grouped, the tracking error is still bounded by the time variation in n time steps. In other

TABLE 3

	$n = 10$	$n = 20$	$n = 50$
$SN = 10$	$2.8100e-02$	$2.0356e-02$	$1.8847e-02$
	$8.2947e-05$	$8.6065e-05$	$5.8207e-05$
	$3.4249e-07$	$5.6355e-07$	$4.2585e-07$
	$1.4534e-09$	$3.7391e-09$	$3.5933e-09$
$SN = 100$	$2.0213e-02$	$2.01087-02$	$1.9357e-02$
	$4.7201e-07$	$5.8845e-07$	$8.9264e-07$
	$2.3465e-11$	$3.5926e-11$	$7.4129e-11$
	$1.1801e-15$	$2.6241e-15$	$6.9677e-15$

words, this latter bound seems to be conservative enough, so that it applies to the (disadvantageous) "nongrouped" case as well.

Remark 3. The above derivation particularly applies to cases where the signal-to-noise ratio is fairly large (e.g., 100 or more). In practice, however, it is observed that the obtained rules of thumb deliver fairly reasonable estimates for even smaller values of SN (e.g., 10; see also Table 3).

Remark 4. For large values of the matrix dimension, it can be expected that the performance slightly declines, much like it has been observed that the number of necessary sweeps in a classical SVD procedure is slightly larger for large matrices (e.g., $n > 100$). In these cases, we can easily double or triple the number of rotations after each QR update accordingly. Strictly speaking, the computational complexity of the updating algorithm could then become $\mathcal{O}(n^{2+\alpha})$, where α is (much) smaller than 1.

The following example from systems theory illustrates the above results.

Example 3 (adaptive system identification). Suppose we are given a simple first-order time-varying system, with state space equations

$$\begin{aligned} x_{[k+1]} &= .8 \cdot \cos\left(\frac{2\pi}{2000}k\right) \cdot x_{[k]} + u_{[k]}, \\ y_{[k]} &= x_{[k]}, \end{aligned}$$

where $u_{[k]}$, $y_{[k]}$, and $x_{[k]}$ are the input, output, and state at time instant k. For every arbitrary input sequence, the output can be computed accordingly, by making use of the state space equations. Conversely, the state space model at a time instant k can (approximately) be computed from the input-output data, by making use of various *system identification* techniques. In [17], it has been shown how a state space model can be computed from the following exponentially weighted block Hankel matrix:

$$A_{[k]} = W_{[k]} \cdot \begin{bmatrix} z_{[1]} & z_{[2]} & \cdots & z_{[i]} \\ z_{[2]} & z_{[3]} & \cdots & z_{[i+1]} \\ z_{[3]} & z_{[4]} & \cdots & z_{[i+2]} \\ \vdots & \vdots & & \vdots \\ z_{[k-i]} & z_{[k-i+1]} & \cdots & z_{[k-1]} \\ z_{[k-i+1]} & z_{[k-i+2]} & \cdots & z_{[k]} \end{bmatrix},$$

$$z_{[k]} = \begin{bmatrix} u_{[k]} & y_{[k]} \end{bmatrix},$$

$$W_{[k]} = \operatorname{diag}\{\lambda^{k-i}, \lambda^{k-i-1}, \cdots, \lambda^1, \lambda^0\}.$$

For a good choice of λ, the "signal" rank (i.e., the number of singular values larger than the noise level) of this weighted block Hankel matrix equals $i+1$ (i is the number

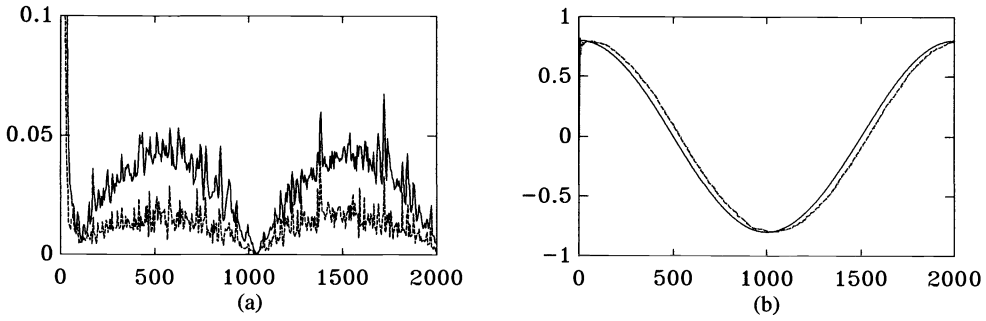


FIG. 1. (a) *Tracking error (lower curve) and time variation (upper curve) versus time* ($\lambda = 1 - 2^{-5}$). (b) *Original pole (solid line) and identified poles (dashed and dotted lines) versus time* ($\lambda = 1 - 2^{-5}$).

of columns with input data, 1 is the system order). The state space model at time instant k can then essentially be computed from the $(i+1)$-dimensional space $\mathcal{R}(Vs^{\circ}_{[k]})$ (see [17] for further details). So, first, the aim is to track the signal subspace of $A_{[k]}$, where in each time step a new row is appended. An adaptive system identification can be performed, by making use of either an exact SVD scheme (in other words, with a complete computation of the SVD at each time instance), or an adaptive SVD scheme with, e.g., $n - 1$ rotations after each QR update. The parameter i was set equal to 5, so that the matrix size equals 10, with a six-dimensional signal subspace.

Figure 1(b) shows the original system pole $0.8 \cdot \cos((2\pi/2000)k)$, solid line, together with the identified pole both for the exact scheme (dashed line) and the adaptive scheme (dotted line almost coinciding with the dashed line). The exponential weighting factor λ was set equal to $1 - 2^{-5}$. Figure 1(a) shows the time variation in $n = 10$ time steps $TV_{[k]\to[k+10]}$ (solid line), together with the tracking error at each time instant $TE_{[k]}$ (dashed line). Clearly, the latter generally remains smaller than the former, confirming the above rule of thumb. Finally, note that the time variation of the data nicely reflects the time variation of the underlying system, viz., the system pole.

In Fig. 2, the same quantities have been plotted for a different choice for the weighting factor $\lambda = 1 - 2^{-8}$. A different choice for λ is seen to hardly influence the time variation and the approximation error. As for the identified pole, the exponential weighting is seen to introduce a kind of time delay, which increases when λ approaches 1. The adaptive scheme, however, still delivers quite the same system pole as the exact scheme.

4. Error analysis. In the previous section, we analyzed the performance of the updating algorithm in infinite precision arithmetic, resulting in an upper bound for the distance between $Vs_{[k]}$ and $Vs^{\circ}_{[k]}$. In finite precision arithmetic, there are a few sources of additional error. Apart from round-off, of course, there is also the reorthogonalization scheme. Both processes change the V-matrix in an arbitrary manner with respect to the original data. As we are only interested in the right singular vectors, together with the singular values, we can define a relevant error matrix in this respect as follows:

$$\Delta_{[k]} = A^T_{[k]} \cdot A_{[k]} - (R_{[k]} \cdot V^T_{[k]})^T \cdot (R_{[k]} \cdot V^T_{[k]}).$$

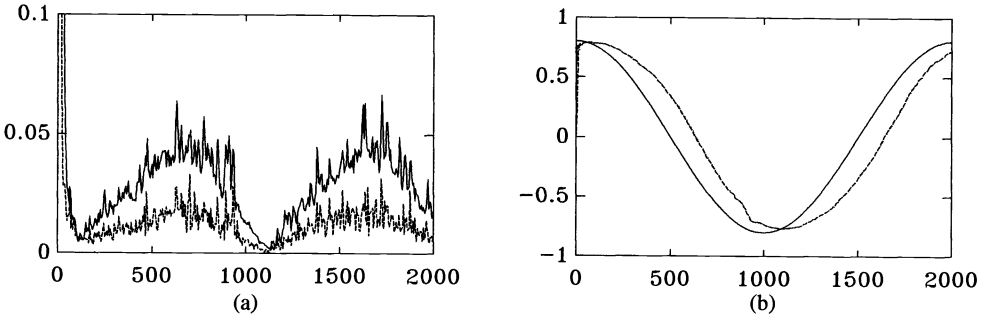


FIG. 2. (a) *Tracking error (lower curve) and time variation (upper curve) versus time* ($\lambda = 1 - 2^{-8}$). (b) *Original pole (solid line) and identified poles (dashed and dotted lines) versus time* ($\lambda = 1 - 2^{-8}$).

This error matrix tells how far the information stored in $R_{[k]} \cdot V_{[k]}^T$ has drifted off from the original data. If $\Delta_{[k]}$ is small, the singular values of $R_{[k]}$ will be close to those of $A_{[k]}$. Furthermore, for large values of SN, the signal subspace of $R_{[k]} \cdot V_{[k]}^T$ will then be close to the signal subspace of $A_{[k]}$.

In the sequel, upper bounds for the propagation of $\Delta_{[k]}$ are derived, resulting in a first-order difference equation

$$\Delta_{[k]} = \lambda^2 \cdot \Delta_{[k-1]} + \delta E_{[k]}$$

where λ is the exponential weighting factor ($\lambda < 1$). As long as $V_{[k]}$ is close to an orthogonal matrix, $\delta E_{[k]}$ contains only bounded local errors, independent of $\Delta_{[k-1]}$. In other words, the norm

$$\|\Delta I_{[k]}\|_F \stackrel{\text{def}}{=} \|V_{[k]} \cdot V_{[k]}^T - I\|_F$$

(where I is the identity matrix) should be kept small, in order to guarantee stability of the error propagation. Therefore, we first derive an estimate for the above norm by investigating the reorthogonalization scheme of §2. By making use of this, we then derive the above error propagation formula, showing that the overall updating procedure is stable.

4.1. An estimate for $\|V_{[k]} \cdot V_{[k]}^T - I\|_F$. In finite precision arithmetic both the updating and the reorthogonalization steps introduce new errors in the stored $V_{[k]}$-matrix. On the other hand, the reorthogonalization itself annihilates accumulated errors up to machine precision. If the number of reorthogonalizations in the updating scheme is chosen to be equal to the number of SVD rotations, one double sweep in the reorthogonalization procedure is performed after each n time steps. If we let $\Phi_{[\text{ac}]}$ and $T_{[\text{ac}]}$ denote the accumulated right and left transformations applied to V in time steps k through $k + n - 1$, we have

$$\begin{aligned} V_{[k+n-1]} &= \mathbf{fl}(T_{[\text{ac}]} \cdot V_{[k-1]} \cdot \Phi_{[\text{ac}]}) \\ &= T_{[\text{ac}]} \cdot V_{[k-1]} \cdot \Phi_{[\text{ac}]} + \delta V_{[\text{ac}]}, \end{aligned}$$

where $\delta V_{[\text{ac}]}$ is an input of local errors in these time steps, and $\mathbf{fl}(\cdot)$ refers to the computer result after a sequence of transformations (in the right order). For a first-order

analysis, the reorthogonalizations can be considered as orthogonal transformations, so that Gentleman's analysis [7] applies. The double sweep (with SVD steps and reorthogonalizations) then consists of $4n-4$ different "stages." Each such stage consists of the simultaneous application of disjoint transformations. This then results in an approximate upper bound for the local errors on V in one double sweep [7]:

$$\begin{aligned}\|\delta V_{[\mathrm{ac}]}\|_F &\leq (4n-4)\cdot k_V\cdot\epsilon\cdot(1+k_V\epsilon)^{4n-5}\cdot\|V\|_F\\ &\simeq 4n\cdot k_V\cdot\epsilon\cdot\|V\|_F,\end{aligned}$$

where ϵ is the largest number such that $\mathbf{fl}(1+\epsilon)=1$ (relative machine precision) and k_V is a constant depending on the specific implementation of both the Givens rotations and the reorthogonalizations.

From the formula for $V_{[k+n-1]}$, we can derive a formal description for the error build-up process as follows:

$$\begin{aligned}V_{[k+n-1]}\cdot V_{[k+n-1]}^T = {}& T_{[\mathrm{ac}]}\cdot V_{[k-1]}\cdot\Phi_{[\mathrm{ac}]}\cdot\Phi_{[\mathrm{ac}]}^T\cdot V_{[k-1]}^T\cdot T_{[\mathrm{ac}]}^T\\ &+T_{[\mathrm{ac}]}\cdot V_{[k-1]}\cdot\Phi_{[\mathrm{ac}]}\cdot\delta V_{[\mathrm{ac}]}^T\\ &+\delta V_{[\mathrm{ac}]}\cdot\Phi_{[\mathrm{ac}]}^T\cdot V_{[k-1]}^T\cdot T_{[\mathrm{ac}]}^T\\ &+\mathcal{O}(\epsilon^2),\end{aligned}$$

$$\underbrace{\|V_{[k+n-1]}\cdot V_{[k+n-1]}^T-I\|_F}_{\|\Delta I_{[k+n-1]}\|_F}\leq\underbrace{\|T_{[\mathrm{ac}]}\cdot V_{[k-1]}\cdot V_{[k-1]}^T\cdot T_{[\mathrm{ac}]}^T-I\|_F}_{\eta_{[\mathrm{ac}]}\cdot\|\Delta I_{[k-1]}\|_F}+2\cdot\|\delta V_{[\mathrm{ac}]}^T\|_F+\mathcal{O}(\epsilon^2),$$

where $\eta_{[\mathrm{ac}]}$ is a factor describing the effect of the reorthogonalization steps.

As long as $\|\Delta I_{[k-1]}\|_F$ is small ($<\epsilon^{1/4}$), it is reduced to machine precision by the double sweep in the reorthogonalization scheme. The input of local errors, however, interferes with this reorthogonalization, and this introduces additional errors of the same order of magnitude. In conclusion, we end up with

$$\begin{aligned}\|\Delta I_{[k+n-1]}\|_F &\leq \underbrace{\eta_{[\mathrm{ac}]}\cdot\|\Delta I_{[k-1]}\|_F}_{\mathcal{O}(n\epsilon)}+2\cdot\|\delta V_{[\mathrm{ac}]}\|_F+\mathcal{O}(\epsilon^2)\\ &\simeq\mathcal{O}(n\epsilon)+8n\cdot k_V\cdot\epsilon\cdot\|V\|_F\\ &\simeq\mathcal{O}(n\epsilon)+8n\sqrt{n}\cdot k_V\cdot\epsilon\\ &\simeq k_I\cdot n\sqrt{n}\cdot\epsilon,\end{aligned}$$

which is then a bound for $\|\Delta I_{[k]}\|_F$ for all values of k. The reorthogonalization procedure thus keeps the stored V-matrix close to an orthogonal one.

4.2. Error propagation formulas. First of all, for the sake of conciseness, let us assume that there exists an upper bound $\|R\|_F$ such that for all k,

$$\begin{aligned}\|R_{[k]}\|_F &\leq\sqrt{n}\|R_{[k]}\|_2\leq\|R\|_F,\\ \|\tilde{R}_{[k]}\|_F &\leq\sqrt{n}\|\tilde{R}_{[k]}\|_2\leq\|R\|_F,\\ \sqrt{n}\|a_{[k]}\|_2 &\leq\|R\|_F.\end{aligned}$$

The error matrix at time step k, viz. $\Delta_{[k]}$, can then be computed from the error matrix $\Delta_{[k-1]}$ at time step $k-1$ as follows.

In a *first step*, appending a new row together with an exponential weighting (weighting factor λ) can be described as

$$\begin{bmatrix} \lambda \cdot R_{[k-1]} \cdot V_{[k-1]}^T \\ a_{[k]}^T \end{bmatrix} = \begin{bmatrix} \lambda \cdot R_{[k-1]} \\ a_{[k]}^T \cdot V_{[k-1]} \end{bmatrix} \cdot V_{[k-1]}^T + \underbrace{\begin{bmatrix} 0 \\ -a_{[k]}^T \cdot \Delta I_{[k-1]} \end{bmatrix}}_{\delta E_{[k]}^1},$$

where $\delta E_{[k]}^1$ is an *"algorithmic"* error due to $V_{[k-1]}$ not being orthogonal. Here we can use the upper bound for ΔI of the previous section:

$$\|\delta E_{[k]}^1\|_2 \leq \|\Delta I_{[k-1]}\|_2 \cdot \|a_{[k]}\|_2 \leq k_I n\sqrt{n}\epsilon \cdot \|a_{[k]}\|_2 \leq k_I n\epsilon \cdot \|R\|_F.$$

In a *second step*, additional round-off errors are introduced by the explicit computation of

$$\begin{aligned} \tilde{R}_{[k-1]} &= \mathbf{fl}\{\lambda \cdot R_{[k-1]}\} = \lambda \cdot R_{[k-1]} + \delta\tilde{R}_{[k-1]}, \\ \tilde{a}_{[k]}^T &= \mathbf{fl}\{a_{[k]}^T \cdot V_{[k-1]}\} = a_{[k]}^T \cdot V_{[k-1]} + \delta\tilde{a}_{[k]}^T. \end{aligned}$$

Substituting this in the above equation, we obtain

$$\begin{aligned} &\begin{bmatrix} \lambda \cdot R_{[k-1]} \cdot V_{[k-1]}^T \\ a_{[k]}^T \end{bmatrix} \\ &= \begin{bmatrix} \tilde{R}_{[k-1]} \\ \tilde{a}_{[k]}^T \end{bmatrix} \cdot V_{[k-1]}^T + \delta E_{[k]}^1 + \underbrace{\begin{bmatrix} -\delta\tilde{R}_{[k-1]} \\ 0 \end{bmatrix} \cdot V_{[k-1]}^T}_{\delta E_{[k]}^2} + \underbrace{\begin{bmatrix} 0 \\ -\delta\tilde{a}_{[k]}^T \end{bmatrix} \cdot V_{[k-1]}^T}_{\delta E_{[k]}^3} \end{aligned}$$

with first-order upper bounds (see [25])

$$\begin{aligned} &\|\delta E_{[k]}^2\|_2 \leq \|\delta E_{[k]}^2\|_F \leq \epsilon \cdot \|\tilde{R}_{[k-1]}\|_F \leq \epsilon \cdot \|R\|_F, \\ &\|\delta E_{[k]}^3\|_2 \leq n\epsilon \cdot \|a_{[k]}\|_2 \cdot \|V_{[k-1]}\|_F \leq n\sqrt{n}\epsilon \cdot \|a_{[k]}\|_2 + \mathcal{O}(\epsilon^2) \simeq n\epsilon\|R\|_F. \end{aligned}$$

In a *third step*, orthogonal row transformations are performed for the QR update, introducing a round-off error $\delta\hat{R}_{[k]}$. The rotation angles are chosen such that the last row of the computer result transforms to zero:

$$\begin{bmatrix} \hat{R}_{[k]} \\ 0 \end{bmatrix} = \mathbf{fl}\left\{Q_{[k]}^T \cdot \begin{bmatrix} \tilde{R}_{[k-1]} \\ \tilde{a}_{[k]}^T \end{bmatrix}\right\} = Q_{[k]}^T \cdot \begin{bmatrix} \tilde{R}_{[k]} \\ \tilde{a}_{[k]}^T \end{bmatrix} + \delta\hat{R}_{[k]}.$$

By making use of this, we obtain

$$\begin{aligned} \begin{bmatrix} \lambda \cdot R_{[k-1]} \cdot V_{[k-1]}^T \\ a_{[k]}^T \end{bmatrix} &= Q_{[k]} \cdot \left(Q_{[k]}^T \cdot \begin{bmatrix} \tilde{R}_{[k-1]} \\ \tilde{a}_{[k]}^T \end{bmatrix}\right) \cdot V_{[k-1]}^T + \delta E_{[k]}^1 + \delta E_{[k]}^2 + \delta E_{[k]}^3 \\ &= Q_{[k]} \cdot \begin{bmatrix} \hat{R}_{[k]} \\ 0 \end{bmatrix} \cdot V_{[k-1]}^T + \delta E_{[k]}^1 + \delta E_{[k]}^2 + \delta E_{[k]}^3 \\ &\quad \underbrace{-Q_{[k]} \cdot \delta\hat{R}_{[k]} \cdot V_{[k-1]}^T}_{\delta E_{[k]}^4} \cdot \end{aligned}$$

Following the error analysis in [7],

$$\|\delta E^4_{[k]}\|_2 \le \|\delta E^4_{[k]}\|_F \le k_G \boldsymbol{\epsilon} \cdot n(1 + k_G \boldsymbol{\epsilon})^{n-1} \cdot \|\hat{R}_{[k]}\|_F \simeq k_G \boldsymbol{\epsilon} \cdot n \cdot \|R\|_F,$$

where k_G is a constant (depending on the specific implementation of the Givens rotations), and n is the number of different stages for one single QR update.

Finally, in a *fourth step* (SVD steps + reorthogonalizations), a sequence of left and right transformations is applied. If we summarize these into left transformations $\Theta_{[k]}$ and $T_{[k]}$, and a right transformation $\Phi_{[k]}$, we can proceed as follows (introducing round-off errors $\delta R_{[k]}$ and $\delta V_{[k]}$):

$$\begin{aligned} R_{[k]} &= \mathbf{fl}\{\Theta^T_{[k]} \cdot \hat{R}_{[k]} \cdot \Phi_{[k]}\} = \Theta^T_{[k]} \cdot \hat{R}_{[k]} \cdot \Phi_{[k]} + \delta R_{[k]}, \\ V_{[k]} &= \mathbf{fl}\{T_{[k]} \cdot V_{[k-1]} \cdot \Phi_{[k]}\} = T_{[k]} \cdot V_{[k-1]} \cdot \Phi_{[k]} + \delta V_{[k]} \\ &= V_{[k-1]} \cdot \Phi_{[k]} + (T_{[k]} - I) \cdot V_{[k-1]} \cdot \Phi_{[k]} + \delta V_{[k]} \end{aligned}$$

and

$$\begin{aligned} \begin{bmatrix} \lambda \cdot R_{[k-1]} \cdot V^T_{[k-1]} \\ a^T_{[k]} \end{bmatrix} &= Q_{[k]} \cdot \begin{bmatrix} \Theta_{[k]} \cdot (\Theta^T_{[k]} \cdot \hat{R}_{[k]} \cdot \Phi_{[k]}) \\ 0 \end{bmatrix} \cdot (\Phi^T_{[k]} \cdot V^T_{[k-1]}) \\ &\quad + \delta E^1_{[k]} + \delta E^2_{[k]} + \delta E^3_{[k]} + \delta E^4_{[k]} \\ &= Q_{[k]} \cdot \left(\begin{bmatrix} \Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} + \begin{bmatrix} -\Theta_{[k]} \cdot \delta R_{[k]} \\ 0 \end{bmatrix} \right) \\ &\quad \cdot (V^T_{[k]} - \delta V^T_{[k]} - \Phi^T_{[k]} V^T_{[k-1]} (T_{[k]} - I)^T) \\ &\quad + \delta E^1_{[k]} + \delta E^2_{[k]} + \delta E^3_{[k]} + \delta E^4_{[k]} \\ &\simeq Q_{[k]} \cdot \begin{bmatrix} \Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} \cdot V^T_{[k]} \\ &\quad + \delta E^1_{[k]} + \delta E^2_{[k]} + \delta E^3_{[k]} + \delta E^4_{[k]} \\ &\quad + \underbrace{Q_{[k]} \cdot \begin{bmatrix} -\Theta_{[k]} \cdot \delta R_{[k]} \\ 0 \end{bmatrix} \cdot V^T_{[k]}}_{\delta E^5_{[k]}} \\ &\quad + \underbrace{Q_{[k]} \cdot \begin{bmatrix} -\Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} \cdot \delta V^T_{[k]}}_{\delta E^6_{[k]}} \\ &\quad + \underbrace{Q_{[k]} \cdot \begin{bmatrix} -\Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} \cdot \Phi^T_{[k]} \cdot V^T_{[k-1]} \cdot (T_{[k]} - I)^T}_{\delta E^7_{[k]}} \\ &\quad + \mathcal{O}(\boldsymbol{\epsilon}^2). \end{aligned}$$

Again applying the error analysis in [7] results in

$$\begin{aligned} \|\delta E^5_{[k]}\|_2 &\le \|\delta E^5_{[k]}\|_F \\ &\simeq k_G \boldsymbol{\epsilon} \cdot (2n - 2)(1 + k_G \boldsymbol{\epsilon})^{2n-3} \cdot \|\hat{R}_{[k]}\|_F \end{aligned}$$

$$\begin{aligned}
&\simeq k_G\epsilon \cdot (2n-2)\|R\|_F, \\
\|\delta E^6_{[k]}\|_2 &\leq \|\delta E^6_{[k]}\|_F \\
&\simeq k_G\epsilon \cdot (2n-2)(1+k_G\epsilon)^{2n-3} \cdot \|V_{[k]}\|_F \cdot \|\hat{R}_{[k]}\|_2 \\
&\simeq k_G\epsilon \cdot (2n-2)\sqrt{n} \cdot \|\hat{R}_{[k]}\|_2 \\
&\simeq k_G\epsilon \cdot (2n-2)\|R\|_F.
\end{aligned}$$

The number of stages in the upper bounds for $\delta E^5_{[k]}$ and $\delta E^6_{[k]}$ equals $2n-2$, as $n-1$ rotations are applied both to the left and to the right.

As for $\delta E^7_{[k]}$, we can estimate an upper bound as follows:

$$\|\delta E^7_{[k]}\|_2 \leq \|T_{[k]} - I\|_F \cdot \|R_{[k]}\|_2.$$

Note that $\delta E^7_{[k]}$ represents an additional error which is introduced by the reorthogonalization scheme. The reorthogonalization indeed changes the V-matrix in an arbitrary manner with respect to the original data, and therefore contributes to $\Delta_{[k]}$. We can easily check that $\|T_{[k]} - I\|_F$ must have an upper bound similar to $\|V_{[k]}V^T_{[k]} - I\|_F$, so that finally

$$\begin{aligned}
\|\delta E^7_{[k]}\|_2 &\leq k_T \cdot n\sqrt{n}\epsilon \cdot \|R_{[k]}\|_2 \\
&\leq k_T \cdot n\epsilon\|R\|_F.
\end{aligned}$$

Adding all the above upper bounds, we obtain

$$\begin{bmatrix} \lambda \cdot R_{[k-1]} \cdot V^T_{[k-1]} \\ a^T_{[k]} \end{bmatrix} = Q_{[k]} \cdot \begin{bmatrix} \Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} \cdot V^T_{[k]} + \underbrace{\delta E^1_{[k]} + \delta E^2_{[k]} + \delta E^3_{[k]} + \delta E^4_{[k]} + \delta E^5_{[k]} + \delta E^6_{[k]} + \delta E^7_{[k]}}_{\delta E^{1\to 7}_{[k]}}$$

with

$$\|\delta E^{1\to 7}_{[k]}\|_2 \leq (k_1 \cdot n + k_2)\epsilon \cdot \|R\|_F.$$

Multiplying the left- and right-hand sides with their transpose now results in

$$\lambda^2 \cdot (R_{[k-1]} \cdot V^T_{[k-1]})^T \cdot (R_{[k-1]} \cdot V^T_{[k-1]}) + a_{[k]} \cdot a^T_{[k]} = (R_{[k]} \cdot V^T_{[k]})^T \cdot (R_{[k]} \cdot V^T_{[k]}) + \delta E_{[k]},$$

where

$$\begin{aligned}
\delta E_{[k]} = (\delta E^{1\to 7}_{[k]})^T &\cdot \left(Q_{[k]} \cdot \begin{bmatrix} \Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} \cdot V^T_{[k]} \right) \\
&+ \left(Q_{[k]} \cdot \begin{bmatrix} \Theta_{[k]} \cdot R_{[k]} \\ 0 \end{bmatrix} \cdot V^T_{[k]} \right)^T \cdot (\delta E^{1\to 7}_{[k]}) + \mathcal{O}(\epsilon^2)
\end{aligned}$$

with an upper bound

$$\begin{aligned}
\|\delta E_{[k]}\|_F &\simeq 2\|R_{[k]}\|_F \cdot \|\delta E^{1\to 7}_{[k]}\|_2 \\
&\simeq 2(k_1 \cdot n + k_2)\epsilon \cdot \|R\|^2_F.
\end{aligned}$$

Substituting the definition of $\Delta_{[\cdot]}$ results in

$$\lambda^2 \cdot A_{[k-1]}^T \cdot A_{[k-1]} - \lambda^2 \cdot \Delta_{[k-1]} + a_{[k]} \cdot a_{[k]}^T = (R_{[k]} \cdot V_{[k]}^T)^T \cdot (R_{[k]} \cdot V_{[k]}^T) + \delta E_{[k]},$$
$$A_{[k]}^T \cdot A_{[k]} - \lambda^2 \cdot \Delta_{[k-1]} = (R_{[k]} \cdot V_{[k]}^T)^T \cdot (R_{[k]} \cdot V_{[k]}^T) + \delta E_{[k]},$$

and finally

$$\Delta_{[k]} = \lambda^2 \cdot \Delta_{[k-1]} + \delta E_{[k]},$$

or if we use norms,

$$\begin{aligned} \|\Delta_{[k]}\|_F &\leq \lambda^2 \cdot \|\Delta_{[k-1]}\|_F + \|\delta E_{[k]}\|_F \\ &\leq \lambda^2 \cdot \|\Delta_{[k-1]}\|_F + 2(k_1 \cdot n + k_2)\epsilon \cdot \|R\|_F^2. \end{aligned}$$

The first term, $\lambda^2 \cdot \Delta_{[k-1]}$, represents the error propagation, which is stable as $\lambda < 1$. The second term is an upper bound for local errors. If we assume that the weighting factor is constant, we finally obtain (for all values of k)

$$\|\Delta_{[k]}\|_F \leq \frac{2(k_1 \cdot n + k_2)}{1 - \lambda^2} \cdot \|R\|_F^2 \cdot \epsilon,$$

or alternatively (a kind of relative error formulation),

$$\frac{\|(R_{[k]} \cdot V_{[k]}^T)^T \cdot (R_{[k]} \cdot V_{[k]}^T) - (R_{[k]} \cdot V_{[k]}^T)^T \cdot (R_{[k]} \cdot V_{[k]}^T)\|_F}{\|R\|_F^2} \leq \frac{2(k_1 \cdot n + k_2)}{1 - \lambda^2} \cdot \epsilon.$$

In conclusion, the overall SVD updating scheme is found to be stable, if an exponential weighting is applied, with weighting factor $\lambda < 1$, and if a reorthogonalization procedure is included, which keeps the stored V-matrix close to orthogonal. The obtained upper bound for the error $\|\Delta_{[k]}\|_F$ is clearly overly conservative, as it consists of an accumulation of several worst-case upper bounds.

5. Conclusion. An SVD updating procedure was constructed as a combination of QR updating and a Jacobi-type SVD algorithm applied to a triangular matrix. As for subspace tracking problems, it was shown how only very few SVD steps after each QR updating can restore an acceptable approximation. Furthermore, the updating is shown to be stable when supplemented with a Jacobi-type reorthogonalization scheme. A systolic array for this updating algorithm is developed in [18].

Appendix A. Let us assume that the initial configuration is as follows (we consider a small 6×6 example, from which the results for the general case obviously follow):

$$R_{[k]} = \begin{bmatrix} Rs_{[k]} & Rsn_{[k]} \\ 0 & Rn_{[k]} \end{bmatrix}$$
$$= \begin{bmatrix} r_{11}^s & r_{12}^s & r_{13}^s & \boxed{\varepsilon_{14}} & \boxed{\varepsilon_{15}} & \boxed{\varepsilon_{16}} \\ & r_{22}^s & r_{23}^s & \boxed{\varepsilon_{24}} & \boxed{\varepsilon_{25}} & \boxed{\varepsilon_{26}} \\ & & r_{33}^s & \boxed{\varepsilon_{34}} & \boxed{\varepsilon_{35}} & \boxed{\varepsilon_{36}} \\ & & & r_{44}^n & r_{45}^n & r_{46}^n \\ & & & & r_{55}^n & r_{56}^n \\ & & & & & r_{66}^n \end{bmatrix}.$$

Time indices are omitted for brevity. We assume that

$$\varepsilon \ll \delta,$$

where

$$\varepsilon = \|Rsn_{[k]}\|_F$$

is the Frobenius norm of the matrix with cross terms, and

$$\delta = \sigma_{\min}\{Rs_{[k]}\} - \sigma_{\max}\{Rn_{[k]}\}$$

is the gap between the singular values of $Rs_{[k]}$ and $Rn_{[k]}$. As these are then known to be (ε^2/δ)-close to the singular values of R [3], we end up with

$$\delta \simeq \sigma_{\min}\{Rs_{[k]}\} \quad \text{for } SN \gg 1.$$

The aim is to investigate the effect on ε of one cyclic-by-rows sweep in Kogbetliantz's SVD algorithm (modified for triangular matrices). This essentially consists of a number of 2×2 SVDs on the main diagonal, where the pivot index takes up the values (see [15] and [23] for details)

$$\begin{aligned} i = {} & 1,2,3,4,5 \\ & 1,2,3,4 \\ & 1,2,3 \\ & 1,2 \\ & 1. \end{aligned}$$

As a reminder, each 2×2 SVD can be described as

$$\begin{bmatrix} r_{i,i} & 0 \\ 0 & r_{i+1,i+1} \end{bmatrix} \Leftarrow \begin{bmatrix} \sin\theta & \cos\theta \\ \cos\theta & -\sin\theta \end{bmatrix} \begin{bmatrix} r_{i,i} & r_{i,i+1} \\ 0 & r_{i+1,i+1} \end{bmatrix} \begin{bmatrix} \sin\phi & \cos\phi \\ \cos\phi & -\sin\phi \end{bmatrix},$$

where

$$\begin{aligned} \tan 2\theta &= \frac{2r_{i+1,i+1} \cdot r_{i,i+1}}{r_{i,i}^2 - r_{i+1,i+1}^2 + r_{i,i+1}^2}, \\ \tan\phi &= \frac{r_{i+1,i+1} \cdot \tan\theta + r_{i,i+1}}{r_{i,i}} \end{aligned}$$

(for the sake of clarity, we prefer to use inner rotations + permutations, instead of outer rotations).

First of all, we can slightly reorder the 2×2 transformations as follows:

$$\begin{aligned} \text{Part (a)} &\begin{cases} i = 1,2 \\ \quad\;\; 1, \end{cases} \\ \text{Part (b)} &\begin{cases} i = .,.,3,4,5 \\ \quad\;\; .,2,3,4 \\ \quad\;\; 1,2,3, \end{cases} \\ \text{Part (c)} &\begin{cases} i = 1,2 \\ \quad\;\; 1. \end{cases} \end{aligned}$$

Part (a) corresponds to rotations within the (approximate) signal subspace, reducing the off-norm in $Rs_{[k]}$. Both ε and δ remain unchanged, so that for the time being, these transformations are irrelevant.

Part (b) corresponds to annihilations of cross terms, and needs further investigation. Referring to the initial configuration (which is basically not changed in Part (a)), let us first remark that the diagonal entries in $Rs_{[k]}$ satisfy

$$r_{ii}^{s} \geq \sigma_{\min}\{Rs_{[k]}\}$$

(as $Rs_{[k]}$ is triangular), while on the other hand the diagonal entries in $Rn_{[k]}$ obviously satisfy

$$r_{ii}^{n} \leq \sigma_{\max}\{Rn_{[k]}\}.$$

We easily verify that the above upper and lower bounds remain valid throughout the computations in Part (b) (r_{ii}^{s} elements always increase; r_{ii}^{n} elements decrease).

The *first series of transformations*, where the pivot index takes up the values

$$i = .,.,3,4,5$$

turns the initial configuration into

$$\begin{bmatrix} r_{11}^{s} & r_{12}^{s} & \boxed{\varepsilon_{13}} & \boxed{\varepsilon_{14}} & \boxed{\varepsilon_{15}} & r_{16}^{s} \\ & r_{22}^{s} & \boxed{\varepsilon_{23}} & \boxed{\varepsilon_{24}} & \boxed{\varepsilon_{25}} & r_{26}^{s} \\ & & & & & \boxed{\varepsilon_{36}^{\star}} \\ & & r_{33}^{n} & r_{34}^{n} & r_{35}^{n} & \boxed{\varepsilon_{46}^{\star}} \\ & & & r_{44}^{n} & r_{45}^{n} & \\ & & & & r_{55}^{n} & \boxed{\varepsilon_{56}^{\star}} \\ & & & & & \\ & & & & & r_{66}^{s} \end{bmatrix}$$

(iteration indices are left out for the sake of clarity; subscripts ij refer to row and column numberings in the full R-matrix).

When $i = 3$, the pivot element ε_{34} is ε-small, from which we can estimate the rotation angles as follows:

$$\sin\theta \simeq \tan\theta \simeq \frac{\sigma_{\max}\{Rn_{[k]}\}}{\sigma_{\min}^{2}\{Rs_{[k]}\}} \cdot \mathcal{O}(\varepsilon),$$

$$\sin\phi \simeq \tan\phi \simeq \frac{1}{\sigma_{\min}\{Rs_{[k]}\}} \cdot \mathcal{O}(\varepsilon).$$

From the fact that θ (row transformation) is particularly small, it follows that the pivot for $i = 4$ is ε-small as well:

$$\begin{aligned} \varepsilon_{35} \cdot \cos\theta - r_{45}^{n} \cdot \sin\theta &\simeq \mathcal{O}(\varepsilon) + \sigma_{\max}\{Rn_{[k]}\} \cdot \frac{\sigma_{\max}\{Rn_{[k]}\}}{\sigma_{\min}^{2}\{Rs_{[k]}\}} \cdot \mathcal{O}(\varepsilon) \\ &\simeq \mathcal{O}(\varepsilon). \end{aligned}$$

Finally, from a similar reasoning it follows that the pivot for $i = 5$ is also ε-small. The estimates for $\tan\theta$ and $\tan\phi$ therefore hold for $i = 3$ as well as for $i = 4, 5$.

From the estimates for the rotation angles, we can estimate the magnitude of r_{ij}^{n}, $\varepsilon_{ij}^{\star}$, ε_{ij} after the first series of transformations.

Elements r_{ij}^n remain $\sigma_{\max}\{Rn_{[k]}\}$-small, which follows from

$$\begin{aligned} r_{ij}^n \cdot \cos\theta + \varepsilon_{kj}^n \cdot \sin\theta &\simeq \mathcal{O}(\sigma_{\max}\{Rn_{[k]}\}) + \frac{\sigma_{\max}\{Rn_{[k]}\}}{\sigma_{\min}^2\{Rs_{[k]}\}} \cdot \mathcal{O}(\varepsilon^2) \\ &\simeq \mathcal{O}(\sigma_{\max}\{Rn_{[k]}\}). \end{aligned}$$

Cross terms $\varepsilon_{ij}^\star$ are (ε/SN)-small, which follows from

$$\begin{aligned} \varepsilon_{ij}^\star &\simeq \sum_k r_{ik}^n \cdot \sin\phi \\ &\simeq \frac{\sigma_{\max}\{Rn_{[k]}\}}{\sigma_{\min}\{Rs_{[k]}\}} \cdot \mathcal{O}(\varepsilon) \\ &\simeq \mathcal{O}\left(\frac{\varepsilon}{SN}\right). \end{aligned}$$

Cross terms ε_{ij} remain ε-small, which follows from

$$\begin{aligned} \varepsilon_{ij} &\simeq \varepsilon_{ij} + \sum_k r_{ik}^s \cdot \sin\phi \\ &\simeq \mathcal{O}(\varepsilon) + \frac{\mathcal{O}(\|Rs_{[k]}\|_{\text{off}})}{\sigma_{\min}\{Rs_{[k]}\}} \cdot \mathcal{O}(\varepsilon) \\ &\simeq \mathcal{O}(\varepsilon), \end{aligned}$$

where we made an opportunist (but mostly fair) assumption, namely, that $r_{ik}^s = \mathcal{O}(\|Rs_{[k]}\|_{\text{off}}) \leq \sigma_{\min}\{Rs_{[k]}\}$. This corresponds to a certain degree of convergence within $Rs_{[k]}$, brought about by Part (a).

We can now repeat this reasoning for the *second and the third series of transformations* in Part (b)

$$\begin{aligned} i = .,&2,3,4 \\ &1,2,3, \end{aligned}$$

in turn transforming the triangular factor into

$$\begin{bmatrix} r_{11}^s & \boxed{\varepsilon_{12}} & \boxed{\varepsilon_{13}} & \boxed{\varepsilon_{14}} & r_{15}^s & r_{16}^s \\ & r_{22}^n & r_{23}^n & r_{24}^n & \boxed{\varepsilon_{25}^\star} & \boxed{\varepsilon_{26}^\star} \\ & & r_{33}^n & r_{34}^n & \boxed{\varepsilon_{35}^\star} & \boxed{\varepsilon_{36}^\star} \\ & & & r_{44}^n & \boxed{\varepsilon_{45}^\star} & \boxed{\varepsilon_{46}^\star} \\ & & & & r_{55}^s & r_{56}^s \\ & & & & & r_{66}^s \end{bmatrix}$$

and

$$\begin{bmatrix} r_{11}^n & r_{12}^n & r_{13}^n & \boxed{\varepsilon_{14}^\star} & \boxed{\varepsilon_{15}^\star} & \boxed{\varepsilon_{16}^\star} \\ & r_{22}^n & r_{23}^n & \boxed{\varepsilon_{24}^\star} & \boxed{\varepsilon_{25}^\star} & \boxed{\varepsilon_{26}^\star} \\ & & r_{33}^n & \boxed{\varepsilon_{34}^\star} & \boxed{\varepsilon_{35}^\star} & \boxed{\varepsilon_{36}^\star} \\ & & & r_{44}^s & r_{45}^s & r_{46}^s \\ & & & & r_{55}^s & r_{56}^s \\ & & & & & r_{66}^s \end{bmatrix},$$

where all cross terms $\varepsilon_{ij}^{\star}$ are seen to remain (ε/SN)-small.

Finally, subsequent transformations in Part (c) do not alter the norm of the submatrix with cross terms.

As a main conclusion, we can state that the matrix with cross terms is (ε/SN)-small after the forward sweep. The backward sweep, returning the triangular matrix to the original configuration, again reduces this norm by a factor $1/SN$ (where this time the column transformations are particularly small). A double sweep thus corresponds to a reduction by a factor $(1/SN)^2$.

REFERENCES

[1] J. R. Bunch and C. P. Nielsen, *Updating the singular value decomposition*, Numer. Math., 31 (1978), pp. 111–129.

[2] J. R. Bunch, C. P. Nielsen, and D. C. Sorensen, *Rank one modification of the symmetric eigenproblem*, Numer. Math., 31 (1978), pp. 31–48.

[3] J. P. Charlier and P. van Dooren, *On Kogbetliantz's* SVD *algorithm in the presence of clusters*, Linear Algebra Appl., 95 (1987), pp. 135–160.

[4] P. Comon and G. H. Golub, *Tracking a few extreme singular values and vectors in signal processing*, Proc. IEEE, 78 (1990), pp. 1327–1343.

[5] K. V. Fernando, *Linear convergence of the cyclic Jacobi and Kogbetliantz methods*, Numer. Math., 56 (1989), pp. 73–91.

[6] G. E. Forsythe and P. Henrici, *The cyclic Jacobi method for computing the principal values of a complex matrix*, Trans. Amer. Math. Soc., 94 (1960), pp. 1–23.

[7] W. M. Gentleman, *Error analysis of* QR *decompositions by Givens transformations*, Linear Algebra Appl., 10 (1975), pp. 189–197.

[8] P. E. Gill, G. H. Golub, W. Murray, and M. A. Saunders, *Methods for modifying matrix factorizations*, Math. Comp., 28 (1974), pp. 505–535.

[9] G. H. Golub, *Some modified eigenvalue problems*, SIAM Rev., 15 (1973), pp. 318–334.

[10] G. H. Golub and C. F. Van Loan, *Matrix computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[11] V. Hari and K. Veselić, *On Jacobi methods for singular value decompositions*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 741–754.

[12] M. T. Heath, A. J. Laub, C. C. Paige, and R. C. Ward, *Computing the singular value decomposition of a product of two matrices*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 1147–1159.

[13] P. Henrici and K. Zimmermann, *An estimate for the norms of certain cyclic Jacobi operators*, Linear Algebra Appl., 1 (1968), pp. 489–501.

[14] E. Kogbetliantz, *Solution of linear equations by diagonalization of coefficient matrices*, Quart. Appl. Math., 13 (1955), pp. 123–132.

[15] F. T. Luk, *A triangular processor array for computing singular values*, Linear Algebra Appl., 77 (1986), pp. 259–273.

[16] F. T. Luk and H. Park, *On the equivalence and convergence of parallel Jacobi* SVD *algorithms*, Proc. SPIE, Vol. 826, Advanced Algorithms and Architectures for Signal Processing II, F. T. Luk, ed., 1987, pp. 152–159.

[17] M. Moonen, B. De Moor, L. Vandenberghe, and J. Vandewalle, *On- and off-line identification of linear state space models*, Internat. J. Control, 49 (1989), pp. 219–232.

[18] M. Moonen, P. van Dooren, and J. Vandewalle, *A systolic array for* SVD *updating*, ESAT-SISTA Report 1989-13b, Department of Electrical Engineering, Katholieke Universiteit Leuven, Belgium, SIAM J. Matrix Anal. Appl., 14 (1993), to appear.

[19] C. C. Paige and P. van Dooren, *On the convergence of Kogbetliantz's algorithm for computing the singular value decomposition*, Linear Algebra Appl., 77 (1986), pp. 301–313.

[20] R. Schreiber, *Implementation of adaptive array algorithms*, IEEE Trans. Acoust. Speech Signal Process., 34 (1986), pp. 1038–1045.

[21] J. M. Speiser, *Signal processing computational needs*, Proc. SPIE, Vol. 696, Advanced Algorithms and Architectures for Signal Processing, J. M. Speiser, ed., 1986, pp. 2–6.

[22] G. W. Stewart, *Error and perturbation bounds for subspaces associated with certain eigenvalue problems*, SIAM Rev., 15 (1973), pp. 727–764.

[23] G. W. STEWART, *A Jacobi-like algorithm for computing the Schur decomposition of a nonhermitian matrix*, SIAM J. Sci. Statist. Comput., 6 (1985), pp. 853–863.
[24] J. H. WILKINSON, *Note on the quadratic convergence of the cyclic Jacobi process*, Numer. Math., 4 (1962), pp. 296–300.
[25] ———*The Algebraic Eigenvalue Problem*, Oxford University Press, London, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1039–1053, October 1992

003

ALGORITHMS FOR MINIMUM TRACE FACTOR ANALYSIS*

G. A. WATSON†

Abstract. Minimum trace factor analysis is a commonly used technique for providing the greatest lower bound to reliability, and a modification of the basic problem involves the maximization of this greatest lower bound with respect to suitably chosen weights. The underlying mathematical problems can be expressed as optimization problems with eigenvalue constraints, and it is well known that these can be nondifferentiable in the presence of multiple eigenvalues. In this paper, some recent developments in methods for working with constraints of this kind are exploited to provide methods which are second-order independent of the eigenvalue multiplicities. The effectiveness of the algorithms is demonstrated on some test problems.

Key words. minimum trace factor analysis, convex programming, eigenvalue constraints, nondifferentiable optimization

AMS(MOS) subject classifications. 65K10, 90C25, 62H25

1. Introduction. Let S be a symmetric positive definite $n \times n$ matrix, and consider the problem of finding a diagonal matrix D to minimize the trace of $S - D$, subject to the condition that $S - D$ is positive semidefinite. The problem corresponds to the minimization of ρ, where

$$\rho = 1 - \frac{\mathbf{e}^T D \mathbf{e}}{\mathbf{e}^T S \mathbf{e}}, \tag{1.1}$$

with $\mathbf{e} = (1, 1, \cdots, 1)^T$. This problem arises in minimum trace factor analysis, and gives the greatest lower bound to reliability in statistical analysis, as measured by the size of ρ. It provides an alternative to the problem in canonical factor analysis of minimizing the rank of $S - D$ (see, for example, Ledermann [7] and Bentler [1]). Negative entries in D cannot be excluded without the addition of nonnegativity constraints, and this modified problem has been considered by a number of authors (see, for example, Jackson and Agunwamba [6], Woodhouse and Jackson [16], Bentler and Woodward [2], and Shapiro [13]); it is referred to as the problem of constrained minimum trace factor analysis. The expression (1.1) may be interpreted as using a vector $\mathbf{e}$ of *unit weights*, and a generalization involves the replacement of $\mathbf{e}$ by $\mathbf{w} \in \mathbb{R}^n$, so that ρ becomes

$$\rho = 1 - \frac{\mathbf{w}^T D \mathbf{w}}{\mathbf{w}^T S \mathbf{w}}. \tag{1.2}$$

Alternatively, trace $W(S - D)W$ is being minimized over appropriate D, where W is the diagonal matrix with the components of $\mathbf{w}$ on the diagonal. When the diagonal elements of D are constrained to be nonnegative, this is referred to as the constrained minimum trace problem with weight vector $\mathbf{w}$.

A difficulty with the above approach is that it is scale dependent, and this has led to the idea of *maximizing* the greatest lower bound to reliability as a *function of the weights*. Following Shapiro [13], define $L(S)$ by

$$L(S) = \{\mathbf{d} \in \mathbb{R}^n, S - D \geqq 0, D \geqq 0\},$$

where $\mathbf{d} = (d_1, \cdots, d_n)^T$, $D = \operatorname{diag}\{d_1, \cdots, d_n\}$, and the convention is adopted that

* Received by the editors May 2, 1990; accepted for publication December 6, 1990.

† Department of Mathematics and Computer Science, University of Dundee, Dundee DD1 4HN, Scotland (gawatson@maths-and-cs.dundee.ac.uk).

$A \geqq 0$ if and only if A is a symmetric positive semidefinite matrix. Then the value ρ_* is sought such that

$$\rho_* = \max_{\mathbf{w} \neq 0} \min_{\mathbf{d} \in L(S)} \rho(\mathbf{d}, \mathbf{w}), \tag{1.3}$$

where ρ is given by (1.2). This problem is referred to as the problem of weighted minimum trace factor analysis.

Shapiro [13] examines the relationship of (1.3) with the *dual* problem of calculating

$$\rho^* = \min_{\mathbf{d} \in L(S)} \max_{\mathbf{w} \neq 0} \rho(\mathbf{d}, \mathbf{w}). \tag{1.4}$$

Under conditions that are frequently satisfied, the values of ρ^* and ρ_* are equal, so that the primal problem of calculating ρ_* can often be solved via the dual problem of calculating ρ^*, a problem that turns out to be much more tractable. A method for obtaining (1.4) is given by Shapiro [13]. However, in common with methods for minimizing (1.2) (see, for example, Bentler [1], Woodhouse and Jackson [16], Bentler and Woodward [2], Ten Berge, Snijders, and Zegers [14]), it is of first order, and it appears that methods which are of higher order have not so far been seriously considered. The purpose of this paper is to show how second-order convergent methods may be applied to (1.2) and (1.4). Since methods for (1.3) require a sequence of minimizations of (1.2), the analysis given here is also relevant to that problem. The solution of problems with (1.2) is considered in § 2, and the treatment of (1.4) is considered in § 3. Finally, in § 4 the solution of the primal problem is considered, using a method suggested by Shapiro [13], and with the subproblems solved by the method developed in § 2. Some numerical results are presented for all the algorithms, using test problems that have appeared in the literature. The techniques presented here exploit recent work on eigenvalue constraints by Fletcher [4]; Friedland, Nocedal, and Overton [5]; Overton [9], [10]; and Watson [15].

2. The solution of the constrained minimum trace problem. Consider the minimization of (1.2) over vectors $\mathbf{d} \in L(S)$, where $\mathbf{w} \in \mathbb{R}^n$ is given. This corresponds to

$$\begin{aligned} &\text{maximize } \mathbf{w}^T D \mathbf{w} \\ &\text{subject to } S - D \geqq 0, \qquad D \geqq 0. \end{aligned} \tag{2.1}$$

Let $\gamma_i(D)$, $i = 1, \cdots, n$, denote the eigenvalues of the matrix $M = S^{-1/2} D S^{-1/2}$. Then (2.1) is equivalent to

$$\begin{aligned} &\text{minimize } -\sum_{i=1}^{n} w_i^2 d_i \\ &\text{subject to } \gamma_i(D) \leqq 1, \qquad i = 1, \cdots, n, \\ &d_i \geqq 0, \qquad i = 1, \cdots, n. \end{aligned} \tag{2.2}$$

Assume that the eigenvalues are ordered so that $\gamma_n \leqq \cdots \leqq \gamma_1$, and at any feasible D, the multiplicity of γ_1 is r. If $r = 1$ at a solution, then (2.2) is just a smooth optimization problem and methods with a fast rate of convergence may be developed in a straightforward way. However, when $r > 1$, techniques of nondifferentiable optimization are required to recover rapid convergence, and it is in the ability to deal with this case in a satisfactory manner that the methods presented here are superior to methods previously applied to the problem. For a particular model problem, the treatment of constraints on eigenvalues is analyzed in the papers by Overton [9], [10] and also in a forthcoming paper by Overton and Womersley [11]. The intention is to use similar techniques here.

For given D, let the eigenvalue/eigenvector decomposition of M be

$$M = Q\Gamma Q^T,$$

where $Q \in \mathbb{R}^{n\times n}$ is orthogonal and $\Gamma = \operatorname{diag}\{\gamma_1, \cdots, \gamma_n\}$. Then letting $Q = [Q_1 : \bar{Q}_1]$, where $Q_1 \in \mathbb{R}^{n\times r}$, it follows that

$$MQ_1 = Q_1$$

or

$$(S - D)V_1 = 0, \tag{2.3}$$

where $V_1 = S^{-1/2}Q_1$. Because

$$\gamma_1(D) = \max_{\mathbf{x}\neq 0} \frac{\mathbf{x}^T M\mathbf{x}}{\mathbf{x}^T\mathbf{x}}, \tag{2.4}$$

it follows that γ_1 is a convex function of D and so (2.2) is a convex programming problem. Necessary and sufficient conditions for D^* to be a solution may be obtained by standard methods of convex analysis (for example, Rockafellar [12, § 28]). The following result is also obtained by Della Riccia and Shapiro [3] and Ten Berge, Snijders, and Zegers [14].

THEOREM 1. *Necessary and sufficient conditions for D^* to solve* (2.2) *are that there exists a matrix $T \in \mathbb{R}^{r\times r}$, $T \geqq 0$, such that for all j, $j = 1, \cdots, n$,*

$$w_j^2 = \mathbf{e}_j^T V_1 T V_1^T \mathbf{e}_j, \qquad d_j^* \neq 0, \tag{2.5a}$$

$$w_j^2 \leqq \mathbf{e}_j^T V_1 T V_1^T \mathbf{e}_j, \qquad d_j^* = 0, \tag{2.5b}$$

where $\mathbf{e}_j$ is the jth coordinate vector in $\mathbb{R}^n$.

If r^* is the multiplicity of the unit eigenvalue at a solution D^*, then locally the problem (2.2) can be posed as

$$\begin{aligned} &\text{minimize } -\sum_{i=1}^{n} w_i^2 d_i \\ &\text{subject to } \gamma_i(D) = 1, \qquad i = 1, \cdots, r^*, \\ &d_i \geqq 0, \qquad i = 1, \cdots, n. \end{aligned} \tag{2.6}$$

Let $g_j = -w_j^2$, $j = 1, \cdots, n$, and, for given D, let $M_j = (\partial M/\partial d_j)$, $j = 1, \cdots, n$, and let Q_1^* be a matrix of order $n \times r^*$ whose columns are orthonormal eigenvectors corresponding to $\gamma_1(D), \cdots, \gamma_{r^*}(D)$. The basic iteration of a method for (2.6) is defined by the quadratic programming problem

$$\begin{aligned} &\text{minimize } \mathbf{g}^T\boldsymbol{\delta} + \tfrac{1}{2}\boldsymbol{\delta}^T H\boldsymbol{\delta} \\ &\text{subject to } \sum_{j=1}^{n} \delta_j Q_1^{*T} M_j Q_1^* = \operatorname{diag}\{1 - \gamma_1, \cdots, 1 - \gamma_{r^*}\}, \\ &d_j + \delta_j \geqq 0, \qquad j = 1, \cdots, n, \end{aligned} \tag{2.7}$$

where δ represents the variation in $\mathbf{d}$, and the equality constraints of (2.7) are a result of the linearization of a differentiable system of $\frac{1}{2}r^*(r^* + 1)$ nonlinear equations characterizing the conditions

$$\gamma_i(D) = 1, \qquad i = 1, \cdots, r^*.$$

The correct system is based on a matrix exponential formulation given by Friedland, Nocedal, and Overton [5]; further details are given in [10] and [11]. In order for an iteration based on (2.7) to give a second-order convergent process, the matrix H should be set to the Hessian matrix of the appropriate Lagrangian function, given by [9, eq. (4.12)]. In this context, the matrix T of Theorem 1 is just a matrix of Lagrange multipliers.

In fact, some modifications of (2.7) are required to ensure convergence from a poor initial approximation. In particular, the correct value of r^* would not normally be known in advance, so a guessed value r' must be used, to be adjusted adaptively at each iteration. An appropriate quadratic programming problem defined at a feasible point D can be stated as

$$\begin{aligned} &\text{minimize } \mathbf{g}^T\boldsymbol{\delta} + \tfrac{1}{2}\boldsymbol{\delta}^T H\boldsymbol{\delta} \\ &\text{subject to } 1 - \gamma_i - \sum_{j=1}^{n} \delta_j \mathbf{q}_i^T M_j \mathbf{q}_i \geqq 0, \qquad i = 1, \cdots, n, \\ &\sum_{j=1}^{n} \delta_j \mathbf{q}_i^T M_j \mathbf{q}_k = 0, \quad i, k = 1, \cdots, r', \quad i < k, \\ &d_j + \delta_j \geqq 0, \qquad j = 1, \cdots, n, \end{aligned} \tag{2.8}$$

where $\mathbf{q}_i$ is the ith column of a current matrix Q of eigenvectors, and where the eigenvalues and coefficients $\mathbf{d}$ are scaled so that $\gamma_1(D) = 1$. The scaling ensures that feasibility of D can be maintained with respect to all the constraints of (2.2). The incorporation of inequality constraints rather than equalities for the diagonal components of the matrix constraints corresponding to (2.7) reflects the original form of the problem (2.2); it may be overly restrictive to force constraints to hold with equality in the event that r' is incorrect. On the other hand, the equality constraints in (2.8) have no meaning if the corresponding inequality constraint is inactive, so there is also merit in treating the first r' inequality constraints as equalities. If care is taken in the strategy that allows r' to increase, there need be no difference in practice in these alternative approaches. The presence of the additional inequality constraints ensures that linearizations of γ_i, $i = r' + 1, \cdots, n$ (assumed simple) give values no greater than 1. This can also be helpful in that activity (or inactivity) of these constraints can be useful in the adjustment of r'.

As already explained, the matrix H is normally set to the Hessian matrix of the appropriate Lagrangian function, with appropriate estimates of the components of T included. These are usually obtained by solving in the least squares sense the system of equations corresponding to (2.5). Using the fact that

$$M_j = A\mathbf{e}_j\mathbf{e}_j^T A, \qquad j = 1, \cdots, n,$$

where $A = S^{-1/2}$, a straightforward calculation using [9, eq. (4.12)] shows that the (i, j) element of H is given by

$$2 \sum_{\alpha,\beta=1}^{r'} T_{\alpha\beta}\left[G_{\alpha j}G_{\beta i} \sum_{l=r'+1}^{n} G_{li}G_{lj}/(1 - \gamma_l)\right],$$

where $G = Q^T A$. This choice permits the solution δ of (2.8) to give the Newton step for satisfying the conditions of Theorem 1, and therefore a second-order rate of convergence is possible (see, for example, Nocedal and Overton [8]). In the special case when the number of constraints of (2.8) eventually holding with equality is precisely n, then a second-order rate of convergence is obtained irrespective of H because the method just becomes Newton's method for solving the equalities holding at a solution to (2.2).

Methods based on the use of (2.8) also have a global convergence capability, as the following theorem illustrates. It is based on the fact that if feasibility with respect to the nonnegativity constraints is maintained, (2.2) is equivalent to the *unconstrained* problem

$$\text{find } \mathbf{d} \in \mathbb{R}^n \quad \text{to minimize } f(D) = \frac{-\sum_{i=1}^n w_i^2 d_i}{\gamma_1(D)}, \tag{2.9}$$

using the homogeneity of the objective function.

THEOREM 2. *Let H be positive definite, and let δ solve* (2.8) *defined at $D \geqq 0$, scaled so that $\gamma_1(D) = 1$ and with r' selected so that $r' \geqq r$ (the actual multiplicity of $\gamma_1(D)$). Then $\Delta = \operatorname{diag}\{\delta_1, \cdots, \delta_n\}$ is a descent direction for f at D.*

Proof. As previously explained, γ_1 is a convex function of D; furthermore, it has subdifferential (see, for example, [13, Lemma 4.3])

$$\partial\gamma_1(D) = \{AQ_1UQ_1^TA,\ U \geqq 0,\ \operatorname{trace}(U) = 1\}.$$

By using the chain rule for the directional derivative of a quotient of convex functions (see, for example, Rockafeller [12, p. 217]), it follows that the directional derivative of $f(D)$ in the direction Δ is given by

$$\max_{F \in \partial\gamma_1(D)} \left[\gamma_1(D)^{-2}\left(\operatorname{trace}(\Delta F)\sum_{i=1}^n w_i^2 d_i - \gamma_1(D)\sum_{i=1}^n w_i^2\delta_i\right)\right]. \tag{2.10}$$

Now Kuhn–Tucker conditions for the problem (2.8) give the existence of nonnegative numbers λ_i, $i = 1, \cdots, n$, β_i, $i = 1, \cdots, n$, and numbers λ_{ik}, $i, k = 1, \cdots, r'$, $i < k$, such that

$$\begin{aligned}\mathbf{g} + H\delta + \sum_{i=1}^n \lambda_i[\mathbf{q}_i^TM_1\mathbf{q}_i, \cdots, \mathbf{q}_i^TM_n\mathbf{q}_i]^T \\ + \sum_{i,k} \lambda_{ik}[\mathbf{q}_i^TM_1\mathbf{q}_k, \cdots, \mathbf{q}_i^TM_n\mathbf{q}_k]^T - \beta = 0\end{aligned} \tag{2.11}$$

so that

$$\delta^T\mathbf{g} + \delta^TH\delta + \sum_{i=1}^n \lambda_i(1 - \gamma_i) + \mathbf{d}^T\beta = 0,$$

using the constraints and the complementary slackness conditions. It follows that

$$\delta^T\mathbf{g} < 0 \quad \text{if } \delta \neq 0. \tag{2.12}$$

Now $Q_1^T\sum_{j=1}^n \delta_jM_jQ_1 = \sum_{j=1}^n \delta_jQ_1^TA\mathbf{e}_j\mathbf{e}_j^TAQ_1$, which, using (2.8), is a diagonal matrix with nonnegative diagonal elements. Therefore, if $U \geqq 0$ is otherwise arbitrary,

$$\operatorname{trace}\left(UQ_1^T\sum_{j=1}^n \delta_jM_jQ_1\right) = \sum_{j=1}^n \delta_j\mathbf{e}_j^TAQ_1UQ_1^TA\mathbf{e}_j$$

is nonpositive. It follows that

$$\operatorname{trace}(\Delta F) \leqq 0 \quad \forall F \in \partial\gamma_1(D), \tag{2.13}$$

and so using (2.12) the directional derivative (2.10) is negative. Therefore, Δ defined in this way is a descent direction for $f(D)$ at D. □

Because H is not necessarily positive definite, some modification to the basic subproblem is required to exploit Theorem 2. One possibility is to add a positive multiple

of the unit matrix to H and to use a line search. However, a more satisfactory strategy is to introduce a step length restriction on δ in (2.8), making a conventional trust region approach. Additional constraints

$$-\tau \leqq \delta_i \leqq \tau, \qquad i = 1, \cdots, n \tag{2.14}$$

are added to the constraints of (2.8), with τ adjusted adaptively and reduced systematically if necessary to ensure that $f(D + \Delta) < f(D)$ at each iteration. This is a standard approach to this kind of problem, and effective prescriptions are available for adjusting the value of τ. If eventually the constraints (2.14) are inactive, then a second-order rate of convergence is not inhibited.

The way in which r' is adjusted is also obviously important. One possibility is to determine the number of eigenvalues satisfying

$$\gamma_1 - \gamma_i < tol\ 1, \qquad i = 2, \cdots, n,$$

for a given value of *tol* 1, but it is also appropriate to check on activity with respect to the inequality constraints of the quadratic programming problem; if the above test is satisfied by an eigenvalue but the corresponding constraint is not active, then it is not necessarily right to include the index in the set defining r'. If δ solving (2.8), (2.14) is forced to zero, with the sequence of values of τ bounded away from zero, then it follows from (2.11) that there is a symmetric matrix T, with nonnegative diagonal elements, such that

$$g_i + \text{trace}\,(TQ_1^T M_i Q_1) - \beta_i = 0, \qquad i = 1, \cdots, n,$$

where $\beta_i \geqq 0$ and $\beta_i = 0$ if $d_i > 0$, and so (2.5) is satisfied. Therefore, if T is also positive semidefinite, the current D solves (2.1).

The algorithm can now be summarized. All the data for the subproblems can be conveniently computed from the elements of the matrix $Q^T A$.

1. Choose an initial diagonal matrix $D \geqq 0$, set r' (usually to 1), and set τ to 1.
(Start iteration number k.)
2. Perform an eigenvalue/eigenvector decomposition of $M = ADA$, and adjust r' if necessary. Scale D and the eigenvalues so that $\gamma_1(D) = 1$. Determine the multipliers T by solving the least squares problem

$$\mathbf{e}_j^T A Q_1 T Q_1^T A \mathbf{e}_j - \beta_j = w_j^2, \qquad j = 1, \cdots, n,$$

where β_j is absent if $d_j > 0$ and form the Hessian matrix H.
3. Solve the quadratic programming problem (2.8), (2.14) for δ. If $\|\delta\|_\infty < tol$ then go to Step 4. If $f(D + \Delta) < f(D)$, set D to $D + \Delta$ and go to Step 2 with τ multiplied by 4; otherwise divide τ by 4 (or set τ to $\frac{1}{2}\|\delta\|_\infty$ if this is necessary) and repeat Step 3.
(End iteration number k.)
4. Check the matrix T of multipliers for positive semidefiniteness, and terminate if this is satisfied. (Otherwise it is necessary to reduce the value of r' by removing an index corresponding to a negative eigenvalue of T.)

In an implementation of this algorithm, no provision was made for the failure of the test in Step 4. However, this situation can be provided for, as in the methods described by Overton [9] or Watson [15]. The value of *tol* 1 was taken to be 0.1, and the following calculations were done on a SUN system 3/50, for which single length precision is about seven decimal places. The eigendecompositions and least squares solutions were obtained using the NAG subroutines F02AMF and F04AMF, respectively, and the quadratic programming problems were solved using the Harwell subroutine VE02A.

TABLE 1

k	$-f(D)$	r'	QSP	$\|\delta\|_\infty$
1	0.007177	1	1	0.365557
2	0.022227	1	1	0.094095
3	0.022871	1	1	0.038634
4	0.023564	1	1	0.005063
5	0.023598	1	1	0.000228
6	0.023598	1	1	0.000001

Example 1. This is the social class data example from Bentler [1], with $n = 6$, and unit weights (scaled so that $\mathbf{w}^T S\mathbf{w} = 1$). The performance of the algorithm is shown in Table 1, from the initial approximation $\mathbf{d} = \mathbf{e}$. The table shows the value of r' at each iteration, the objective function $f(D)$, the number of times (2.8) and (2.14) have to be solved (QSP), and the value of $\|\delta\|_\infty$. The solution is

$$\mathbf{d}^* = (0.0, 0.0, 0.249525, 0.188007, 0.221789, 0.0)^T,$$

and the value of ρ is 0.976402. This differs from the value quoted by Bentler [1]; however, it is readily established that the conditions of Theorem 1 are satisfied: the eigenvector of $S^{-1}D^*$ corresponding to the unit eigenvalue is

$$\mathbf{v}_1 = (-1.470138, 3.328967, -1.231549, -1.231549, -1.231549, 1.618697)^T,$$

so that (2.5a), (2.5b) are satisfied with T the scalar 0.023598. In fact, if $\mathbf{r}$ is the vector with ith component $w_i^2 - \mathbf{e}_i^T V_1 T V_1^T \mathbf{e}_i$, $i = 1, \cdots, n$, then

$$\mathbf{r} = (-0.015211, -0.225722, 0.0, 0.0, 0.0, -0.026040)^T.$$

Note that the value of T is just $-f(D^*)$. This is no coincidence and, in fact, it follows from the complementary slackness conditions for (2.4) that

$$\text{trace}\,(T) = -f(D^*). \tag{2.15}$$

Example 2. This is the WAIS intelligence example from Bentler [1], which has $n = 11$ with unit weights. This turns out to be quite a difficult example for the algorithm because the value of r^* is not easy to identify. The correct value is $r^* = 2$, although a

TABLE 2

k	$-f(D)$	r'	QSP	$\|\delta\|_\infty$
1	0.026594	1	1	1.0
2	0.039900	1	2	1.0
3	0.045432	1	2	1.0
4	0.048964	1	4	0.0625
5	0.049535	2	2	0.0625
6	0.049825	2	3	0.015625
7	0.049904	2	1	0.0625
8	0.049980	2	1	0.25
9	0.050252	2	2	0.25
10	0.050446	2	1	0.245339
11	0.050488	2	1	0.029942
12	0.050547	2	1	0.004118
13	0.050552	2	1	0.000360
14	0.050552	2	1	0.000001

third eigenvalue is very close to 1 at the solution. Starting from $\mathbf{d} = \mathbf{e}$, the algorithm sets r' to 2 at iteration number 5, but the trust region bound remains active until iteration number 9, after which fast convergence is achieved. Table 2 shows the progress of the algorithm, with the information displayed similar to that given in Table 1. The final vector $\mathbf{d}^*$ is $(0.987033, 2.300717, 3.782821, 2.540044, 4.769281, 1.385407, 4.045103, 2.976816, 2.916875, 2.290365, 3.030018)^T$, and the value of ρ is 0.949448. The final multiplier matrix is

$$T = \begin{bmatrix} 0.022189 & 0.000069 \\ 0.000069 & 0.028364 \end{bmatrix},$$

which is positive definite, and satisfies (2.15).

3. The solution of the dual problem. This section is concerned with the determination of ρ^* defined by (1.4). Letting as before, $\gamma_i(D)$ denote the eigenvalues of $S^{-1}D$ in descending order of magnitude, it is clear that this problem can be solved by means of the solution to the problem

$$\text{maximize } \gamma_n(D) \quad \text{subject to } \gamma_1(D) \leqq 1. \tag{3.1}$$

If D^* solves (3.1), then

$$\rho^* = 1 - \frac{\gamma_n(D^*)}{\gamma_1(D^*)}.$$

As before, γ_1 is a convex function of D. Furthermore,

$$-\gamma_n = \max_{\mathbf{x} \neq 0} \left\{ \frac{-\mathbf{x}^T M \mathbf{x}}{\mathbf{x}^T \mathbf{x}} \right\},$$

which is also convex. Thus (3.1) is a convex optimization problem. Strictly speaking, (3.1) should be supplemented by the restrictions that the diagonal elements of D be nonnegative; however, it is clear from the formulation of (3.1) that no diagonal component can be zero, so the nonnegativity constraints play a much lesser role than they did in the problem considered in § 2. For convenience, in what follows, these conditions will simply be ignored. The problem (3.1) is equivalent to the problem of best l_2 scaling of the matrix LD with respect to all diagonal matrices D, where LL^T is the Cholesky factorization of S. The algorithm presented here is therefore similar to the algorithm for such problems given in Watson [15].

An alternative formulation of (3.1) is

$$\begin{aligned} &\text{maximize } h \\ &\text{subject to } \gamma_i(D) \leqq 1, \qquad i = 1, \cdots, n, \\ &\qquad\qquad \gamma_i(D) \geqq h, \qquad i = 1, \cdots, n. \end{aligned} \tag{3.2}$$

For given D, let the multiplicity of γ_1 be r and the multiplicity of γ_n be s. Let V_1 be defined as in (2.3), an $n \times r$ matrix of orthonormal eigenvectors of $S^{-1}D$ corresponding to the eigenvalue γ_1, and let V_2 denote an $n \times s$ matrix of orthonormal eigenvectors corresponding to γ_n. Then necessary and sufficient conditions for a solution can be given in the following form (see, for example, Shapiro [13]).

THEOREM 3. *D^* solves (3.1) if and only if there exists $T \in \mathbb{R}^{r \times r}$, $T \geqq 0$, $Z \in \mathbb{R}^{s \times s}$, $Z \geqq 0$, with* trace $(Z) = 1$, *such that*

$$\mathbf{e}_j^T V_2 Z V_2^T \mathbf{e}_j = \mathbf{e}_j^T V_1 T V_1^T \mathbf{e}_j, \qquad j = 1, \cdots, n. \tag{3.3}$$

The problem (3.2) can be solved through a sequence of quadratic programming problems analogous to (2.8). An appropriate problem can have the form

$$\begin{aligned}
&\text{minimize } -p + \tfrac{1}{2}\delta^T H\delta \\
&\text{subject to } \gamma_i(M) - h + \sum_{j=1}^{n} \delta_j \mathbf{q}_i^T M_j \mathbf{q}_i - p \geqq 0, \qquad i = n - s' + 1, \cdots, n, \\
&\qquad \sum_{j=1}^{n} \delta_j \mathbf{q}_i^T M_j \mathbf{q}_k = 0, \quad i, k = n - s' + 1, \cdots, n, \quad i < k, \\
&\qquad 1 - \gamma_i(M) - \sum_{j=1}^{n} \delta_j \mathbf{q}_i^T M_j \mathbf{q}_i \geqq 0, \qquad i = 1, \cdots, r', \\
&\qquad \sum_{i=1}^{n} \delta_j \mathbf{q}_i^T M_j \mathbf{q}_k = 0, \quad i, k = 1, \cdots, r', \quad i < k.
\end{aligned} \tag{3.4}$$

In this case, two estimates must be kept of the multiplicity r' of γ_1 and s' of γ_n; these are approximations to the true multiplicity r^* and s^*, respectively, at a solution. If the matrix H is set to the Hessian matrix of the appropriate Lagrangian function, then a second-order rate of convergence to a point satisfying (3.3) can be obtained. For the special case when $\frac{1}{2}r^*(r^* + 1) + \frac{1}{2}s^*(s^* + 1) = n + 1$, then a second-order rate of convergence is obtained irrespective of the choice of H, for the method just becomes Newton's method for solving the system of equalities defined by constraints of (3.4) active at a solution.

There are some modifications of (3.4) that could be used. For example, n inequality constraints could be maintained for each set of inequalities in (3.2) (rather than just r' and s', respectively, as is done here). There could be some merit in a compromise, keeping two separate estimates that are adjusted at each iteration. Also, just as for (2.8), some of the inequalities (here $r' + s'$) could be treated as equalities.

If the eigenvalues (and parameters) are scaled so that $\gamma_1 = 1$ and h is set to γ_n, feasibility is maintained with respect to the original problem constraints and progress can be monitored with respect to the quotient $q(D) = -\gamma_n(D)/\gamma_1(D)$. The following theorem is important in this context; it is similar to Theorem 1, and also to [15, Thm. 2]; the details are therefore omitted.

THEOREM 4. *Let H be positive definite, and let δ solve* (3.4) *defined at the positive definite diagonal matrix D, where D is scaled so that $\gamma_1(D) = 1$, where $h = \gamma_n(D)$, and where r' and s' are chosen so that they are no less than the current actual multiplicities. Then $p \geqq 0$; furthermore,* (i) *if $p > 0$, δ is a descent direction for q at D, and* (ii) *if $p = 0$, then there exist symmetric matrices T and Z such that* (3.3) *is satisfied.*

As before, descent is achieved using a trust region strategy, so that the additional constraints (2.14) are incorporated. When estimates of r' and s' have been established based on the proximity of eigenvalues and activity in appropriate inequality constraints, Lagrange multiplier estimates (that is, estimates of T and Z) can be obtained by solving in the least squares sense the system

$$\mathbf{e}_j^T A Q_1 T Q_1^T A \mathbf{e}_j - \mathbf{e}_j^T A Q_2 Z Q_2^T A \mathbf{e}_j = 0, \quad j = 1, \cdots, n, \quad \text{trace}\,(Z) = 1. \tag{3.5}$$

Because of the simple form of M, all quantities can again be calculated from the elements of the matrix $Q^T A$, where $A = S^{-1/2}$. A summary of the algorithm is now given.

1. Choose an initial positive definite approximation D, and set r' and s' (usually to 1), and τ to 1.

(Start iteration number k.)

2. Perform an eigenvalue/eigenvector decomposition of $M = ADA$, and adjust r' and s' if necessary. Scale D and the eigenvalues so that $\gamma_1 = 1$, and set h to γ_n. Determine multipliers by solving the least squares problem (3.5), and form the Hessian matrix H.
3. Solve the quadratic programming problem (3.4), (2.14) for δ and p. If $\|\delta\|_\infty < tol$, then go to Step 4. If $q(D + \Delta) < q(D)$, and $D + \Delta > 0$, set D to $D + \Delta$ and go to Step 2 with τ multiplied by 4; otherwise divide τ by 4 (or set τ to $\frac{1}{2}\|\delta\|_\infty$ if necessary) and repeat Step 3.

(End iteration number k.)

4. Check the multipliers T and Z for positive semidefiniteness, and terminate if this is satisfied. (Otherwise it is necessary to reduce r' or s' by removing an index corresponding to a negative eigenvalue.)

Eigenvalues were deemed to be coalescing if they became closer than $0.1 \times \gamma_n$ (or γ_1). No provision was made in the algorithm for the failure of the test in Step 4, although a suitable strategy is given by Watson [15]. Some numerical examples are now given to illustrate the performance of an implementation of the algorithm. The problems are those solved by Shapiro [13], using a first-order method base on linearization of the constraints of (3.2).

Example 3(a). This is the same social class data example from Bentler [1], with $n = 6$, as is used for Example 1. The performance of the algorithm is shown in Table 3, from the initial approximation $D = I$. The table shows the values of r' and s' used for each iteration, the value $q = -\gamma_n/\gamma_1$, the number of quadratic programming problems required to be solved, and the value of $\|\delta\|_\infty$, where δ solves the quadratic programming problem. This is a completely straightforward smooth example; because both γ_1 and γ_n are simple at the solution, they are differentiable functions of D. The solution obtained was

$$\mathbf{d}^* = (0.033388, 0.023388, 0.103285, 0.061308, 0.072562, 0.037834)^T,$$

$$\mathbf{w}^* = (0.252975, 0.364487, 0.074181, 0.127305, 0.105158, 0.191948)^T,$$

and $\rho^* = \rho_* = 0.990998$. In this example, the vector of weights is unique up to a scalar multiplier; the scaling used here was such that $\mathbf{w}^T S\mathbf{w} = 1$. The multiplier matrix T (just a scalar in this case) is 0.0090021, which is just $-q(D^*)$. In general it is the case that

$$\text{trace}\,(T) = -q(D^*). \tag{3.6}$$

Example 3(b). This is the artificial example used by Shapiro [13] having

$$S = \begin{bmatrix} 1.00 & -0.05 & -0.35 & -0.4 \\ -0.05 & 1.00 & -0.2 & -0.05 \\ -0.35 & -0.2 & 1.00 & -0.05 \\ -0.4 & -0.05 & -0.05 & 1.00 \end{bmatrix}.$$

TABLE 3

k	$-q(D)$	r'	s'	QSP	$\|\delta\|_\infty$
1	0.0071660	1	1	1	0.028508
2	0.0085330	1	1	1	0.026587
3	0.0089700	1	1	1	0.013846
4	0.0090017	1	1	1	0.002089
5	0.0090021	1	1	1	0.000033
6	0.0090021	1	1	1	0.000000005

Again, this is a straightforward example, and the algorithm converges from $\mathbf{d} = \mathbf{e}$ ($q(D) = -0.262750$) in six iterations ($\|\delta\|_\infty = 10^{-8}$), with one quadratic programming solution required per iteration, to

$$\mathbf{d}^* = (0.409278, 0.362083, 0.396618, 0.396011)^T,$$

$$\mathbf{w}^* = (-0.534555, -0.202388, 0.410702, 0.404770)^T.$$

Both γ_1 and γ_n are simple at D^*, and $q(D^*) = -0.263564$, so that $\rho^* = \rho_* = 0.736436$.

Example 4. This is the WAIS intelligence example from Bentler [1] used in Example 2, with $n = 11$. Table 4 shows similar results to those shown in Table 3. Note that in contrast to Examples 3(a) and 3(b), this is not a smooth problem. For the same reasons as before, it is also a difficult problem inasmuch as the multiplicity of γ_1 could easily be taken to be 3 rather than 2. No additional information is available because of the absence of extra inequality constraints, and this example illustrates that a more sophisticated strategy for choosing r' and s' would generally be required. The results shown in Table 4 are based on favorable decisions about the absence of a third coalescing eigenvalue. The trust region radius was active until iteration number 12, and the final solution obtained was

$$\mathbf{d}^* = (1.265351, 2.573287, 3.587663, 2.327083, 4.521483, 1.382124,$$
$$3.802840, 2.638874, 3.173902, 2.332701, 2.666162)^T,$$

$$\mathbf{w}^* = (0.083006, 0.036680, 0.024414, 0.041380, 0.015716, 0.077872,$$
$$0.021939, 0.033439, 0.027752, 0.034520, 0.026087)^T,$$

and $\rho^* = \rho_* = 0.960379$. Again $\mathbf{w}^*$ is unique to a scalar multiplier. The matrix T at the solution was

$$T = \begin{bmatrix} 0.011271 & 0.004899 \\ 0.004899 & 0.028350 \end{bmatrix},$$

which is positive definite and satisfies (3.6).

Example 5. Let S be the 3×3 matrix $I - \frac{1}{4}\mathbf{e}\mathbf{e}^T$. Then the performance of the algorithm is shown in Table 5. The point of this relatively simple example is to show

TABLE 4

k	$-q(D)$	r'	s'	QSP	$\|\delta\|_\infty$
1	0.0263452	1	1	1	1.0
2	0.0362394	1	1	3	0.25
3	0.0375337	2	1	1	1.0
4	0.0384112	2	1	4	0.0625
5	0.0392416	2	1	2	0.0625
6	0.0393531	2	1	3	0.015625
7	0.0394118	2	1	2	0.015625
8	0.0394294	2	1	1	0.0625
9	0.0394533	2	1	1	0.25
10	0.0395221	2	1	2	0.25
11	0.0395790	2	1	2	0.25
12	0.0396018	2	1	1	0.095843
13	0.0396165	2	1	1	0.004885
14	0.0396210	2	1	1	0.000933
15	0.0396213	2	1	1	0.000019
16	0.0396213	2	1	1	0.00000005

TABLE 5

k	γ_2	γ_3	r'	s'	QSP	$\|\delta\|_\infty$
1	0.2835037	0.1418721	1	1	2	0.25
2	0.2974130	0.1576195	1	1	1	0.269836
3	0.2625305	0.1822586	1	1	3	0.207583
4	0.2664593	0.2231516	1	1	2	0.051895
5	0.2528271	0.2470431	1	2	1	0.005600
6	0.2500420	0.2499580	1	2	1	0.000079
7	0.2500000	0.2500000	1	2	1	0.00000002

that it is possible for the multiplicity of γ_n to exceed 1, and the actual values of the eigenvalues are given (the value of γ_1 is always scaled to be 1, and so the quotient value is just γ_3). This has consequences for the relationship between ρ^* and ρ_*. Shapiro [13] shows that if γ_n is simple at the solution to (3.2), then $\rho^* = \rho_*$, but otherwise equality need not (and generally will not) hold. For this example,

$$\mathbf{d}^* = (0.250000, 0.250000, 0.250000)^T,$$

$\rho^* = 0.750000$, and $\mathbf{w}^*$ is any eigenvector of $S^{-1}D^*$ corresponding to γ_3. The multiplier matrices at the solution were

$$Z = \begin{bmatrix} 0.5 & 0.0 \\ 0.0 & 0.5 \end{bmatrix}, \qquad T = [0.25].$$

4. The solution of the primal problem. Shapiro [13] considers in detail the primal problem (1.3) and its relationship to the dual problem (1.4). The primal problem is not a convex problem, and so a complete characterization of solutions is not possible; the following theorem gives necessary conditions.

THEOREM 5 (Shapiro [13]). *Let* $\mathbf{w}^*$, $\mathbf{d}^*$ *solve* (1.3). *Then there exists* $\mu > 0$ *such that*

$$D^*\mathbf{w}^* = \mu S\mathbf{w}^*, \tag{4.1}$$

and further, $\mathbf{d}^*$ *satisfies Theorem* 1 *with* $\mathbf{w} = \mathbf{w}^*$.

Because $\mathbf{d}^*$ can have no zero component, the required conditions from Theorem 1 are just

$$w_j^{*2} = \mathbf{e}_j^T V_1 T V_1^T \mathbf{e}_j, \qquad j = 1, \cdots, n. \tag{4.2}$$

If $\mathbf{d}^*$, $\mathbf{w}^*$ solve the dual problem (1.4), then (4.1) is satisfied with $\mu = \gamma_n$. If γ_n is simple, then $V_2 = [\mathbf{w}^*]$. The conditions of Theorem 3 coincide with (4.2), and (1.3) is also solved. However, if the multiplicity of γ_n exceeds 1, this will not be the case unless the matrix Z of Theorem 3 has rank 1.

Shapiro [13] gives an algorithm for the primal problem based on the fact that

$$\max_{\mathbf{d} \in L(S)} \mathbf{w}^T D\mathbf{w} \leqq \min_{\operatorname{diag}(V_1 V_1^T) \geqq W^2} \operatorname{trace}(V_1^T S V_1), \tag{4.3}$$

where V_1 satisfies (2.3), with equality holding when

$$\operatorname{trace}(V_1^T D V_1) = \mathbf{w}^T D\mathbf{w}. \tag{4.4}$$

With this reformulation of the problem substituted into the expression for ρ_*, the presence of two minimizations (of the quotient on the right-hand side of (1.2)) means that the calculation of ρ_* can be achieved iteratively. Fix $\mathbf{w}$ and minimize with respect to V_1 (a

problem of the form (2.1)); then for this V_1, minimize with respect to $\mathbf{w}$ (a minimum eigenvalue calculation) and so on. The calculation of V_1 can be achieved as follows: if W is the diagonal matrix with the current components of $\mathbf{w}$ on the diagonal, then solve (2.1) with S replaced by WSW and $\mathbf{w} = \mathbf{e}$ (unit weights); then use (2.3) and (4.2). The algorithm can be summarized as follows (for full details, see Shapiro [13]).

1. Choose $\mathbf{w}^{(1)} \in \mathbb{R}^n$, normalized so that $\mathbf{w}^{(1)T} S \mathbf{w}^{(1)} = 1$. Set $k = 1$.
2. Solve (2.1) with $\mathbf{w} = \mathbf{e}$ (normalized so that $\mathbf{w}^T S \mathbf{w} = 1$) and S replaced by $W^{(k)} S W^{(k)}$. Let the solution be $D^{(k)}$ and the objective function value be $f^{(k)}$.
3. Identify $V_1^{(k)}$ satisfying (2.3) in this case (a matrix of eigenvectors of $W^{(k)} S W^{(k)-1} D^{(k)}$ corresponding to the unit eigenvalue; the dimensions of $V_1^{(k)}$ will be $n \times r$ if the unit eigenvalue has multiplicity r), normalized so that
$$\mathbf{w}^{(k)T} D^{(k)} \mathbf{w}^{(k)} = \operatorname{trace}(V_1^{(k)T} D^{(k)} V_1^{(k)}).$$
4. Compute $X^{(k)} \in \mathbb{R}^{n \times n}$ such that
$$X^{(k)} = \sum_{i=1}^{r} \operatorname{diag}\{(\mathbf{v}_i^{(k)})_1, \cdots, (\mathbf{v}_i^{(k)})_n\} S \operatorname{diag}\{(\mathbf{v}_i^{(k)})_1, \cdots, (\mathbf{v}_i^{(k)})_n\}.$$
5. Determine $\gamma^{(k)}$ as the smallest eigenvalue of $S^{-1} X^{(k)}$. If $|\gamma^{(k-1)} - \gamma^{(k)}| < \varepsilon$, then terminate the iteration. Otherwise set $\mathbf{w}^{(k+1)}$ to an eigenvector corresponding to the smallest eigenvalue, correctly normalized, and go to Step 2 with k increased by 1.

The above algorithm will descend in the limit to vectors $\mathbf{d}^*$ and $\mathbf{w}^*$ satisfying the conditions of Theorem 5, as shown by Shapiro [13]. Note, however, that it is possible for this to correspond to a local rather than a global minimum, so that ρ_* need not be attained. Essentially the algorithm corresponds to the solution of a sequence of problems of the type considered in § 2, and their efficient treatment is important to the effectiveness of the algorithm.

To illustrate, consider its application to Example 5 above, starting from $\mathbf{w} = \mathbf{e}$ (correctly normalized) with $\mathbf{d}$ initially set also to $\mathbf{e}$. For subsequent iterations, the initial $\mathbf{d}$ for Step 2 is the current optimal value; in all iterations, $r = 1$, and the smallest eigenvalue in Step 4 is simple so that the new $\mathbf{w}$ is uniquely defined. Table 6 shows the progress of the algorithm, including the number of iterations (No.) at each inner iteration (which is in fact equal to the total number of problems (2.8) and (2.14) solved) to obtain six-figure accuracy. The final vector $\mathbf{d}^*$ was $(0.211324, 0.211324, 0.316988)^T$, with $\rho_* = 0.732051$. As explained before, this example is of interest mainly because $\rho^* \neq \rho_*$, so that solving the dual problem is not sufficient to give the primal solution. The eigenvalues of $S^{-1} D^*$ are 0.211325, 0.267949, and 1.0. Of course, $\mu = 0.267949$ is not the smallest eigenvalue, otherwise the primal and dual solutions would coincide, which is not possible

TABLE 6

k	$w_1^{(k)}$	$w_2^{(k)}$	$w_3^{(k)}$	No.	$f^{(k)}$	$\gamma^{(k)}$
1	1.154700	1.154700	1.154700	1	1.0	1.0
2	−0.595141	−1.074397	0.192350	6	0.579403	0.509164
3	−0.860357	−0.860357	0.248234	5	0.493477	0.470450
4	−0.819342	0.819342	0.312027	1	0.447548	0.412164
5	−0.752661	−0.752661	0.409148	1	0.377666	0.322777
6	0.639188	0.639187	−0.558846	5	0.285585	0.267949
7	0.563016	0.563016	−0.650115	4	0.267949	0.267949

TABLE 7

k	$w_1^{(k)}$	$w_2^{(k)}$	$w_3^{(k)}$	No.	$f^{(k)}$	$\gamma^{(k)}$
1	0.150958	−0.770395	0.619437	5	0.333333	0.326935
2	−0.157792	0.714938	−0.684101	4	0.326694	0.326425
3	−0.164185	0.714903	−0.682879	2	0.326145	0.325829
4	−0.171395	0.714831	−0.681472	2	0.325500	0.325124
5	−0.179610	0.714703	−0.679836	2	0.324731	0.324277
6	−0.189080	0.714498	−0.677905	2	0.323800	0.323242
7	−0.200148	0.714176	−0.675588	2	0.322646	0.321949
8	−0.213307	0.713679	−0.672748	3	0.321201	0.320292
9	−0.229278	0.712904	−0.669184	2	0.319317	0.318098
10	−0.249176	0.711679	−0.664552	3	0.316779	0.315071
11	0.274814	−0.709667	0.658278	3	0.313199	0.310658
12	0.309351	−0.706167	0.649280	3	0.307824	0.303719
13	0.358780	−0.699541	0.635301	3	0.299031	0.291572
14	0.435686	−0.685273	0.610916	3	0.282792	0.267949
15	0.563016	−0.650115	0.563016	5	0.267949	0.267949

for this example. In general, if the second smallest eigenvalue is obtained in a situation where the dual solution does not also provide the primal solution, then it may be assumed that the primal problem has been correctly solved. However, the primal solution process may well provide some other eigenvalue, having converged to a local minimum.

Since it is usually much easier to solve the dual problem, then it is reasonable to suppose that that problem is always solved first. Therefore, a dual solution is available to provide starting information to the primal algorithm. If the primal solution is not obtained, then, as explained above, the smallest eigenvalue of $S^{-1}D$ is not simple, and the corresponding eigenvector is not uniquely defined. However, it might seem appropriate to take one of the eigenvectors as $\mathbf{w}^{(1)}$ for the primal solution. The effect of doing this is illustrated in Table 7, and it is seen that convergence in this case is actually much slower, although again only one solution of (2.8), (2.14) was required at each iteration. This example demonstrates, therefore, that the method has the potential for slow convergence. Notice that the answer differs only in the ordering of the components.

5. Concluding remarks. The purpose of this paper has been to demonstrate how recent developments in the provision of methods for nondifferentiable optimization problems involving eigenvalue constraints can be used to facilitate the solution of problems of minimum trace factor analysis. Details of some possible algorithms have been presented and numerical results given for some problems that have appeared in the literature. The intention at this stage has not been to provide extremely robust methods that will cope in a satisfactory manner with all eventualities. However, it is hoped that the analysis and examples illustrate that the methods are potentially very powerful for solving the classes of problems considered. In fact, for problems with a fixed weight vector and for the dual problem, it would seem that algorithms of the general type described are likely to be optimal. However, the method used to solve the primal problem involves a sequence of infinite problems, and this is not entirely satisfactory. Also, the rate of convergence of the outer algorithm can be slow, although in the examples tried only a small number of problems (2.1) was required in total. It would be interesting to establish whether or not the primal can be solved by the application of a sequence of finite problems (for example, quadratic programming problems), although at the present time this remains an open question. Another aspect of the primal is that it is not a convex problem (unlike the dual) and so the possibility of convergence to a local minimum is left open.

REFERENCES

[1] P. M. BENTLER, *A lower-bound method for the dimension-free measurement of internal consistency*, Social Science Research, 1 (1972), pp. 343–357.

[2] P. M. BENTLER AND J. A. WOODWARD, *Inequalities among lower bounds to reliability: With applications to test construction and factor analysis*, Psychometrica, 45 (1980), pp. 249–267.

[3] G. DELLA RICCIA AND A. SHAPIRO, *Minimum rank and minimum trace of covariance matrices*, Department of Mathematics, Ben-Gurion University of the Negev, Beer Sheva, Israel, 1980.

[4] R. FLETCHER, *Semidefinite matrix constraints in optimization*, SIAM J. Control Optim., 23 (1985), pp. 493–513.

[5] S. FRIEDLAND, J. NOCEDAL, AND M. L. OVERTON, *The formulation and analysis of numerical methods for inverse eigenvalue problems*, SIAM J. Numer. Anal., 24 (1987), pp. 634–667.

[6] P. H. JACKSON AND C. C. AGUNWAMBA, *Lower bounds for the reliability of the total score on a test composed of non-homogeneous items: I. Algebraic lower bounds*, Psychometrica, 42 (1977), pp. 567–578.

[7] W. LEDERMANN, *On a problem concerning matrices with variable diagonal elements*, Proc. Roy. Soc. Edinburgh, 60 (1939), pp. 1–17.

[8] J. NOCEDAL AND M. L. OVERTON, *Numerical methods for solving inverse eigenvalue problems*, in Numerical Methods, V. Pereyra and A. Reinoza, eds., Lecture Notes in Mathematics 1005, Springer-Verlag, Berlin, 1983, pp. 212–226.

[9] M. L. OVERTON, *On minimizing the maximum eigenvalue of a symmetric matrix*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 256–268.

[10] ———, *Large scale optimization of eigenvalues*, Tech. Report 505, Courant Institute of Mathematical Sciences, New York University, New York, 1990.

[11] M. L. OVERTON AND R. S. WOMERSLEY, *Second derivatives for optimizing eigenvalues of symmetric matrices*, manuscript.

[12] R. T. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, New York, 1970.

[13] A. SHAPIRO, *Weighted minimum trace factor analysis*, Psychometrica, 47 (1982), pp. 243–264.

[14] J. M. F. TEN BERGE, T. A. B. SNIJDERS, AND F. E. ZEGERS, *Computational aspects of the greatest lower bound to the reliability and constrained minimum trace factor analysis*, Psychometrica, 46 (1981), pp. 201–213.

[15] G. A. WATSON, *An algorithm for optimal l_2 scaling of matrices*, IMA J. Numer. Anal., 11 (1991), pp. 481–492.

[16] B. WOODHOUSE AND P. H. JACKSON, *Lower bounds for the reliability of the total score on a test composed of nonhomogeneous items: A search procedure to locate the greatest lower bound*, Psychometrica, 42 (1977), pp. 579–591.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1054–1066, October 1992

004

COMPUTING THE STRUCTURED SINGULAR VALUE*

G. A. WATSON†

Abstract. The concept of the structured singular value was introduced by J. C. Doyle as a tool for the analysis and synthesis of feedback systems with structured uncertainties. There are different ways in which this quantity can be computed and some methods are considered here. One method, due to Doyle, involves minimizing the largest singular value of a given matrix: this is a convex problem, which can give the structured singular value in some cases; otherwise it provides an upper bound. Another approach leads to a real minimax problem: this is nonconvex, and solves the original problem if the global solution is found; otherwise it gives a lower bound. Different formulations of this latter problem have been presented. In this paper, the minimax problem is developed as a special case of a class of problems whose solutions lead to generalizations of the structured singular value, when different matrix norms are permitted in the original problem statement. Efficient computational methods are developed, and some numerical results are presented.

Key words. structured singular value, minimizing maximum singular value, minimax optimization

AMS(MOS) subject classifications. 65F30, 65K10

1. Introduction. In multivariable control theory, methods based on singular values are well known for the treatment of feedback systems. For the development of multiloop generalizations of classical single-loop techniques, however, conventional singular value methods have serious limitations because they do not exploit information on the structure of the underlying perturbations. Therefore, the concept of structured singular value was introduced by Doyle [2] for the analysis and synthesis of feedback systems with structured uncertainties. It permits the design of control systems under joint robustness and performance specifications, and it complements the H^∞ approach to control system design. For some appropriate references, see [2].

There are different ways in which the structured singular value can be computed. The main purpose of this paper is to examine some algorithms which appear to be more efficient than others that have been used, and also to give some numerical examples. In the rest of this introductory section, the structured singular value is defined, and some notation is introduced. In § 2, a particular approach to the solution of the problem, due to Doyle [2], is considered, and some numerical results of the application of an algorithm to some problems are given. The problem solved is a convex problem, which always provides an upper bound for the structured singular value, and is equal to it in some cases. In § 3, a method is considered based on exploiting the equivalence of the original problem to a (smooth) minimax problem. This result, in a slightly different form, first appeared in Fan and Tits [4], but the development given here is one that is relevant to a much wider class of problems and contains the known results as a special case. The minimax problem is no longer convex, but any local solution gives a lower bound for the structured singular value, and is equal to it if the global solution is obtained. Numerical results are given for an algorithm based on this approach, which may be used as a comparison with those previously obtained. However, it may be desirable to use both techniques to generate both upper and lower bounds, so the methods need not necessarily be regarded as competitors.

Let $M \in C^{n\times n}$ be any complex square matrix, and let $\mathbf{k} = (k_1, \cdots, k_m)^T \in Z_+^m$ be a vector of positive integers summing to n. Let $\|\cdot\|$ denote the l_2 operator or spectral

* Received by the editors September 8, 1989; accepted for publication (in revised form) March 26, 1991.
† Department of Mathematics and Computer Science, University of Dundee, Dundee DD1 4HN, Scotland (gawatson@maths-and-cs.dundee.ac.uk).

norm on matrices of any dimension, and for any positive scalar δ (possibly ∞) define

$$X_\delta = \{\text{block diag}\,(\Delta_1, \cdots, \Delta_m)\colon \Delta_i \in C^{k_i \times k_i} \text{ satisfying } \|\Delta_i\| < \delta\}.$$

DEFINITION 1. The structured singular value $\mu(M)$ of $M \in C^{n\times n}$ with respect to block structure $\mathbf{k}$ is the positive number μ having the property that

$$\det(I + M\Delta) \neq 0 \quad \forall \Delta \in X_\delta \quad \text{iff } \delta\mu < 1.$$

Alternatively, $\mu(M)$ is zero if there is no $\Delta \in X_\infty$ such that $\det(I + M\Delta) = 0$. Otherwise,

$$\mu(M) = (\min_{\Delta \in X_\infty}\{\|\Delta\|\colon \det(I + M\Delta) = 0\})^{-1}. \tag{1.1}$$

It is possible to extend the above definition to allow repeated blocks; in other words, to force equality in some of the Δ_i: this case is not considered here. It follows from (1.1) that the structured singular value can be computed by considering the problem

$$\text{find } \Delta \in X_\infty \quad \text{to minimize } \|\Delta\| \text{ subject to } \det(I + M\Delta) = 0. \tag{1.2}$$

For the special case when $m = 1$, it may be shown that $\mu(M) = \|M\|$, in other words, the largest singular value of M, and this explains the naming of $\mu(M)$ in the more general structured situation described above. It is assumed in what follows that the feasible region of (1.2) is nonempty.

2. The method of Doyle. Let D be the set of diagonal matrices

$$\mathrm{D} = \{\text{block diag}\,(d_1 I_{k_1}, \cdots, d_m I_{k_m})\colon d_i \in \mathbb{R},\ i = 1, \cdots, m\}.$$

Then for any $D \in \mathrm{D}$, and any $\Delta \in X_\infty$ feasible in (1.2), there exists $\mathbf{x}$, $\mathbf{x}^H\mathbf{x} = 1$ such that

$$(I + e^D M e^{-D}\Delta)\mathbf{x} = 0.$$

Thus

$$\|\Delta\|^{-1} \leqq \|e^D M e^{-D}\|,$$

and it follows that

$$\mu(M) \leqq \inf_{D\in\mathrm{D}} \|e^D M e^{-D}\|. \tag{2.1}$$

The following result is given by Doyle [2].

THEOREM 1. *Equality holds in* (2.1) *if* $m \leqq 3$ *or the minimum on the right-hand side is attained with the largest singular value being simple.*

It follows that the structured singular value can often be computed by solving the problem

$$\text{minimize}_{D\in\mathrm{D}} \|e^D M e^{-D}\|. \tag{2.2}$$

Let $\mathbf{d} = (d_1, \cdots, d_{m-1})^T$ and $A\,(=A(\mathbf{d})) = e^D M e^{-D}$, and let

$$A_i = \frac{\partial A}{\partial d_i}, \qquad i = 1, \cdots, m-1,$$

$$A_{ij} = \frac{\partial^2 A}{\partial d_i \partial d_j}, \qquad i, j = 1, 2, \cdots, m-1,$$

where it is assumed that the degree of freedom in (2.2) is used up by fixing $d_m = 0$. Let the singular value decomposition of A be

$$A = U\Sigma V^H,$$

where $\Sigma = \text{diag}\{\sigma_1 = \cdots = \sigma_k > \sigma_{k+1} \geqq \cdots \geqq \sigma_n\}$, U and V are unitary matrices, and the superscript H denotes the complex conjugate transpose. Let $\mathbf{u}_i$, $\mathbf{v}_i$ denote the ith column of U and V, respectively, and let U_1, V_1 denote the first k columns of U, V, respectively (which are, of course, not uniquely defined if $k > 1$). Let $\partial\|\cdot\|$ denote the subdifferential of $\|\cdot\|$,

$$\partial\|A\| = \{G \in C^{n\times n}: \|B\| \geqq \|A\| + Re\,\text{trace}\,G^H(B - A), \text{ for all } B \in C^{n\times n}\}.$$

Then it is readily established that $G \in \partial\|A\|$ if and only if

(i) $\|G\|^* \leqq 1$,

(ii) $Re\,\text{trace}\,(G^HA) = \|A\|$,

where $\|\cdot\|^*$ denotes the dual norm.

The following result may be obtained from standard analysis carried over from the real vector case (see, for example, Rockafellar [10]).

LEMMA 1 (*directional derivative*).

$$\lim_{\gamma\to 0+} \frac{\|A(\mathbf{d} + \gamma\mathbf{s})\| - \|A(\mathbf{d})\|}{\gamma} = \max_{G\in\partial\|A\|} \sum_{j=1}^{m-1} d_j Re\,\text{trace}\,(G^HA_j).$$

LEMMA 2 (*characterization of* $\partial\|A\|$; *see, for example, Berens and Finzel* [1]). *$G \in \partial\|A\|$ if and only if $G = U_1\Lambda V_1^H$, where Λ is a $k \times k$ Hermitian positive semidefinite matrix with trace* $(\Lambda) = 1$.

Now define the $k \times k$ Hermitian matrices

$$H_j = \tfrac{1}{2}(U_1^HA_jV_1 + V_1^HA_j^HU_1), \qquad j = 1, 2, \cdots, m-1.$$

Also, let $M_j \in C^{k_j\times n}$ refer to the jth block row of M, let $\mathbf{x}_j \in C^{k_j}$ denote the corresponding elements of $\mathbf{x} \in C^n$, and let U_{1j}, V_{1j} denote the jth block row of U_1 and V_1, respectively.

THEOREM 2. *A vector $\mathbf{d} \in \mathbb{R}^{m-1}$ solves* (2.2) *if and only if there exists a $k \times k$ Hermitian positive semidefinite matrix Λ, with trace* $(\Lambda) = 1$, *such that*

$$\text{trace}\,(\Lambda H_j) = 0, \qquad j = 1, 2, \cdots, m-1. \tag{2.3}$$

Proof. Using Lemma 1, it follows that necessary conditions for $\mathbf{d}$ to be a solution are that there exists $G \in \partial\|A\|$ such that

$$Re\,\text{trace}\,(G^HA_j) = 0, \qquad j = 1, \cdots, m-1.$$

That (2.3) is necessary follows from Lemma 2 and the definition of the matrices H_j. The result is completed from the fact that the objective function of (2.2) may be shown to be a convex function of $\mathbf{d}$ [11], so that the necessary conditions are also sufficient. □

COROLLARY. *A vector $\mathbf{d} \in \mathbb{R}^{m-1}$ solves* (2.2) *if and only if there exist real numbers $\gamma_i \geqq 0$, $i = 1, \cdots, k$, summing to* 1 *and vectors $\mathbf{c}_i \in C^k$, $\mathbf{c}_i^H\mathbf{c}_i = 1$, $i = 1, \cdots, k$ such that*

$$\sum_{i=1}^{k} \gamma_i\|U_{1j}\mathbf{c}_i\|^2 = \sum_{i=1}^{k} \gamma_i\|V_{1j}\mathbf{c}_i\|^2, \qquad j = 1, \cdots, m.$$

Proof. This follows by writing the singular value decomposition of Λ as

$$\Lambda = \sum_{i=1}^{k} \gamma_i\mathbf{c}_i\mathbf{c}_i^H,$$

and using the specific form of A. Note that the result for $j = m$ is a consequence of the others. □

THEOREM 3. *Let* $\mathbf{d}$ *solve* (2.2) *and assume that there is a rank-one matrix* Λ *satisfying* (2.3), *say* $\Lambda = \mathbf{c}\mathbf{c}^H$, *where* $\mathbf{c}^H\mathbf{c} = 1$. *Let*

$$\Delta_j = -\frac{e^{-d_j}V_{1j}\mathbf{c}(M_j e^{-D}V_1\mathbf{c})^H}{\|M_j e^{-D}V_1\mathbf{c}\|^2}, \qquad j = 1, \cdots, m, \tag{2.4}$$

except that Δ_j *is set to zero if the corresponding denominator is zero. Then equality holds in* (2.1) *and* Δ *defined by* (2.4) *solves* (1.2).

Proof. By virtue of the fact that $\mathbf{d}$ solves (2.2), and the assumption about Λ, it follows from the above corollary that

$$\|V_{1j}\mathbf{c}\| = \|U_{1j}\mathbf{c}\|, \qquad j = 1, \cdots, m. \tag{2.5}$$

Let

$$\mathbf{x} = \frac{e^{-D}V_1\mathbf{c}}{\|e^{-D}V_1\mathbf{c}\|}.$$

Now

$$e^D M e^{-D}V_1 = \sigma_1 U_1,$$

so that

$$Me^{-D}V_1\mathbf{c} = \sigma_1 e^{-D}U_1\mathbf{c}.$$

Thus if $M_j e^{-D}V_1\mathbf{c} = 0$ for any j, it follows that $U_{1j}\mathbf{c} = \mathbf{0}$ so that $\mathbf{x}_j = 0$. A straightforward calculation shows that

$$\Delta_j M_j\mathbf{x} = -\mathbf{x}_j, \qquad j = 1, \cdots, m,$$

so that Δ is feasible for (1.2). Furthermore, for any j, $j = 1, \cdots, m$, with $U_{1j}\mathbf{c} \neq \mathbf{0}$,

$$\|\Delta_j\| = \frac{e^{-d_j}\|V_{1j}\mathbf{c}\|}{\|M_j e^{-D}V_1\mathbf{c}\|}$$

$$= \frac{\|V_{1j}\mathbf{c}\|}{\sigma_1\|U_{1j}\mathbf{c}\|},$$

so that $\|\Delta\| = \sigma_1^{-1}$. Thus the inequality in (2.1) is also reversed, and the result follows. □

Theorem 1 may be interpreted as a consequence of Theorem 3: Λ has rank one trivially if $k = 1$; in general, what is required for (2.5) to hold is the convexity of the set

$$\{\mathbf{s}: s_j = \mathbf{c}^H(U_{1j}^H U_{1j} - V_{1j}^H V_{1j})\mathbf{c}, j = 1, \cdots, m, \mathbf{c}^H\mathbf{c} = 1\}.$$

There is nothing to prove when $m = 1$, because $\|V_{1j}\| = \|U_{1j}\| = 1$, and convexity is established by Doyle [2] for $m = 2$ and 3. An advantage of the particular form of the statement in Theorem 3 is that if a general algorithm is used to solve (2.2), then an explicit Λ is likely to be available, so that it would be possible to check (when $k > 1$) for a solution to (1.2). In addition, an explicit form of the minimizing Δ is provided.

Note that at any $\mathbf{d}$ with $k = 1$, G is unique, and

$$H_j = \mathbf{u}_1^H A_j \mathbf{v}_1 = \frac{\partial}{\partial d_j}\|A\|, \qquad j = 1, 2, \cdots, m-1,$$

so that the conditions of Theorem 2 just reduce to the usual zero derivative conditions. In particular, it is easy to define descent directions for $\|A\|$ at such points. Descent

directions may also be defined for arbitrary points $\mathbf{d}$: for any $\mathbf{d} \in \mathbb{R}^{m-1}$, let

$$K = \{\mathbf{s} \in \mathbb{R}^{m-1} : s_j = \text{trace}(\Lambda H_j),\ j = 1, \cdots, m-1,\ \Lambda \geqq 0,\ \text{trace}(\Lambda) = 1\},$$

where here (and subsequently) we use the convention that $\Lambda \geqq 0$ means that the matrix Λ is Hermitian, positive semidefinite.

THEOREM 4. *For given* $\mathbf{d}$, *let* $-\mathbf{s}^* \in \mathbb{R}^{m-1}$ *solve*

$$\text{find } \mathbf{s} \in K \quad \text{to minimize } \|\mathbf{s}\|. \tag{2.6}$$

Then either $\mathbf{s}^* = \mathbf{0}$, *in which case* $\mathbf{d}$ *solves* (2.2), *or* $\mathbf{s}^*$ *is a descent direction for* $\|A\|$ *at* $\mathbf{d}$.

Proof. The first part follows immediately from Lemma 2. Assume that $-\mathbf{s}^* \neq 0$ solves (2.6). Then using convexity,

$$\mathbf{s}^{*T}\mathbf{t} < 0 \quad \forall \mathbf{t} \in K,$$

so that

$$\max_{\substack{\Lambda \geqq 0 \\ \text{trace}(\Lambda) = 1}} \sum_{j=1}^{m-1} s_j^* \,\text{trace}(\Lambda H_j) < 0,$$

and the descent property is established using Lemmas 1 and 2. □

At any point $\mathbf{d}$ where the largest singular value of A is simple, first and second derivatives of the objective function in (2.2) exist and may be calculated. These are

$$\frac{\partial \sigma_1}{\partial d_i} = \mathbf{u}_1^H A_i \mathbf{v}_1,$$

$$\frac{\partial^2 \sigma_1}{\partial d_i \partial d_j} = Re \sum_{l=2}^{n} \frac{\sigma_1(\alpha_{1il}\overline{\alpha_{1jl}} + \alpha_{li1}\overline{\alpha_{lj1}}) + \sigma_l(\alpha_{1il}\alpha_{lj1} + \alpha_{li1}\alpha_{1jl})}{\sigma_1^2 - \sigma_l^2} + Re\,\mathbf{u}_1^H A_{ij} \mathbf{v}_1,$$

where

$$\alpha_{ijk} = \mathbf{u}_i^H A_j \mathbf{v}_k,$$

for all relevant i, j, k, and the bar denotes the complex conjugate. Thus, if the largest singular value is simple at a solution, Newton's method is available, and so locally a second-order rate of convergence is possible. Because the objective function in (2.2) is convex, the Newton step is a descent step, so that a globally convergent algorithm can be obtained by using a line search. However, if the largest singular value at the solution has multiplicity greater than one, these derivatives are not defined there and, in particular, the solution is no longer a point of attraction for Newton's method.

The problem of dealing with multiplicity of σ_1 can be approached in various ways. For example, the steepest descent step of Theorem 4 is always available (at least in theory), leading to a generally applicable first-order method. The problem (2.6) of computing the steepest descent step is, of course, not a finite problem, but different methods may be used for its solution: for example, algorithms are given in [2] and [3]. A first-order method for (2.2) is also given by Polak and Wardi [9]. An algorithm for (2.2) based on the use of Newton's method, which reverts to the steepest descent step in the presence of multiple singular values, is given by Fan [3]. The problem of recovering a second-order convergence rate, when there is multiplicity of the largest singular value at the solution of problems similar to (2.2), has recently received some attention. An algorithm based on work of Friedland, Nocedal, and Overton [7] on numerical methods for inverse eigenvalue problems, and involving the solution of a sequence of quadratic

programming problems, has been developed by Overton [8]: it seems likely to lead to a very efficient method for solving problems of the type (2.2) under very general circumstances.

However, the interesting thing about the class of problems (2.2) is that the usual situation appears to be that the second alternative in Theorem 1 holds, so that the solution to (2.2) occurs for a simple maximum singular value, and this seems to be a phenomenon that is independent of the size of m. Not only does this make the computational process easier (and obviate, or at least reduce, the need for sophisticated methods), it also guarantees that the structured singular value is found even when $m > 3$. Therefore, variants of Newton's method appear to be very effective in solving (1.2), and numerical results were obtained for the application of Newton's method to a range of problems with real M. In the event that the full Newton step did not reduce the function, a strategy involving successive halving of the step length was used to ensure a reduction. The initial approximation $\mathbf{d} = \mathbf{0}$ appeared to be an excellent one for the examples tried, so that only local properties of the method were being tested, and the expected rapid convergence to the solution was obtained in a few iterations.

Some numerical results are displayed in Table 1 for M given by the Fortran random number generator RAND. The singular value decompositions were obtained from the NAG subroutine F02WEF, and the system of equations defining the Newton step was solved by the NAG subroutine F04ASF. In the results given, k_i was constant for each pair of values of n, m. Initial and final values of σ_1 are given (the blanks imply that the same initial approximation as in the line above was used), together with the number of iterations (I) and seconds of CPU time for a Fortran program run on a SUN 3/50 to achieve the stopping criterion of $\|\mathbf{s}\|_\infty < 0.000001$, where $\mathbf{s}$ is the Newton step. Double precision was used because this was required by the system for calls to the NAG subroutines. The times were obtained from the TIME function.

It is known that counterexamples exist to the conjecture that $k = 1$ *always* at solutions to (2.2); therefore, using a method based on this assumption, holding is not a completely foolproof strategy. However, for the purposes of numerical comparisons with alternative methods for (1.2), the quoted computer times give lower bounds on the times taken for any more sophisticated algorithm to solve the problems; also, the occurrence of multiple σ_1 when $m > 3$ means that even if (2.2) is solved correctly, the structured singular value may not be obtained.

TABLE 1

k_i	n	m	I	CPU	σ_1^0	σ_1^*
5	10	2	4	2.2	5.06351	5.05416
2	10	5	5	4.8		5.04482
1	10	10	5	9.0		5.01613
10	20	2	3	8.2	10.2930	10.2904
5	20	4	4	16.2		10.2828
4	20	5	4	19.0		10.2614
2	20	10	5	43.7		10.2330
1	20	20	6	129.1		10.2096
15	30	2	3	25.4	15.4437	15.4412
10	30	3	3	31.1		15.4419
6	30	5	4	56.5		15.4282
3	30	10	5	125.5		15.3970
2	30	15	5	194.3		15.3164
1	30	30	7	703.1		15.2664

3. An equivalent smooth minimax problem and some generalizations. The solution of the problem (2.2) seems an attractive way of computing the structured singular value, particularly if it can be treated effectively as a smooth convex unconstrained optimization problem in just $(m-1)$ variables. However, it involves a complete singular value decomposition of the $n \times n$ matrix A at each iteration, and this becomes increasingly expensive with increasing n. In this section, an alternative approach is considered that avoids these expensive decompositions. In fact, the method of development given here has the additional advantage that it readily generalizes to enable (1.2) to be solved for other norms on the matrix Δ. Of interest here is an important range of unitarily invariant norms; in particular, we consider the problem

$$\text{(3.1)} \qquad \text{find } \Delta \in X_\infty \quad \text{to minimize } \|\Delta\|_s \text{ subject to } \det(I + M\Delta) = 0,$$

with $\|\cdot\|_s$ defined by

$$\|\Delta\|_s = \|\sigma(\Delta)\|_p,$$

where $\sigma(\Delta)$ is a vector in $\mathbb{R}^n$ whose components are the singular values of Δ, and where the norm on $\mathbb{R}^n$ is the usual l_p vector norm, $p \geqq 1$: this matrix norm is usually referred to as the Schatten-p norm. For this more general problem, $\mu_s(M)$ denotes the expression analogous to that defined by (1.1). Clearly, the special case $p = \infty$ gives (1.2); another important norm is the Frobenius norm, which corresponds to $p = 2$. As additional pieces of notation, let $\sigma^{(i)}(\Delta_i)$ denote the vector in $\mathbb{R}^{k_i}$ whose components are the k_i singular values of Δ_i for each i, $i = 1, \cdots, m$, and define

$$\|\Delta_i\|_s = \|\sigma^{(i)}(\Delta_i)\|_p, \qquad i = 1, \cdots, m,$$

where the subscript s is used to reflect the fact that although the matrices here are of different dimension, the norm is being defined in the same way. It follows from these definitions that $\|\Delta\|_s = \|\mathbf{z}\|_p$, where $\mathbf{z} \in \mathbb{R}^m$ with $z_i = \|\Delta_i\|_s$, $i = 1, \cdots, m$.

Now consider the problem

$$\begin{aligned} &\text{minimize } \|\mathbf{h}\|_p \\ \text{(3.2)} \qquad &\text{subject to } \|\mathbf{x}_i\| \leqq h_i \|M_i \mathbf{x}\|, \qquad i = 1, \cdots, m, \\ &\mathbf{x}^H \mathbf{x} = 1, \end{aligned}$$

where unadorned norms are l_2 norms just as before, and where $\mathbf{h} \in \mathbb{R}^m$, with components h_i, $i = 1, \cdots, m$. The connection of (3.2) with (3.1) is the subject of the following theorem.

THEOREM 5. *Let* $\hat{\mathbf{x}}$, $\hat{\mathbf{h}}$ *solve* (3.2). *Then* $\mu_s(M) = \|\hat{\mathbf{h}}\|_p^{-1}$, *and* (3.1) *is solved by taking* $\Delta = \hat{\Delta}$, *where*

$$\hat{\Delta}_i = -\frac{\hat{\mathbf{x}}_i \hat{\mathbf{x}}^H M_i^H}{\|M_i \hat{\mathbf{x}}\|^2} \quad \text{if } M_i \hat{\mathbf{x}} \neq \mathbf{0};$$

otherwise,

$$\hat{\Delta}_i = 0.$$

Proof. Let Δ^* solve (3.1). Then since $\det(I + M\Delta^*) = 0$, it follows that $\det(I + \Delta^* M) = 0$, and so there exists $\mathbf{x}^*$, $\mathbf{x}^{*H}\mathbf{x}^* = 1$, such that

$$\Delta_i^* M_i \mathbf{x}^* + \mathbf{x}_i^* = \mathbf{0}, \qquad i = 1, \cdots, m.$$

Thus

$$\text{(3.3)} \qquad \|\mathbf{x}_i^*\| \leqq \|\sigma^{(i)}(\Delta_i^*)\|_p \|M_i \mathbf{x}^*\|, \qquad i = 1, \cdots, m,$$

so that $\mathbf{x}^*$ and $h_i^* = \|\sigma^{(i)}(\Delta_i^*)\|_p$, $i = 1, \cdots, m$, are feasible for (3.2). Now a straightforward calculation shows that

$$(I + \hat{\Delta}M)\hat{\mathbf{x}} = \mathbf{0}$$

so that

$$\det(I + M\hat{\Delta}) = 0,$$

and $\hat{\Delta}$ is feasible for (3.1). Furthermore,

$$\|\hat{\Delta}_i\|_s = \frac{\|\hat{\mathbf{x}}_i\|}{\|M_i\hat{\mathbf{x}}\|} \quad \text{if } M_i\hat{\mathbf{x}} \neq \mathbf{0}, \tag{3.4}$$

because $\hat{\Delta}_i$ is a rank-one matrix. Thus

$$\begin{aligned}
\|\Delta^*\|_s &\leqq \|\hat{\Delta}\|_s \\
&= \|\hat{\mathbf{z}}\|_p, \quad \text{say, where } \hat{z}_i = \|\hat{\Delta}_i\|_s, \quad i = 1, \cdots, m, \\
&\leqq \|\hat{\mathbf{h}}\|_p \\
&\leqq \|\mathbf{h}^*\|_p, \quad \text{by definition of } \hat{\mathbf{h}}, \\
&= \|\sigma(\Delta^*)\|_p \\
&= \|\Delta^*\|_s.
\end{aligned}$$

The result follows. □

It is a consequence of this theorem that $\hat{\Delta}_i$ is always a rank-one matrix, so that $\hat{\Delta}$ is rank m. However, the solution to (3.1) is not unique: for example, let $\hat{\mathbf{x}}$ again solve (3.2) and let R_i be $k_i \times k_i$ unitary matrices satisfying

$$\|\hat{\mathbf{x}}_i\| R_i M_i \hat{\mathbf{x}} = \|M_i\hat{\mathbf{x}}\|\hat{\mathbf{x}}_i \quad \text{if } M_i\hat{\mathbf{x}} \neq \mathbf{0}.$$

Let

$$\tilde{\Delta}_i = -\frac{\|\hat{\mathbf{x}}_i\|}{\|M_i\hat{\mathbf{x}}\|} R_i \quad \text{if } M_i\hat{\mathbf{x}} \neq \mathbf{0};$$

otherwise,

$$\tilde{\Delta}_i = 0.$$

Then

$$\begin{aligned}
&(I + \tilde{\Delta}M)\hat{\mathbf{x}} = \mathbf{0}, \\
&\|\tilde{\Delta}_i\|_s = \frac{\|\hat{\mathbf{x}}_i\|}{\|M_i\hat{\mathbf{x}}\|} \quad \text{if } M_i\hat{\mathbf{x}} \neq \mathbf{0},
\end{aligned}$$

and a similar argument to that used in Theorem 5 shows that $\tilde{\Delta}$ is also a solution of (3.1).

It is clear that equality will hold in the constraints of (3.2) at a solution. Indeed, if we define

$$h_i(\mathbf{x}) = \frac{\|\mathbf{x}_i\|}{\|M_i\mathbf{x}\|}, \qquad i = 1, 2, \cdots, m,$$

then (3.2) may be restated in the form

$$\text{minimize } \|\mathbf{h}(\mathbf{x})\|_p \quad \text{subject to } \mathbf{x}^H\mathbf{x} = 1,$$

where $\mathbf{h}(\mathbf{x})$ denotes the vector in $\mathbb{R}^m$ with components $h_i(\mathbf{x})$, $i = 1, \cdots, m$. Reverting to the original problem (1.2) (the case $p = \infty$), the structured singular value can therefore be computed by minimizing

$$\max \left\{ \frac{\|\mathbf{x}_i\|}{\|M_i\mathbf{x}\|}, \; i = 1, \cdots, m \right\}.$$

This is a real minimax problem in $2n$ real unknowns, these being the real and imaginary parts of $\mathbf{x} \in C^n$. Note that this formula immediately shows that $\mu(M)$ is the largest singular value of M when $m = 1$; it also provides an intuitively pleasing alternative definition of the structured singular value. An equivalent statement of this problem is

$$\begin{aligned} &\text{minimize } h \\ &\text{subject to } \|\mathbf{x}_i\| \leqq h\|M_i\mathbf{x}\|, \qquad i = 1, \cdots, m, \\ &\mathbf{x}^H\mathbf{x} = 1. \end{aligned} \tag{3.5}$$

The formulation (3.5) is also given in Fan, Tits, and Doyle [5], [6], where it is extended to deal with the case of more general uncertainty structures. In fact, the inequalities in (3.5) actually hold with equality at a solution, and this is very useful from the point of view of the actual calculation. A key result is the following lemma.

LEMMA 3 (see, for example, [2]). *Let $p(\mathbf{d})$ be a polynomial of degree q in $\mathbf{d} \in \mathbb{R}^m$, and consider the problem*:

$$\text{minimize } \|\mathbf{d}\|_\infty \quad \text{subject to } p(\mathbf{d}) = 0.$$

Let $\hat{\mathbf{d}}$ be a solution. Then there exists a solution $\mathbf{d}^*$ such that

$$|d_i^*| = \|\hat{\mathbf{d}}\|_\infty, \qquad i = 1, \cdots, m.$$

The result of the next theorem is essentially a reformulation of results in [4], but a proof is given here for completeness.

THEOREM 6. *There exists a solution to* (3.5) *with the constraints holding with equality.*

Proof. Let $\hat{\mathbf{x}}$ solve (3.5) and define

$$\hat{d}_i = \frac{\|\hat{\mathbf{x}}_i\|}{\|M_i\hat{\mathbf{x}}\|} \quad \text{if } M_i\hat{\mathbf{x}} \neq \mathbf{0};$$

otherwise, $\hat{d}_i = 0$. Then there exists a $k_i \times k_i$ unitary matrix R_i such that

$$\hat{d}_i R_i M_i \hat{\mathbf{x}} = \hat{\mathbf{x}}_i, \qquad i = 1, \cdots, m,$$

or

$$\hat{D}RM\hat{\mathbf{x}} = \hat{\mathbf{x}},$$

where $\hat{D} \in \mathrm{D}$ and $R = \text{block diag}\{R_1, \cdots, R_m\}$. Therefore, $\det(I - \hat{D}RM) = 0$, or $p(\hat{\mathbf{d}}) = 0$, where $\hat{\mathbf{d}} = (\hat{d}_1, \cdots, \hat{d}_m)^T$, with p a polynomial of degree n. Thus $\|\mathbf{d}\|_\infty$ is minimized subject to $p(\mathbf{d}) = 0$ by $\mathbf{d} = \hat{\mathbf{d}}$. By Lemma 3, there is a solution with

$$d_i = \|\hat{\mathbf{d}}\|_\infty, \qquad i = 1, \cdots, m.$$

Thus

$$\det(I - hRM) = 0,$$

where $h = \|\hat{\mathbf{d}}\|_\infty$, It follows that there exists $\mathbf{x}^*$, $\|\mathbf{x}^*\| = 1$ such that

$$(I - hRM)\mathbf{x}^* = \mathbf{0},$$

so that

$$\|\mathbf{x}_i^*\| = h\|M_i\mathbf{x}^*\|, \qquad i = 1, \cdots, m.$$

The result is proved. □

We may consider, therefore, the problem (3.5) posed in the form

$$\begin{aligned} &\text{minimize } \gamma \\ &\text{subject to } \|\mathbf{x}_i\|^2 = \gamma\|M_i\mathbf{x}\|^2, \qquad i = 1, \cdots, m, \\ &\|\mathbf{x}\|^2 = 1. \end{aligned} \tag{3.6}$$

This is a smooth optimization problem in $(2n + 1)$ real variables (γ and the real and imaginary parts of $\mathbf{x}$), and may be solved in a standard way through the solution of a sequence of equality-constrained quadratic programming problems. There is no need for an artificial penalty function, because descent can be obtained with respect to the natural merit function

$$\max_i \frac{\|\mathbf{x}_i\|^2}{\|M_i\mathbf{x}\|^2}. \tag{3.7}$$

A version of this problem is given in [4], having the form

$$\begin{aligned} &\text{maximize } \|M\mathbf{x}\|^2 \\ &\text{subject to } \|\mathbf{x}_i\|^2\|M\mathbf{x}\|^2 = \|M_i\mathbf{x}\|^2, \qquad i = 1, \cdots, m, \\ &\|\mathbf{x}\|^2 = 1. \end{aligned} \tag{3.8}$$

If $\hat{\mathbf{x}}$ solves (3.8), then

$$\mu(M) = \|M\hat{\mathbf{x}}\|.$$

The solution of (3.8) is considered in [4], using a general method for solving constrained optimization problems. Thus advantage is not being taken of the minimax formulation, but nevertheless the numerical results show that this process can still be considerably more efficient than solving (2.2).

An algorithm for (3.6) when M is a real matrix has been implemented as a FORTRAN program to run on the same SUN 3/50 system used for the results of the previous section. At each approximation to the solution, γ was reset to the current value of (3.7), and an equality-constrained quadratic programming problem was generated based on linearizations of the constraints and a quadratic approximation to the objective function of (3.6) with Hessian matrix that of the Lagrangian function. Approximations to the Lagrange multipliers were obtained in a standard way by solving a least-squares problem derived from the Kuhn–Tucker conditions. The linear equality constraints may be eliminated, again in a standard way, by a QR factorization technique, so that it is necessary to work only with the reduced Hessian matrix, and an efficient method can therefore be developed. To improve the global convergence capabilities of the method, at first the Hessian matrix was, in fact, augmented by adding a positive multiple of the unit matrix (initially one), and this multiple was adjusted as the computation proceeded based on the progress of the method.

The solution of the quadratic programming problem is a descent direction for the merit function (3.7), provided that the quadratic Hessian is positive definite (although this is not a necessary condition), and on a successful full step in this direction, the multiple of the unit matrix was reduced by a factor of 4. When less than 0.1, it was set to zero. Descent can be obtained, if necessary, by a line search (here based on successive halving of the step length), or if the solution of the quadratic programming problem is

TABLE 2

k_i	n	m	I	CPU	γ^0	γ^*	$\sqrt{1/\gamma^*}$
5	10	2	6	1.0	0.042554	0.039147	5.05416
2	10	5	6	1.3	0.043077	0.039292	5.04482
1	10	10	4	1.1	0.057499	0.039743	5.01613
10	20	2	5	3.4	0.010195	0.009443	10.2904
5	20	4	5	3.9	0.010836	0.009457	10.2828
4	20	5	5	4.4	0.010821	0.009497	10.2614
2	20	10	6	6.3	0.019213	0.009550	10.2330
1	20	20	4	5.7	0.021693	0.009594	10.2096
15	30	2	5	9.3	0.004355	0.004194	15.4412
10	30	3	6	11.9	0.004246	0.004193	15.4419
6	30	5	6	12.9	0.004506	0.004201	15.4282
3	30	10	6	16.1	0.005398	0.004218	15.3970
2	30	15	6	18.3	0.007870	0.004263	15.3164
1	30	30	4	16.3	0.008553	0.004291	15.2664

not a descent direction, by repeating the iteration with the multiplier restored to one (or multiplied by 4 if appropriate). Eventually, the original Hessian can be used, resulting in a process which may be shown to be asymptotically equivalent to Newton's method for satisfying the Kuhn–Tucker conditions, and so a second-order rate of convergence may be obtained.

Some numerical results are presented in Table 2, which may be used for comparison with those obtained in Table 1. The initial approximation was always taken to be $\mathbf{x} = (1, 1, \cdots, 1)^T$ (suitably normalized), and a comparable stopping criterion was used. Here, too, a very good initial approximation appears to be available, as this choice gives an excellent starting value in every case. For the purposes of comparison, note that the optimal value of γ is the square of the inverse of the structured singular value as defined before, provided that the global (rather than just a local) minimum has been obtained. There is, of course, no guarantee of termination at a global minimum, because (3.6) is not a convex problem, and, in this respect, the method is not as good as that of the previous section. Nevertheless, it would appear from numerical experiments that it is usual for the global minimum to be found.

A few results using the initial approximation $\mathbf{x} = (1, 0, \cdots, 0)^T$ are given in Table 3 to give a better idea of the global capability of the algorithm. This is important, as the matrices used here are not necessarily typical of the kind of matrices that occur in practice. It was found to be useful to allow the multiple of the unit matrix added to the Hessian to go more slowly to zero and, in the results quoted, this value was set to zero if otherwise it would be less than 0.01. Results for some slightly larger problems with different vectors $\mathbf{k}$ are also given in Table 4, again with the initial approximation $\mathbf{x} = (1, 1, \cdots, 1)^T$.

TABLE 3

k_i	n	m	I	CPU	γ^0	γ^*
5	10	2	9	1.5	0.315962	0.039147
4	20	5	15	12.2	0.444655	0.009497
6	30	5	16	33.4	0.342729	0.004201
3	30	10	19	49.2	0.998851	0.004218

TABLE 4

k	n	m	I	CPU	γ^0	γ^*	$\sqrt{1/\gamma^*}$
$\mathbf{k}_1$	40	6	5	23.1	0.002474	0.002395	20.4337
$\mathbf{k}_2$	50	5	6	47.8	0.001585	0.001544	25.4493
$\mathbf{k}_3$	50	8	6	52.6	0.001627	0.001545	25.4411
$\mathbf{k}_4$	60	8	5	70.9	0.001168	0.001079	30.4431
$\mathbf{k}_5$	100	5	5	270.2	0.000423	0.000398	50.1255

For the examples, the vectors $\mathbf{k}$ were chosen, for no particular reason, to be

$$\mathbf{k}_1 = (6, 5, 5, 8, 7, 9),$$

$$\mathbf{k}_2 = (12, 5, 10, 8, 15)^T,$$

$$\mathbf{k}_3 = (8, 5, 7, 4, 9, 3, 6, 8)^T,$$

$$\mathbf{k}_4 = (7, 9, 11, 4, 6, 14, 3, 6)^T,$$

$$\mathbf{k}_5 = (18, 31, 16, 23, 12)^T.$$

It is clear from Tables 2 and 4 that, for the problems considered here, the approach based on (3.6) appears to give a very efficient way of solving (1.2).

4. Concluding remarks. Some methods for computing the structured singular value have been discussed in this paper, and some numerical evidence given to indicate the efficiency of two of these methods. The stopping criteria used for the calculations were based on the step lengths in the increments in $\mathbf{d}$ and $\mathbf{x}$ being sufficiently small, and while this is appropriate if these quantities are required to a particular accuracy, it is generally an exaggeration of the amount of computation required to obtain the same accuracy in the structured singular values; therefore these quantities are being obtained to greater accuracy than the number of figures displayed. Algorithms based on solving (2.2) have the advantage that if this problem is correctly solved, then the structured singular value is always obtained if $m \leqq 3$. Otherwise, it is possible for a solution to (2.2) to fail to provide the correct value, although numerical evidence supports the view that this does not happen much in practice with real matrices. However, a referee has made the comment that the phenomenon of repeated maximum singular values does occur with more frequency for large matrices when the matrices are complex. At worst, an upper bound will be obtained.

The numerical results presented here suggest that algorithms based on (3.6) are considerably more efficient, mainly (although not entirely) because they avoid computationally expensive singular value decompositions, and this offsets the effect of the larger number of unknowns. Of course, the difference becomes more pronounced as m becomes large relative to n. It is clear that the structure of the problem (3.6) makes it computationally attractive, and may be exploited to permit accurate solutions to be obtained in an efficient manner. Another advantage of methods based on this problem is that they have no restriction on the value of m. On the other hand, (3.6) is not a convex problem, so that there is no guarantee that a local and not a global solution is obtained. For the problems used here, an excellent initial approximation is available, and the numerical evidence is that this potential difficulty does not arise. However, the method will, at worst, provide a lower bound.

Although the numerical results quoted here are all for real problems, they should be useful as a guide to the more general situation, and should be a valid basis for a

comparison of algorithms. It seems likely that the relative efficiency of the use of (3.6) over (2.2) would be increased in more general cases, but the extent of this remains to be established. In any event, it may be desirable to include both techniques in any package for solving (1.2) so as to generate both upper and lower bounds.

Finally, the analysis given in § 3 may be further generalized in different directions. For example, the restriction to the particular class of unitarily invariant norms is not necessary, and the results can extend to wider classes of norms occurring in (1.2). There is also no difficulty, in principle, in extending the analysis to the treatment of rectangular matrices M, and with randomly structured matrices Δ.

Acknowledgment. I am grateful to the referees for a number of helpful comments and suggestions.

REFERENCES

[1] H. BERENS AND M. FINZEL, *A continuous selection of the metric projection in matrix spaces*, in Numerical Methods of Approximation Theory, Vol. 8, L. Collatz, G. Meinardus, and G. Nurnberger, eds., Birkhauser-Verlag, Basel, Germany, 1987, pp. 21–29.

[2] J. C. DOYLE, *Analysis of feedback systems with structured uncertainties*, Proc. IEE-D, 129 (1982), pp. 242–250.

[3] M. K.-H. FAN, *An algorithm to compute the structured singular value*, Electrical Engineering Department and Systems Research Center, University of Maryland, College Park, MD, 1988, preprint.

[4] M. K.-H. FAN AND A. L. TITS, *Characterization and efficient computation of the structured singular value*, IEEE Trans. Automat. Control, AC-31 (1986), pp. 734–743.

[5] M. K.-H. FAN, A. L. TITS, AND J. C. DOYLE, *Robustness in the presence of joint parametric uncertainty and unmodeled dynamics,* in Proc. 1988 IEEE American Control Conference, 1988, pp. 1195–1200.

[6] ———, *Robustness in the presence of mixed parametric uncertainty and unmodeled dynamics*, University of Maryland Systems Research Center Tech. Report SRC TR 88-55R1, College Park, MD, 1988; IEEE Trans. Automat. Control, to appear.

[7] S. FRIEDLAND, J. NOCEDAL, AND M. L. OVERTON, *The formulation and analysis of numerical methods for inverse eigenvalue problems*, SIAM J. Numer. Anal., 24 (1987), pp. 634–667.

[8] M. L. OVERTON, *On minimizing the maximum eigenvalue of a symmetric matrix*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 256–268.

[9] E. POLAK AND Y. WARDI, *Nondifferentiable optimization algorithm for designing control systems having singular value inequalities*, Automatica, 18 (1982), pp. 267–283.

[10] R. T. ROCKAFELLAR, *Convex Analysis*, Princeton University Press, Princeton, NJ, 1970.

[11] R. SEZGINER AND M. L. OVERTON, *The largest singular value of $e^{x}A_0e^{-x}$ is convex on convex sets of commuting matrices*, IEEE Trans. Automat. Control, 35 (1990), pp. 229–230.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1067–1084, October 1992

005

EXPONENTIAL CONVERGENCE PROPERTIES OF AUTOCOVARIANCE MATRIX INVERSES AND LATENT VECTOR PREDICTION COEFFICIENTS*

JOHN L. ELTINGE†

Abstract. This paper presents some convergence properties of the inverse of the autocovariance matrix of a vector autoregressive moving average model. Under causality and invertibility conditions, the inverse has bounded on-diagonal blocks and exponentially declining off-diagonal blocks and some elements of the inverse converge to limiting values at exponential rates. These properties lead to similar convergence rates for filter weights and prediction error autocovariances in a latent-variable prediction problem. Uniform convergence rates and the convergence properties of some matrix derivatives are also discussed. These results are useful in developing some distributional properties of estimated autocovariance matrix inverses and estimated latent-variable filter weights.

Key words. time series, vector autoregressive moving average model, filter weights, geometric convergence, latent variables

AMS(MOS) subject classifications. primary 62M10; secondary 62M20

1. Introduction and notation. The mathematical and statistical literature has often noted the exponential convergence properties of some matrix inverses and filter coefficients related to time series analysis. For example, Demko [1] found conditions under which the elements of the inverse of a band matrix decay exponentially away from the main diagonal. Also, de Boor [2, p. 906] discussed the exponential rate of convergence to limiting values of the interior elements of the inverse of an m-banded, bounded, and boundedly invertible matrix; he also [2, pp. 894, 900] reviewed some exponential convergence properties of inverses of banded Toeplitz matrices. In addition, Harrison and Stevens [3, p. 216], Smith [4, pp. 375–376], Anderson and Moore [5, p. 83], and others have noted the close relation between state-space model predictors and exponentially weighted moving average predictors, and have discussed the convergence of optimal filter weights and prediction error variances to time-invariant values. See also [6] and [7].

The purpose of this paper is to obtain similar exponential convergence rates, uniform over a certain parameter set, for the inverse of the autocovariance matrix of a causal and invertible vector autoregressive moving average model, for some related latent-variable filter weights, and for derivatives of the inverse and filter matrices.

These results are based on the following model and definitions. Let $\{\mathbf{Z}_t\}$ be a k-dimensional time series which follows an autoregressive moving average (p, q) model

$$\mathbf{\Phi}(B)\mathbf{Z}_t = \mathbf{\Theta}(B)\mathbf{g}_t, \qquad t \in \mathscr{Z}, \tag{1.1}$$

where the shocks $\{\mathbf{g}_t\}$ are k-dimensional, $\mathbf{g}_t \overset{\text{i.i.d.}}{\sim} (\mathbf{0}, \mathbf{\Sigma}_{gg})$; $\det(\mathbf{\Sigma}_{gg}) \neq 0$; the autoregressive and moving average polynomials are defined by the expressions $\mathbf{\Phi}(B) = \sum_{j=0}^{p} \mathbf{\Phi}_j B^j$ and $\mathbf{\Theta}(B) = \sum_{i=0}^{q} \mathbf{\Theta}_i B^i$, respectively; B is the backshift operator; $\{\mathbf{\Phi}_j, j = 0, 1, \cdots, p\}$ and $\{\mathbf{\Theta}_i, i = 0, 1, \cdots, q\}$ are sequences of $k \times k$ real matrices with $\mathbf{\Phi}_0 = \mathbf{I}_k = \mathbf{\Theta}_0$; and $\mathscr{Z}$ is the set of all integers. Following [8, pp. 408–409], assume that the coefficient matrices

* Received by the editors July 5, 1989; accepted for publication (in revised form) January 15, 1991. Initial work for this paper was done while the author was at the Department of Statistics, Iowa State University, Ames, Iowa. This research was supported in part by National Institutes of Health grant GM39015-01A1.

† Department of Statistics, Texas A&M University, College Station, Texas 77843-3143 (jeltinge@stat.tamu.edu).

of model (1.1) satisfy the causality criterion

$$\det[\Phi(\omega)] \neq 0 \quad \text{for all } \omega \in \mathscr{C} \quad \text{such that } |\omega| \leqq 1, \tag{1.2a}$$

and the invertibility criterion

$$\det[\Theta(\omega)] \neq 0 \quad \text{for all } \omega \in \mathscr{C} \quad \text{such that } |\omega| \leqq 1, \tag{1.2b}$$

where $\mathscr{C}$ is the set of complex numbers. Then $\mathbf{Z}_{(T)} \overset{\text{def}}{=} (\mathbf{Z}_T', \mathbf{Z}_{T-1}', \cdots, \mathbf{Z}_1')'$ has an invertible autocovariance matrix $\operatorname{Var}(\mathbf{Z}_{(T)}) = \mathbf{\Gamma}_{(TT)}$, say.

Convergence results developed below for $\mathbf{\Gamma}_{(TT)}^{-1}$ and related matrices use the following definitions of exponentially declining sequences of matrices and of matrices with bounded on-diagonal blocks and exponentially declining off-diagonal blocks. These definitions use the standard matrix norm $\|\mathbf{B}\| = \sup_{|\mathbf{x}| \leqq 1} |\mathbf{Bx}|$, where $|\mathbf{x}|$ denotes the Euclidean norm of $\mathbf{x}$ (see, e.g., [9, p. 176]). This norm, however, is applied only to fixed-dimensional matrices and blocks, and the causality and invertibility conditions will impose bounds and coefficients of exponential decline that are uniform over all elements of a given block. Consequently, the results developed below also apply if we use other matrix norms, e.g., $\|\mathbf{B}\| = [\operatorname{tr}(\mathbf{B}'\mathbf{B})]^{1/2}$ [10, p. 10].

DEFINITION 1.1. Let $\mathscr{N}$ be the set of nonnegative integers and let $\mathscr{Z}^+$ be the set of positive integers.

(a) A sequence of $r \times k$-dimensional real matrices $\{\mathbf{B}_j, j \in \mathscr{N}\}$ is said to be declining exponentially in j if there exist finite positive real numbers K and c such that for all $j \in \mathscr{N}$, $\|\mathbf{B}_j\| \leqq K \exp(-cj)$.

(b) A set of sequences of $r \times k$-dimensional real matrices $\{\mathbf{B}_{Tj}, j \in \mathscr{N}\}$, $T \in \mathscr{Z}^+$, is said to be declining exponentially in j, uniformly in T, if there exist finite positive real numbers K and c such that for all $j \in \mathscr{N}$, $\sup_{T \in \mathscr{Z}^+} \|\mathbf{B}_{Tj}\| \leqq K \exp(-cj)$.

(c) Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_Q)' \in \mathscr{A}$ be a vector of fixed dimension Q, where $\mathscr{A}$ is a fixed subset of the Q-dimensional real numbers $\mathscr{R}^Q$. Let $\mathscr{A}_0 \subseteq \mathscr{A}$. For each $\boldsymbol{\alpha} \in \mathscr{A}$ and $T \in \mathscr{Z}^+$, let $\{\mathbf{B}_{Tj}(\boldsymbol{\alpha}), j \in \mathscr{N}\}$ be a set of sequences of $r \times k$ real matrices. The set $\{\mathbf{B}_{Tj}(\boldsymbol{\alpha}), j \in \mathscr{N}, T \in \mathscr{Z}^+, \boldsymbol{\alpha} \in \mathscr{A}\}$ is said to be declining exponentially in j, uniformly in $T \in \mathscr{Z}^+$ and $\boldsymbol{\alpha} \in \mathscr{A}_0$, if there exist finite positive real numbers K and c such that for all $j \in \mathscr{N}$, $\sup_{T \in \mathscr{Z}^+} \sup_{\boldsymbol{\alpha} \in \mathscr{A}_0} \|\mathbf{B}_{Tj}(\boldsymbol{\alpha})\| \leqq K \exp(-cj)$.

DEFINITION 1.2. (a) Let $\{\mathbf{A}_T, T \in \mathscr{Z}^+\}$ be a sequence of $Tr \times Tk$-dimensional matrices with (i,j)th $r \times k$ block denoted $\mathbf{A}_{Tij}$, $i, j = 1, 2, \cdots, T$.

(a.i) The sequence $\{\mathbf{A}_T\}$ is said to have bounded on-diagonal blocks and exponentially declining off-diagonal blocks if there exist some positive real numbers K and c such that $\|\mathbf{A}_{Tij}\| \leqq K \exp(-c|i-j|)$ for all $T \in \mathscr{Z}^+$ and all $i, j = 1, 2, \cdots, T$.

(a.ii) The sequence $\{\mathbf{A}_T\}$ is said to be declining exponentially in T if there exist some positive real numbers K and c such that $\|\mathbf{A}_{Tij}\| \leqq K \exp(-cT)$ for all $T \in \mathscr{Z}^+$ and all $i, j = 1, 2, \cdots, T$.

(b) Let $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \cdots, \alpha_Q)' \in \mathscr{A}$ be a vector of fixed dimension Q, where $\mathscr{A}$ is a fixed subset of $\mathscr{R}^Q$. For each $\boldsymbol{\alpha} \in \mathscr{A}$, let $\{\mathbf{A}_T(\boldsymbol{\alpha}), T \in \mathscr{Z}^+\}$ be a sequence of $Tr \times Tk$ matrices with (i,j)th $r \times k$ block denoted $\mathbf{A}_{Tij}(\alpha)$, $i, j = 1, 2, \cdots, T$. Let $\mathscr{A}_0 \subseteq \mathscr{A}$.

(b.i) The set of sequences $\{\mathbf{A}_T(\boldsymbol{\alpha}), T \in \mathscr{Z}^+\}$, $\boldsymbol{\alpha} \in \mathscr{A}$, is said to have bounded on-diagonal blocks and exponentially declining off-diagonal blocks uniformly in $\boldsymbol{\alpha} \in \mathscr{A}_0$ if there exist positive real numbers K and c such that $\sup_{\boldsymbol{\alpha} \in \mathscr{A}_0} \|\mathbf{A}_{Tij}(\boldsymbol{\alpha})\| \leqq K \exp(-c|i-j|)$ for all $T \in \mathscr{Z}^+$ and all $i, j = 1, 2, \cdots, T$.

(b.ii) The set of sequences $\{\mathbf{A}_T(\boldsymbol{\alpha}), T \in \mathscr{Z}^+\}$, $\boldsymbol{\alpha} \in \mathscr{A}$, is said to be declining exponentially in T, uniformly in $\boldsymbol{\alpha} \in \mathscr{A}_0$, if there exist positive real numbers K and c such that $\sup_{\boldsymbol{\alpha} \in \mathscr{A}_0} \|A_{Tij}(\boldsymbol{\alpha})\| \leqq K \exp(-cT)$ for all $T \in \mathscr{Z}^+$, and all $i, j = 1, 2, \cdots, T$.

The constants K and c will be referred to as the "coefficients of exponential decline," and sequences which satisfy Definitions 1.1(c) or 1.2(b) will be said to have coefficients of exponential decline that are uniform in $\alpha \in \mathcal{A}_0$.

The remainder of this paper is organized as follows. Section 2 uses the ideas of [11] and [12] to express the inverse of $\mathbf{\Gamma}_{(TT)}$ as a function of the matrices $\mathbf{\Phi}_j$, $\mathbf{\Theta}_i$, and $\mathbf{\Sigma}_{gg}$. For large T, the decomposition of $\mathbf{\Gamma}_{(TT)}^{-1}$ in § 2 corresponds approximately to an infinite autoregressive form of model (1.1)–(1.2). Section 3 uses the details of this approximation to show that with increasing T, the inverse $\mathbf{\Gamma}_{(TT)}^{-1}$ has bounded on-diagonal blocks and exponentially declining off-diagonal blocks, and that some elements of $\mathbf{\Gamma}_{(TT)}^{-1}$ converge to limiting values at exponential rates. In § 4, the convergence results for autocovariance matrix inverses lead to similar convergence properties for the filter weights used in the minimum mean squared error linear prediction of a time series $\{\mathbf{x}_t\}$ related to the observed sequence $\{\mathbf{Z}_t\}$. Section 5 considers a general parameterization of the $\{\mathbf{Z}_t\}$ and $\{\mathbf{x}_t\}$ models, discusses rates of convergence that are uniform over a compact subspace of the parameter space, and extends the results of the previous two sections to some derivative matrices. Section 6 summarizes the main ideas of the paper. Appendix A presents some implications of the causality and invertibility criteria, and Appendix B gives two lemmas associated with Definitions 1.1 and 1.2, and presents the proofs of the main results in § 5.

2. Inversion of the covariance matrix of a vector autoregressive moving average model. Exact expressions for the inverse of the autocovariance matrix of a univariate autoregressive moving average model may be found in [11]–[14]. Section 2.1 uses slight variants of the arguments and notation of [11] and [12] to extend their results to the multivariate case. Sections 2.2 and 2.3 then discuss properties of two terms of the matrix inverse expression.

2.1. The inverse of the autocovariance matrix. For $s \geqq T > 0$, define the Tk-dimensional random vectors

$$\mathbf{Z}_{s,T} = (\mathbf{Z}_s', \mathbf{Z}_{s-1}', \cdots, \mathbf{Z}_{s-T+1}')' \tag{2.1}$$

and $\mathbf{g}_{s,T} = (\mathbf{g}_s', \mathbf{g}_{s-1}', \cdots, \mathbf{g}_{s-T+1}')'$. Note that T is the number of periods represented in $\mathbf{Z}_{s,T}$ and $\mathbf{g}_{s,T}$, and s is the index of the final period represented. Given a fixed set of $k \times k$ matrices $\{\beta_0, \beta_1, \cdots, \beta_r\}$, define $\mathbf{U}_{n,r}(\beta)$ to be the $nk \times nk$-dimensional block upper triangular matrix with ith $k \times nk$-dimensional row block equal to $[\mathbf{0}_{k\times(i-1)k}, \beta_0, \beta_1, \cdots, \beta_{\min(r,n-i)}, \mathbf{0}_{k\times[n-i-\min(r,n-i)]}]$ and define $\mathbf{L}_r(\beta)$ to be the $rk \times rk$-dimensional block lower triangular matrix with ith $k \times kr$-dimensional row block equal to $[\beta_{r-i+1}, \beta_{r-i+2}, \cdots, \beta_r, \mathbf{0}_{k\times(r-i)k}]$, $i = 1, 2, \cdots, r$. Let $m = \max(p, q)$. Then under model (1.1), for any $s > T > m$,

$$\mathbf{U}_{T,p}(\mathbf{\Phi})\mathbf{Z}_{s,T} = \mathbf{U}_{T,q}(\mathbf{\Theta})\mathbf{g}_{s,T} + \begin{pmatrix} \mathbf{0}_{k(T-q)\times kq} \\ \mathbf{L}_q(\mathbf{\Theta}) \end{pmatrix}\mathbf{g}_{s-T,q} - \begin{pmatrix} \mathbf{0}_{k(T-p)\times kp} \\ \mathbf{L}_p(\mathbf{\Phi}) \end{pmatrix}\mathbf{Z}_{s-T,p}. \tag{2.2}$$

Moreover, $\mathbf{g}_{s,T}$ is uncorrelated with $\mathbf{g}_{s-T,q}$ and with $\mathbf{Z}_{s-T,p}$, so $\mathbf{\Gamma}_{(TT)} = \operatorname{Var}(\mathbf{Z}_{s,T})$ satisfies the relation

$$\begin{aligned} &\mathbf{U}_{T,p}(\mathbf{\Phi})\mathbf{\Gamma}_{(TT)}[\mathbf{U}_{T,p}(\mathbf{\Phi})]' \\ &\quad = \mathbf{U}_{T,q}(\mathbf{\Theta})(\mathbf{I}_T \otimes \mathbf{\Sigma}_{gg})[\mathbf{U}_{T,q}(\mathbf{\Theta})]' + \begin{pmatrix} \mathbf{0}_{k(T-m)\times k(T-m)} & \mathbf{0}_{k(T-m)\times km} \\ \mathbf{0}_{km\times k(T-m)} & \mathbf{V} \end{pmatrix} \end{aligned} \tag{2.3}$$

where $\mathbf{V}$ is the $km \times km$-dimensional covariance matrix of $\mathbf{L}_1\mathbf{g}_{s-T,q} - \mathbf{L}_2\mathbf{Z}_{s-T,p}$,

$$\mathbf{L}_1 = \begin{pmatrix} \mathbf{0}_{k(m-q)\times kq} \\ \mathbf{L}_q(\Theta) \end{pmatrix} \quad \text{and} \quad \mathbf{L}_2 = \begin{pmatrix} \mathbf{0}_{k(m-p)\times kp} \\ \mathbf{L}_p(\Phi) \end{pmatrix}.$$

Define $\mathbf{A} = \mathbf{U}_{T,q}(\Theta)(\mathbf{I}_T \otimes \Sigma_{gg})[\mathbf{U}_{T,q}(\Theta)]'$. Theorem 2.1 presents a general expression for $\Gamma^{-1}_{(TT)}$ in terms of $\mathbf{U}_{T,p}(\Phi)$, $\mathbf{A}^{-1}$, and $\mathbf{V}$.

THEOREM 2.1 (cf. [12, (2.7)]). *Let the sequence* $\{\mathbf{Z}_t, t \in \mathscr{Z}\}$ *follow model* (1.1)–(1.2). *Then*

$$\Gamma^{-1}_{(TT)} = \mathbf{U}'_{T,p}(\Phi)[\mathbf{C} - \mathbf{C}\mathbf{E}\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{C}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}\mathbf{E}'\mathbf{C}]\mathbf{U}_{T,p}(\Phi), \tag{2.4}$$

where

$$\mathbf{C} \overset{\text{def}}{=} \mathbf{A}^{-1} = [\mathbf{U}'_{T,q}(\Theta)]^{-1}(\mathbf{I}_T \otimes \Sigma^{-1}_{gg})[\mathbf{U}_{T,q}(\Theta)]^{-1}, \tag{2.5}$$

$\mathbf{E} = [\mathbf{0}_{km\times k(T-m)}, \mathbf{I}_{km}]'$, *and* $\mathbf{V}^{1/2}$ *is the symmetric square root of the matrix* $\mathbf{V}$.

Proof of Theorem 2.1. Recall the standard algebraic result (cf. [12, (2.6)]),

$$(\mathbf{D}_1 + \mathbf{D}_2\mathbf{D}_2')^{-1} = \mathbf{D}_1^{-1} - \mathbf{D}_1^{-1}\mathbf{D}_2(\mathbf{D}_2'\mathbf{D}_1^{-1}\mathbf{D}_2 + \mathbf{I})^{-1}\mathbf{D}_2'\mathbf{D}_1^{-1},$$

where $\mathbf{D}_1$ is an invertible square symmetric matrix and $\mathbf{D}_2$ is conformable in the multiplication $\mathbf{D}_1\mathbf{D}_2$. Rewrite (2.3) in the form $\mathbf{U}_{T,p}(\Phi)\Gamma_{(TT)}\mathbf{U}'_{T,p}(\Phi) = \mathbf{A} + \mathbf{E}\mathbf{V}\mathbf{E}'$. Since $\mathbf{U}_{T,p}(\Phi)$ and $\mathbf{U}_{T,q}(\Theta)$ are nonsingular by construction and Σ_{gg} is nonsingular by assumption, (2.3) implies that $\mathbf{A}$ and $\Gamma_{(TT)}$ are invertible. Then with $\mathbf{D}_2 = \mathbf{E}\mathbf{V}^{1/2}$,

$$\{\mathbf{U}_{T,p}(\Phi)\Gamma_{(TT)}\mathbf{U}'_{T,p}(\Phi)\}^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}$$

and the results follows. □

Expression (2.4) leads us to consider the role of the matrices $\mathbf{C}$ and $\mathbf{V}$ in the inverse $\Gamma^{-1}_{(TT)}$. First, the convergence properties of $\Gamma^{-1}_{(TT)}$ will depend primarily on the convergence properties of the matrix $\mathbf{C}$ and the related matrix $\mathbf{B} \overset{\text{def}}{=} \mathbf{U}'_{T,p}(\Phi)\mathbf{C}\mathbf{U}_{T,p}(\Phi)$. Section 2.2 discusses these matrices in more detail. Second, the inverse expression (2.4) is more complex than inverse expressions developed previously for the univariate case. This additional complexity arises from the fact that for general dimension k, the matrix $\mathbf{V}$ may be singular. Section 2.3 discusses the representation and invertibility of $\mathbf{V}$.

2.2. Convergence properties of the matrices C and B. We may approach the convergence properties of the matrices $\mathbf{C}$ and $\mathbf{B}$ as follows. A multivariate form of [12, (2.12)] indicates that the matrix $\mathbf{P} = [\mathbf{U}_{T,q}(\Theta)]^{-1}$ has the same block upper triangular form as $\mathbf{U}_{T,q}(\Theta)$, with $k \times k$ main diagonal blocks equal to $\mathbf{P}_0 = \mathbf{I}_k$ and $(i, i+d+1)$th $k \times k$ submatrix equal to

$$\mathbf{P}_{d+1} = -\sum_{l=1}^{\min(q,d)} \Theta_{d+1-l}\mathbf{P}_l, \tag{2.6}$$

$d = 0, 1, \cdots, T-2$. Note that the elements of $\mathbf{P}_d$ do not depend on T. Moreover, standard arguments associated with the inversion of a matrix polynomial indicate that under condition (1.2b), the elements of $\mathbf{P}_d$ are declining exponentially in d.

By expressions (2.5) and (2.6), $\mathbf{C} = \mathbf{P}'(\mathbf{I}_T \otimes \Sigma^{-1}_{gg})\mathbf{P}$ has (r, s)th $k \times k$ block equal to

$$\mathbf{C}_{rs} = \sum_{i=1}^{\min(r,s)} \mathbf{P}'_{r-i}\Sigma^{-1}_{gg}\mathbf{P}_{s-i}, \tag{2.7}$$

so for fixed $r, s \in \mathscr{Z}^+$, $\mathbf{C}_{rs}$ is not a function of T. In addition, $\mathbf{B} = \mathbf{U}'_{T,p}(\Phi)\mathbf{C}\mathbf{U}_{T,p}(\Phi)$

has (r, s)th $k \times k$ block equal to

$$\mathbf{B}_{rs} = \sum_{i=r}^{\min(T,p+r)} \sum_{j=s}^{\min(T,p+s)} \sum_{l=1}^{\min(i,j)} \mathbf{\Phi}_{i-r}\mathbf{P}'_{i-l}\mathbf{\Sigma}_{gg}^{-1}\mathbf{P}_{j-l}\mathbf{\Phi}'_{j-s} \tag{2.8}$$
$$= \sum_{i=0}^{\min(T-r,p)} \sum_{j=0}^{\min(T-s,p)} \mathbf{\Phi}_i\mathbf{C}_{i+r,j+s}\mathbf{\Phi}'_j.$$

Thus, some elements of $\mathbf{B}$ are functions of T, but the exponentially declining property of the $\mathbf{P}_d$ matrices implies that the matrices $\mathbf{C}$ and $\mathbf{B}$ have bounded on-diagonal blocks and exponentially declining off-diagonal blocks.

2.3. The residual matrix V. In light of Theorem 2.1, it is useful to express the matrix $\mathbf{V}$ as a function of the parameters of model (1.1)–(1.2), and to discuss the invertibility of $\mathbf{V}$. Although [11] and [12] do not present an explicit expression for $\mathbf{V}$, we may do so as follows. Appendix A gives the infinite moving average representation

$$\mathbf{Z}_t = \sum_{i=0}^{\infty} \mathbf{\Psi}_i\mathbf{g}_{t-i}, \tag{2.9}$$

so that $\text{Cov}(\mathbf{Z}_t, \mathbf{g}_{t-i}) = \mathbf{\Psi}_i\mathbf{\Sigma}_{gg}$, where $\mathbf{\Psi}_i = \mathbf{0}$ for $i < 0$. Let $\mathbf{R}_\Psi$ be a $kp \times kq$-dimensional matrix with ith $k \times kq$-dimensional row block equal to $[\mathbf{\Psi}_{1-i}, \mathbf{\Psi}_{2-i}, \cdots, \mathbf{\Psi}_{q-i}]$, $i = 1, 2, \cdots, p$. Then $\text{Cov}(\mathbf{Z}_{s-T,p}, \mathbf{g}_{s-T,q}) = \mathbf{R}_\Psi(\mathbf{I}_q \otimes \mathbf{\Sigma}_{gg})$, so

$$\mathbf{V} = \text{Var}(\mathbf{L}_1\mathbf{g}_{s-T,q} - \mathbf{L}_2\mathbf{Z}_{s-T,p})$$
$$= \mathbf{L}_1(\mathbf{I}_q \otimes \mathbf{\Sigma}_{gg})\mathbf{L}'_1 - \mathbf{L}_2\mathbf{R}_\Psi(\mathbf{I}_q \otimes \mathbf{\Sigma}_{gg})\mathbf{L}'_1 - \mathbf{L}_1(\mathbf{I}_q \otimes \mathbf{\Sigma}_{gg})\mathbf{R}'_\Psi\mathbf{L}'_2 + \mathbf{L}_2\mathbf{\Gamma}_{(pp)}\mathbf{L}'_2.$$

Stationarity of model (1.1) implies that $\mathbf{V}$ is not a function of T.

Now consider the invertibility of $\mathbf{V}$. For the case $k = 1$, it follows immediately that for an autoregressive moving average model with minimal orders p and q, the matrix $\mathbf{V}$ is invertible. For the case $k > 1$, this may not be true (cf. the discussion of identifiability of vector ARMA models in [15]). For a pure moving average process, $\mathbf{V} = \mathbf{L}_q(\mathbf{\Theta})(\mathbf{I} \otimes \mathbf{\Sigma}_{gg})\mathbf{L}_q(\mathbf{\Theta})'$, so for nonsingular $\mathbf{\Sigma}_{gg}$, $\mathbf{V}$ is nonsingular if and only if $\det(\mathbf{\Theta}_q) \neq 0$. For the general case, let $\mathbf{h}_{s-T,p}$ be a kp-dimensional vector with ith $k \times 1$ block equal to $\sum_{l=q-i+1}^{\infty} \mathbf{\Psi}_l\mathbf{g}_{s-T-i-l+1}$, $i = 1, 2, \cdots, p$. Note that $\mathbf{h}_{s-T,p}$ is independent of $\mathbf{g}_{s-T,q}$. Then by expression (2.9),

$$\begin{pmatrix} \mathbf{g}_{s-T,q} \\ \mathbf{Z}_{s-T,p} \end{pmatrix} = \begin{pmatrix} \mathbf{I}_{kq} \\ \mathbf{R}_\Psi \end{pmatrix}\mathbf{g}_{s-T,q} + \begin{pmatrix} \mathbf{0}_{kq\times 1} \\ \mathbf{h}_{s-T,p} \end{pmatrix}$$

so

$$\mathbf{V} = (\mathbf{L}_1 - \mathbf{L}_2\mathbf{R}_\Psi)(\mathbf{I}_q \otimes \mathbf{\Sigma}_{gg})(\mathbf{L}_1 - \mathbf{L}_2\mathbf{R}_\Psi)' + \mathbf{L}_2\,\text{Var}(\mathbf{h}_{s-T,p})\mathbf{L}'_2. \tag{2.10}$$

Thus, two sets of sufficient conditions for the invertibility of $\mathbf{V}$ are as follows.

(1) Assume that $\det(\mathbf{\Sigma}_{gg}) \neq 0$ and $\mathbf{L}_1 - \mathbf{L}_2\mathbf{R}_\Psi$ has full row rank. Then the first addend of expression (2.10) is nonsingular.

(2) Assume that $p \geqq q$, $\det[\text{Var}(\mathbf{h}_{s-T,p})] \neq 0$, and $\det(\mathbf{\Phi}_p) \neq 0$. Then $\det(\mathbf{L}_2) = [\det(\mathbf{\Phi}_p)]^p \neq 0$, so the second addend of expression (2.10) is nonsingular.

3. Convergence properties of the inverse. This section uses the results of § 2 to develop some limiting properties of the inverse matrix $\mathbf{\Gamma}_{(TT)}^{-1}$ and some associated matrix functions. These properties depend on the $Tk \times Tk$-dimensional symmetric matrices $\bar{\mathbf{C}}$ and $\bar{\mathbf{B}}$, where $\bar{\mathbf{C}}$ has $(r, r+d)$th $k \times k$ block equal to $\bar{\mathbf{C}}_d = \sum_{l=0}^{\infty} \mathbf{P}'_l\mathbf{\Sigma}_{gg}^{-1}\mathbf{P}_{l+d}$, $d \geqq 0$,

and $\bar{\mathbf{B}}$ has (r, s)th $k \times k$ block equal to

$$\bar{\mathbf{B}}_{rs} = \sum_{i=0}^{\min(T-r,p)} \sum_{j=0}^{\min(T-s,p)} \Phi_i \bar{\mathbf{C}}_{j+s-i-r} \Phi_j'. \tag{3.1}$$

The matrices $\bar{\mathbf{C}}_d$ are declining exponentially in d, so the matrices $\bar{\mathbf{B}}$ have bounded on-diagonal blocks and exponentially declining off-diagonal blocks. Moreover, if $r \leqq T - p$ and $d \geqq 0$, then

$$\bar{\mathbf{B}}_{r+d,r}' = \bar{\mathbf{B}}_{r,r+d} = \sum_{i=0}^{p} \sum_{j=0}^{p} \Phi_i \bar{\mathbf{C}}_{j+d-i} \Phi_j', \tag{3.2}$$

and we may define $\bar{\bar{\mathbf{B}}}_d = \bar{\bar{\mathbf{B}}}_{-d}'$ to equal expression (3.2). Theorem 3.1 presents some convergence properties associated with the matrices $\mathbf{\Gamma}_{(TT)}^{-1}$, $\mathbf{B}$, and $\bar{\mathbf{B}}$.

THEOREM 3.1. *Let* $\{\mathbf{Z}_t, t \in \mathscr{Z}\}$ *satisfy model* (1.1)–(1.2). *Define* $\mathbf{B}$ *and* $\bar{\mathbf{B}}$ *by expressions* (2.8) *and* (3.1), *respectively. Let* $\mathbf{G}_{TT} = \mathbf{\Gamma}_{(TT)}^{-1}$ *have* (r, s)*th* $k \times k$ *block* $\mathbf{G}_{TTrs}$. *Then*

(a) *The sequence* $\{\mathbf{B}_{r,r+d} - \bar{\mathbf{B}}_{r,r+d}\}$ *is declining exponentially in* r, *uniformly in* T *and* d, *and is declining exponentially in* d, *uniformly in* T *and* r.

(b) *For fixed* $r \in \mathscr{Z}^+$, *the sequence* $\{\mathbf{G}_{TTrs} - \mathbf{B}_{rs}\}$ *is declining exponentially in* T, *uniformly in* $s \in \mathscr{Z}^+$.

(c) *Let* $\{r(T), T \in \mathscr{Z}^+\}$ *be a sequence of positive integers such that* $T - r(T)$ *is increasing without bound. Then the sequence* $\{\mathbf{G}_{TTr(T)s} - \mathbf{B}_{r(T)s}\}$ *is declining exponentially in* $T - r(T)$, *uniformly in* $s \in \mathscr{Z}^+$.

(d) *The elements of* $\mathbf{G}_{TT}$ *are bounded uniformly in* T.

(e) *The matrices in the sequence* $\{\mathbf{G}_{TT}\}$ *have bounded on-diagonal blocks and exponentially declining off-diagonal blocks.*

Proof of Theorem 3.1. (a) Note first that for $d \geqq 0$,

$$\mathbf{C}_{r,r+d} - \bar{\mathbf{C}}_d = -\sum_{l=r+1}^{\infty} \mathbf{P}_l' \mathbf{\Sigma}_{gg}^{-1} \mathbf{P}_{l+d}, \tag{3.3}$$

which is declining exponentially in r, uniformly in d, and is also declining exponentially in d, uniformly in r. For $d \geqq 0$,

$$\mathbf{B}_{r,r+d} - \bar{\mathbf{B}}_{r,r+d} = \sum_{i=0}^{\min(T-r,p)} \sum_{j=0}^{\min(T-r-d,p)} \Phi_i (\mathbf{C}_{i+r,j+r+d} - \bar{\mathbf{C}}_{j+d-i}) \Phi_j', \tag{3.4}$$

which for some $K > 0$ and $c > 0$ is bounded in norm by $K \exp[-c(|r| + |r + d|)]$. Thus $\mathbf{B}_{r,r+d} - \bar{\mathbf{B}}_{r,r+d}$ is declining exponentially in r, uniformly in T and d, and is declining exponentially in d, uniformly in T and r.

(b) By expression (2.4) and the definition of $\mathbf{B}$,

$$\mathbf{G}_{TTrs} - \mathbf{B}_{rs} = -\mathbf{D}_r \mathbf{V}^{1/2} (\mathbf{V}^{1/2} \mathbf{E}' \mathbf{A}^{-1} \mathbf{E} \mathbf{V}^{1/2} + \mathbf{I})^{-1} \mathbf{V}^{1/2} \mathbf{D}_s', \tag{3.5}$$

where

$$\mathbf{D}_r = \sum_{j=r}^{r+p} \sum_{l=1}^{j} \Phi_{j-r} \mathbf{P}_{j-l}' \mathbf{\Sigma}_{gg}^{-1} \mathbf{P}_{T-l} \tag{3.6}$$

is the rth $k \times k$-dimensional block of $\mathbf{U}_{T,p}'(\Phi) \mathbf{A}^{-1} \mathbf{E}$. For fixed r, $\mathbf{D}_r$ is declining exponentially in T due to the factor $\mathbf{P}_{T-l}$. Also, $\|\mathbf{D}_r\|$ is bounded uniformly in r and T. Now

$$\mathbf{E}' \mathbf{A}^{-1} \mathbf{E} = \sum_{l=0}^{T-1} \mathbf{P}_l' \mathbf{\Sigma}_{gg}^{-1} \mathbf{P}_l \tag{3.7}$$

is positive definite, $\mathbf{V}^{1/2}$ is nonnegative definite, and both terms are bounded uniformly in T. Thus $\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}$ is bounded uniformly in T. Part (b) then follows from expression (3.5).

(c) The proof of part (c) is identical to the proof of part (b), except that the index r is replaced by $r(T)$, and the following property of $\mathbf{D}_{r(T)}$ is noted. Since the $\mathbf{P}_d$ are declining exponentially in d, (3.6) implies that there exist $K > 0$, $c > 0$ such that

$$\begin{aligned}\|\mathbf{D}_{r(T)}\| &\leqq K \sum_{j=r(T)}^{r(T)+p} \sum_{l=1}^{j} \exp\left[-c(j-l+T-l)\right] \\ &\leqq Kp[1-\exp(-2c)]^{-1} \exp\{-c[T-r(T)]\}.\end{aligned} \tag{3.8}$$

Thus, $\mathbf{D}_{r(T)}$ is declining exponentially in $T - r(T)$.

(d) By expression (2.7) and the exponentially declining property of the matrices $\mathbf{P}_d$, the elements of the matrices $\mathbf{C}_{rs}$ and $\mathbf{B}_{rs}$ are bounded uniformly in r, s, and T. Part (d) then follows from expression (3.5) and the uniform boundedness of $\mathbf{\Gamma}$ and $\mathbf{D}_r$ in r and T.

(e) Finally, note that for $d \geqq 0$, the inequalities $1 \leqq r$, $r + d \leqq T$ imply that $1 \leqq r \leqq T - d$, so by (3.6), there exist $K > 0$ and $c > 0$ such that $\|\mathbf{D}_r\| \leqq pK \sum_{l=1}^{r+p} \exp[-c(T-l)] \leqq pK[1-\exp(-c)]^{-1} \exp[-c(d-p)]$. Then by (3.5), $\mathbf{G}_{TTr,r+d} - \mathbf{B}_{r,r+d} = \mathbf{D}_r\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}\mathbf{D}'_{r+d}$ is declining exponentially in $d \geqq 0$, uniformly in T, and in $1 \leqq r \leqq T - d$. Part (e) then follows from the fact that the matrices $\mathbf{B}$ have bounded on-diagonal blocks and exponentially declining off-diagonal blocks. □

Note that in the proof of Theorem 3.1, the causality criterion ensures the existence of the autocovariances $\mathbf{\Gamma}_{ZZ}(d)$, and the invertibility criterion ensures the existence of the inverse $\mathbf{\Gamma}_{(TT)}^{-1}$ and the exponentially declining properties of the matrices $\mathbf{P}_d$, and thus of $\mathbf{C}$, $\mathbf{B}$, and $\bar{\mathbf{B}}$, as well.

As noted in § 1, [1] and [2] have developed results similar to Theorem 3.1 for the inverses of certain banded matrices. The results of Theorem 3.1 may also be approached through a study of the exponential rates of convergence of innovation sequence variances and coefficients to the associated moving average model terms. See, for example, [8, pp. 414, 445]. Also, following [8, p. 384], we may give an intuitive interpretation of the limiting values $\bar{\bar{\mathbf{B}}}_d$ of the autocovariance matrix inverse. Appendix A shows that the invertibility criterion leads to the infinite autoregressive representation $\mathbf{\Upsilon}(B)\mathbf{Z}_t = \mathbf{\Theta}^{-1}(B)\mathbf{\Phi}(B)\mathbf{Z}_t = \mathbf{g}_t$. This leads to the approximation $\mathbf{\Upsilon}_{(T)}\mathbf{Z}_{(T)} \doteq \mathbf{g}_{(T)}$, where $\mathbf{\Upsilon}_{(T)}$ is a $kT \times kT$ upper-triangular matrix with ith $k \times kT$ block row equal to $\mathbf{\Upsilon}_{(T)i\cdot} = (\mathbf{0}_{k\times k(i-1)}, \mathbf{\Upsilon}_0, \mathbf{\Upsilon}_1, \cdots, \mathbf{\Upsilon}_{T-i})$. Then for large T and upper-triangular $\mathbf{\Sigma}_{gg}^{-1/2}$, the matrix $\mathbf{\Sigma}_{gg}^{-1/2}\mathbf{\Upsilon}_{(T)i\cdot}$ approximates the ith row of the Cholesky decomposition matrix $\mathbf{\Gamma}_{(TT)}^{-1/2}$. Consequently, for large i and T and fixed $d \geqq 0$, the $(i, i+d)$th $k \times k$ block of $\mathbf{\Gamma}_{(TT)}^{-1}$ is approximated by $\sum_{l=0}^{i-1} \mathbf{\Upsilon}_l'\mathbf{\Sigma}_{gg}^{-1}\mathbf{\Upsilon}_{l+d}$, which converges to $\sum_{l=0}^{\infty} \mathbf{\Upsilon}_l'\mathbf{\Sigma}_{gg}^{-1}\mathbf{\Upsilon}_{l+d} = \bar{\bar{\mathbf{B}}}_d$ as $i \to \infty$. Thus we may view the convergence properties of the autocovariance matrix inverse as dependent on the convergence properties of the rational matrix function $\mathbf{\Upsilon}(B)$, or equivalently, as dependent on the roots of the matrix polynomials $\mathbf{\Phi}(B)$ and $\mathbf{\Theta}(B)$.

4. Application to a latent-variable prediction problem. Theorem 3.1 may be applied to the following latent-variable prediction problem. In addition to the observed vectors $\mathbf{Z}_t$ following model (1.1)–(1.2), let $\{\mathbf{x}_t, t \in \mathscr{Z}\}$ be a sequence of r-dimensional random vectors such that $\mathbf{W}_t = (\mathbf{x}_t', \mathbf{Z}_t')'$ follows an $(r+k)$-dimensional stationary autoregressive moving average (p_W, q_W) model,

$$\mathbf{\Phi}_W(B)\mathbf{W}_t = \theta_W(B)\mathbf{g}_{Wt}, \tag{4.1}$$

where $\mathbf{\Sigma}_{ggWW} \overset{\text{def}}{=} \operatorname{Var}(\mathbf{g}_{Wt})$, $t \in \mathscr{Z}$, is invertible, and the coefficient matrices of the poly-

nomials $\Phi_W(B)$ and $\theta_W(B)$ satisfy the causality and invertibility criteria, respectively. Using the notation of expression (2.1), define $\mathbf{x}_{t+T,s+t+1} = (\mathbf{x}'_{t+T}, \mathbf{x}'_{t+T-1}, \cdots, \mathbf{x}'_{T-s})'$, the values of the latent vector from s periods before the current period T to t periods ahead of T. Define $\mathbf{\Gamma}_{xZst} = \text{Cov}\,(\mathbf{x}_{t+T,s+t+1}, \mathbf{Z}_{(T)})$, which has typical $r \times k$ block $\mathbf{\Gamma}_{xZ}(i-j) = \text{Cov}\,(\mathbf{x}_{l-i}, \mathbf{Z}_{l-j})$, $l \in \mathscr{Z}$, and define $\mathbf{\Gamma}_{xx} = \text{Var}\,(\mathbf{x}_{t+T,s+t+1})$. Then the minimum mean squared error predictor of $\mathbf{x}_{t+T,s+t+1}$, which is linear in $\mathbf{Z}_{(T)}$, is $\ddot{\mathbf{x}} = \mathbf{\Gamma}_{xZst}\mathbf{\Gamma}^{-1}_{(TT)}\mathbf{Z}_{(T)}$, and $\mathbf{V}_F \overset{\text{def}}{=} \text{Var}\,(\ddot{\mathbf{x}} - \mathbf{x}) = \mathbf{\Gamma}_{xx} - \mathbf{\Gamma}_{xZst}\mathbf{\Gamma}^{-1}_{(TT)}\mathbf{\Gamma}'_{xZst}$. Define the $(s+t+1)r \times Tk$-dimensional matrix $\mathbf{F}_{(T)} = \mathbf{\Gamma}_{xZst}\mathbf{\Gamma}^{-1}_{(TT)}$ and let $\mathbf{F}_{(T)ij}$ be the $r \times k$ block of $\mathbf{F}_{(T)}$ which gives the coefficients of $\mathbf{Z}_{T-j+1}$ in the prediction of $\mathbf{x}_{T-i}$:

$$\mathbf{F}_{(T)ij} = \sum_{l=1}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)\mathbf{G}_{TTlj}, \tag{4.2}$$

$i = -t, -t+1, \cdots, s; j = 1, 2, \cdots, T$. Also, note that $\mathbf{V}_F$ has $(i+t+1, j+t+1)$th $k \times k$ block equal to

$$\begin{aligned}\mathbf{V}_{Fij} &\overset{\text{def}}{=} \mathbf{\Gamma}_{xx}(i-j) - \sum_{n=1}^{T} \mathbf{F}_{(T)in}\mathbf{\Gamma}'_{xZ}(j-n+1) \\ &= \mathbf{\Gamma}_{xx}(i-j) - \sum_{l=1}^{T}\sum_{n=1}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)\mathbf{G}_{TTln}\mathbf{\Gamma}'_{xZ}(j-n+1),\end{aligned} \tag{4.3}$$

the covariance matrix of the prediction errors for $\mathbf{x}_{T-i}$ and $\mathbf{x}_{T-j}$, $i, j = -t, -t+1, \cdots, s$. Define

$$\mathbf{F}_{Bij} = \sum_{l=1}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)\mathbf{B}_{lj}, \tag{4.4}$$

$$\mathbf{V}_{Bij} = \mathbf{\Gamma}_{xx}(i-j) - \sum_{n=1}^{T} \mathbf{F}_{Bin}\mathbf{\Gamma}'_{xZ}(j-n+1), \tag{4.5}$$

$$\bar{\mathbf{F}}_{ij} = \sum_{l=1}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)\bar{\mathbf{B}}_{lj}, \tag{4.6}$$

$$\bar{\mathbf{V}}_{ij} = \mathbf{\Gamma}_{xx}(i-j) - \sum_{l=1}^{T}\sum_{n=1}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)\bar{\mathbf{B}}_{ln}\mathbf{\Gamma}'_{xZ}(j-n+1), \tag{4.7}$$

$$\bar{\bar{\mathbf{F}}}(i-j) = \sum_{l=1}^{\infty} \mathbf{\Gamma}_{xZ}(i-l+1)\bar{\bar{\mathbf{B}}}_{j-l}, \tag{4.8}$$

and

$$\bar{\bar{\mathbf{V}}}(i-j) = \mathbf{\Gamma}_{xx}(i-j) - \sum_{l=0}^{\infty}\sum_{n=0}^{\infty} \mathbf{\Gamma}_{xZ}(i-l)\bar{\bar{\mathbf{B}}}_{n-l}\mathbf{\Gamma}'_{xZ}(j-n). \tag{4.9}$$

Theorem 4.1 describes the convergence properties of the elements of the **F** and **V** matrices.

THEOREM 4.1. *Let* $\{\mathbf{W}_t = (\mathbf{x}'_t, \mathbf{Z}'_t)', t \in \mathscr{Z}\}$ *satisfy model* (4.1), *where the autoregressive coefficients* Φ_{wj} *satisfy the causality criterion, and the* $\{\mathbf{Z}_t\}$ *vectors satisfy model* (1.1)–(1.2). *Define* $\mathbf{F}_{(T)ij}$, $\mathbf{V}_{Fij}$, $\mathbf{F}_{Bij}$, $\mathbf{V}_{Bij}$, $\bar{\mathbf{F}}_{ij}$, $\bar{\mathbf{V}}_{ij}$, $\bar{\bar{\mathbf{F}}}(i-j)$, *and* $\bar{\bar{\mathbf{V}}}(i-j)$ *by expressions* (4.2)–(4.9), *respectively. Let* $0 < \varepsilon < 1$ *and let* $\{r(t), T \in \mathscr{Z}^+\}$ *be a sequence of positive integers such that* $T^{\varepsilon} \leqq r(T) < 2^{-1}T$ *for sufficiently large* $T \in \mathscr{Z}^+$. *Then*

(a) *The matrices* $\mathbf{F}_{(T)ij} - \mathbf{F}_{Bij}$ *are declining exponentially in* T, *uniformly in* j *and in* $T - i \geqq T^{\varepsilon}$.

(b) *The matrices* $\mathbf{V}_{Fij} - \mathbf{V}_{Bij}$ *are declining exponentially in* T, *uniformly in* $T - i \geqq T^{\epsilon}$ *and in* $T - j \geqq T^{\epsilon}$.

(c) *For fixed* $T - i$, $\mathbf{F}_{(T)ij} - \bar{\mathbf{F}}_{ij}$ *is declining exponentially in* T, *uniformly in* j.

(d) *For fixed* $T - i$ *and* $T - j$, $\mathbf{V}_{Fij} - \bar{\mathbf{V}}_{ij}$ *is declining exponentially in* T.

(e) *The matrices* $\mathbf{F}_{(T)ij} - \bar{\bar{\mathbf{F}}}(i - j)$ *are declining exponentially in* T, *uniformly in* j *and in* $r(T) \leqq i \leqq T - r(T)$.

(f) *The matrices* $\mathbf{V}_{Fij} - \bar{\bar{\mathbf{V}}}(i - j)$ *are declining exponentially in* $r(T)$, *uniformly in* $r(T) \leqq i, j \leqq T - r(T)$.

(g) *For fixed* $T - i$, $F_{(T)ij}$ *is declining exponentially in* $|i - j|$, *uniformly in* T.

(h) *For* $s = T - 1$ *and for fixed* t, $\mathbf{V}_F$ *has exponentially declining off-diagonal blocks*.

Proof of Theorem 4.1. Detailed proofs are given for parts (a), (c), (e), and (g) only. Parts (b), (d), (f), and (h) then follow immediately from expressions (4.3), (4.5), (4.7), and (4.9).

(a) First,

$$\mathbf{F}_{(T)ij} - \mathbf{F}_{Bij} = \sum_{l=1}^{T_1} \mathbf{\Gamma}_{xZ}(i - l + 1)(\mathbf{G}_{TTlj} - \mathbf{B}_{lj}) \tag{4.10a}$$

$$+ \sum_{l=T_1+1}^{T} \mathbf{\Gamma}_{xZ}(i - l + 1)(\mathbf{G}_{TTlj} - \mathbf{B}_{lj}), \tag{4.10b}$$

where $T_1 = \lceil T - 2^{-1}T^{\epsilon} \rceil$ and $\lceil \cdot \rceil$ denotes the least-integer function. Then by the uniform boundedness of $\mathbf{\Gamma}_{xZ}(i - l + 1)$ in l and i and by Theorem 3.1(c), there exist $K_1 > 0$, $c_1 > 0$ such that expression (4.10a) is bounded in norm by $K_1(T - 2^{-1}T^{\epsilon}) \exp[-c_1(T - 2^{-1}T^{\epsilon})]$ and thus is declining exponentially in T. Moreover, by the uniform boundedness of $\mathbf{G}_{TTlj} - \mathbf{B}_{lj}$ and the exponentially declining property of $\mathbf{\Gamma}_{xZ}(d)$, the inequalities $T - i \geqq T^{\epsilon}$ and $l \geqq T - 2^{-1}T^{\epsilon}$ imply that there exist constants $K_2 > 0$, $c_2 > 0$ such that expression (4.10b) is bounded in norm by

$$K_2 \sum_{l=T_1+1}^{T} \exp[-c_2(i - l + 1)] \leqq K_2 2^{-1} T^{\epsilon} \exp[-c_2(T - 2^{-1}T^{\epsilon})]$$

and is thus declining exponentially in T.

(c) Note that $\mathbf{F}_{(T)ij} - \bar{\mathbf{F}}_{ij} = \mathbf{F}_{(T)ij} - \mathbf{F}_{Bij} + \mathbf{F}_{Bij} - \bar{\mathbf{F}}_{ij}$. By Theorem 4.1(a), $\mathbf{F}_{(T)ij} - \mathbf{F}_{Bij}$ is declining exponentially in T, uniformly in j. Also,

$$\mathbf{F}_{Bij} - \bar{\mathbf{F}}_{ij} = \sum_{d=0}^{T} \mathbf{\Gamma}_{xZ}(d + 1 - T + i)[\mathbf{B}_{T-d,j} - \bar{\mathbf{B}}_{T-d,j}]. \tag{4.11}$$

By Theorem 3.1(a) and the exponentially declining property of $\mathbf{\Gamma}_{xZ}(l)$, there exist constants $K_3 > 0$, $c_3 > 0$ such that expression (4.11) is bounded in norm by

$$K_3 \sum_{d=0}^{T-1} \exp[-c_3(|T - i - d - 1| + |T - d - j| + |T - d|)]$$

$$\leqq K_3 \sum_{d=0}^{\lceil 2^{-1}T \rceil} \exp(-c_3|T - d|) + K_3 \sum_{d=\lceil 2^{-1}T \rceil}^{T} \exp(-c_3|T - i - d - 1|). \tag{4.12}$$

The first sum of expression (4.12) is bounded above by $K_3 T \exp(-c_3 2^{-1}T)$ and, since $T - i$ is fixed, the second sum of expression (4.12) is bounded above by $K_3 T \exp(-c_3 4^{-1}T)$ for sufficiently large T, so part (c) follows.

(e) Now

$$\mathbf{F}_{(T)ij} - \bar{\bar{\mathbf{F}}}(i-j) = \mathbf{F}_{(T)ij} - \mathbf{F}_{Bij}$$

$$+ \sum_{l=1}^{T_2-1} \mathbf{\Gamma}_{xZ}(i-l+1)(\mathbf{B}_{lj} - \bar{\mathbf{B}}_{lj}) \tag{4.13a}$$

$$+ \sum_{l=T_2}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)(\mathbf{B}_{lj} - \bar{\mathbf{B}}_{lj})$$

$$+ \sum_{l=1}^{T_3-1} \mathbf{\Gamma}_{xZ}(i-l+1)(\bar{\mathbf{B}}_{lj} - \bar{\bar{\mathbf{B}}}_{j-l}) \tag{4.13b}$$

$$+ \sum_{l=T_3}^{T} \mathbf{\Gamma}_{xZ}(i-l+1)(\bar{\mathbf{B}}_{lj} - \bar{\bar{\mathbf{B}}}_{j-l}),$$

where $T_2 = \lceil 2^{-1}r(T)\rceil$ and $T_3 = T - \lceil 2^{-1}r(T)\rceil$. By Theorem 3.1(a), $\mathbf{F}_{(T)ij} - \mathbf{F}_{Bij}$ is declining exponentially in T, uniformly in j and in $T - i \geqq r(T)$. Since $\mathbf{\Gamma}_{xZ}(d)$ is declining exponentially in $|d|$, and $\mathbf{B}_{lj} - \bar{\mathbf{B}}_{lj}$ and $\bar{\mathbf{B}}_{lj} - \bar{\bar{\mathbf{B}}}_{j-l}$ are bounded in norm uniformly in l and j, there exist $K_4 > 0$ and $c_4 > 0$ such that the first sum of (4.13a) is bounded in norm by

$$K_4 \sum_{l=1}^{T_2-1} \exp[-c_4(i-l+1)] \leqq K_4 r(T) \exp[-c_4 2^{-1} r(T)]$$

for all $r(T) \leqq i, j \leqq T - r(T)$. A similar argument shows that the second sum in expression (4.13b) is bounded in norm by $K_5 r(T) \exp[-c_5 2^{-1} r(T)]$ for some $K_5 > 0$, $c_5 > 0$ and all $r(T) \leqq i, j \leqq T - r(T)$. Also, by (3.4) and the uniform boundedness of $\mathbf{\Gamma}_{xZ}(d)$ in d, there exist $K_6 > 0$ and $c_6 > 0$ such that the second sum in (4.13a) is bounded in norm by

$$K_6 \sum_{l=T_2}^{T} \exp[-c_6(|l| + |j|)] \leqq K_6 T \exp[-c_6 2^{-1} r(T)].$$

Finally, note that for sufficiently large T, $l \leqq T_3$ and $r(T) \leqq j \leqq T - r(T)$ imply that l, $j \leqq T - p$, so by (3.2), the first sum in (4.13b) is identically zero, and part (e) is established.

(g) Part (g) follows immediately from expression (4.2), Theorem 4.1(e), and Lemma B.2. □

Theorem 4.1 may be interpreted as follows. Part (a) indicates that for large T, and $T - i \geqq T^{\epsilon}$, the "filter weight" $\mathbf{F}_{(T)ij}$ in the prediction of $\mathbf{x}_{T-i}$ may be approximated uniformly in j by the weights $\mathbf{F}_{Bij}$, which are dependent on $\mathbf{B}$ rather than on $\mathbf{G}_{TT}$. Similarly, the corresponding mean squared prediction error $\mathbf{V}_{Fij}$ is approximated by $\mathbf{V}_{Bij}$, which depends on $\mathbf{B}$, T, i, and j.

Part (c) indicates that for fixed values of $T - i$, the $\mathbf{F}_{(T)ij}$ converge to "limiting weights" $\bar{\mathbf{F}}_{ij}$. These limiting weights depend only on the specific values of $T - i$ and $T - j$ employed. However, for fixed $T - i$, we do not have a similar convergence of $\mathbf{F}_{(T)ij}$ to $\bar{\bar{\mathbf{F}}}(i-j)$, say. In part (d), the corresponding mean squared prediction error $\mathbf{V}_{Fij}$ converges to $\bar{\mathbf{V}}_{ij}$, which is dependent on $T - i$ and $T - j$, and not just on $i - j$.

Part (e) indicates that for a period $T - i$ distant from the two ends of the observed series, the "filter weight" $\mathbf{F}_{(T)ij}$ in the prediction of $\mathbf{x}_{T-i}$ is approximated by $\bar{\bar{\mathbf{F}}}(i-j)$, which depends only on $i - j$ and not on T. Part (f) shows that for periods $T - i$ and

$T - j$ both distant from the two ends of the observed series, the corresponding mean squared prediction error $\mathbf{V}_{Fij}$ is approximated by $\bar{\bar{\mathbf{V}}}(i-j)$, which again depends only on $i-j$. For parts (e) and (f), the rate of exponential convergence $r(T)$ depends on the distance of i and j from either end of the observed series; two examples of the rate are $r(T) = T^{\epsilon}$ and $r(T) = \beta T$, $0 < \beta < 2^{-1}$.

Part (g) gives a form of the commonly encountered result that observations have a declining effect on prediction of values far away in time. Similarly, part (h) notes the decreasing correlation between predicted values at increasingly distant periods.

Theorem 4.1 generalizes comments by Harrison and Stevens [3, p. 216], Smith [4, pp. 375–376], and others on the relation of state-space model predictors to exponentially weighted moving average predictors and on the convergence of predictor weights and prediction error variances to values independent of T. See also [16, Chap. 7]. In addition, the duality between autoregressive moving average models and state-space models implies that the expression $F_{(T)ij} - \bar{\bar{F}}(i-j)$ gives the difference between the optimal filter weight and the (suboptimal) time-invariant filter weight discussed in [5, p. 83]. Similarly, the matrices $\mathbf{V}_{Fij} - \bar{\bar{\mathbf{V}}}(i-j)$ give the difference between the mean squared prediction error for $x_{t+T,s+t+1}$ under optimal filtering and the nominal mean squared prediction error under time-invariant filtering. The principal distinction between such optimal filtering results and Theorem 4.1 is that the former work focuses on convergence of the coefficients of one-step-ahead prediction and updating equations, while the latter considers prediction for any period and emphasizes the dependence of the convergence of the direct filter weights on the corresponding convergence properties of the autocovariance matrix inverse. The relationship between Theorem 4.1 and convergence and stability results in the optimal filtering literature will not be discussed further here. Instead, § 5 extends Theorems 3.1 and 4.1 to a general parameterization of model (4.1).

5. Derivatives and uniform rates of convergence. Sections 2–4 presented some properties of $\mathbf{\Gamma}^{-1}_{(TT)}$ and associated filter weights. In practice (see, e.g., [17]), these matrices may depend on a fixed-dimensional parameter $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_Q)'$, which must be estimated. Extension of §§ 2–4 to estimated inverses and filter weights thus requires additional results on matrix derivatives and uniform rates of convergence. This section develops the necessary mathematical results. The application of these results will be considered elsewhere. Let $\mathbf{M}^{(m)}$ denote a matrix of first derivatives of a real matrix $\mathbf{M}$ taken with respect to α_m, and let $\mathbf{M}^{(m,n)}$ denote the corresponding matrix of second derivatives taken with respect to α_m and α_n. Theorem 5.1 extends Theorem 3.1 to the approximation of $\mathbf{G}^{(m)}_{TT}$ and $\mathbf{G}^{(m,n)}_{TT}$ by derivatives of $\mathbf{B}$, $\bar{\mathbf{B}}$, and $\bar{\bar{\mathbf{B}}}$. Theorem 5.2 presents some convergence properties of the first and second derivatives of $\mathbf{F}_{(T)}$ and $\mathbf{V}_F$. Theorem 5.3 extends Theorem 4.1 to the approximation of $\mathbf{F}^{(m)}_{(T)}$, $\mathbf{F}^{(m,n)}_{(T)}$, $\mathbf{V}^{(m)}_F$, and $\mathbf{V}^{(m,n)}_F$ by derivatives of $\mathbf{F}_B$, $\bar{\mathbf{F}}$, $\bar{\bar{\mathbf{F}}}$, $\mathbf{V}_B$, $\bar{\mathbf{V}}$, and $\bar{\bar{\mathbf{V}}}$. Finally, Theorem 5.4 discusses rates of exponential convergence that are uniform in some neighborhood of α. The proofs of these theorems are similar to the proofs of the theorems in §§ 3 and 4, and thus are placed in Appendix B.

THEOREM 5.1. *Let* $\{\mathbf{Z}_t, t \in \mathscr{Z}\}$ *satisfy model* (1.1). *Assume that the coefficient matrices and the error covariance matrix are twice continuously differentiable functions* $\{\mathbf{\Phi}_j(\alpha), j = 0, 1, \cdots, p\}$, $\{\mathbf{\Theta}_i(\alpha), i = 0, 1, \cdots, q\}$, *and* $\mathbf{\Sigma}_{gg}(\alpha)$ *of some parameter* α *of fixed dimension* Q. *Let* $\mathscr{A}$ *be an open subset of* $\mathscr{R}^Q$ *such that for all* $\alpha \in \mathscr{A}$, *the conditions of Theorem* 3.1 *are satisfied. Then for all* $\alpha \in \mathscr{A}$ *and all* $1 \leqq m, n \leqq Q$

(a) *Conclusions* (a) *through* (e) *of Theorem* 3.1 *are satisfied when we replace* $\mathbf{G}_{TT}$ *with* $\mathbf{G}^{(m)}_{TT}$, $\mathbf{B}$ *with* $\mathbf{B}^{(m)}$, $\bar{\mathbf{B}}$ *with* $\bar{\mathbf{B}}^{(m)}$, *and*

$$\bar{\bar{\mathbf{B}}} \quad \textit{with } \bar{\bar{\mathbf{B}}}^{(m)}.$$

(b) *Conclusions* (a) *through* (e) *of Theorem* 3.1 *are satisfied when we replace* $\mathbf{G}_{TT}$ *with* $\mathbf{G}_{TT}^{(m,n)}$, $\mathbf{B}$ *with* $\mathbf{B}^{(m,n)}$, $\bar{\mathbf{B}}$ *with* $\bar{\mathbf{B}}^{(m,n)}$, *and*

$$\bar{\bar{\mathbf{B}}} \quad \text{with } \bar{\bar{\mathbf{B}}}^{(m,n)}.$$

THEOREM 5.2. *Let* $\{\mathbf{W}_t = (\mathbf{x}_t', \mathbf{Z}_t')', t \in \mathscr{Z}\}$ *satisfy model* (4.1). *Assume that the coefficient matrices and the error covariance matrix are twice continuously differentiable functions* $\{\Phi_{Wj}(\alpha), j = 0, 1, \cdots, p_W\}$, $\{\Theta_{Wi}(\alpha), j = 0, 1, \cdots, q_W\}$, *and* $\Sigma_{ggWW}(\alpha)$ *of some parameter* α *of fixed dimension* Q. *Let* $\mathscr{A}$ *be an open subset of* $\mathscr{R}^Q$ *such that for all* $\alpha \in \mathscr{A}$, *the coefficient matrices* $\Phi_{Wj}(\alpha)$ *satisfy the causality criterion and the* $\{\mathbf{Z}_t\}$ *vectors satisfy model* (1.1)–(1.2). *Then for all* $\alpha \in \mathscr{A}$:

(a) *For fixed* $T - i$, *the matrices* $\mathbf{F}_{ij}^{(m)}$ *and* $\mathbf{F}_{ij}^{(m,n)}$ *are declining exponentially in* $|i - j|$, *uniformly in* T, $1 \leqq m, n \leqq Q$.

(b) *For* $s = T$ *and fixed* t, *the matrices* $\mathbf{V}_F^{(m)}$ *and* $\mathbf{V}_F^{(m,n)}$ *have exponentially declining off-diagonal blocks*.

Theorem 5.3 extends the convergence results (a) through (f) of Theorem 4.1 to the matrices of first and second derivatives of the "filter weights" $\mathbf{F}_{(T)}$, the prediction error variances $\mathbf{V}_F$, and their approximations. The proof of Theorem 5.3 is very similar to the proofs of Theorems 5.1 and 5.2 and is therefore omitted.

THEOREM 5.3. *Assume the conditions of Theorem* 5.2. *Then for all* $\alpha \in \mathscr{A}$ *and all* $1 \leqq m, n \leqq Q$, *the following statements hold*.

(a) *Conclusions* (a) *through* (f) *of Theorem* 4.1 *are satisfied when we replace* $\mathbf{F}_{(T)}$ *with* $\mathbf{F}_{(T)}^{(m)}$, $\mathbf{V}_F$ *with* $\mathbf{V}_F^{(m)}$, $\mathbf{F}_B$ *with* $\mathbf{F}_B^{(m)}$, $\mathbf{V}_B$ *with* $\mathbf{V}_B^{(m)}$, $\bar{\mathbf{F}}$ *with* $\bar{\mathbf{F}}^{(m)}$, $\bar{\mathbf{V}}$ *with* $\bar{\mathbf{V}}^{(m)}$, $\bar{\bar{\mathbf{F}}}$ *with* $\bar{\bar{\mathbf{F}}}^{(m)}$, *and*

$$\bar{\bar{\mathbf{V}}} \quad \text{with } \bar{\bar{\mathbf{V}}}^{(m)}.$$

(b) *Conclusions* (a) *through* (f) *of Theorem* 4.1 *are satisfied when we replace* $\mathbf{F}_{(T)}$ *with* $\mathbf{F}_{(T)}^{(m,n)}$, $\mathbf{V}_F$ *with* $\mathbf{V}_F^{(m,n)}$, $\mathbf{F}_B$ *with* $\mathbf{F}_B^{(m,n)}$, $\mathbf{V}_B$ *with* $\mathbf{V}_B^{(m,n)}$, $\bar{\mathbf{F}}$ *with* $\bar{\mathbf{F}}^{(m,n)}$, $\bar{\mathbf{V}}$ *with* $\bar{\mathbf{V}}^{(m,n)}$, $\bar{\bar{\mathbf{F}}}$ *with* $\bar{\bar{\mathbf{F}}}^{(m,n)}$, *and*

$$\bar{\bar{\mathbf{V}}} \quad \text{with } \bar{\bar{\mathbf{V}}}^{(m,n)}.$$

Finally, Theorem 5.4 extends the conclusions of Theorems 3.1, 4.1, 5.1, 5.2, and 5.3 to indicate that for each $\alpha \in \mathscr{A}$ the coefficients of exponential decline are uniform in some neighborhood of α. As with Theorem 5.3, this extension is useful in the discussion of estimated autocovariance matrices.

THEOREM 5.4. *Assume the conditions of Theorem* 5.2. *Then for all* $\alpha \in \mathscr{A}$, *there exists some neighborhood* A_α *of* α *such that in the conclusions of Theorems* 3.1, 4.1, 5.1, 5.2, *and* 5.3 *the coefficients of exponential decline are uniform in* $\beta \in \mathscr{A}_\alpha$.

6. Conclusion. This paper amplifies and extends several authors' work on the inverse of an autocovariance matrix and on the convergence of weights in the prediction of unobserved linear processes. In § 2, slight extensions of [11] and [12] to multiple time series lead to general expressions for the inverse of the autocovariance matrix of a causal and invertible vector autoregressive moving average model. Section 3 outlines the limiting properties of the elements of the autocovariance matrix inverse. In particular, the inverse has exponentially declining off-diagonal blocks, and some elements of the inverse converge to limiting values at exponential rates. Demko [1], de Boor [2], and others have noted similar properties of the inverses of some band matrices. Section 3 also notes the relation of its results to similar convergence properties of innovation sequence coefficients and variances discussed in [8] and elsewhere. For some comments on the relation of the inverse $\mathbf{\Gamma}_{(TT)}^{-1}$ to the inverse autocorrelation function [18] and other frequency-domain

ideas, see [19] and [20]. Section 4 uses the results of § 3 to develop similar convergence properties for some latent-variable prediction coefficients, where the latent and observed variables follow a vector autoregressive moving average model. These results extend similar ideas on one-step-ahead prediction and updating in the optimal filtering literature (e.g., [5]) to prediction of a latent variable for any period. Finally, § 5 considers the parameterization of a vector autoregressive moving average model by a fixed-dimensional vector α. Uniform convergence rates for parameters in a neighborhood of α are developed. In addition, convergence of some associated matrix derivatives is discussed. These results are useful in the estimation of autocovariance matrix inverses and the estimation of latent-variable filter coefficients.

Appendix A. Implications of the causality and invertibility criteria. Assume that the k-dimensional random vectors $\{\mathbf{Z}_t, t = 1, 2, \cdots, T\}$ follow model (1.1) and that conditions (1.2a, b) are satisfied. Then we may make the following remarks about the covariance matrix $\mathbf{\Gamma}_{(TT)}$ and associated derivatives.

First, Lemma A.1 extends [8, Thms. 11.3.1, 11.3.2] to a functionally dependent parameterization of a vector autoregressive moving average model.

LEMMA A.1. *Assume model* (1.1). *Let the coefficient and variance matrices be continuous functions* $\{\mathbf{\Phi}_j(\alpha), j = 0, 1, \cdots, p\}$, $\{\mathbf{\Theta}_i(\alpha), i = 0, 1, \cdots, q\}$, *and* $\mathbf{\Sigma}_{gg}(\alpha)$ *of a fixed-dimensional parameter vector* $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_Q)'$ *and define the matrix polynomials* $\mathbf{\Phi}(\alpha, \omega)$ *and* $\mathbf{\Theta}(\alpha, \omega)$ *accordingly. Let* $\mathscr{A}_0$ *be a compact subset of* $\mathscr{R}^Q$.

(a) *If condition* (1.2a) *is satisfied for all* $\alpha \in \mathscr{A}_0$, *then there exists a finite positive real number* ε *such that*:

(a.i) *the polynomial* $\mathbf{\Phi}(\alpha, \omega)$ *is invertible for all* $\alpha \in \mathscr{A}_0$ *and all* $\omega \in \mathscr{C}$ *such that* $|\omega| \leqq 1 + \varepsilon$;

(a.ii) *for each* $\alpha \in \mathscr{A}_0$, *the rational function of* ω, $[\mathbf{\Phi}(\alpha, \omega)]^{-1}\mathbf{\Theta}(\alpha, \omega)$ *has a unique power series expansion*

$$[\mathbf{\Phi}(\alpha, \omega)]^{-1}\mathbf{\Theta}(\alpha, \omega) = \sum_{i=0}^{\infty} \mathbf{\Psi}_i(\alpha)\omega^i = \mathbf{\Psi}(\alpha, \omega),$$

say, for all $\omega \in \mathscr{C}$ *such that* $|\omega| \leqq 1 + \varepsilon$; *and*

(a.iii) *the sequence* $\{\mathbf{\Psi}_j(\alpha), j \in \mathscr{N}\}$ *is declining exponentially in* j, *uniformly in* $\alpha \in \mathscr{A}_0$.

(b) *If condition* (1.2b) *is satisfied for all* $\alpha \in \mathscr{A}_0$, *then there exists a finite positive real number* ε *such that*:

(b.i) *the polynomial* $\mathbf{\Theta}(\alpha, \omega)$ *is invertible for all* $\alpha \in \mathscr{A}_0$ *and for all* $\omega \in \mathscr{C}$ *such that* $|\omega| \leqq 1 + \varepsilon$;

(b.ii) *for all* $\alpha \in \mathscr{A}_0$, *the rational function of* z, $[\mathbf{\Theta}(\alpha, \omega)]^{-1}\mathbf{\Phi}(\alpha, \omega)$ *has a unique power series expansion*

$$[\mathbf{\Theta}(\alpha, \omega)]^{-1}\mathbf{\Phi}(\alpha, \omega) = \sum_{j=0}^{\infty} \mathbf{\Upsilon}_j(\alpha)\omega^j = \mathbf{\Upsilon}(\alpha, \omega),$$

say, for all $\omega \in \mathscr{C}$ *such that* $|\omega| \leqq 1 + \varepsilon$; *and*

(b.iii) *the sequence* $\{\mathbf{\Upsilon}_j(\alpha), j \in \mathscr{Z}\}$ *is declining exponentially in* j, *uniformly in* $\alpha \in \mathscr{A}_0$.

(c) *Let* $\mathscr{A}$ *be an open subset of* $\mathscr{R}^Q$ *such that for all* $\alpha \in \mathscr{A}$, (1.2a, b) *are satisfied and* $\det[\mathbf{\Sigma}_{gg}(\alpha)] > 0$. *Then for all* $\alpha \in \mathscr{A}$, *there exists some* $\delta > 0$ *such that for all elements* β *of the set* $\mathscr{A}_\delta \overset{\text{def}}{=} \{\beta \in \mathscr{R}^Q : |\beta - \alpha| \leqq \delta\}$, *conclusions* (a) *and* (b) *hold and* $\inf\{\det[\mathbf{\Sigma}_{gg}(\beta)], \beta \in \mathscr{A}_\delta\} > 0$.

The proof of Lemma A.1 follows from the proofs of [8, Thms. 11.3.1, 11.3.2] and some additional uniform continuity arguments.

Proof of Lemma A.1. Only the proof for parts (a) and (c) will be given. The proof for parts (a) and (b) is identical except for changes in notation.

(a) Note first that for any finite $\varepsilon \geqq 0$, the function $f(\alpha, \omega) = |\det[\Phi(\alpha, \omega)]|$ is uniformly continuous in (α, ω) on the compact set $\mathscr{A}_0 \times \mathscr{C}_\varepsilon$, where $\mathscr{C}_\varepsilon = \{\omega \in \mathscr{C} : |\omega| \leqq 1 + \varepsilon\}$ and $\times$ denotes the usual Cartesian product of two spaces. Thus, the image of $\mathscr{A}_0 \times \mathscr{C}_\varepsilon$ under $f(\alpha, \omega)$ is compact. For $\varepsilon = 0$, (1.2a) implies that the image of $\mathscr{A}_0 \times \mathscr{C}_\varepsilon$ under $f(\alpha, \omega)$ is bounded away from zero. Hence, there also exists some $\varepsilon > 0$ such that the image of $\mathscr{A}_0 \times \mathscr{C}_\varepsilon$ under $f(\alpha, \omega)$ is bounded away from zero. For this ε and all $\alpha \in \mathscr{A}_0$, $\Phi^{-1}(\alpha, \omega)$ exists for all $\omega \in \mathscr{C}_\varepsilon$ and $\Phi^{-1}(\alpha, \omega)$ has the power series expansion $\Phi^{-1}(\alpha, \omega) = \sum_{i=0}^{\infty} \mathbf{A}_i(\alpha)\omega^i = \mathbf{A}(\alpha, \omega)$, $\omega \in \mathscr{C}_\varepsilon$, say. Moreover, the uniform continuity of $\Phi^{-1}(\alpha, \omega)$ in (α, ω) on $\mathscr{A}_0 \times \mathscr{C}_\varepsilon$ implies that the elements of $\mathbf{A}_i(\alpha)$ are continuous in α, uniformly in $\alpha \in \mathscr{A}_0$ and in $i \in \mathscr{N}$. Thus $\lim_{i \to \infty} \|\mathbf{A}_i(\alpha)\|(1 + \varepsilon)^i = 0$ uniformly in $\alpha \in \mathscr{A}_0$, so there exists some $K > 0$ such that $\|\mathbf{A}_i(\alpha)\| \leqq K(1 + \varepsilon)^{-i}$ for all $\alpha \in \mathscr{A}_0$ and all $i \in \mathscr{N}$. Taking $c = \ln(1 + \varepsilon) > 0$, it follows that the sequence $\{\mathbf{A}_i(\alpha), i \in \mathscr{N}\}$ is declining exponentially in i, uniformly in $\alpha \in \mathscr{A}_0$. Since (trivially) the sequence $\{\Theta_i(\alpha), i = 0, 1, \cdots, q\}$ is declining exponentially in i, uniformly in $\alpha \in \mathscr{A}_0$, it follows from Lemma B.1 that the sequence $\{\Psi_i(\alpha), i \in \mathscr{N}\}$ is declining exponentially in i, uniformly in $\alpha \in \mathscr{A}_0$.

(c) The polynomial results for part (c) follow immediately from parts (a) and (b) and the observation that for all $\alpha \in \mathscr{A}$, there exists $\delta > 0$ such that the compact set $\{\beta : |\alpha - \beta| \leqq \delta\}$ is contained in $\mathscr{A}$. The determinant result for part (c) follows from the observation that on any compact subset $\mathscr{A}_0$ of $\mathscr{A}$, $\det[\Sigma_{gg}(\beta)]$ is uniformly continuous in $\beta \in \mathscr{A}_0$ and thus has a compact image bounded away from zero. □

Lemma A.1(a) leads to an infinite moving average representation of model (1.1), while Lemma A.1(b) leads to an infinite autoregressive representation of the same model. For a given $\alpha \in \mathscr{A}_0$, computational formulas for the matrices $\{\Psi_i(\alpha)\}$ and $\{\Upsilon_j(\alpha)\}$ are given in [8, p. 409].

Note that Lemma A.1(a) ensures the covariance stationarity of the observations $\{\mathbf{Z}_t\}$. Moreover, the relationship,

$$\text{(A.1)} \qquad \Gamma_{ZZ}(l) = \operatorname{Cov}(\mathbf{Z}_t, \mathbf{Z}_{t+l}) = \sum_{i=0}^{\infty} \Psi_i \Sigma_{gg} \Psi'_{i+l},$$

$l \in \mathscr{N}$, leads to the following result.

LEMMA A.2. *Assume model* (1.1). *Let the coefficient and variance matrices be continuous functions* $\{\Phi_j(\alpha), j = 1, 2, \cdots, p\}$, $\{\Theta_i(\alpha), i = 1, 2, \cdots, q\}$, *and* $\Sigma_{gg}(\alpha)$ *of a fixed-dimensional parameter vector* $\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_Q)'$. *Let* $\mathscr{A}$ *be some open subset of* $\mathscr{R}^Q$ *such that for all* $\alpha \in \mathscr{A}$, (1.2a) *is satisfied and* $\{\Phi_j(\alpha)\}$, $\{\Theta_i(\alpha)\}$, *and* $\Sigma_{gg}(\alpha)$ *are twice continuously differentiable in* α. *Let* $\mathscr{A}_0$ *be some compact subset of* $\mathscr{A}$.

(a) *Then the lagged autocovariance matrices* $\{\Gamma_{ZZ}(l)\}$ *are twice continuously differentiable functions of* α.

Moreover, the following sequences of matrices are all declining exponentially in the lags $l \in \mathscr{Z}^+$, *uniformly in* $\alpha \in \mathscr{A}_0$:

(b) *the lagged autocovariances* $\Gamma_{ZZ}(l)$;

(c) *the first derivative matrices* $[\Gamma_{ZZ}(l)]^{(i)}$, $i = 1, 2, \cdots, Q$; and

(d) *the second derivative matrices* $[\Gamma_{ZZ}(l)]^{(ij)}$, $i, j = 1, 2, \cdots, Q$.

Proof of Lemma A.2. (a) By [8, (11.3.12)],

$$\mathbf{\Psi}_j(\alpha) = \sum_{i=1}^{j} \mathbf{\Phi}_j(\alpha)\mathbf{\Psi}_{j-i}(\alpha) + \mathbf{\Theta}_j(\alpha), \qquad j \in \mathscr{Z}^+,$$

where $\mathbf{\Psi}_j(\alpha) = \mathbf{0}$, $j < 0$, $\mathbf{\Phi}_j(\alpha) = \mathbf{0}$, $j > p$, and $\mathbf{\Theta}_i(\alpha) = \mathbf{0}$, $i > q$. Lemma A.2(a) then follows from (A.1) and Fubini's Theorem.

(b) By Lemma A.1(a), let K and c be positive numbers such that $\|\mathbf{\Psi}_i(\alpha)\| \leqq K \exp(-ci)$ for all $i \in \mathscr{N}$ and all $\alpha \in \mathscr{A}_0$. Then

$$\|\mathbf{\Gamma}_{ZZ}(l)\| \leqq \sum_{i=0}^{\infty} \|\mathbf{\Psi}_i(\alpha)\| \cdot \|\mathbf{\Sigma}_{gg}(\alpha)\| \cdot \|\mathbf{\Psi}_{i+l}(\alpha)\|$$

$$\leqq \|\mathbf{\Sigma}_{gg}(\alpha)\| K^2 \exp(-cl) \sum_{i=0}^{\infty} \exp(-2ci)$$

and Lemma A.2(b) follows.

(c) and (d) Again using the notation of [8, p. 408], let $\mathbf{A}(\alpha, \omega) = \sum_{j=0}^{\infty} \mathbf{A}_j(\alpha)\omega^j = [\mathbf{\Phi}(\alpha, \omega)]^{-1}$ and note that the matrices $\{\mathbf{A}_j(\alpha)\}$ are declining exponentially in j, uniformly in $\alpha \in \mathscr{A}_0$. Repeating the arguments for Lemma A.2(a), the matrices $\{\mathbf{A}_j(\alpha)\}$ are twice continuously differentiable in α. Fubini's Theorem, the relations,

$$\sum_{l=0}^{\infty} \left[\frac{d}{d\alpha_k}(\mathbf{A}_{\alpha l})\right]\omega^l = \frac{d}{d\alpha_k}[\mathbf{A}(\alpha, \omega)] = \mathbf{A}(\alpha, \omega)\left\{\frac{d}{d\alpha_k}[\mathbf{\Phi}(\alpha, \omega)]\right\}\mathbf{A}(\alpha, \omega),$$

collection of the coefficients of ω^l, $l \in \mathscr{Z}^+$, and Lemma B.1 imply that the matrices $\{(d/d\alpha_k)[\mathbf{\Gamma}_{ZZ}(l)]\}$ are declining exponentially in l, uniformly in $\alpha \in \mathscr{A}_0$. Lemma A.2(d) follows from similar arguments. □

Appendix B. Lemmas and proofs. Proofs of the main theorems in this paper use the following two algebraic results associated with Definitions 1.1 and 1.2.

LEMMA B.1. *Using the notation of Definition* 1.1(c), *for each* $T \in \mathscr{Z}^+$ *and each* $\alpha \in \mathscr{A}$, *let* $\{\mathbf{A}_{Tj}(\alpha), j \in \mathscr{N}\}$ *and* $\{\mathbf{B}_{Tj}(\alpha), j \in \mathscr{N}\}$ *be two sequences of matrices of dimensions* $r \times k$ *and* $k \times n$, *respectively, which are declining exponentially in* j, *uniformly in* T, *and in* $\alpha \in \mathscr{A}_0$. *Then the convolution*

$$\mathbf{C}_{Tk}(\alpha) = \sum_{j=0}^{k} \mathbf{A}_{T,k-j}(\alpha)\mathbf{B}_{Tj}(\alpha), \qquad k \in \mathscr{N}$$

is declining exponentially in k, *uniformly in* T *and in* $\alpha \in \mathscr{A}_0$.

Proof of Lemma B.1. Let K_A, K_B, c_A, and c_B be the coefficients of exponential decline for $\{\mathbf{A}_{Tj}(\alpha)\}$ and $\{\mathbf{B}_{Tj}(\alpha)\}$ required by Definition 1.1(c). Assume first that $c_B > c_A$. Then the norm inequality $\|\mathbf{A}_{T,k-j}(\alpha)\mathbf{B}_{Tj}(\alpha)\| \leqq \|\mathbf{A}_{T,k-j}(\alpha)\| \cdot \|\mathbf{B}_{Tj}(\alpha)\|$ [9, p. 197] implies that

$$\|\mathbf{C}_{Tk}(\alpha)\| \leqq \sum_{j=0}^{k} \|\mathbf{A}_{T,k-j}(\alpha)\| \cdot \|\mathbf{B}_{Tj}(\alpha)\| \leqq K_A K_B\{1 - \exp[-(c_B - c_A)]\}^{-1} \exp(-c_A k)$$

and Lemma B.1 follows. The cases $c_A > c_B$ and $c_A = c_B$ are argued similarly. □

LEMMA B.2. *Using the notation of Definition* 1.2(b), *for each* $\alpha \in \mathscr{A}$, *let* $\{\mathbf{A}_T(\alpha), T \in \mathscr{Z}^+\}$ *and* $\{\mathbf{B}_T(\alpha), T \in \mathscr{Z}^+\}$ *be two sequences of real matrices of dimensions* $Tr \times$

Tk and Tk × Tn, respectively, which have bounded on-diagonal blocks and exponentially declining off-diagonal blocks uniformly in $\alpha \in \mathscr{A}_0$. *Define* $\mathbf{C}_T(\alpha) = \mathbf{A}_T(\alpha)\mathbf{B}_T(\alpha)$. *Then the sequences* $\{\mathbf{C}_T(\alpha), T \in \mathscr{Z}^+\}$ *also have bounded on-diagonal blocks and exponentially declining off-diagonal blocks uniformly in* $\alpha \in \mathscr{A}_0$.

Proof of Lemma B.2. The result follows immediately from the proof of Lemma B.1 and the inequalities,

$$\|\mathbf{C}_{Tik}(\alpha)\| \leqq \sum_{j=1}^{T} \|\mathbf{A}_{Tij}(\alpha)\| \cdot \|\mathbf{B}_{Tjk}(\alpha)\| \leqq K_A K_B \sum_{j=1}^{T} \exp\{-(c_A|i-j| + c_B|j-k|)\},$$

where K_A, K_B, c_A, and c_B are the K and c constants of $\{\mathbf{A}_T(\alpha)\}$ and $\{\mathbf{B}_T(\alpha)\}$ required by Definition 1.2(b). □

Proof of Theorem 5.1. Only proofs for the first derivative results are presented here. The second derivative results follow from similar arguments. Also, Fubini's Theorem will be used repeatedly without further comment.

(a) Note from (3.3) that for $d \geqq 0$,

$$\text{(B.1)} \quad \mathbf{C}_{r,r+d}^{(m)} - \bar{\mathbf{C}}_d^{(m)} = -\sum_{l=r+1}^{\infty} (\mathbf{P}_l^{(m)\prime}\boldsymbol{\Sigma}_{gg}^{-1}\mathbf{P}_{l+d} + \mathbf{P}_l'\boldsymbol{\Sigma}_{gg}^{-1}\mathbf{P}_{l+d}^{(m)} - \mathbf{P}_l'\boldsymbol{\Sigma}_{gg}^{-1}\boldsymbol{\Sigma}_{gg}^{(m)}\boldsymbol{\Sigma}_{gg}^{-1}\mathbf{P}_{l+d}).$$

By (2.6) and the proof of Lemma A.2(c), the matrices $\mathbf{P}_l^{(m)}$ are declining exponentially in l. Also, $\boldsymbol{\Sigma}_{gg}^{-1}\boldsymbol{\Sigma}_{gg}^{(m)}\boldsymbol{\Sigma}_{gg}^{-1}$ is bounded in norm for fixed $\alpha \in \mathscr{A}$. Thus expression (B.1) is declining exponentially in r, uniformly in d, and declining exponentially in d, uniformly in r. Next, note from (3.4) that for $d \geqq 0$,

$$\begin{aligned}\|\mathbf{B}_{r,r+d}^{(m)} - \bar{\mathbf{B}}_{r,r+d}^{(m)}\| \leqq{}& \sum_{i,j=0}^{p} \|\boldsymbol{\Phi}_i^{(m)}(\mathbf{C}_{i+r,j+r+d} - \bar{\mathbf{C}}_{j+d-i})\boldsymbol{\Phi}_j'\| \\ &+ \sum_{i,j=0}^{p} \|\boldsymbol{\Phi}_i(\mathbf{C}_{i+r,j+r+d} - \bar{\mathbf{C}}_{j+d-i})\boldsymbol{\Phi}_j^{(m)\prime}\| \\ &+ \sum_{i,j=0}^{p} \|\boldsymbol{\Phi}_i(\mathbf{C}_{i+r,j+r+d}^{(m)} - \bar{\mathbf{C}}_{j+d-i}^{(m)})\boldsymbol{\Phi}_j'\|.\end{aligned} \quad \text{(B.2)}$$

For fixed $\alpha \in \mathscr{A}$, $\boldsymbol{\Phi}_i^{(m)}$ is bounded uniformly in $1 \leqq i \leqq p$, so three separate repetitions of the arguments for (3.4) indicate that there exist $K > 0$ and $c > 0$ such that each of the sums on the right-hand side of (B.2) is bounded by $K \exp[-c(|r| + |r+d|)]$. Thus, $\mathbf{B}_{r,r+d}^{(m)} - \bar{\mathbf{B}}_{r,r+d}^{(m)}$ is declining exponentially in r, uniformly in T and d, and is declining exponentially in d, uniformly in T and r.

(b) By expression (3.5),

$$\text{(B.3a)} \quad \mathbf{G}_{TTrs}^{(m)} - \mathbf{B}_{rs}^{(m)} = \mathbf{D}_r^{(m)}\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}\mathbf{D}_s'$$

$$\text{(B.3b)} \quad + \mathbf{D}_r\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}[\mathbf{D}_s^{(m)}]'$$

$$\text{(B.3c)} \quad + \mathbf{D}_r[\mathbf{V}^{1/2}(\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}]^{(m)}\mathbf{D}_s'.$$

Note that by (3.6), there exists some $K > 0$ such that $\mathbf{D}_r^{(m)}$ is bounded in norm by

$$\text{(B.4)} \quad K\sum_{j=r}^{r+p}\sum_{l=1}^{j} (\|\mathbf{P}_{j-l}'\boldsymbol{\Sigma}_{gg}^{-1}\mathbf{P}_{T-l}^{(m)}\| + \|\mathbf{P}_{j-l}^{(m)\prime}\boldsymbol{\Sigma}_{gg}^{-1}\mathbf{P}_{T-l}\| + \|\mathbf{P}_{j-l}'\boldsymbol{\Sigma}_{gg}^{-1}\boldsymbol{\Sigma}_{gg}^{(m)}\boldsymbol{\Sigma}_{gg}^{-1}\mathbf{P}_{T-l}\|).$$

Repetition of the arguments in the proof of Theorem 3.1(b) indicates that (B.4) and (B.3a) are declining exponentially in T for fixed r. These same arguments in-

dicate that $\mathbf{D}_s^{(m)}$ is bounded uniformly in s and T, so (B.3b) is declining exponentially in T. Finally, note that by (3.7) and the functional independence of $\mathbf{V}$ from T, $[\mathbf{V}^{1/2} \times (\mathbf{V}^{1/2}\mathbf{E}'\mathbf{A}^{-1}\mathbf{E}\mathbf{V}^{1/2} + \mathbf{I})^{-1}\mathbf{V}^{1/2}]^{(m)}$ is bounded in norm uniformly in T. Thus (B.3c) is declining exponentially in T, uniformly in $s \in \mathscr{L}^+$.

(c) The proof of part (c) is identical to the proof of part (b), except that the index r is replaced by $r(T)$ and, using arguments similar to those for (3.8), the derivative matrices $\mathbf{D}_{r(T)}^{(m)}$ are seen to be declining exponentially in $T - r(T)$ and to be bounded uniformly in $r(T)$ and T.

(d) and (e) Note that $\mathbf{G}_{TT}^{(m)} = -\mathbf{G}_{TT}\boldsymbol{\Gamma}_{(TT)}^{(m)}\mathbf{G}_{TT}$, and recall from Theorem 3.1(e) and Lemma A.2 that $\mathbf{G}_{TT}$ and $\boldsymbol{\Gamma}_{(TT)}^{(m)}$ have exponentially declining off-diagonal blocks. Part (e) then follows immediately from Lemma B.2. Part (d) follows from the additional remark that if a sequence of matrices have exponentially declining off-diagonal blocks, then their elements are uniformly bounded. □

Proof of Theorem 5.2. The theorem follows immediately from the derivative expressions

$$\mathbf{F}^{(m)} = (\boldsymbol{\Gamma}_{xZ}^{(m)}\boldsymbol{\Gamma}_{(TT)}^{-1} - \boldsymbol{\Gamma}_{xZ}\boldsymbol{\Gamma}_{(TT)}^{-1}\boldsymbol{\Gamma}_{(TT)}^{(m)}\boldsymbol{\Gamma}_{(TT)}^{-1}),$$

$$\mathbf{V}_F^{(m)} = \boldsymbol{\Gamma}_{xx}^{(m)} - \boldsymbol{\Gamma}_{xZst}^{(m)}\boldsymbol{\Gamma}_{(TT)}^{-1}\boldsymbol{\Gamma}_{xZst}' - \boldsymbol{\Gamma}_{xZst}\boldsymbol{\Gamma}_{(TT)}^{-1}[\boldsymbol{\Gamma}_{xZst}^{(m)}]' + \boldsymbol{\Gamma}_{xZst}\boldsymbol{\Gamma}_{(TT)}^{-1}\boldsymbol{\Gamma}_{(TT)}^{(m)}\boldsymbol{\Gamma}_{(TT)}^{-1}\boldsymbol{\Gamma}_{xZst}',$$

$1 \leqq m \leqq Q$, similar expressions for $\mathbf{F}^{(m,n)}$ and $\mathbf{V}_F^{(m,n)}$, Lemma A.2, Lemma B.2, and parts (g) and (h) of Theorem 4.1. □

Proof of Theorem 5.4. For any $\alpha \in \mathscr{A}$, there exists $\delta > 0$ such that $\mathscr{A}_{\alpha\delta} \overset{\text{def}}{=} \{\beta \in \mathscr{R}^Q : |\beta - \alpha| < \delta\} \subset \mathscr{A}$. For this δ, the arguments in the proofs of Lemmas A.1 and A.2 indicate that the coefficients of exponential decline for the matrices $\boldsymbol{\Gamma}_{xZ}(l)$ and $\mathbf{P}_l$ are uniform in $\beta \in \mathscr{A}_{\alpha\delta}$, so each of the coefficients of exponential decline in the proofs of Theorems 3.1, 4.1, 5.1–5.3 may be replaced by coefficients that are uniform in $\beta \in \mathscr{A}_{\alpha\delta}$. Theorem 5.4 then follows immediately from Lemmas A.1, A.2, B.1, and B.2. □

Acknowledgments. The author thanks Wayne A. Fuller for suggesting a study of the exponentially declining properties of autocovariance matrix inverses, Richard A. Davis for helpful comments on exponential convergence rates for prediction coefficients and error variances, and Joe Newton and Cliff Spiegelman for comments on an earlier manuscript. The author also thanks the associate editor for very helpful suggestions.

REFERENCES

[1] S. DEMKO, *Inverses of band matrices and local convergence of spline projections*, SIAM J. Numer. Anal., 14 (1977), pp. 616–619.

[2] C. DE BOOR, *Dichotomies for band matrices*, SIAM J. Numer. Anal., 17 (1980), pp. 894–907.

[3] P. J. HARRISON AND C. F. STEVENS, *Bayesian forecasting* (with discussion), J. Roy. Statist. Soc. Ser. B, 38 (1976), pp. 205–247.

[4] J. Q. SMITH, *A generalization of the Bayesian steady forecasting model*, J. Roy. Statist. Soc. Ser. B, 41 (1979), pp. 375–387.

[5] B. D. O. ANDERSON AND J. B. MOORE, *Optimal Filtering*, Prentice-Hall, Englewood Cliffs, NJ, 1979.

[6] W. W. BARRET, *Toeplitz matrices with banded inverses*, Linear Algebra Appl., 57 (1984), pp. 131–145.

[7] R. P. MENTZ, *A note on inverting a covariance matrix of Toeplitz type*, Bull. Inst. Internat. Statist., 46 (4) (1975), pp. 151–155.

[8] P. J. BROCKWELL AND R. A. DAVIS, *Time Series: Theory and Methods*, Springer-Verlag, New York, 1987.

[9] P. R. HALMOS, *Finite-Dimensional Vector Spaces*, Springer-Verlag, New York, 1974.

[10] J. R. MAGNUS AND H. NEUDECKER, *Matrix Differential Calculus with Applications in Statistics and Econometrics*, John Wiley, New York, 1988.

[11] O. D. ANDERSON, *An improved approach to inverting the autocovariance matrix of a general mixed autoregressive moving average time process*, Austral. J. Statist., 18 (1976), pp. 73–75.
[12] J. G. DE GOOIJER, *On the inverse of the autocovariance matrix for a general mixed autoregressive moving average process*, Statist. Hefte, 19 (1978), pp. 114–123.
[13] R. F. GALBRAITH AND J. I. GALBRAITH, *On the inverses of some patterned matrices arising in the theory of stationary time series*, J. Appl. Probab., 11 (1974), pp. 63–71.
[14] O. D. ANDERSON, *On the inversion of autocovariance matrices for general autoregressive moving average* (p, q) *processes*, Metron, 35 (1977), pp. 243–254.
[15] E. J. HANNAN, *The identification of vector mixed autoregressive-moving average systems*, Biometrika, 56 (1969), pp. 223–225.
[16] A. H. JAZWINSKI, *Stochastic Processes and Filtering Theory*, Academic Press, New York, 1970.
[17] J. L. ELTINGE, *Measurement error models for time series*, unpublished Ph.D. thesis, Department of Statistics, Iowa State University, Ames, IA, 1987.
[18] W. S. CLEVELAND, *The inverse autocovariances of a time series and their applications*, Technometrics, 14 (1972), pp. 277–293.
[19] P. SHAMAN, *An approximate inverse for the covariance matrix of moving average and autoregressive processes*, Ann. Statist., 3 (1975), pp. 532–538.
[20] ———, *Approximations for stationary covariance matrices and their inverses with application to* ARMA *models*, Ann. Statist., 4 (1976), pp. 292–301.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1085–1093, October 1992

006

A TWO-DIMENSIONAL BISECTION METHOD FOR SOLVING TWO-PARAMETER EIGENVALUE PROBLEMS*

XINGZHI JI†

Abstract. It has long been known that separation of variables can be applied to the Helmholtz equation in 11 three-dimensional coordinate systems. As a result, a multiparameter eigenvalue problem is formed. In this paper, the well-known bisection method for ordinary eigenvalue problems is generalized to a special class of discrete two-parameter eigenvalue problems. The range of the real roots of the problem is discussed and some numerical results are given.

Key words. two-parameter eigenvalue problem, two-dimensional bisection, range of real roots

AMS(MOS) subject classifications. 65F15, 65L10, 65N25

1. Introduction. It is well known that the bisection method is very effective for some eigenvalue problems of a real symmetric matrix [8, p. 305]. The main objective of this paper is to extend the bisection method to the following two-parameter eigenvalue problem:

$$(T_1 + \lambda B_1 + \mu C_1)x_1 = 0, \tag{1.1}$$

$$(T_2 + \lambda B_2 + \mu C_2)x_2 = 0, \tag{1.2}$$

where T_i, B_i, C_i $(i = 1, 2)$ satisfy the following condition (TBC for short): $T_i \in \mathbb{R}^{n\times n}$ are irreducible symmetric tridiagonal matrices; B_i, $C_i \in \mathbb{R}^{n\times n}$ are nonsingular diagonal matrices: $B_i = \operatorname{diag}(b_{i,1}, \cdots, b_{i,n})$, $C_i = \operatorname{diag}(c_{i,1}, \cdots, c_{i,n})$, $\operatorname{sign}(b_{i,1}) = \operatorname{sign}(b_{i,j})$, $\operatorname{sign}(c_{i,1}) = \operatorname{sign}(c_{i,j})$, $j = 2, \cdots, n$. The real pairs (λ, μ) and x_1, x_2 are the eigenvalues and the corresponding eigenvectors to be found.

The motivation for investigating the problem (1.1), (1.2) is the numerical study of two-parameter Sturm–Liouville (S–L) eigenvalue problems such as

$$\begin{aligned} &-X'' + q_1(x)X = (\lambda s_1(x) + \mu t_1(x))X, \\ &-Y'' + q_2(y)Y = (\lambda s_2(y) + \mu t_2(y))Y, \\ &X(a_1) = X(b_1) = 0, \\ &Y(a_2) = Y(b_2) = 0, \end{aligned} \tag{1.3}$$

where q_i, s_i, t_i $(i = 1, 2)$ are real-valued, piecewise continuous functions. The above problems arise from solving the Helmholtz equation by separation of variables. For example, consider the problem

$$\begin{cases} -\Delta u = \lambda u & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega, \end{cases} \tag{1.4}$$

where $\Omega \in \mathbb{R}^2$ is bounded by symmetric orthogonal parabolas confocal to the origin, and $\partial\Omega$ is the boundary of Ω. Applying the method of separation of variables to the problem (1.4) in a parabolic coordinate system will lead to the following two-parameter S–L problem (cf. [9]):

* Received by the editors April 30, 1990; accepted for publication (in revised form) March 5, 1991.
† Department of Mathematics, University of Alberta, Edmonton, Alberta, Canada T6G 2G1 (ji@qucis.queensu.ca).

$$
\begin{aligned}
&-X'' = (\lambda + \mu x^2)X,\\
&-Y'' = (-\lambda + \mu y^2)Y,\\
&X(0) = X(1) = 0,\\
&Y(-1) = Y(1) = 0.
\end{aligned}
\tag{1.5}
$$

Using the second central-difference approximation for X'' and Y'', (1.3) leads to the two-parameter discrete eigenvalue problem (1.1), (1.2). It is easy to see that the condition *TBC* will hold if s_i, t_i $(i = 1, 2)$ do not change sign.

It is well known that there are 11 three-dimensional coordinate systems in which the Helmholtz equation (also the Laplace equation) is directly separable [1], [10], [13]. In some cases (e.g., rectangular coordinate), separation constants are not coupled. Then the method of Blum and Reid [4] can be used with inhomogeneous boundary conditions. In other cases, however, the separation constants cannot themselves be decoupled. They appear as spectral parameters in each of the attendant ordinary differential equations, and multiparameter S–L problems are consequently formed [10], [12].

In [2], through Prüfer phase transformation, Bailey presented an alternating parameter iterative method for general two-parameter S–L problems and gave some numerical examples that show the algorithm to be rather effective. Fox, Blum, and others also investigated three-point boundary value problems very similar to problems (1.3) or (1.1), (1.2) (see [5]–[7]).

The two-dimensional bisection method presented in the following section can locate the eigenvalues of the problem (1.1) and (1.2) in any specified rectangle with sides parallel to the coordinate axes. Being a generalization of the ordinary bisection method, the algorithm is stable and the results are accurate. In § 3, we discuss the range of the real roots of the problem (1.1), (1.2). Some numerical results are given in § 4.

For the sake of simplicity, we have let the order of the matrices in (1.1) be equal to that in (1.2). But all results follow analogously for the nonequal case.

Throughout the paper, we use the notation $[t_{i,i-1}, t_{ii}, t_{i,i+1}]_{i=1}^{n}$ to denote the tridiagonal matrix

$$
\begin{pmatrix}
t_{11} & t_{12} & & & \\
t_{21} & t_{22} & \cdot & & \\
 & \cdot & \cdot & \cdot & \\
 & & \cdot & \cdot & t_{n-1,n}\\
 & & t_{n,n-1} & & t_{nn}
\end{pmatrix}.
$$

2. Two-dimensional bisection method. Consider the double eigenvalue problem (1.1), (1.2). We define

$$f_1(\lambda, \mu) = \det(T_1 + \lambda B_1 + \mu C_1),$$

$$f_2(\lambda, \mu) = \det(T_2 + \lambda B_2 + \mu C_2).$$

Then our aim is to locate the intersection points of the two families of eigencurves $f_1(\lambda, \mu) = 0$ and $f_2(\lambda, \mu) = 0$ in the $\lambda - \mu$ plane. Since the eigenvalue is the continuous function of the entries of a matrix, the following properties of the eigencurves, $f_1 = 0$ and $f_2 = 0$, can be easily derived due to the condition *TBC*.

PROPERTY 1. Any straight line $\lambda = \lambda_0$ or $\mu = \mu_0$ has n noncoincident intersection points with eigencurves of $f_1 = 0$ or $f_2 = 0$.

PROPERTY 2. Each $f_i = 0$ $(i = 1, 2)$ has n eigencurves $\lambda_1^{(i)}(\mu) = 0, \cdots, \lambda_n^{(i)}(\mu) = 0$, which are continuous in $\{-\infty < \lambda < +\infty, -\infty < \mu < +\infty\}$.

PROPERTY 3. The n eigencurves of $f_1 = 0$ or $f_2 = 0$ are strictly monotone.

Let $[a_1, a_2; b_1, b_2]$ denote a rectangle that is made of four straight lines: $\lambda = a_1$, $\lambda = a_2$, $\mu = b_1$, and $\mu = b_2$. It follows from the above properties that if an eigenpair (λ, μ) of the problem (1.1) and (1.2) lies in the rectangle $[a_1, a_2; b_1, b_2]$, there must be eigencurves of both $f_1 = 0$ and $f_2 = 0$ that cross through at least one of any three sides of that rectangle. By constructing Sturm sequences, we can check whether there are eigencurves through any given side of the rectangle.

Let us denote $f_{i,k}(\lambda, \mu)$ as the leading $k \times k$ principal minors of $T_i + \lambda B_i + \mu C_i$ $(i = 1, 2; k = 1, 2, \cdots, n)$. And let $f_{i,0} = 1, f_{i,-1} = 0$, $\lambda = \lambda_0$. We have the three-term recurrence formulas:

$$f_{i,j}(\lambda_0, \mu) = [(t_{jj}^{(i)} + b_{i,j}\lambda_0) + c_{i,j}\mu]f_{i,j-1}(\lambda_0, \mu) - t_{j,j-1}^{(i)2} f_{i,j-2}(\lambda_0, \mu),$$

$$i = 1, 2; \qquad j = 1, 2, \cdots, n.$$

As the polynomial sequences in μ, $\{f_{i,j}(\lambda_0, \mu)\}_{j=0}^{n}$ $(i = 1, 2)$ satisfy the following: (1) $f_{i,0}$ has no real roots; (2) For $j = 1, \cdots, n-1, f_{i,j-1}(\lambda_0, \mu_0)f_{i,j+1}(\lambda_0, \mu_0) < 0$ if μ_0 is a real root of $f_{i,j}(\lambda_0, \mu)$.

Hence, $\{f_{i,j}(\lambda_0, \mu)\}_0^n$ $(i = 1, 2)$ are Sturm sequences about μ. Furthermore, the number of roots of $f_{i,n}(\lambda_0, \mu)$ in (a, b) equals $|V(a) - V(b)|$, where neither a nor b is a root of $f_{i,n}(\lambda_0, \mu)$, and $V(a)$ and $V(b)$ are the numbers of sign changes of $\{f_{i,j}(\lambda_0, \mu)\}_0^n$ at $\mu = a$ and $\mu = b$, respectively [11].

Therefore, we obtain the following two-dimensional bisection algorithm for solving the eigenvalues of the problem (1.1) and (1.2) in a rectangle $[a_1, a_2; b_1, b_2]$, where ε is a suitable criterion for termination.

ALGORITHM.

(1) Calculate respectively for $\{f_{1,j}\}_0^n$, $\{f_{2,j}\}_0^n$ the number of sign changes at the vertices of three sides $\lambda = a_1$, a_2, and $\mu = b_1$ and form differences between the two corresponding numbers for each side. Let V_{11}, V_{12}, V_{13} and V_{21}, V_{22}, V_{23} denote the absolute values of these differences for $\{f_{1,j}\}_0^n$ and $\{f_{2,j}\}_0^n$, respectively. Set $V_1 = V_{11} + V_{12} + V_{13}$, $V_2 = V_{21} + V_{22} + V_{23}$.

(2) If $V_1 * V_2 = 0$, then there is no solution in $[a_1, a_2; b_1, b_2]$.

(3) If $V_1 * V_2 \neq 0$:

(i) if $\max\{|a_1 - a_2|, |b_1 - b_2|\} \leqq \varepsilon$, then we have an approximate eigenvalue pair $\lambda_0 = (a_1 + a_2)/2$, $\mu_0 = (b_1 + b_2)/2$; otherwise,

(ii) taking $a_3 = (a_1 + a_2)/2$, $b_3 = (b_1 + b_2)/2$, we obtain four smaller rectangles.

(4) Repeat the above steps (1)–(3) for each smaller rectangle obtained.

During computation, we may adopt adjustable arrays to store the coordinates of increasing vertices. If $|a_{i+1} - a_i| \leqq \varepsilon$ but $|b_{i+1} - b_i| > \varepsilon$, then we use bisection for $\overline{b_i b_{i+1}}$ and not for $\overline{a_i a_{i+1}}$.

In order to prevent underflow and overflow, we can adopt the technique described in [14]. That is, we apply the algorithm to the sequence

$$s_{i,j}(\lambda, \mu) = f_{i,j}(\lambda, \mu)/f_{i,j-1}(\lambda, \mu), \quad i = 1, 2; \quad j = 0, 1, \cdots, n.$$

It is worth noting that even if $V_1 * V_2 \neq 0$ in step (3) of the above algorithm, one cannot assert that there exist solutions in that rectangle. Then further computation is needed. For this reason, the two-dimensional bisection method is inferior to the ordinary bisection. However, the two-dimensional bisection is still very stable and can be extended

to the full matrix cases where T_i are symmetric and B_i, C_i are positive/negative definite ($i = 1, 2$) (see [3]).

3. Range of the roots. In application, an initial rectangle may be chosen according to various practical needs. However, we hope to obtain a rectangle that contains all the roots of the problem. We expect, as in the Gershgorin circle theorem, that the range of real roots can be found in advance by simple computation. In this section, we discuss the problem.

Let us consider a tridiagonal matrix containing two parameters λ and μ:

$$A_1(\lambda, \mu) = [\eta_{i-1}, \alpha_i + \beta_i\lambda + \gamma_i\mu, \xi_i]_{i=1}^n, \tag{3.1}$$

and define for $i = 1, 2, \cdots, n$

$$\nu_i = -\beta_i/\gamma_i, \quad M_i = (|\eta_{i-1}| + |\alpha_i| + |\xi_i|)/|\gamma_i|, \quad \text{where } \eta_0 = \eta_n = 0.$$
$$\rho = \min_{1 \leqq i \leqq n} \{\nu_i\}, \quad \sigma = \max_{1 \leqq i \leqq n} \{\nu_i\}; \quad M = \max_{1 \leqq i \leqq n} \{M_i\}. \tag{3.2}$$

Obviously, ρ, σ, and M are easily computed. We use these values to give the range of the real roots.

LEMMA 3.1. *Assume that* $\gamma_i \neq 0$, $i = 1, \cdots, n$. *The range of the real roots of* $\det A_1(\lambda, \mu) = 0$ *are bounded by the following four half-lines* (*cf. Fig.* 3.1):

On $\lambda \geqq 0$ *half-plane* l_1: $\mu = \rho\lambda - M$, (3.3)

l_2: $\mu = \sigma\lambda + M$, (3.4)

On $\lambda \leqq 0$ *half-plane* l_3: $\mu = \rho\lambda + M$, (3.5)

l_4: $\mu = \sigma\lambda - M$. (3.6)

Proof. Applying the Gerschgorin theorem to the matrix $A_1(\lambda, \mu)$, we obtain the following inequalities:

$$|\alpha_i + \beta_i\lambda + \gamma_i\mu| \leqq |\eta_{i-1}| + |\xi_i|, \qquad i = 1, 2, \cdots, n.$$

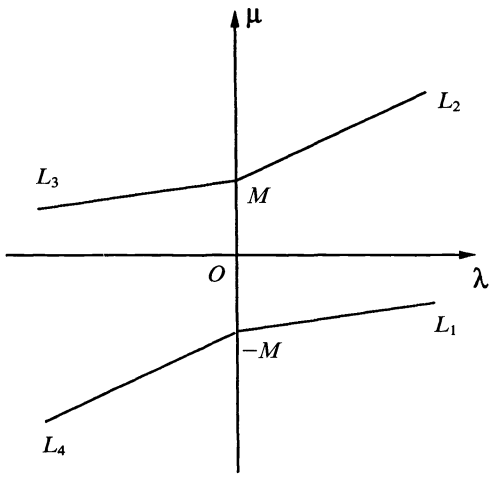


FIG. 3.1

Hence

$$|\beta_i\lambda + \gamma_i\mu| \leqq |\eta_{i-1}| + |\alpha_i| + |\xi_i|, \qquad i = 1, 2, \cdots, n.$$

Since $\gamma_i \neq 0$ $(i = 1, 2, \cdots, n)$,

$$-(|\eta_{i-1}| + |\alpha_i| + |\xi_i|)/|\gamma_i| \leqq \beta_i\lambda/\gamma_i + \mu \leqq (|\eta_{i-1}| + |\alpha_i| + |\xi_i|)/|\gamma_i|, \qquad i = 1, 2, \cdots, n.$$

Or

$$\nu_i\lambda - M_i \leqq \mu \leqq \nu_i\lambda + M_i, \qquad i = 1, 2, \cdots, n.$$

All the real eigenvalues of $A_1(\lambda, \mu)$ are located in the union of the regions determined by these n inequalities. Hence it follows that they fall into the region bounded by the four half-lines:

$$\text{On } \lambda \geqq 0 \text{ half-plane: } \rho\lambda - M \leqq \mu \leqq \sigma\lambda + M,$$

$$\text{On } \lambda \leqq 0 \text{ half-plane: } \sigma\lambda - M \leqq \mu \leqq \rho\lambda + M. \qquad \square$$

LEMMA 3.2. *The region bounded by the rays l_1, l_2, l_3, and l_4 is symmetric about the origin.*

Similarly, we can define ρ', σ', and M' for

$$A_2(\lambda, \mu) = [\eta'_{i-1}, \alpha'_i + \beta'_i\lambda + \gamma'_i\mu, \xi'_i]_{i=1}^n. \tag{3.7}$$

The real roots of $\det A_2(\lambda, \mu) = 0$ in the right half-plane fall between l'_1 and l'_2, where

$$l'_1: \quad \mu = \rho'\lambda - M', \tag{3.8}$$

$$l'_2: \quad \mu = \sigma'\lambda + M'. \tag{3.9}$$

In line with Lemma 3.2, it is sufficient to discuss the range of real roots of

$$\det A_1(\lambda, \mu) = 0, \qquad \det A_2(\lambda, \mu) = 0$$

in the $\lambda \geqq 0$ half-plane.

Let $P\{l_1 - l_2; l'_1 - l'_2\}$ denote the intersection set of the two regions on the right half-plane formed by l_1, l_2 and l'_1, l'_2, respectively.

DEFINITION. $P\{l_1 - l_2; l'_1 - l'_2\}$ is said to be closed if it is bounded; otherwise it is said to be nonclosed.

Without loss of generality, we assume that $M' \geqq M$. We also suppose that M and M' are not equal to 0, because $P\{l_1 - l_2; l'_1 - l'_2\}$ is always nonclosed if $M = M' = 0$.

LEMMA 3.3. *If $P\{l_1 - l_2; l'_1 - l'_2\}$ is closed, then either l'_1 crosses l_1 and l_2 (when $M = M'$, l'_2 and l_2 cross at point $(0, M)$), or l'_2 crosses l_1 and l_2 (when $M = M'$, l_1 and l'_1 cross at point $(0, -M)$). l'_1 and l'_2 cannot cross l_1 or l_2 at the same time.*

THEOREM 3.4. *$P\{l_1 - l_2; l'_1 - l'_2\}$ is closed if and only if*

$$[\rho, \sigma] \cap [\rho', \sigma'] = 0.$$

Proof. (1) *Proof of sufficiency of the condition by constructive method.* If $[\rho, \sigma] \cap [\rho', \sigma'] = 0$, then either $\rho' > \sigma$ or $\rho > \sigma'$.

Suppose $\rho' > \sigma$; then l'_1 crosses l_1 and l_2. Solving systems of linear equations (3.8), (3.3) and (3.8), (3.4) we obtain two intersection points:

$$\begin{cases} \lambda = (M' - M)/(\rho' - \rho) \geqq 0, \\ \mu = (\rho M' - \rho' M)/(\rho' - \rho), \end{cases} \quad \text{and} \quad \begin{cases} \lambda = (M' + M)/(\rho' - \rho) \geqq 0, \\ \mu = (\sigma M' + \rho' M)/(\rho' - \sigma). \end{cases} \tag{3.10}$$

If $\rho > \sigma'$, then l_2' crosses l_1 and l_2. Solving systems (3.9), (3.3) and (3.9), (3.4), we obtain two intersection points

$$(3.11)\quad \begin{cases} \lambda = (M' + M)/(\rho - \sigma') > 0, \\ \mu = (\rho' M + \rho M')/(\rho - \sigma'), \end{cases} \quad \text{and} \quad \begin{cases} \lambda = (M' - M)/(\sigma - \sigma') \geqq 0, \\ \mu = (\sigma M' - \sigma' M)/(\sigma - \sigma'). \end{cases}$$

We thus obtain a quadrilateral, the vertices of which are the above-computed points and $(0, M)$ and $(0, -M)$. This quadrilateral and its symmetric region about the origin contain all the real roots of

$$\det A_1 = 0, \qquad \det A_2 = 0.$$

(2) *Proof of necessity of the condition by reduction to absurdity*. Assume that $[\rho, \sigma] \cap [\rho', \sigma'] \neq 0$. Without loss of generality, we suppose that $\rho' \in [\rho, \sigma]$. It divides into two cases:

(i) $\rho' = \rho \Rightarrow l_1' // l_1$ or $\rho' = \sigma \Rightarrow l_1' // l_2$. So l_1' cannot cross l_2; at the same time, l_2' cannot cross l_1.

(ii) $\rho' \in (\rho, \sigma)$; then l_1' cannot cross l_2, and l_2' cannot cross l_1.

By Lemma 3.3, either (i) or (ii) shows that $P\{l_1 - l_2; l_1' - l_2'\}$ is nonclosed, which is a contradiction. □

An an example, let us determine the range of the real roots of the following model problem:

$$(3.12)\quad \det \begin{pmatrix} \lambda - \mu & 1 \\ 1 & \lambda - \mu \end{pmatrix} = 0,$$

$$(3.13)\quad \det \begin{pmatrix} \lambda + \mu & 1 \\ 1 & \lambda + \mu \end{pmatrix} = 0.$$

For (3.12), we have from (3.2)

$$\rho = 1, \quad \sigma = 1, \quad M = 1.$$

Consequently,

$$(3.14)\quad l_1: \quad \mu = \lambda - 1,$$

$$(3.15)\quad l_2: \quad \mu = \lambda + 1.$$

And for (3.13), we have

$$\rho' = -1, \quad \sigma' = -1, \quad M' = 1,$$

$$(3.16)\quad l_1': \quad \mu = -\lambda - 1,$$

$$(3.17)\quad l_2': \quad \mu = -\lambda + 1.$$

Since $1 = \rho > \sigma' = -1$, we solve (by Theorem 3.4) (3.14), (3.17) and (3.15), (3.17), to obtain two points $(\lambda, \mu) = (1, 0)$ and $(0, 1)$.

Adding two points $(0, M) = (0, 1)$ and $(0, -M) = (0, -1)$, we have a triangle. This is a degenerate case. Therefore, all the real roots of (3.12) and (3.13) are located in a rhombus, as shown in Fig. 3.2.

By simple computation, the four exact solutions of the problem (3.12), (3.13) are just four vertices of this rhombus. This shows that, in general, the theorem cannot be improved.

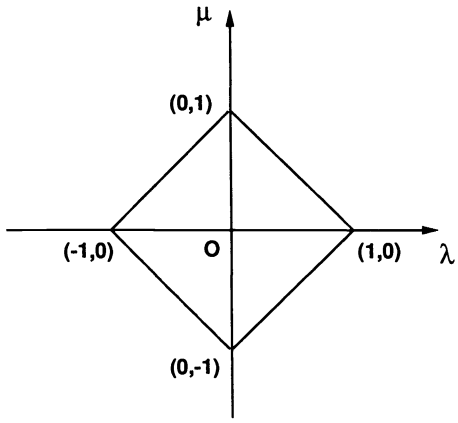


FIG. 3.2

We should point out that, even if $P\{l_1 - l_2; l_1' - l_2'\}$ is nonclosed, we cannot assume that the solutions of the problem run to infinity. This shows the limitation of Theorem 3.4.

4. Numerical results. To test the bisection method described above, a program for the problem (1.1), (1.2) has been written in FORTRAN 77 on Honeywell DPS 8. The program is subdivided into three subroutines:

SUBROUTINE (i). Compute the number of sign changes of a given Sturm sequence at two vertices of a segment.

SUBROUTINE (ii). Check whether eigencurves of $f_1(\lambda, \mu) = 0$ or $f_2(\lambda, \mu) = 0$ pass through the rectangle $ABCD$, where $ABCD = \{a, b; c, d\}$ denotes the rectangle formed by $\lambda = a$, $\lambda = b$, $\mu = c$ and $\mu = d$.

SUBROUTINE (iii). Compute all the roots of the problem in the given rectangle.

Example 1. First consider a model problem

$$\begin{pmatrix} \lambda + \mu & 1 \\ 1 & \lambda + \mu \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = 0, \tag{4.1}$$

$$\begin{pmatrix} \lambda - \mu & 1 \\ 1 & \lambda - \mu \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = 0.$$

The exact solutions are $(\lambda, \mu) = (0, \pm 1), (\pm 1, 0)$.

Now given $\varepsilon = 10^{-6}$, $ABCD = \{0.5, 2.0; -1.0, 0.5\}$. Using the two-dimensional bisection algorithm, we obtain $\lambda = 0.999\ 999\ 9E + 00$ and $\mu = 0.119\ 209\ 3E - 06$.

Adopting the root range $\{-1, 1; -1, 1\}$ given in § 3, we solve for all the roots:

$$(\lambda_1, \mu_1) \approx (-0.100\ 000E + 01, 0.476\ 837E - 06),$$

$$(\lambda_2, \mu_2) \approx (-0.476\ 837E - 06, 0.100\ 000E + 01),$$

$$(\lambda_3, \mu_3) \approx (0.100\ 000E + 01, -0.476\ 837E - 06),$$

$$(\lambda_4, \mu_4) \approx (-0.476\ 837E - 06, -0.100\ 000E + 01).$$

Example 2 (see [5] and [7]). Consider the two-parameter Sturm–Liouville problem

$$\begin{aligned}
&(1 + x + x^2)y_1'' + (1 + \mu x + x^2)y_1 = 0, \quad \text{on } (-1, 0),\\
&(1 + x + x^2)y_2'' + (1 + \mu x + x^2)y_2 = 0, \quad \text{on } (0, 1),\\
&y_1(-1) = y_1(0),\\
&y_2(0) = y_2(1).
\end{aligned} \tag{4.2}$$

It is known that this problem has the eigenpair $(\lambda, \mu) \approx (12.135, 9.604)$.

We introduce $p(x) = (1 + x + x^2)$, $q(x) = 1/p(x)$, $s(x) = x/p(x)$, $r(x) = x^2/p(x)$; then (4.2) becomes

$$\begin{aligned}
&y_1'' + (\lambda q(x) + \mu s(x) + r(x))y_1 = 0, \quad \text{on } (-1, 0)\\
&y_2'' + (\lambda q(x) + \mu s(x) + r(x))y_2 = 0, \quad \text{on } (0, 1)\\
&y_1(-1) = y_1(0),\\
&y_2(0) = y_2(1).
\end{aligned} \tag{4.3}$$

Taking step size $h = 1/100$ and grid points $x_{-i} = -1 + i*h$ in $[-1, 0]$, $x_i = i*h$ in $[0, 1]$ $(i = 1, 2, \cdots, 99)$, and replacing y'' by the second central difference, we obtain a two-parameter algebraic eigenvalue problem:

$$\begin{aligned}
&[1, -2 + h^2(\lambda q_{-i} + \mu s_{-i} + r_{-i}), 1]_{i=1}^{n} y_1 = 0, \quad \text{in } [-1, 0],\\
&[1, -2 + h_2(\lambda q_i + \mu s_i + r_i), 1]_{i=1}^{n} y_2 = 0, \quad \text{in } [0, 1],\\
&y_{1,n} = -y_{2,1}
\end{aligned} \tag{4.4}$$

where $y_{1,n}$ and $y_{2,1}$ are the first and last components of y_1 and y_2, respectively, $q_{-i} = q(x_{-i})$, $s_{-i} = s(x_{-i})$, $r_{-i} = r(x_{-i})$, $q_i = q(x_i)$, $s_i = s(x_i)$, $r_i = r(x_i)$.

For $\varepsilon = 10^{-4}$ and $ABCD = \{11, 14; 8, 12\}$, we get an eigenvalue pair $(\lambda, \mu) = (12.1351, 9.6037)$.

Now let $\varepsilon = 10^{-6}$, $ABCD = \{-100, 100; -100, 100\}$. We have

$$\begin{array}{cc}
\lambda & \mu\\
(0.364\,987E + 02, & 0.683\,313E + 02),\\
(0.121\,250E + 02, & 0.959\,622E + 01),\\
(0.503\,453E + 02, & 0.362\,203E + 02),\\
(0.235\,932E + 02, & -0.167\,225E + 02),\\
(0.417\,165E + 02, & -0.628\,952E + 02),\\
(0.691\,972E + 02, & -0.623\,948E + 01),\\
(0.963\,565E + 02, & -0.652\,450E + 02).
\end{array}$$

The corresponding eigenvectors $y_{1,i}$, $y_{2,i}$ $(i = 1, 2, \cdots, 7)$ can be obtained by the inverse iteration. This problem will be discussed in detail in a future paper. Here we only use the double precision routine DEVFSB in IMSL to compute eigenvectors and let $\|y_{1,i}\|_\infty = \|y_{2,i}\|_\infty = 1$ $(i = 1, 2, \cdots, 7)$; then the maximum residual of all seven eigenpairs is

$$\max_{1 \le i \le 7} (\|(T_1 + \lambda_i B_1 + \mu_i C_1)y_{1,i}\|_\infty, \|(T_2 + \lambda_i B_2 + \mu_i C_2)y_{2,i}\|_\infty) = 1.838 \times 10^{-4}.$$

Example 3. Let us consider the problem (1.5). Take grid points $x_i = i*h (i = 1, 2, \cdots, 100)$ for the first equation, and grid points $y_j = -1 + j*h$ $(j = 1, 2, \cdots, 201)$ for the second equation, where $h = 1/101$, and use central differences to approximate

TABLE 1
Results of Example 3.

m \ n	0	1	2	3
0	1.8782	−6.199E-4	−8.3373	−17.6674
	25.5327	30.9153	52.6877	74.3302
1	17.8523	17.6674		
	73.6537	74.3426		

X'' and Y''. The order of the two discrete equations obtained are not the same. We adopt $\varepsilon = 10^{-4}$, and calculate the eigenvalues in the rectangle $ABCD = \{-100, 100; -100, 100,\}$. As a result, six eigenpairs (λ_i, μ_i), $i = 1, 2, \cdots, 6$ are obtained and listed in Table 1, in which m and n stand for the oscillation numbers of the eigenvectors X and Y, respectively, corresponding to the numbers of zero points of the eigenfunctions $X(x)$ and $Y(y)$ in the open intervals $(0, 1)$ and $(-1, 1)$.

The corresponding eigenvectors X_i, Y_i $(i = 1, 2, \cdots, 6)$ are computed as in Example 2. The maximum residual of all six eigenpairs is

$$\max_{1 \le i \le 6} (\|(T_1 + \lambda_i B_1 + \mu_i C_1)X_i\|_\infty, \|(T_2 + \lambda_i B_2 + \mu_i C_2)Y_i\|_\infty) = 1.258 \times 10^{-5}.$$

The discretization error of replacing (1.3) by (1.1), (1.2) is discussed in [3].

Acknowledgments. The author is indebted to Professor H. F. Weinberger and the referee for their helpful comments and suggestions.

REFERENCES

[1] F. M. ARSCOTT AND A. DARAI, *Curvilinear co-ordinate systems in which the Helmholtz equation separates*, IMA J. Appl. Math., 27 (1981), pp. 33–70.
[2] P. B. BAILEY, *The automatic solution of two-parameter Sturm–Liouville eigenvalue problems in ordinary differential equations*, Appl. Math. Comp., 8 (1981), pp. 251–259.
[3] P. A. BINDING, P. J. BROWNE, AND XINGZHI JI, *Numerical solution of a class of double eigenvalue problems*, in preparation.
[4] E. K. BLUM AND G. J. REID, *On the numerical solution of three-dimensional boundary value problems by separation of variables*, SIAM J. Numer. Anal., 25 (1988), pp. 75–90.
[5] E. K. BLUM AND A. F. CHANG, *A numerical method for the solution of the double eigenvalue problem*, J. Inst. Maths. Appl., 22 (1978), pp. 29–42.
[6] E. K. BLUM AND P. B. GELTNER, *Numerical solution of eigentuple-eigenvector problems in Hilbert spaces by a gradient method*, Numer. Math., 31 (1978), pp. 231–246.
[7] L. FOX, L. HAYES, AND D. F. MAYERS, *The double eigenvalue problem*, in Topics in Numerical Analysis, Proc. Royal Irish Academy Conference on Numerical Analysis, J. Miller, ed., Academic Press, New York, 1973, pp. 93–112.
[8] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, The Johns Hopkins University Press, Baltimore, MD, 1983.
[9] J. R. KUTTLER AND V. G. SIGILLITO, *Eigenvalues of the Laplacian in two dimensions*, SIAM Rev., 26 (1984), pp. 163–193.
[10] P. M. MORSE AND H. FESHBACH, *Methods of Theoretical Physics*, McGraw-Hill, New York, 1953.
[11] M. PEASE, *Methods of Matrix Algebra*, Academic Press, New York, 1965.
[12] B. D. SLEEMAN, *Multiparameter periodic differential equations*, in Ordinary and Partial Differential Equations, W. N. Everitt, ed., Lecture Notes in Mathematics 827, pp. 229–250, Springer-Verlag, Berlin, 1978.
[13] H. F. WEINBERGER, *A First Course in Partial Differential Equations*, Blaisdell, New York, 1965.
[14] J. H. WILKINSON AND C. REINSCH, *Handbook for Automatic Computation*, Vol. II, Springer-Verlag, New York, 1971.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1094–1122, October 1992

007

ESTIMATING THE LARGEST EIGENVALUE BY THE POWER AND LANCZOS ALGORITHMS WITH A RANDOM START*

J. KUCZYŃSKI† AND H. WOŹNIAKOWSKI‡

Abstract. This paper addresses the problem of computing an approximation to the largest eigenvalue of an $n \times n$ large symmetric positive definite matrix with relative error at most ε. Only algorithms that use Krylov information $[b, Ab, \cdots, A^k b]$ consisting of k matrix-vector multiplications for some unit vector b are considered. If the vector b is chosen deterministically, then the problem cannot be solved no matter how many matrix-vector multiplications are performed and what algorithm is used. If, however, the vector b is chosen randomly with respect to the uniform distribution over the unit sphere, then the problem can be solved on the average and probabilistically. More precisely, for a randomly chosen vector b, the power and Lanczos algorithms are studied. For the power algorithm (method), sharp bounds on the average relative error and on the probabilistic relative failure are proven. For the Lanczos algorithm only upper bounds are presented. In particular, $\ln(n)/k$ characterizes the average relative error of the power algorithm, whereas $O((\ln(n)/k)^2)$ is an upper bound on the average relative error of the Lanczos algorithm. In the probabilistic case, the algorithm is characterized by its probabilistic relative failure, which is defined as the measure of the set of vectors b for which the algorithm fails. It is shown that the probabilistic relative failure goes to zero roughly as $\sqrt{n}(1-\varepsilon)^k$ for the power algorithm and at most as $\sqrt{n}e^{-(2k-1)\sqrt{\varepsilon}}$ for the Lanczos algorithm. These bounds are for a worst case distribution of eigenvalues which may depend on k. The behavior in the average and probabilistic cases of the two algorithms for a fixed matrix A is also studied as the number of matrix-vector multiplications k increases. The bounds for the power algorithm depend then on the ratio of the two largest eigenvalues and their multiplicities. The bounds for the Lanczos algorithm depend on the ratio between the difference of the two largest eigenvalues and the difference of the largest and the smallest eigenvalues.

Key words. largest eigenvalue, power and Lanczos algorithms, random start

AMS(MOS) subject classification. 65

1. Introduction.

In this paper we address the problem of approximating the largest eigenvalue λ_1 of an $n \times n$ large symmetric positive definite matrix A. We wish to compute an approximation ξ with relative error at most ε, i.e., $|\lambda_1 - \xi| \leqq \varepsilon\lambda_1$. Typically the matrix A is sparse and it is reasonable to use Krylov information consisting of k matrix-vector multiplications $[b, Ab, \cdots, A^k b]$ for some unit vector b. Examples of algorithms for this problem are the power algorithm, which has rather limited practical value, and the far superior Lanczos algorithm. It is well known that convergence of both algorithms depends on the distribution of eigenvalues and on the angle between the vector b and the eigenvector η_1 corresponding to the largest eigenvalue (see § 2 for references). In particular, if the vector b is chosen deterministically and independently on the matrix A, then it may happen that b is orthogonal to η_1. In such a case the two algorithms fail to approximate the largest eigenvalue. It is easy to extend this negative result by showing that as long as Krylov information is used with a deterministic unit vector b, then there exists no algorithm that can approximate the largest eigenvalue for all symmetric positive matrices (see § 2 for details). Also if Krylov information is replaced by any k matrix-vector multiplications, then the problem cannot be solved for all symmetric positive matrices as long as $k \leqq n-1$ since all the vectors might be orthogonal to η_1 (see Remark 7.1).

* Received by the editors August 10, 1989; accepted for publication (in revised form) January 4, 1991.

† Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland.

‡ Department of Computer Science, Columbia University, New York, New York (henryk@ground.cs.columbia.edu) and Institute of Informatics, University of Warsaw, Warsaw, Poland. This author's research was supported in part by National Science Foundation grant DCR-86-03674.

On the other hand, a closer look at the analysis of convergence of the power or Lanczos algorithm yields the impression that it is very unlikely that the position of the vector b will be so unfortunate and that it should not really happen with a randomly chosen vector b. This is exactly the point of departure of our paper. We assume that the vector b is chosen randomly with uniform distribution over the unit sphere of n-dimensional space. Then we define the average relative error of an algorithm as the expected relative error while integrating over the vectors b of the unit sphere. We also analyze the probabilistic relative failure, which is defined as the measure of the set of vectors b for which the algorithm fails to approximate the largest eigenvalue with relative error at most ε.

For the average case we find sharp bounds on the relative error of the power algorithm (see Theorem 3.1). Namely, no matter what the distribution of eigenvalues of the matrix A, the relative error is bounded from above, for large n, by roughly $0.564 \ln(n)/(k-1)$. This bound is sharp in the sense that for each k there exists a symmetric positive definite matrix A for which the relative error is at least roughly $0.5 \ln(n)/(k-1)$. Hence, the relative error of the power algorithm tends to zero as k goes to $+\infty$, although the speed of convergence is quite slow. Observe that the dimension n of the problem affects the speed of convergence only logarithmically.

For the Lanczos algorithm, we are only able to present upper bounds on its average relative error (see Theorem 3.2). We show that independent of the distribution of eigenvalues of the matrix A, the relative error is bounded by $2.575(\ln(n)/(k-1))^2$ for $k \in [4, n-1]$, and that the relative error is zero if k is no less than the total number of distinct eigenvalues. To check the quality of this upper bound, we performed many numerical tests. They are reported in § 6. Numerical tests for the matrix whose eigenvalues are shifted zeros of the Chebyshev polynomial of the first kind of degree n seem to indicate that the relative error of the Lanczos algorithm behaves like k^{-2}. If so, then the factor $\ln^2(n)$ in our upper bound is an overestimate.

Comparing the two algorithms, we see, not surprisingly, the superiority of the Lanczos algorithm. The ratio of steps of the power and Lanczos algorithms needed to achieve error at most ε is roughly at least equal to $0.35\varepsilon^{-1/2}$. Thus the smaller the ε, the more superior the Lanczos algorithm.

So far we have discussed the bounds for a worst case distribution of eigenvalues. We also study the behavior of the average relative errors for a fixed matrix A and increasing k. For the power algorithm, we obtain formulas for the rate of convergence, which depends on the ratio ρ of the two largest eigenvalues and on their multiplicities (see Theorem 3.1(c)). In particular, the best rate is obtained if the multiplicity p of the largest eigenvalue is at least 3 and then it is equal to $\rho^{2(k-1)}$. For $p = 1$, the rate is ρ^{k-1}. Observe that for a deterministic vector b which is not orthogonal to the eigenvector η_1, the rate is $\rho^{2(k-1)}$. In § 3 we explain why for $p \leqq 2$ the rate decreases in the average case. For the Lanczos algorithm we obtain only an upper bound on the ratio, which depends on the difference of the two largest eigenvalues over the difference of the largest and the smallest eigenvalues (see Theorem 3.2(b)).

We now turn to the probabilistic case. As before, we find sharp bounds for the probabilistic relative failure of the power algorithm that are independent of the distribution of eigenvalues (see Theorem 4.1). The failure goes to zero roughly as $\sqrt{n}(1-\varepsilon)^k$. Note that now the dimension n affects the failure much more substantially than in the average case. Although the failure goes to zero exponentially, for small ε the speed of convergence is quite slow.

The failure of the Lanczos algorithm is zero if k is no less than the total number of distinct eigenvalues, and is bounded by roughly $1.648\sqrt{n}e^{-\sqrt{\varepsilon}(2k-1)}$ for any k (see Theorem

4.2). Hence, we have the same dependence on the dimension n, but the dependence on ε is much improved.

If we compare the number of steps needed to obtain a failure of at most δ, then the ratio between the steps of the power and Lanczos algorithms is independent of δ and is roughly at least $2\varepsilon^{-1/2}$. Thus, in both the average and probabilistic cases the ratio is proportional to $\varepsilon^{-1/2}$.

We also study the probabilistic relative failure for a fixed matrix A and increasing k. The rate of convergence of the power algorithm depends on multiplicity p and is given by $\rho^{p(k-1)}$. Hence, the rate increases with multiplicity. On the other hand, the asymptotic constant for large p and small ε is huge (see Theorem 4.1(c)). As before, for the Lanczos algorithm we only obtain an upper bound on the ratio, which depends on the two largest and the smallest eigenvalues.

The proofs of theorems from §§ 3 and 4 are presented in § 5. It turns out that the proof technique for the power algorithm can be applied for the Lanczos algorithm with the use of Chebyshev polynomials of the first kind for the average case and of the second kind for the probabilistic case. We think that getting a sharp lower bound on the error or failure of the Lanczos algorithm will require a more sophisticated analysis.

In Remark 7.3 we briefly mention a modified power algorithm which was analyzed in the probabilistic case by Dixon [3]. We extend his analysis to the average case and conclude that the power algorithm is better.

In this paper, we do not address the termination criterion. Termination is inherently difficult due to the negative result for deterministic vectors b. Furthermore, for the Lanczos algorithm a "misconvergence phenomenon" takes place as indicated in Parlett, Simon, and Stringer [15]. We also experienced this in our tests as reported in § 6. Nevertheless we hope that average and probabilistic bounds can be useful in deriving a reliable termination criterion for which we can prove how the algorithm works on the average or probabilistically. It should be added that it is often the case in engineering that the quality of the computed approximation ξ can be verified for moderate n by performing triangular factorization of $\xi_1 I - A$ and checking that no negative pivot occurs. Here, ξ_1 is a computed upper bound on the largest eigenvalue λ_1. For example, if we believe that ξ is an approximation to λ_1 with relative error at most ε, then $\lambda_1 \leqq \xi/(1-\varepsilon)$, and we can set $\xi_1 = \xi/(1-\varepsilon)$.

Of course, approximating the largest eigenvalue is only one of many interesting eigenvalue problems. To list a few, we mention approximating the mth largest eigenvalue, the smallest eigenvalue, or corresponding eigenvectors. Since the negative result for deterministic vectors b extends also to these new problems, it is quite natural to use random vectors and, hopefully, to get positive results on the average or probabilistically. In particular, it seems to us that a similar proof technique can work for approximating the smallest eigenvalue and the condition number of a symmetric positive definite matrix. We hope to report this in the near future.

Finally, we add a remark on replacing the standard relative error criterion by one depending on the eigenvalue spread $\lambda_1 - \lambda_n$, where λ_1 is the rightmost and λ_n is the leftmost eigenvalue (see Parlett [14]).

The gap ratio is defined (see Parlett [13]) as $(\lambda_1 - \lambda_2)/(\lambda_2 - \lambda_n)$, where λ_2 is the second largest eigenvalue, and it occurs in the analysis of the rate of convergence of Krylov subspace methods to λ_1. The natural extension for our problem is to seek an approximation ξ to λ_1 that satisfies

$$|\lambda_1 - \xi| \leqq \varepsilon(\lambda_1 - \lambda_n).$$

Since this error criterion for the Lanczos algorithm is shift invariant, the bounds presented

in this paper for standard relative error also hold for the error relative to the eigenvalue spread $\lambda_1 - \lambda_n$, as given above. The significant advantage of using the spread is that our results apply to all symmetric matrices, positive definite or not. In contrast to the Lanczos algorithm, the error of the power algorithm relative to the eigenvalue spread is not shift invariant and there is no sense in modifying the standard relative error criterion. Details are given in Remark 7.5.

2. Definition of the problem. Let A be an $n \times n$ large symmetric positive definite matrix. Let $\lambda_i = \lambda_i(A)$ denote the eigenvalues of the matrix A, $\lambda_1(A) \geqq \lambda_2(A) \geqq \cdots \geqq \lambda_n(A) > 0$. We want to compute an approximation to the largest eigenvalue $\lambda_1(A)$. More precisely, for a given (presumably small) positive number ε, we want to compute a number $\xi = \xi(A)$ such that the relative error between $\lambda_1(A)$ and $\xi(A)$ does not exceed ε:

$$\left|\frac{\xi(A) - \lambda_1(A)}{\lambda_1(A)}\right| \leqq \varepsilon. \tag{1}$$

Obviously, if $\varepsilon \geqq 1$, $\xi(A) = 0$ satisfies (1). To avoid this trivial case, we assume that $\varepsilon \in [0, 1)$.

If n is large, say, of order 10^{+3} or 10^{+4}, then it is prohibitively expensive to use well-known algorithms such as QR or QL. Instead, it is reasonable to assume that the information about the matrix A is supplied by a subroutine that computes Az for any vector z. If A is sparse, which often is the case, the time and storage needed to perform the matrix-vector multiplication Az is proportional to n.

We therefore assume that *Krylov* information consisting of k matrix-vector multiplications, $k \geqq 1$,

$$N_k(A, b) = [b, Ab, \cdots, A^k b], \tag{2}$$

is used to compute the approximation $\xi(A)$. That is, $\xi(A) = \phi_k(N_k(A, b))$ for some mapping $\phi_k : \mathbf{R}^{n(k+1)} \to \mathbf{R}$. Here, b is a nonzero vector which, without loss of generality, may be normalized such that $\|b\| = 1$, where $\|\cdot\|$ stands for the Euclidean norm of vectors.

Krylov information can be written as $[z_1, z_2, \cdots, z_{k+1}]$ with $z_1 = b$ and $z_i = Az_{i-1}$. This shows that it can be computed using k matrix-vector multiplications.

Examples of algorithms that use Krylov information include the power and (simple) Lanczos algorithms. For the power algorithm ξ^{pow} we have

$$\xi(A) = \xi^{\text{pow}}(A, b, k) = \frac{(Ax, x)}{(x, x)} \quad \text{with } x = A^{k-1}b = z_k, \tag{3}$$

whereas for the Lanczos algorithm ξ^{Lan} we have

$$\xi(A) = \xi^{\text{Lan}}(A, b, k) = \max\left\{\frac{(Ax, x)}{(x, x)} : 0 \neq x \in \text{span}\,(b, \cdots, A^{k-1}b)\right\}. \tag{4}$$

The analysis of convergence of the power algorithm is straightforward and may be found in most books on numerical analysis. The analysis of convergence of the Lanczos algorithm is more complex and some of it may be found in, e.g., Wilkinson [20], Kaniel [6], Paige [11], [12], Kahan and Parlett [5], Scott [17], Parlett [13], and Saad [16].

In both cases, convergence depends on distributions of eigenvalues of the matrix A and on the vector b. In particular, if b is orthogonal to the eigenvector η_1, $A\eta_1 = \lambda_1\eta_1$, then both algorithms fail to converge to λ_1. This means that (1) cannot always be satisfied.

It is then natural to ask if there exists an algorithm using Krylov information (with sufficiently large k) for which (1) is satisfied for some ε and for *all* symmetric and positive definite matrices. It is easy to verify that, unfortunately, this is *not* the case.

We now present a simple argument why this is so (see also Remark 7.1, where further discussion may be found). For arbitrary A, b, and k, let

$$d = d(A, b, k) = \dim \operatorname{span}(b, Ab, \cdots, A^{k-1}b).$$

Clearly, $1 \leqq d \leqq \min(k, n)$ and both bounds can be achieved. Let $\xi(A) = \phi_k(N_k(A, b))$, where ϕ_k is an arbitrary mapping.

Assume that $d \leqq n - 1$. Then there exists a matrix $\bar{A}$, $\bar{A} = \bar{A}^T > 0$, such that $\xi(\bar{A}) = \xi(A)$ and

$$\left| \frac{\xi(\bar{A}) - \lambda_1(\bar{A})}{\lambda_1(\bar{A})} \right| > \varepsilon, \tag{5}$$

that is, $\xi(\bar{A})$ does *not* satisfy (1) for the matrix $\bar{A}$. The matrix $\bar{A}$ is of the form $\bar{A} = A + \alpha uu^T$, where α is a positive constant and u is a nonzero vector orthogonal to $b, Ab, \cdots, A^{k-1}b$. Such a vector exists since $d \leqq n - 1$. By induction we get

$$\bar{A}^j b = A^j b \quad \text{for } j = 0, 1, \cdots, k.$$

Thus $N_k(\bar{A}, b) = N_k(A, b)$ and therefore $\xi(\bar{A}) = \xi(A)$. Observe that the trace of $\bar{A}$ is given by

$$\operatorname{trace}(\bar{A}) = \operatorname{trace}(A) + \alpha \|u\|^2$$

and it goes to infinity as $\alpha \to +\infty$. Therefore the largest eigenvalue $\lambda_1(\bar{A})$ goes to infinity as well. We thus have

$$\left| \frac{\xi(\bar{A}) - \lambda_1(\bar{A})}{\lambda_1(\bar{A})} \right| = \left| \frac{\xi(A) - \lambda_1(\bar{A})}{\lambda_1(\bar{A})} \right| \to 1 \quad \text{as } \alpha \to +\infty.$$

Hence, there exists a positive α for which (5) holds, as claimed.

Observe that for large α, the largest eigenvalue $\lambda_1(\bar{A})$ of $\bar{A}$ is close to α and the eigenvector corresponding to $\lambda_1(\bar{A})$ is close to u. The vector u is orthogonal to all but the last vectors of Krylov information. Thus $N_k(\bar{A}, b)$ contains almost no information on the vector u and therefore no matter how ϕ_k is chosen, $\xi(\bar{A}) = \phi_k(N_k(\bar{A}, b))$ cannot approximate $\lambda_1(\bar{A})$ with relative error at most ε.

To prove (5), we needed to assume that $d(A, b, k) \leqq n - 1$. Observe that this inequality holds for all A and b as long as $k \leqq n - 1$. Thus, if we perform fewer than n matrix-vector multiplications, there always exists a symmetric and positive definite matrix $\bar{A}$ which shares the same information as A and for which it is impossible to approximate its largest eigenvalue with relative error at most ε. We stress that ε need not be small. The only assumption is $\varepsilon < 1$.

Clearly, if for any A we have $d(A, b, k) = n$, then it is possible to satisfy (1). Indeed, the vectors $b, Ab, \cdots, A^{k-1}b$ span the whole space and the matrix A can be uniquely recovered from the computed Krylov information $N_k(A, b)$. Knowing A, we have, at least conceptually, enough information to recover the largest eigenvalue $\lambda_1(A)$ even exactly.

Can we thus guarantee that $d(A, b, k) = n$ for some $k \geqq n$? Clearly, not always. For any vector b, there exists a matrix $A = A^T > 0$ such that b is its eigenvector, say, $Ab = \alpha b$. Then $d(A, b, k) \equiv 1$ for all k, and no matter how many matrix-vector multiplications are performed, (1) cannot be satisfied for some symmetric and positive definite

matrices. It can also happen that $d(A, b, p) = d(A, b, p+1)$ for some p, where $1 \leqq p \leqq n-1$. Then $d(A, b, k) = d(A, b, p)$ for all $k \geqq p$, and still (1) cannot always hold. We have $d(A, b, k) = n$ if and only if $k \geqq n$ and vectors $b, Ab, \cdots, A^{n-1}b$ are linearly independent.

Observe that $b, Ab, \cdots, A^{n-1}b$ are linearly independent if and only if all the eigenvalues of A are distinct and the projections of the vector b onto the eigenvectors η_i of the matrix A are nonzero, that is, $(b, \eta_i) \neq 0$ for $i = 1, 2, \cdots, n$. This property is guaranteed if, for example, A is unreduced tridiagonal and $b = [1, 0, \cdots, 0]$.

Although it is impossible to guarantee that $(b, \eta_i) \neq 0$ for all $i \in [1, n]$, it is intuitively clear that $(b, \eta_i) \neq 0$ should hold for "almost all" vectors b. This is definitely the case if the vector b is chosen randomly, say, with uniform distribution μ on the n-dimensional sphere of radius 1. The reader may consult Knuth [7, p. 130], where it is explained how such a vector can be generated computationally. Then $(b, \eta_i) = 0$ holds with probability 0 and $d(A, b, k) = k$ with probability 1 if and only if A has at least k distinct eigenvalues.

The last fact follows by noting that $[b, Ab, \cdots, A^{k-1}b]$ in the basis of eigenvectors of A is equal to the product of the diagonal matrix D whose entries are components of b, and the Vandermonde matrix V whose entries are powers of eigenvalues of A. The matrix D is nonsingular with probability 1 whereas the matrix V has rank k if and only if A has at least k distinct eigenvalues.

This discussion suggests that although (1) cannot be satisfied for *all* symmetric and positive definite matrices with a deterministically chosen vector b, there is hope that this problem can be solved by introducing a random initial vector b of Krylov information, that is, for *all* symmetric and positive definite matrices we wish to have the average relative error with respect to vectors b to be at most ε. Or we may wish to solve the problem with high probability, i.e., for vectors b that form a set of measure close to 1.

We now formalize this idea. Let μ be a distribution of the initial vectors b. How should we select μ for the power and Lanczos algorithms? Observe that for both of them we have

$$\xi(A, b, k) = \xi(A, \alpha b, k) \quad \forall \alpha \neq 0,$$

$$\xi(A, b, k) = \xi(Q^T A Q, Q^T b, k) \quad \forall Q \text{ orthogonal}.$$

It is then natural to choose μ which shares these two properties, that is, μ should be concentrated on the unit sphere of $\mathbf{R}^n$ and should be orthogonally invariant. Therefore, we assume in this paper that μ is a *uniform* distribution over the unit sphere of $\mathbf{R}^n$, $\mu(\{b \in \mathbf{R}^n : \|b\| = 1\}) = 1$.

For any symmetric and positive definite matrix A, we select a *random* vector b according to the distribution μ. Then we compute Krylov information $N_k(A, b)$ and the approximation $\xi(A, b, k)$ of the largest eigenvalue $\lambda_1(A)$ by the power or Lanczos algorithm (3) or (4). Then

$$e^{\text{avg}}(\xi, A, k) = \int_{\|b\|=1} \left| \frac{\xi(A, b, k) - \lambda_1(A)}{\lambda_1(A)} \right| \mu(db) \tag{6}$$

denotes the average relative error. Let

$$f^{\text{prob}}(\xi, A, k, \varepsilon) = \mu\left\{ b \in \mathbf{R}^n : \|b\| = 1, \left| \frac{\xi(A, b, k) - \lambda_1(A)}{\lambda_1(A)} \right| > \varepsilon \right\} \tag{7}$$

denote the probability that the algorithm fails to approximate the largest eigenvalue with relative error at most ε. We call (7) the probabilistic relative failure of ξ.

In Remark 7.2 we show that (6) and (7) remain the same if we integrate over the unit ball B_n with respect to normalized Lebesgue measure.

3. Average case. In this section, we present bounds on the average relative error (6) for both the power and Lanczos algorithms. Proofs are given in § 5. To simplify some estimates, we assume that $n \geqq 8$. We begin with the power algorithm.

THEOREM 3.1. *Let ξ^{pow} be the power algorithm defined by* (3).

(a) *For any symmetric positive definite matrix A and for any $k \geqq 2$, we have*

$$e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k) \leqq \alpha(n) \frac{\ln n}{k-1},$$

where $\pi^{-1/2} \leqq \alpha(n) \leqq 0.871$ and, for large n, $\alpha(n) \simeq \pi^{-1/2} = 0.564 \cdots$.

(b) *For any $k > 1 + \frac{1}{2} \ln (n/\ln n)$, let A be any symmetric matrix with exactly two distinct eigenvalues $\lambda_1 > 0$ and $\lambda_i = \lambda_1(1 - \ln (n/\ln n)/(2(k-1)))$, for $i = 2, 3, \cdots, n$. Then for large n and k,*

$$e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k) \geqq 0.5 \frac{\ln n}{k-1}(1 + o(1)).$$

(c) *For any symmetric positive definite matrix A, let p, $p < n$, and q denote the multiplicities of the two largest eigenvalues λ_1 and λ_{p+1}. Then*

$$\lim_{k \to +\infty} \frac{e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k)}{(\lambda_{p+1}/\lambda_1)^{2(k-1)}} = \frac{q}{p-2}\left(1 - \frac{\lambda_{p+1}}{\lambda_1}\right) \quad \text{for } p \geqq 3,$$

$$\lim_{k \to +\infty} \frac{e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k)}{(k-1)(\lambda_3/\lambda_1)^{2(k-1)}} = q\left(1 - \frac{\lambda_3}{\lambda_1}\right) \ln \frac{\lambda_1}{\lambda_3} \quad \text{for } p = 2,$$

$$\lim_{k \to +\infty} \frac{e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k)}{(\lambda_2/\lambda_1)^{k-1}} = \sqrt{\pi}\, \frac{\Gamma((q+1)/2)}{\Gamma(q/2)}\left(1 - \frac{\lambda_2}{\lambda_1}\right) \quad \text{for } p = 1.$$

Part (a) of Theorem 3.1 states that no matter what the distribution of eigenvalues of A is nor how poorly the dominant eigenvalue is separated from the next largest eigenvalue, the average relative error of the power algorithm is bounded by $0.871 \ln (n)/(k-1)$. For large n, the constant 0.871 can be replaced by roughly 0.564.

Part (b) of Theorem 3.1 states that this upper bound is essentially sharp since for each k there exists a matrix $A = A^T > 0$ with only two distinct eigenvalues for which the average relative error of the power algorithm is at least roughly $0.5 \ln (n)/(k-1)$.

The average relative error of the power algorithm depends only logarithmically on the dimension n. Thus, even for large n, the constant $0.564 \ln (n)$ is quite moderate and the error is a modest multiple of $(k-1)^{-1}$. Of course, $(k-1)^{-1}$ tends to zero slowly, and to guarantee

$$e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k) \leqq \varepsilon \quad \forall A = A^T > 0,$$

we have to perform roughly $k = \lceil 1 + 0.564 \ln (n)/\varepsilon \rceil$ steps. For small ε, such a number of steps cannot be realistically done. As we shall see in Theorem 3.2, the Lanczos algorithm is, not surprisingly, much better and therefore the power algorithm is of limited value in numerical practice.

We now comment on the paper of O'Leary, Stewart, and Vandergraft [10]. They analyzed the power algorithm for fixed eigenvectors $\eta_1, \eta_2, \cdots, \eta_n$ and for a fixed vector b, $\|b\| = 1$. They showed that for a worst case distribution of eigenvalues, the power algorithm takes roughly $k = \ln (\tau)/\varepsilon$ steps to compute an ε-approximation to the largest

eigenvalue. Here $\tau = \tan|\theta|$, where θ is the angle between b and η_1. If all $b_i = (b, \eta_i)$ are more or less equal, then $\tau \simeq \sqrt{n}$ and $k \simeq \ln(n)/\varepsilon$. Hence, also in this case, $\ln(n)/\varepsilon$ exhibits the behavior of the power algorithm.

We turn to Theorem 3.1(c), which explains the asymptotic behavior of the average relative errors of the power algorithm. The rate of convergence depends on the multiplicity p of the largest eigenvalue. We assumed that $p < n$. Note that the case $p = n$ is not interesting since then A is proportional to the identity matrix and one step of the power algorithm recovers exactly the largest eigenvalue.

The worst rate is for $p = 1$ and in this case is proportional to $(\lambda_2/\lambda_1)^{k-1}$. This should be compared with the deterministic case for which the rate is proportional to $(\lambda_2/\lambda_1)^{2(k-1)}$ whenever $b_1 = (b, \eta_1) \neq 0$. More precisely, for any vector b, let $\rho_k(b) = (\lambda_1 - \xi^{\mathrm{pow}}(A, b, k))/\lambda_1$. As before, let $b_i = (b, \eta_i)$. Assuming that $b_1 \neq 0$, we have

$$\rho_k(b) = \left(\frac{\lambda_2}{\lambda_1}\right)^{2(k-1)} \left(\frac{b_2^2 + \cdots + b_{q+1}^2}{b_1^2}\right)\left(1 - \frac{\lambda_2}{\lambda_1}\right) + o\left(\left(\frac{\lambda_2}{\lambda_1}\right)^{2(k-1)}\right),$$

where q is the multiplicity of the second largest eigenvalue.

To explain the difference in the rate of convergence, note that the average value of $\rho_k(b)$ with respect to b cannot be proportional to $(\lambda_2/\lambda_1)^{2(k-1)}$ since

$$\int_{\|b\|=1} \frac{b_2^2 + \cdots + b_{q+1}^2}{b_1^2}\, \mu(db) = +\infty.$$

The complete analysis shows that we lose a factor $(\lambda_2/\lambda_1)^{k-1}$ when integrating $\rho_k(b)$, and therefore the average value of $\rho_k(b)$ is proportional to $(\lambda_2/\lambda_1)^{k-1}$, as claimed in part (c) for $p = 1$.

For $p \geqq 2$, the situation is different since

$$\rho_k(b) = \left(\frac{\lambda_{p+1}}{\lambda_1}\right)^{2(k-1)} \left(\frac{b_{p+1}^2 + \cdots + b_{p+q}^2}{b_1^2 + \cdots + b_p^2}\right)\left(1 - \frac{\lambda^{p+1}}{\lambda_1}\right) + o\left(\left(\frac{\lambda_{p+1}}{\lambda_1}\right)^{2(k-1)}\right)$$

whenever $b_1^2 + \cdots + b_p^2 \neq 0$. For $p \geq 3$, the integral

$$\int_{\|b\|=1} \frac{b_{p+1}^2 + \cdots + b_{p+q}^2}{b_1^2 + \cdots + b_p^2}\, \mu(db) < +\infty,$$

which explains why the rate of convergence is proportional to $(\lambda_{p+1}/\lambda_1)^{2(k-1)}$.

For $p = 2$, the integral above is "barely" infinite and the complete analysis shows that we lose the factor $\ln(\lambda_3/\lambda_1)^{2(k-1)} = 2(k-1)\ln(\lambda_3/\lambda_1)$ when integrating $\rho_k(b)$. As claimed in part (c) for $p = 2$, the rate of convergence is therefore proportional to $(k-1)(\lambda_3/\lambda_2)^{2(k-1)}$.

Theorem 3.1(c) shows that the asymptotic constant depends also on the multiplicity q of the second largest eigenvalue and on the ratio λ_{p+1}/λ_1. The multiplicity q may depend on the dimension n, and it can happen that $q = n - p$. In this case and for λ_{p+1}/λ_1 not too close to 1, the asymptotic constant is huge.

We wish to add that a similar analysis may be performed for a modified power algorithm ξ^{mpow}, where

$$\xi^{\mathrm{mpow}}(A, b, k) = (A^k b, b)^{1/k}, \qquad \|b\| = 1.$$

For the modified power algorithm, $\ln(n)/(k-1)$ is a sharp upper bound on the average relative error, which is roughly 1.8 times worse than the corresponding error bound of the power algorithm. Unlike the power algorithm, $\ln(n)/(k-1)$ is also a sharp upper bound on the asymptotic behavior of the average relative error of the modified power

algorithm. This shows that the power algorithm is superior to the modified power algorithm. Details are presented in Remark 7.3.

We now proceed to the Lanczos algorithm. The analysis of this algorithm is much more complex and we are able to present only upper bounds. We verify some of our estimates by numerical tests, which will be reported here and in more detail in § 6. Obviously

$$e^{\text{avg}}(\xi^{\text{Lan}}, A, k) \leqq e^{\text{avg}}(\xi^{\text{pow}}, A, k) \quad \forall A \text{ and } k. \tag{8}$$

Therefore we can also apply upper estimates of the power algorithm to the Lanczos algorithm. Of course, since the Lanczos algorithm is much more powerful than the power algorithm, we hope to get much better estimates of convergence. This will be confirmed by the following theorem. To simplify some formulas, we assume that $k \geqq 4$ and (as before) that $n \geqq 8$.

THEOREM 3.2. *Let ξ^{Lan} be the Lanczos algorithm defined by* (4).

(a) *For any symmetric positive definite matrix A, let m denote the number of distinct eigenvalues of A. Then for $k \geqq m$,*

$$e^{\text{avg}}(\xi^{\text{Lan}}, A, k) = 0;$$

for $k \in [4, m-1]$,

$$e^{\text{avg}}(\xi^{\text{Lan}}, A, k) \leqq 0.103\left(\frac{\ln\,(n(k-1)^4)}{k-1}\right)^2 \leqq 2.575\left(\frac{\ln n}{k-1}\right)^2.$$

(b) *For any symmetric positive definite matrix A, let p, $p < n$, denote the multiplicity of the largest eigenvalue λ_1, and let λ_{p+1} and λ_n be the second largest and the smallest eigenvalues of A. Then*

$$e^{\text{avg}}(\xi_{\text{Lan}}, A, k) \leqq 2.589\sqrt{n}\left(\frac{1-\sqrt{(\lambda_1-\lambda_{p+1})/(\lambda_1-\lambda_n)}}{1+\sqrt{(\lambda_1-\lambda_{p+1})/(\lambda_1-\lambda_n)}}\right)^{k-1}.$$

Theorem 3.2 states that the Lanczos algorithm converges in m steps, $m \leqq n$, which confirms our intuition that it can fail only on a set of vectors b of measure 0. For k essentially less than n, the average relative error of the Lanczos algorithm is roughly bounded by $0.1(\ln\,(n)/k)^2$. Since $\ln\,(n)/k$ is a sharp estimate of the average relative error of the power algorithm, we see that the Lanczos algorithm is far superior. If we want to guarantee that $e^{\text{avg}}(\xi, A, k) \leqq \varepsilon$, then the power algorithm needs to perform roughly $k^{\text{pow}} = 0.564 \ln\,(n)/\varepsilon$ steps, whereas the Lanczos algorithm will take roughly $k^{\text{Lan}} \leqq 1.605 \ln\,(n)/\sqrt{\varepsilon}$ steps. Thus

$$\frac{k^{\text{pow}}}{k^{\text{Lan}}} \geqq \frac{0.35}{\sqrt{\varepsilon}}.$$

As already indicated, we do not know if the upper bound for the Lanczos algorithm presented in part (a) is sharp. We verify the sharpness of this bound by many numerical tests. These tests seem to indicate that

$$e^{\text{avg}}(\xi^{\text{Lan}}, A, k) = \Theta(k^{-2})$$

with the constant in the Θ notation independent of n. If this is the case, then the bound in part (a) is an overestimate by the factor $\ln^2 n$. Details of numerical tests are reported in § 6.

Theorem 3.2(b) yields a nonasymptotic estimate in terms of the two largest eigenvalues and the smallest eigenvalue of A. Observe that the bound in part (b) is better than the bound in (a) if $(\lambda_1 - \lambda_{p+1})/(\lambda_1 - \lambda_n)$ is not too close to 0.

4. Probabilistic case. In this section, we present bounds for the probabilistic relative failure (7) for the power and Lanczos algorithms. Proofs are given in § 5. As in § 3, we begin with the power algorithm.

It is easy to check that for $\varepsilon = 0$, the probabilistic relative failure of the power algorithm is $f^{\text{prob}}(\xi^{\text{pow}}, A, k, 0) = 1$ for all matrices A with at least two distinct eigenvalues, and $f^{\text{prob}}(\xi^{\text{pow}}, A, k, 0) = 0$ for all matrices A having only one distinct eigenvalue. That is why we assume in Theorem 4.1 that $\varepsilon > 0$. The probabilistic relative failure of the power algorithm depends on the function g defined by

$$(9) \qquad g(\varepsilon, k) = \frac{(1-\varepsilon)^{k-1/2}(1-1/(2k-1))^{k-1}}{\sqrt{(1-\varepsilon)^{2k-1}(1-1/(2k-1))^{2(k-1)} + 2(k-1)\varepsilon}}, \qquad k \geqq 2.$$

Note that

$$g(\varepsilon, k) \leqq (1-\varepsilon)^{k-1/2},$$

$$g(\varepsilon, k) = \frac{1}{\sqrt{2e}} \frac{(1-\varepsilon)^{k-1/2}}{\sqrt{\varepsilon(k-1)}} (1 + o(1)) \quad \text{as } k \to +\infty \quad \text{for } \varepsilon > 0,$$

with a negative $o(1)$ term.

THEOREM 4.1. *Let ξ^{pow} be the power algorithm defined by (3) and let $\varepsilon > 0$.*

(a) *For any symmetric positive definite matrix A and for any $k \geqq 2$ we have*

$$f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon) \leqq 0.824\sqrt{n} \int_0^{g(\varepsilon,k)} (1-t^2)^{(n-1)/2}\, dt$$

$$\leqq \min\left\{0.824, \frac{0.354}{\sqrt{\varepsilon(k-1)}}\right\}\sqrt{n}(1-\varepsilon)^{k-1/2}.$$

(b) *For any integer $k \geqq 2$, let $\bar{A}$ be any symmetric matrix with two distinct eigenvalues $\lambda_1 > 0$ and $\lambda_i = \lambda_1(1-\varepsilon)(1-1/(2k-1))$ for $i = 2, 3, \cdots, n$. Then*

$$\max_{A=A^T>0} f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon) = f^{\text{prob}}(\xi^{\text{pow}}, \bar{A}, k, \varepsilon)$$

$$\geqq 0.797\sqrt{n}(1-1/n) \int_0^{g(\varepsilon,k)} (1-t^2)^{(n-1)/2}\, dt,$$

and for large n and k,

$$f^{\text{prob}}(\xi^{\text{pow}}, \bar{A}, k, \varepsilon) = a\sqrt{\frac{n}{\varepsilon}} \frac{(1-\varepsilon)^{k-1/2}}{\sqrt{k-1}} (1 + o(1)),$$

where $a = 1/\sqrt{\pi e} = 0.342\cdots$.

(c) *For any symmetric and positive definite matrix A, let p, $p < n$, and q denote the multiplicities of the two largest eigenvalues λ_1 and λ_{p+1}. If $\lambda_{p+1}/\lambda_1 < 1 - \varepsilon$, then*

$$\lim_{k\to+\infty} \frac{f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon)}{(\lambda_{p+1}/\lambda_1)^{p(k-1)}} = \frac{2(1-\varepsilon-\lambda_{p+1}/\lambda_1)^{p/2}}{p\varepsilon^{p/2}} \frac{\Gamma((p+q)/2)}{\Gamma(p/2)\Gamma(q/2)}.$$

Parts (a) and (b) of Theorem 4.1 present sharp bounds on the probabilistic relative failure of the power algorithm. The failure tends to 0 with the rate of convergence roughly $(1-\varepsilon)^{k-1/2}$. For small ε, this is quite unsatisfactory. On the other hand, if we are interested in a rough estimate of the largest eigenvalue, say $\varepsilon = 0.5$, then the rate is quite good.

The dependence of the probabilistic relative failure on the dimension n is through $\sqrt{n}$. This shows that the dimension n affects the probabilistic case for the power algorithm

in a much more substantial way than the average case, which depends only through $\ln n$.

Consider now the minimal number of steps needed to get

$$f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon) \leqq \delta \quad \forall A = A^T > 0,$$

where δ denotes the measure of a set for which the power algorithm may fail.

Then $k \simeq \ln(n/\delta^2)/(2\varepsilon)$. Hence the dimension n and the parameter δ affect the number of steps only logarithmically. Even for huge n and very small δ, the factor $\ln(n/\delta^2)/2$ is quite moderate. The dependence on ε is much more crucial since k goes linearly to infinity with ε^{-1}. Observe that the dimension n and the parameter ε affect the number of steps in the same way in the average and probabilistic cases.

Theorem 4.1(c) presents the asymptotic behavior of the probabilistic relative failure of the power algorithm. The rate of convergence depends on the multiplicity p of the largest eigenvalue, and the rate improves as p increases. On the other hand, the asymptotic constant gets huge for large p and small ε.

Part (c) holds under the assumption that the ratio of two largest eigenvalues is not too close to one, $\lambda_{p+1}/\lambda_1 < 1 - \varepsilon$. Of course, this holds for sufficiently small ε. If, however, $\lambda_{p+1}/\lambda_1 \geqq 1 - \varepsilon$, then we do not know the asymptotic behavior of the probabilistic relative failure of the power algorithm and we suspect that its behavior may be quite different from that presented in part (c).

We wish to add that the modified power algorithm in the probabilistic case was analyzed by Dixon [3]. We present his result in Remark 7.3.

We now turn to the Lanczos algorithm. As was the case for the average case, we are able to present only upper bounds. Also in the probabilistic case we have

$$f^{\text{prob}}(\xi^{\text{Lan}}, A, k, \varepsilon) \leqq f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon) \quad \forall A \text{ and } k \tag{10}$$

and upper bounds of Theorem 4.1 can be used for the Lanczos algorithm. The following theorem presents some better bounds.

THEOREM 4.2. *Let ξ^{Lan} be the Lanczos algorithm defined by* (4) *and let $\varepsilon \in [0, 1)$.*

(a) *For any symmetric positive definite matrix A, let m denote the number of distinct eigenvalues of A. Then for $k \geqq m$,*

$$f^{\text{prob}}(\xi^{\text{Lan}}, A, k, \varepsilon) = 0;$$

for any k,

$$f^{\text{prob}}(\xi^{\text{Lan}}, A, k, \varepsilon) \leqq 1.648\sqrt{n}\, e^{-\sqrt{\varepsilon}(2k-1)}.$$

(b) *For any symmetric positive definite matrix A, let p, $p < n$, denote the multiplicity of the largest eigenvalue λ_1, and let λ_{p+1} and λ_n be the second largest and the smallest eigenvalues of A. Then for $\varepsilon > 0$,*

$$f^{\text{prob}}(\xi^{\text{Lan}}, A, k, \varepsilon) \leqq 1.648\sqrt{\frac{n}{\varepsilon}}\left(\frac{1 - \sqrt{(\lambda_1 - \lambda_{p+1})/(\lambda_1 - \lambda_n)}}{1 + \sqrt{(\lambda_1 - \lambda_{p+1})/(\lambda_1 - \lambda_n)}}\right)^{k-1}.$$

Theorem 4.2 states that also in the probabilistic case the Lanczos algorithm converges in m steps. For any k and for small ε the probabilistic relative failure of the Lanczos algorithm is roughly bounded by $\sqrt{n}\exp(-\sqrt{\varepsilon}(2k-1))$. This should be compared with a sharp bound for the power algorithm given by $\sqrt{n/\varepsilon}(1-\varepsilon)^k/\sqrt{k}$. Once more we see the superiority of the Lanczos algorithm. If we want to guarantee a δ-failure, $f^{\text{prob}}(\xi, A, k, \varepsilon) \leqq \delta$, then we have to perform roughly $k^{\text{pow}} = \ln(n/(\delta^2))/(2\varepsilon)$ steps by the power algorithm and roughly $k^{\text{Lan}} \leqq \ln(n/\delta^2)/(4\sqrt{\varepsilon})$ by the Lanczos algorithm.

Thus

$$\frac{k^{\text{pow}}}{k^{\text{Lan}}} \geqq \frac{2}{\sqrt{\varepsilon}}.$$

Observe a weak dependence on δ which only logarithmically affects the number of steps. The dependence on ε is more crucial.

As in the average case, Theorem 4.2(b) presents a nonasymptotic bound on the probabilistic relative failure of the Lanczos algorithm. Observe that the bound in part (b) is better then the bound in part (a) if $(\lambda_1 - \lambda_{p+1})/(\lambda_1 - \lambda_n) \geqq \varepsilon$.

5. Proofs of theorems. In this section, we present proofs of theorems from §§ 3 and 4. We begin with Theorem 3.1, which deals with convergence of the power algorithm in the average case.

Proof of Theorem 3.1. Let A be any symmetric positive matrix with eigenpairs (λ_i, η_i), where the eigenvectors η_i form an orthonormal basis of $\mathbf{R}^n$ and $\lambda_1 \geqq \lambda_2 \geqq \cdots \geqq \lambda_n > 0$, that is,

$$A\eta_i = \lambda_i \eta_i, \quad (\eta_i, \eta_j) = \delta_{i,j}, \quad i, j = 1, 2, \cdots, n.$$

Let $b = \sum_{i=1}^n b_i \eta_i$. From (3) we get

$$\xi^{\text{pow}} = \xi^{\text{pow}}(A, b, k) = \sum_{i=1}^n b_i^2 \lambda_i^{2k-1} \Big/ \sum_{i=1}^n b_i^2 \lambda_i^{2(k-1)}.$$

Let $x_i = \lambda_i/\lambda_1 \in (0, 1]$. Then

$$\frac{\lambda_1 - \xi^{\text{pow}}}{\lambda_1} = \frac{\sum_{i=2}^n b_i^2 x_i^{2(k-1)}(1 - x_i)}{b_1^2 + \sum_{i=2}^n b_i^2 x_i^{2(k-1)}}.$$

From Remark 7.2 we know that the average relative error can be defined through the integration over the unit ball B_n,

$$e_k^{\text{pow}} = e^{\text{avg}}(\xi^{\text{pow}}, A, k) = \frac{1}{c_n} \int_{B_n} \frac{\lambda_1 - \xi^{\text{pow}}(A, b, k)}{\lambda_1} db, \tag{11}$$

where $c_n = \pi^{n/2}/\Gamma(1 + n/2)$ is the Lebesgue measure of the unit ball B_n. Since Lebesgue measure is orthogonally invariant, we can integrate in (11) with respect to b_i:

$$e_k^{\text{pow}} = \frac{1}{c_n} \int_{B'} \sum_{i=2}^n b_i^2 x_i^{2(k-1)}(1 - x_i) \left(\int_{b_1^2 \leqq 1 - \|b\|_{n-1}^2} \frac{db_1}{b_1^2 + \sum_{i=2}^n b_i^2 x_i^{2(k-1)}} \right) d\vec{b}$$

$$= \frac{2}{c_n} \int_{B'} \sum_{i=2}^n b_i^2 x_i^{2(k-1)}(1 - x_i) \left(\sum_{i=2}^n b_i^2 x_i^{2(k-1)} \right)^{-1/2} \arctan(h(b))\, d\vec{b},$$

where B' is the $(n-1)$-dimensional unit ball $\|b\|_{n-1}^2 = \sum_{i=2}^n b_i^2$, $d\vec{b}$ stands for $db_2 \cdots db_n$, and

$$h(b) = \sqrt{(1 - \|b\|_{n-1}^2)/\sum_{i=2}^n b_i^2 x_i^{2(k-1)}}.$$

Schwartz's inequality for sums,

$$\sum_{i=2}^n y_i z_i \leqq \left(\sum_{i=2}^n y_i^2 \right)^{1/2} \left(\sum_{i=2}^n z_i^2 \right)^{1/2},$$

with $y_i = b_i x_i^{k-1}$ and $z_i = b_i x_i^{k-1}(1 - x_i)$, yields

$$e_k^{\text{pow}} \leqq \frac{2}{c_n} \int_{B'} \left(\sum_{i=2}^{n} b_i^2 x_i^{2(k-1)} (1 - x_i)^2 \right)^{1/2} \arctan(h(b))\, d\vec{b}.$$

Now, using Schwartz's inequality for integrals, we get

$$e_k^{\text{pow}} = \frac{2}{c_n} \left(\int_{B'} d\vec{b} \right)^{1/2} \left(\int_{B'} \sum_{i=2}^{n} b_i^2 x_i^{2(k-1)} (1 - x_i)^2 \arctan^2(h(b))\, d\vec{b} \right)^{1/2}$$

$$= \frac{2c_{n-1}}{c_n} \left(\frac{1}{c_{n-1}} \int_{B'} \arctan^2(h(b)) \left(\sum_{i:\, x_i \leqq \beta} b_i^2 x_i^{2(k-1)} (1 - x_i)^2 \right.\right.$$

$$\left.\left. + \sum_{i:\, x_i > \beta} b_i^2 x_i^{2(k-1)} (1 - x_i)^2 \right) d\vec{b} \right)^{1/2}$$

for any number $\beta \in [0, 1]$. Here, $c_{n-1} = \pi^{(n-1)/2}/\Gamma(1 + (n-1)/2)$ is the Lebesgue measure of the $(n-1)$-dimensional unit ball.

Consider the function $H(t) = (1 - t)^2 t^{2(k-1)}$. The maximum of H is attained at $t_0 = 1 - 1/k$ and H is increasing in $[0, t_0]$. Let $\beta \leqq t_0$. Since $\arctan z \leqq \pi/2$ and $\arctan z \leqq z$, then

$$\arctan^2(h(b)) \sum_{i:\, x_i \leqq \beta}^{n} b_i^2 x_i^{2(k-1)} (1 - x_i)^2 \leqq \left(\frac{\pi}{2} \right)^2 \sum_{i=2}^{n} b_i^2 \beta^{2(k-1)} (1 - \beta)^2,$$

$$\arctan^2(h(b)) \sum_{i:\, x_i > \beta}^{n} b_i^2 x_i^{2(k-1)} (1 - x_i)^2 \leqq h^2(b) \sum_{i=2}^{n} b_i^2 x_i^{2(k-1)} (1 - \beta)^2$$

$$= (1 - \|b\|_{n-1}^2)(1 - \beta)^2.$$

Combining these bounds, we obtain

$$e_k^{\text{pow}} \leqq \frac{2c_{n-1}}{c_n} \left(\frac{1}{c_{n-1}} \int_{B'} (1 - \beta)^2 \left(\frac{\pi^2}{4} \|b\|_{n-1}^2 \beta^{2(k-1)} + (1 - \|b\|_{n-1}^2) \right) d\vec{b} \right)^{1/2}$$

$$= \frac{2c_{n-1}}{c_n} (1 - \beta) \left(\frac{1}{c_{n-1}} \left(\frac{\pi^2}{4} \beta^{2(k-1)} - 1 \right) \int_{B'} \|b\|_{n-1}^2\, d\vec{b} + 1 \right)^{1/2}.$$

Recall that for any measurable function $f\colon [0, r] \to \mathbf{R}$, we have

$$\int_{b \in \mathbf{R}^i, \|b\| \leqq r} f(\|b\|)\, db = i c_i \int_0^r t^{i-1} f(t)\, dt, \tag{12}$$

where $c_i = \pi^{i/2}/\Gamma(1 + i/2)$ (see, e.g., Gradshteyn and Ryzhik [4, 4.642]). For $f(t) = t^2$ we get

$$\int_{B'} \|b\|_{n-1}^2\, d\vec{b} = \frac{n-1}{n+1} c_{n-1}.$$

From this we have

$$e_k^{\text{pow}} \leqq \frac{2c_{n-1}}{c_n} (1 - \beta) \sqrt{\frac{\pi^2}{4} \beta^{2(k-1)} \frac{n-1}{n+1} + \frac{2}{n+1}}$$

$$\leqq \frac{2c_{n-1}}{c_n} (1 - \beta) \sqrt{\frac{\pi^2}{4} \beta^{2(k-1)} + \frac{2}{n}}.$$

Observe that

$$(13)\qquad \sqrt{\frac{n}{2\pi}} \leqq \frac{c_{n-1}}{c_n} = \sqrt{\frac{n}{2\pi}}\,\frac{\sqrt{n/2}\,\Gamma(n/2)}{\Gamma(n/2+\frac{1}{2})} \leqq \sigma\sqrt{\frac{n}{2\pi}}, \qquad \sigma = \frac{192}{105\sqrt{\pi}} \leqq 1.032.$$

Indeed, it is enough to show that $\sqrt{x}\,\Gamma(x)/\Gamma(x+\frac{1}{2}) \in [1, \sigma]$ for $x = n/2 \geqq 4$. To show this, consider

$$H(x) = \ln \Gamma(x) + \tfrac{1}{2}\ln x - \ln \Gamma(x+\tfrac{1}{2}).$$

From Gradshteyn and Ryzhik [4, 8.360, 8.365, 8.372], we have $H'(x) = \psi(x) + 1/(2x) - \psi(x+\frac{1}{2}) \leqq 0$ for the psi function ψ. Thus

$$0 = H(+\infty) \leqq H(x) \leqq H(4) = \ln \sigma,$$

as claimed.

Note that for $k - 1 \leqq \pi^{-1/2}\ln n$, Theorem 3.1(a) trivially holds since

$$e_k^{\text{pow}} \leqq 1 \leqq \pi^{-1/2}\ln n/(k-1).$$

Assume thus that $k - 1 > \pi^{-1/2}\ln n$. Now take $\beta = 1 - \ln n/(2(k-1))$. Then $\beta \in (0, t_0]$ for $n \geqq 8$ and $\beta^{2(k-1)} \leqq 1/n$. Thus we have

$$e_k^{\text{pow}} \leqq 2\sigma\sqrt{\frac{n}{2\pi}}\,\frac{\ln n}{2(k-1)}\sqrt{\frac{\pi^2}{4n}+\frac{2}{n}} = \sigma\sqrt{\frac{\pi^2+8}{8\pi}}\,\frac{\ln n}{k-1} \leqq 0.871\,\frac{\ln n}{k-1}.$$

This proves that $\alpha(n) \leqq 0.871$ for all $n \geqq 8$.

For large n, take $\beta = 1 - a\ln n/(2(k-1))$ with $a = 1 + 1/\ln\ln n$. Then $\beta^{2(k-1)} \leqq 1/n^a$ and

$$e_k^{\text{pow}} \leqq \frac{2c_{n-1}}{c_n}\,a\,\frac{\ln n}{2(k-1)}\sqrt{\frac{\pi^2}{4n^a}+\frac{2}{n}} = \alpha(n)\,\frac{\ln n}{k-1}.$$

Since $c_{n-1}/c_n \simeq \sqrt{n/2\pi}$ and $1/n^{a-1}$ goes to 0, we have

$$\alpha(n) \simeq \frac{1}{\sqrt{\pi}} = 0.564\cdots,$$

which completes the proof of Theorem 3.1(a).

We now prove part (b) of Theorem 3.1. Consider the matrix A from part (b). Then

$$x_i = \frac{\lambda_i}{\lambda_1} = 1 - \frac{\alpha}{2(k-1)}, \quad i = 2, 3, \cdots, n, \quad \alpha = \ln\left(\frac{n}{\ln n}\right).$$

We have $1 - x_i = \alpha/(2(k-1))$ and $\beta = x_i^{2(k-1)} = \ln(n)/n(1+o(1))$. For the matrix A, (11) takes the form

$$\begin{aligned} e^{\text{avg}}(\xi^{\text{pow}}, A, k) &= \frac{1}{c_n}\,\frac{\alpha}{2(k-1)}\int_{B_n}\frac{\beta\sum_{i=2}^n b_i^2 + b_1^2 - b_1^2}{b_1^2+\beta\sum_{i=2}^n b_i^2}\,db \\ &= \frac{1}{c_n}\,\frac{\alpha}{2(k-1)}\left(c_n - \int_{B_n}\frac{b_1^2}{b_1^2+\beta\sum_{i=2}^n b_i^2}\,db\right) \\ &= \frac{1}{c_n}\,\frac{\alpha}{2(k-1)}\left(c_n - \int_{B_n}\frac{\beta^{-1}b_1^2}{\beta^{-1}b_1^2+\sum_{i=2}^n b_i^2}\,db\right). \end{aligned}$$

Since $\beta^{-1} \geq 1$, then

$$e^{\text{avg}}(\xi^{\text{pow}}, A, k) \geq \frac{1}{c_n} \frac{\alpha}{2(k-1)} \left(c_n - \beta^{-1} \int_{B_n} \frac{b_1^2}{\sum_{i=1}^n b_i^2} \, db \right)$$

$$= \frac{1}{c_n} \frac{\alpha}{2(k-1)} \left(c_n - \beta^{-1} \frac{c_n}{n} \right)$$

$$= \frac{\ln (n/\ln n)}{2(k-1)} \left(1 - \frac{1 + o(1)}{\ln n} \right) = 0.5 \frac{\ln n}{k-1} (1 + o(1)),$$

as claimed.

We proceed to prove Theorem 3.1(c). Recall that p and q are multiplicities of the two largest distinct eigenvalues of A. From (11) we can write

$$e_k^{\text{pow}} = \frac{1}{c_n} \int_{B_n} \frac{x_{p+1}^{2(k-1)} (1 - x_{p+1}) \sum_{i=p+1}^{p+q} b_i^2}{\sum_{i=1}^p b_i^2 + x_{p+1}^{2(k-1)} \sum_{i=p+1}^{p+q} b_i^2} \, db (1 + o(1)) \quad \text{as } k \to +\infty.$$

Let $a = x_{p+1}^{2(k-1)} = (\lambda_{p+1}/\lambda_1)^{2(k-1)}$ and $\alpha = (1 - x_{p+1})a$. Integrating with respect to $b_{p+q+1}, \cdots, b_n$, we get

$$\frac{e_k^{\text{pow}}}{1 + o(1)} = \frac{\alpha c_{n-p-q}}{c_n} \int_{B_{p+q}} \frac{\sum_{i=p+1}^{p+q} b_i^2 (1 - \sum_{j=1}^{p+q} b_i^2)^{(n-p-q)/2}}{\sum_{i=1}^p b_i^2 + a \sum_{i=p+1}^{p+q} b_i^2} \, db,$$

where B_i is the i-dimensional unit ball and c_i is its measure. We rewrite the last integral as an integral over the unit ball B_p and the ball $\sum_{i=p+1}^{p+q} b_i^2 \leq 1 - \sum_{i=1}^p b_i^2$. Let $t_i = b_i/(1 - \sum_{j=1}^p b_j^2)^{1/2}$ for $i = p+1, \cdots, p+q$ and let $\|b\|^2 = \sum_{i=1}^p b_i^2$, $\|t\|^2 = \sum_{i=p+1}^{p+q} t_i^2$. Then we have

$$\frac{e_k^{\text{pow}}}{1 + o(1)} \frac{c_n}{\alpha c_{n-p-q}}$$

$$= \int_{B_p} \int_{B_q} \frac{(1 - \|b\|^2)^{1+q/2} \|t\|^2 (1 - \|b\|^2 - (1 - \|b\|^2)\|t\|^2)^{(n-p-q)/2}}{\|b\|^2 + a(1 - \|b\|^2)\|t\|^2} \, dt \, db.$$

Using (12) first for the second integral and then for the first integral, we get

$$\frac{e_k^{\text{pow}}}{1 + o(1)} = \gamma_1 \int_{B_p} \int_0^1 \frac{(1 - \|b\|^2)^{1+(n-p)/2} t^{q+1} (1 - t^2)^{(n-p-q)/2}}{\|b\|^2 + a(1 - \|b\|^2)t^2} \, dt \, db$$

$$= \gamma_2 \int_0^1 \int_0^1 \frac{(1 - x^2)^{1+(n-p)/2} x^{p-1} t^{q+1} (1 - t^2)^{(n-p-q)/2}}{x^2 + a(1 - x^2)t^2} \, dt \, dx,$$

where $\gamma_1 = (\alpha q c_{n-p-q} c_q)/c_n$ and $\gamma_2 = (\alpha p q c_{n-p-q} c_p c_q)/c_n$.

Consider now the case $p \geq 3$. Then the last double integral is finite even for $a = 0$. Recalling the definition of the beta function,

$$B(i, j) = 2 \int_0^1 t^{2i-1} (1 - t^2)^{j-1} \, dt = \frac{\Gamma(i)\Gamma(j)}{\Gamma(i+j)} \tag{14}$$

(see, e.g., Gradshteyn and Ryzhik [4, 8.380 and 8.384]), we have

$$\frac{e_k^{\text{pow}}}{1 + o(1)} = \gamma_2 \left(\int_0^1 (1 - x^2)^{1+(n-p)/2} x^{p-3} \, dx \right) \left(\int_0^1 (1 - t^2)^{(n-p-q)/2} t^{q+1} \, dt \right)$$

$$= \frac{\gamma_2}{4} B\left(\frac{p-2}{2}, \frac{n-p}{2} + 2 \right) B\left(\frac{q}{2} + 1, \frac{n-p-q}{2} + 1 \right).$$

Expressing c_i's and B's in terms of the gamma function, we finally get

$$\frac{e_k^{\text{pow}}}{1+o(1)} = \frac{\alpha pq}{4}\frac{\Gamma(-1+p/2)}{\Gamma(1+p/2)} = \left(1-\frac{\lambda_{p+1}}{\lambda_1}\right)\left(\frac{\lambda_{p+1}}{\lambda_1}\right)^{2(k-1)}\frac{q}{p-2},$$

which proves Theorem 3.1(c) for $p \geqq 3$.

Assume now that $p = 2$. Observe that for $a \to 0$ we have

$$\begin{aligned}
&\int_0^1 \frac{x(1-x^2)^{n/2}}{x^2 + a(1-x^2)t^2}\,dx \\
&= \int_0^1 \frac{x}{(1-at^2)x^2 + at^2}\,dx + O\left(\int_0^1 \frac{x^3}{(1-at^2)x^2+at^2}\,dx\right) \\
&= \int_0^1 \frac{x}{x^2+at^2}\,dx + O\left(\int_0^1 x\,dx\right) = \tfrac{1}{2}\ln(x^2+at^2)\Big|_0^1 + O(1) \\
&= \ln\left(\frac{1}{t\sqrt{a}}\right) + O(1).
\end{aligned}$$

Therefore, we have

$$\begin{aligned}
\frac{e_k^{\text{pow}}}{1+o(1)} &= \gamma_2 \int_0^1 t^{q+1}(1-t^2)^{(n-q-2)/2}\ln\left(\frac{1}{t\sqrt{a}}\right)dt \\
&= \frac{\gamma_2}{2}\ln\left(\frac{1}{\sqrt{a}}\right)B\left(\frac{q}{2}+1, \frac{n-q}{2}\right)(1+o(1)),
\end{aligned}$$

where now $\gamma_2 = (2q\alpha c_{n-q-2}c_q c_2)/c_n$. This yields

$$\frac{e_k^{\text{pow}}}{1+o(1)} = q\alpha \ln\frac{1}{\sqrt{a}} = q\left(1-\frac{\lambda_3}{\lambda_1}\right)\left(\frac{\lambda_3}{\lambda_1}\right)^{2(k-1)}(k-1)\ln\frac{\lambda_1}{\lambda_3},$$

which proves Theorem 3.1(c) for $p = 2$.

Finally, assume that $p = 1$. Then for $a \to 0$ we have

$$\begin{aligned}
\int_0^1 \frac{(1-x^2)^{(n+1)/2}}{x^2+a(1-x^2)t^2}\,dx &= \int_0^1 \frac{1}{x^2+at^2}\,dx + O(1) \\
&= \frac{1}{t\sqrt{a}}\arctan\frac{x}{t\sqrt{a}}\Big|_0^1 + O(1) = \frac{1}{t\sqrt{a}}\frac{\pi}{2} + O(1).
\end{aligned}$$

Thus we have

$$\begin{aligned}
\frac{e_k^{\text{pow}}}{1+o(1)} &= \gamma_2 \frac{\pi}{2\sqrt{a}}\int_0^1 t^q(1-t^2)^{(n-q-1)/2}\,dt \\
&= \gamma_2 \frac{\pi}{4\sqrt{a}}B\left(\frac{q+1}{2}, \frac{n-q+1}{2}\right) = \sqrt{\pi}\left(1-\frac{\lambda_2}{\lambda_1}\right)\left(\frac{\lambda_2}{\lambda_1}\right)^{k-1}\frac{\Gamma((q+1)/2)}{\Gamma(q/2)},
\end{aligned}$$

with $\gamma_2 = (\alpha q c_{n-q-1}c_q c_1)/c_n$. This completes the proof of Theorem 3.1. □

Proof of Theorem 3.2. The Lanczos algorithm takes the maximum of $(Ax, x)/(x, x)$ for $0 \neq x \in \text{span}(b, Ab, \cdots, A^{k-1}b)$; see (4). This means that $x = P(A)b$ for a nonzero polynomial from the class $\mathscr{P}_k$ of polynomials of degree less than or equal to

$k-1$. We have

$$\xi^{\mathrm{Lan}} = \xi^{\mathrm{Lan}}(A, b, k) = \max_{P \in \mathscr{P}_k} \frac{\sum_{i=1}^n b_i^2 \lambda_i P^2(\lambda_i)}{\sum_{i=1}^n b_i^2 P^2(\lambda_i)}.$$

The relative error of the Lanczos algorithm is given by

$$\frac{\lambda_1 - \xi^{\mathrm{Lan}}}{\lambda_1} = \min_{P \in \mathscr{P}_k} \frac{\sum_{i=2}^n b_i^2 P^2(\lambda_i)(1 - \lambda_i/\lambda_1)}{\sum_{i=1}^n b_i^2 P^2(\lambda_i)}.$$

Using a continuity argument, we may restrict ourselves to polynomials P such that $P(\lambda_1) \neq 0$. Let $Q(t) = P(\lambda_1 t)/P(\lambda_1)$. Then $Q \in \mathscr{P}_k$ and $Q(1) = 1$. Let $\mathscr{P}_k(1)$ denote such polynomials. Thus, for $x_i = \lambda_i/\lambda_1 \in (0, 1]$, we have

$$\frac{\lambda_1 - \xi^{\mathrm{Lan}}}{\lambda_1} = \inf_{Q \in \mathscr{P}_k(1)} \frac{\sum_{i=2}^n b_i^2 Q^2(x_i)(1 - x_i)}{b_1^2 + \sum_{i=2}^n b_i^2 Q^2(x_i)}. \tag{15}$$

As for the power algorithm, we conclude that the average relative error of the Lanczos algorithm is given by

$$e_k^{\mathrm{Lan}} = e_k^{\mathrm{avg}}(\xi^{\mathrm{Lan}}, A, k) = \frac{1}{c_n} \int_{B_n} \inf_{Q \in \mathscr{P}_k(1)} \frac{\sum_{i=2}^n b_i^2 Q^2(x_i)(1 - x_i)}{b_1^2 + \sum_{i=2}^n b_i^2 Q^2(x_i)}\, db. \tag{16}$$

Assume first that $k \geqq m$. This means that the set $\{x_1, x_2, \cdots, x_n\}$ contains m distinct elements $\{t_1, t_2, \cdots, t_m\}$ with $t_1 = 1$. Take

$$Q(x) = \prod_{i=2}^m (x - t_i)/(1 - t_i).$$

Then $Q \in \mathscr{P}_k(1)$ and the integrand in (16) vanishes for $b_1 \neq 0$. Since $b_1 = 0$ for a set of measure zero, we have $e_k^{\mathrm{Lan}} = 0$, as claimed.

Assume now that $k \in [4, m-1]$. We find an upper bound on e_k^{Lan} by changing the order of integration and taking the infimum

$$e_k^{\mathrm{Lan}} \leqq \frac{1}{c_n} \inf_{Q \in \mathscr{P}_k(1)} \int_{B_n} \frac{\sum_{i=2}^n b_i^2 Q^2(x_i)(1 - x_i)}{b_1^2 + \sum_{i=2}^n b_i^2 Q^2(x_i)}\, db.$$

Observe that to estimate the integral we can repeat the same reasoning as for the power algorithm with the polynomial Q instead of x^{k-1}. Therefore, for any $\beta \in [0, 1]$ we have

$$e_k^{\mathrm{Lan}} \leqq \frac{2c_{n-1}}{c_n} \inf_{Q \in \mathscr{P}_k(1)} \left(\frac{1}{c_{n-1}} \frac{\pi^2}{4} \int_{B'} \sum_{i:\, x_i \leqq \beta} b_i^2 Q^2(x_i)(1 - x_i)^2\, d\vec{b} + \frac{1}{c_{n-1}} (1 - \beta)^2 \int_{B'} (1 - \|b\|_{n-1}^2)\, d\vec{b} \right)^{1/2}.$$

Let

$$w(\beta) = \inf_{Q \in \mathscr{P}_k(1)} \max_{0 \leqq x \leqq \beta} Q^2(x)(1 - x)^2. \tag{17}$$

Then

$$e_k^{\mathrm{Lan}} \leqq \frac{2c_{n-1}}{c_n} \left(\frac{\pi^2}{4} w(\beta) + \frac{2(1 - \beta)^2}{n} \right)^{1/2},$$

and using (13) we have

$$e_k^{\mathrm{Lan}} \leqq 0.412\sqrt{\pi^2 n w(\beta) + 8(1-\beta)^2}. \tag{18}$$

To get an upper bound on e_k^{Lan} we thus need to find an upper bound on $w(\beta)$ and select a proper β (see also Remark 7.4). Take

$$Q(x) = T_{k-1}((2/\beta)x - 1)/T_{k-1}((2/\beta) - 1),$$

where T_{k-1} is the Chebyshev polynomial of the first kind of degree $k-1$. Then

$$w(\beta) \leqq T_{k-1}^{-2}((2/\beta) - 1) \leqq 4\left(\frac{1-\sqrt{1-\beta}}{1+\sqrt{1-\beta}}\right)^{2(k-1)} \leqq 4e^{-4(k-1)\sqrt{1-\beta}}. \tag{19}$$

Let $\gamma = \sqrt{1-\beta} \in [0, 1]$. Then

$$e_k^{\mathrm{Lan}} \leqq 0.824\sqrt{\pi^2 n e^{-4(k-1)\gamma} + 2\gamma^4}.$$

Note that for $k - 1 \leqq \sqrt{0.103}\ln(n(k-1)^4)$, Theorem 3.2(a) trivially holds since

$$e_k^{\mathrm{Lan}} \leqq 1 \leqq 0.103\left(\frac{\ln n(k-1)^4}{k-1}\right)^2.$$

Assume thus that $k - 1 > \sqrt{0.103}\ln n(k-1)^4$. Now take

$$\gamma = \frac{1}{4(k-1)}\ln\frac{128\pi^2 n(k-1)^4}{(\ln n(k-1)^4)^4}.$$

Since $128\pi^2 \leqq (\ln n(k-1)^4)^4$ for $n \geqq 8$ and $k \geqq 4$, we have $\gamma \leqq 1$. Clearly, $\gamma \geqq 0$. A simple calculation yields

$$e_k^{\mathrm{Lan}} \leqq 0.103\left(\frac{\ln n(k-1)^4}{k-1}\right)^2,$$

as claimed in Theorem 3.2(a).

To prove part (b), define $\beta_1 = \lambda_n/\lambda_1$ and $\beta_2 = \lambda_{p+1}/\lambda_1$. Repeating the same reasoning that led to (18), we conclude that the sum for $x_i > \beta_2$ of the upper bound on e_k^{Lan} disappears and

$$e_k^{\mathrm{Lan}} \leqq 0.412\sqrt{\pi^2 n w(\beta_1, \beta_2)},$$

where

$$w(\beta_1, \beta_2) = \inf_{Q \in \mathscr{P}_k(1)} \max_{\beta_1 \leqq x \leqq \beta_2} Q^2(x)(1-x)^2.$$

For $\beta = (\lambda_{p+1} - \lambda_n)/(\lambda_1 - \lambda_n)$, take

$$Q(x) = T_{k-1}\left(\frac{2(x-\beta_1)}{\beta_2 - \beta_1} - 1\right) \Big/ T_{k-1}((2/\beta) - 1).$$

Then $w(\beta_1, \beta_2) \leqq T_{k-1}^{-2}((2/\beta) - 1)$ and using the second inequality of (19), we get part (b). This completes the proof of Theorem 3.2. □

Proof of Theorem 4.1. We need to find the measure of the set

$$Z = \left\{b \in \mathbf{R}^n : \|b\| = 1, \frac{\sum_{i=2}^n b_i^2 x_i^{2(k-1)}(1-x_i)}{b_1^2 + \sum_{i=2}^n b_i^2 x_i^{2(k-1)}} > \varepsilon\right\}$$

$$= \left\{b \in \mathbf{R}^n : \|b\| = 1, \sum_{i=2}^n b_i^2(1-\varepsilon-x_i)x_i^{2(k-1)} > \varepsilon b_1^2\right\},$$

where, as before, $x_i = \lambda_i/\lambda_1$.

Note that $H(x) = (1 - \varepsilon - x)x^{2(k-1)}$ for $x \in [0, 1]$ attains its maximum value at $x^* = (1 - \varepsilon)(1 - 1/(2k - 1))$ and $H(x^*) = (1 - \varepsilon)^{2k-1}(1 - 1/(2k - 1))^{2(k-1)}/(2k - 1)$. Then

$$\sum_{i=2}^{n} b_i^2 (1 - \varepsilon - x_i) x_i^{2(k-1)} \leqq H(x^*) \sum_{i=2}^{n} b_i^2$$

and $Z \subset Z^*$, where

$$Z^* = \left\{ b \in \mathbf{R}^n : \|b\| = 1, \sum_{i=2}^{n} b_i^2 > \alpha b_1^2 \right\}$$

with

$$\alpha = \frac{\varepsilon}{H(x^*)} = \frac{(2k - 1)\varepsilon}{(1 - \varepsilon)^{2k-1}(1 - 1/(2k - 1))^{2(k-1)}}.$$

Obviously,

$$f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon) = \mu(Z) \leqq \mu(Z^*).$$

We have

$$1 - \mu(Z^*) = \frac{2}{c_n} \int_0^1 \int_{\sum_{i=2}^n b_i^2 \leqq \min\{1 - b_1^2, \alpha b_1^2\}} db$$

$$= \frac{2c_{n-1}}{c_n} \int_0^1 \min\{1 - t^2, \alpha t^2\}^{(n-1)/2} \, dt.$$

Observe that $\min\{1 - t^2, \alpha t^2\} = \alpha t^2$ for $t \leqq 1/\sqrt{1 + \alpha} = g(k, \varepsilon)$ (see (9) for the definition of g), and $\min\{1 - t^2, \alpha t^2\} = 1 - t^2$ for $t \geqq g(k, \varepsilon)$. Therefore,

$$1 - \mu(Z^*) = \gamma\left(\int_0^{g(k,\varepsilon)} (\alpha t^2)^j \, dt + \int_{g(k,\varepsilon)}^1 (1 - t^2)^j \, dt \right)$$

$$= \gamma\left(\frac{1}{n\sqrt{1 + \alpha}} \left(\frac{\alpha}{1 + \alpha} \right)^j + \int_0^1 (1 - t^2)^j \, dt - \int_0^{g(k,\varepsilon)} (1 - t^2)^j \, dt \right),$$

where $j = (n - 1)/2$ and $\gamma = 2c_{n-1}/c_n$. Since $c_n = 2c_{n-1} \int_0^1 (1 - t^2)^{(n-1)/2} \, dt$, we get

$$\mu(Z^*) = \frac{2c_{n-1}}{c_n} \left(\int_0^{g(k,\varepsilon)} (1 - t^2)^{(n-1)/2} \, dt - \frac{g(k, \varepsilon)}{n} \left(\frac{\alpha}{1 + \alpha} \right)^{(n-1)/2} \right). \tag{20}$$

From (13) we have $\gamma \leqq 2.064\sqrt{n/(2\pi)}$ and

$$\mu(Z^*) \leqq 0.824\sqrt{n} \int_0^{g(k,\varepsilon)} (1 - t^2)^{(n-1)/2} \, dt \leqq 0.824\sqrt{n} g(k, \varepsilon). \tag{21}$$

This and (9) complete the proof of Theorem 4.1(a).

We proceed to part (b). It is clear that

$$f^{\text{prob}}(\xi^{\text{pow}}, \bar{A}, k, \varepsilon) = \mu(Z^*) = \max_{A = A^T > 0} f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon).$$

To estimate $\mu(Z^*)$ from below, note that $\gamma \geqq \sqrt{2n/\pi}$ due to (13), and

$$\int_0^{g(k,\varepsilon)} (1 - t^2)^{(n-1)/2} \, dt \geqq g(k, \varepsilon) \left(\frac{\alpha}{1 + \alpha} \right)^{(n-1)/2}.$$

Therefore,

$$\mu(Z^*) \geqq 0.797\sqrt{n}\left(1 - \frac{1}{n}\right)\int_0^{g(k,\varepsilon)} (1 - t^2)^{(n-1)/2}\, dt,$$

as claimed. The asymptotic formula follows from the estimates of (9).

To prove Theorem 4.1(c), note that we need to find the measure of the set

$$W = \left\{ b \in \mathbf{R}^n : \|b\| = 1, \sum_{i=p+1}^{p+q} b_i^2 (1 - \varepsilon - x_{p+1}) x_{p+1}^{2(k-1)} > \varepsilon \sum_{i=1}^{p} b_i^2 \right\}$$

since $f^{\text{prob}}(\xi^{\text{pow}}, A, k, \varepsilon) = \mu(W)(1 + o(1))$ as $k \to \infty$.

Denote by $\beta = \varepsilon/((1 - \varepsilon - x_{p+1})x_{p+1}^{2(k-1)})$, $a_p = \sum_{i=1}^{p} b_i^2$, $a_{p+q} = \sum_{i=1}^{p+q} b_i^2$, $a_p' = \sum_{i=p+1}^{p+q} b_i^2$. We have

$$1 - \mu(W) = \frac{1}{c_n}\int_{a_p \leqq 1}\int_{a_p' \leqq \min\{1 - a_p, \beta a_p\}}\int_{\sum_{i=p+q+1}^{n} b_i^2 \leqq 1 - a_{p+q}} db$$

$$= \frac{c_{n-p-q}}{c_n}\int_{a_p \leqq 1}\int_{a_p' \leqq \min\{1 - a_p, \beta a_p\}} (1 - a_p - a_p')^{(n-p-q)/2}\, db',$$

where $db' = db_1 \cdots db_{p+q}$. Using (12) twice we get

$$1 - \mu(W) = \frac{qc_q c_{n-p-q}}{c_n}\int_{a_p \leqq 1}\int_0^{\min\{1 - a_p, \beta a_p\}^{1/2}} t^{q-1}(1 - a_p - t^2)^{(n-p-q)/2}\, dt$$

$$= \omega\int_0^1 x^{p-1}\int_0^{\min\{1 - x^2, \beta x^2\}^{1/2}} t^{q-1}(1 - x^2 - t^2)^{(n-p-q)/2}\, dt\, dx,$$

with $\omega = pqc_p c_q c_{n-p-q}/c_n$.

Observe that by formally setting $\beta = +\infty$, we get $\mu(W) = 0$ and

$$\omega\int_0^1 x^{p-1}\left(\int_0^{\sqrt{1-x^2}} t^{q-1}(1 - x^2 - t^2)^{(n-p-q)/2}\, dt\right) dx = 1. \tag{22}$$

We thus have for $h(t, x) = t^{q-1}(1 - x^2 - t^2)^{(n-p-q)/2}$ and $a = (1 - \mu(W))/\omega$,

$$a = \int_0^{1/\sqrt{1+\beta}} x^{p-1}\int_0^{x\sqrt{\beta}} h(t, x)\, dt\, dx + \int_{1/\sqrt{1+\beta}}^1 x^{p-1}\int_0^{\sqrt{1-x^2}} h(t, x)\, dt\, dx$$

$$= \int_0^1 x^{p-1}\int_0^{\sqrt{1-x^2}} h(t, x)\, dt\, dx - \int_0^{1/\sqrt{1+\beta}} x^{p-1}\int_{x\sqrt{\beta}}^{\sqrt{1-x^2}} h(t, x)\, dt\, dx.$$

Due to (22) we get

$$\mu(W) = \omega\int_0^{1/\sqrt{1+\beta}} x^{p-1}\int_{x\sqrt{\beta}}^{\sqrt{1-x^2}} t^{q-1}(1 - x^2 - t^2)^{(n-p-q)/2}\, dt\, dx.$$

Changing variables by $\nu = x\sqrt{1 + \beta}$, we obtain

$$\mu(W) = \frac{\omega}{(1 + \beta)^{p/2}}\int_0^1 \nu^{p-1}\int_{\nu\sqrt{\beta/(1+\beta)}}^{\sqrt{1 - \nu^2/(1+\beta)}} t^{q-1}\left(1 - \frac{\nu^2}{1 + \beta} - t^2\right)^{(n-p-q)/2} dt\, d\nu.$$

Note that $\beta \to +\infty$ as $k \to +\infty$. Therefore, we have

$$\begin{aligned}\frac{\mu(W)}{1+o(1)} &= \omega\beta^{-p/2}\int_0^1 \nu^{p-1}\int_\nu^1 t^{q-1}(1-t^2)^{(n-p-q)/2}\,dt\,d\nu \\ &= \omega\beta^{-p/2}\int_0^1 \left(\int_0^t \nu^{p-1}\,d\nu\right) t^{q-1}(1-t^2)^{(n-p-q)/2}\,dt \\ &= \frac{\omega}{p\beta^{p/2}}\int_0^1 t^{p+q-1}(1-t^2)^{(n-p-q)/2}\,dt \\ &= \frac{\omega}{2p\beta^{p/2}} B\left(\frac{p+q}{2}, \frac{n-p-q}{2}+1\right),\end{aligned}$$

the last equality due to (14). To complete the proof it is enough to observe that

$$\begin{aligned}\frac{\omega}{2p} B\left(\frac{p+q}{2}, \frac{n-p-q}{2}+1\right) &= \frac{pq\Gamma(1+n/2)\Gamma((p+q)/2)\Gamma(j)}{2p\Gamma(1+p/2)\Gamma(1+q/2)\Gamma(j)\Gamma(1+n/2)} \\ &= \frac{q\Gamma((p+q)/2)}{2\frac{p}{2}\Gamma(p/2)\frac{q}{2}\Gamma(q/2)} = \frac{2}{p}\frac{\Gamma((p+q)/2)}{\Gamma(p/2)\Gamma(q/2)},\end{aligned}$$

where $j = 1 + (n-p-q)/2$. □

Proof of Theorem 4.2. We need to find an upper bound on the measure of the set

$$Z = \{b : \|b\| = 1,\ \lambda_1 - \xi^{\mathrm{Lan}}(A, b, k) > \varepsilon\lambda_1\}.$$

Due to (15) we have

$$Z = \left\{b : \|b\| = 1,\ \inf_{Q\in\mathscr{P}_k(1)} \sum_{i=2}^n b_i^2 Q^2(x_i)(1-\varepsilon-x_i) > b_1^2\varepsilon\right\}.$$

Obviously

$$f_k^{\mathrm{Lan}} = f^{\mathrm{prob}}(\xi^{\mathrm{Lan}}, A, k, \varepsilon) = \mu(Z).$$

Assume first that $k \geq m$. As in the proof of Theorem 3.2, $\{x_1, x_2, \cdots, x_n\} = \{t_1, t_2, \cdots, t_m\}$ with distinct t_i and $t_1 = 1$. Setting $Q(x) = \prod_{i=2}^m (x - t_i)/(1 - t_i)$, we get $Z = \varnothing$. Thus $f_k^{\mathrm{Lan}} = 0$, as claimed.

Take now an arbitrary k. For $\varepsilon = 0$, the remaining bound of Theorem 4.2(a) trivially holds. (In fact, it is easy to see that for $k < m$, we have $f^{\mathrm{prob}}(\xi^{\mathrm{Lan}}, A, k, 0) = 1$.) Assume thus that $\varepsilon > 0$ and let

$$w_k = \inf_{Q\in\mathscr{P}_k(1)} \max_{0\leq x\leq 1} Q^2(x)(1-\varepsilon-x). \tag{23}$$

Then

$$Z \subset Z^* = \left\{b : \|b\| = 1,\ \sum_{i=2}^n b_i^2 > b_i^2\varepsilon/w_k\right\}$$

and $f_k^{\mathrm{Lan}} \leq \mu(Z^*)$.

Observe that an upper bound on the measure of the set Z^* was found in (21):

$$f_k^{\mathrm{Lan}} \leq 0.824\sqrt{n}\,g(k, \varepsilon) = 0.824\sqrt{\frac{n}{1+\varepsilon/w_k}}, \tag{24}$$

where now $g(k, \varepsilon) = 1/\sqrt{1+\alpha}$ with $\alpha = \varepsilon / w_k$. We prove that

$$w_k = 4\varepsilon \left(\frac{1-\sqrt{\varepsilon}}{1+\sqrt{\varepsilon}} \right)^{2k-1} \left(1 - \left(\frac{1-\sqrt{\varepsilon}}{1+\sqrt{\varepsilon}} \right)^{2k-1} \right)^{-2}.$$

Let $U_{2(k-1)}$ be the Chebyshev polynomial of the second kind of degree $2(k-1)$. Consider

$$Q(x) = U_{2(k-1)}(\sqrt{x/(1-\varepsilon)}) // U_{2(k-1)}(1/\sqrt{1-\varepsilon}), \qquad x \in [0, 1].$$

Since $U_{2(k-1)}$ is even, Q is a polynomial of degree $k-1$. Clearly, $Q(1) = 1$, so $Q \in \mathscr{P}_k(1)$. Let

$$H(x) = \sqrt{1-\varepsilon-x}\, Q(x), \qquad x \in [0, 1-\varepsilon].$$

For

$$t_i = (1-\varepsilon) \cos^2 \frac{(2i-1)\pi}{2(2k-1)}, \qquad i = 1, 2, \cdots, k,$$

the extremal points of $U_{2(k-1)}$ yield

$$H(t_i) = \frac{\sqrt{1-\varepsilon}}{U_{2(k-1)}(1/\sqrt{1-\varepsilon})} (-1)^{i-1}.$$

Note that

$$w_k \leqq a := \max_{0 \leqq x \leqq 1-\varepsilon} H^2(x) = \frac{1-\varepsilon}{U^2_{2(k-1)}(1/\sqrt{1-\varepsilon})} = 4\varepsilon c (1-c)^{-2},$$

where $c = ((1-\sqrt{\varepsilon})/(1+\sqrt{\varepsilon}))^{2k-1}$.

Assume that $w_k < a$. Then there exists a polynomial $P \in \mathscr{P}_k(1)$ such that $\max_{x \in [0,1]} P^2(x)(1-\varepsilon-x) < w_k$. The sign of the function

$$h(x) = \sqrt{1-\varepsilon-x}\,(Q(x) - P(x)), \qquad x \in [0, 1-\varepsilon],$$

alternates at t_i for $i = 1, 2, \cdots, k$. Thus, $Q - P$ has at least $k-1$ zeros in $[0, 1, -\varepsilon)$. Since $x = 1$ is also a zero of $Q - P$, we conclude that $Q = P$, which is a contradiction. Hence $w_k = a$, as claimed.

From this and (24) we finally get

$$\begin{aligned} f_k^{\text{Lan}} &\leqq 0.824\sqrt{4n/(4 + (1-c)^2/c)} \\ &\leqq 0.824\sqrt{4n/(2 + 1/c)} \leqq 1.648\sqrt{cn}. \end{aligned}$$

Theorem 4.2(a) follows by noting that $\sqrt{c} \leqq \exp(-\sqrt{\varepsilon})$.

To prove part (b), let $\beta_1 = \lambda_n/\lambda_1$, $\beta_2 = \lambda_{p+1}/\lambda_1$, and

$$u(\beta_1, \beta_2) = \inf_{Q \in \mathscr{P}_k(1)} \max_{\beta_1 \leqq x \leqq \beta_2} Q^2(x)(1-\varepsilon-x).$$

(Observe that $u(0, \beta_2) = w_k$ for $\beta_2 \geqq 1-\varepsilon$.) Then

$$Z \subset \left\{ b : \|b\| = 1, \sum_{i=2}^{n} b_i^2 > b_1^2 \varepsilon / u(\beta_1, \beta_2) \right\}$$

and $f_k^{\text{Lan}} \leqq 0.824\sqrt{n/(1+\varepsilon/u(\beta_1, \beta_2))} \leqq 0.824\sqrt{nu(\beta_1, \beta_2)/\varepsilon}$. We need to estimate $u(\beta_1, \beta_2)$. Changing the variables $x = (1-\beta_1)t + \beta_1$, we get

$$u(\beta_1, \beta_2) \leqq \max_{0 \leqq t \leqq 1-\lambda^*} Q^2(t) \quad \forall Q \in \mathscr{P}_k(1),$$

where $\lambda^* = (\lambda_1 - \lambda_{p+1})/(\lambda_1 - \lambda_n)$. We can use now the estimate (19) with $\beta = 1 - \lambda^*$ to get

$$u(\beta_1, \beta_2) \leq 4\left(\frac{1 - \sqrt{\lambda^*}}{1 + \sqrt{\lambda^*}}\right)^{2(k-1)},$$

which yields Theorem 4.2(b) and completes the proof. □

6. Numerical tests. We have tested the Lanczos algorithm with reorthogonalization in single precision for several matrices and many (pseudo)random vectors b. We report numerical results for one matrix A for which the relative errors of the Lanczos algorithm were the largest. The matrix A was chosen as follows. Observe that for any orthogonal matrix Q we have $\xi^{\text{Lan}}(Q^TAQ, Q^Tb, k) = \xi^{\text{Lan}}(A, b, k)$. Note also that the uniform distribution on the unit sphere of vectors b implies the same distribution of vectors Q^Tb. This shows that without loss of generality we can restrict ourselves to diagonal matrices while testing the Lanczos algorithm. Therefore, the matrix A was taken as diagonal. We chose the dimension $n = 250$ and the eigenvalues of A as

$$\lambda_i = 1 + \cos\frac{(2i-1)\pi}{2n}, \qquad i = 1, 2, \cdots, n,$$

that is, the eigenvalues of A are shifted zeros of the Chebyshev polynomial T_n and $\lambda_1 = 1 + \cos(\pi/(2n)) \simeq 2$. (The shift by 1 is needed to guarantee that A is positive definite.)

We have performed numerical tests for this matrix with 30 pseudorandom vectors b uniformly distributed over the unit sphere of $\mathbf{R}^n$. To get such a distribution we used the fact that if $X = (X_1, X_2, \cdots, X_n)$ is a random variable whose components are independent random variables with a normal distribution $N(0, 1)$, then $X/\|X\|$ is uniformly distributed over the unit sphere (see Knuth [7, p. 116]). The normal distribution was in turn generated from the uniform distribution over (0, 1) using the formula $Z = (-2 \ln R_1)^{1/2} \cos 2\pi R_2$, where R_1 and R_2 are independent random variables uniformly distributed over (0, 1) (see Box and Muller [1]). The variables R_i were produced using a number generator similar to the one used for testing EISPACK procedures (see Smith et al. [18]).

For each pseudorandom vector b we performed the Lanczos algorithm for $k = 1, 2, \cdots, k^*$, where k^* was chosen as the minimal k for which the relative error $(\lambda_1 - \xi^{\text{Lan}}(A, b, k))/\lambda_1$ was no greater than ε. For some tests k^* was around 150. We compared the relative error with k^{-2}. For all tested b and k, we obtained

$$0.1241 \leq \frac{\lambda_1 - \xi^{\text{Lan}}(A, b, k)}{\lambda_1} k^2 \leq 1.62.$$

In fact, in most cases, $(\lambda_1 - \xi^{\text{Lan}}(A, b, k))/\lambda_1 k^2$ was between 0.286 and 1.25.

In Table 6.1 we report the average errors achieved after $k - 1$ steps of the Lanczos algorithm for ten different values of k which are listed in the first column. The second column contains the average errors defined as

$$\xi^{\text{ave}} = \frac{1}{30}\sum_{i=1}^{30} \frac{\lambda_1 - \xi^{\text{Lan}}(A, b_i, k)}{\lambda_1},$$

where b_i is the ith pseudorandom vector. The third column presents upper bounds on the Lanczos errors from Theorem 3.2, i.e.,

$$\varepsilon^{\text{up}} = 0.103\left(\frac{\ln(n(k-1)^4)}{k-1}\right)^2.$$

TABLE 6.1

$k-1$	ε^{ave}	ε^{up}	r_1	r_2	$\varepsilon^{\min}$	$\varepsilon^{\max}$
10	0.004830	0.2235	46.273	2.070	0.003782	0.006868
20	0.001333	0.0789	59.185	1.875	0.000853	0.002109
30	0.000623	0.0419	67.288	1.786	0.000329	0.001454
40	0.000360	0.0265	73.632	1.739	0.000145	0.000943
50	0.000226	0.0185	81.714	1.770	0.000080	0.000463
60	0.000156	0.0137	87.596	1.773	0.000051	0.000295
70	0.000115	0.0107	92.897	1.779	0.000040	0.000230
80	0.000090	0.0086	95.875	1.752	0.000034	0.000182
90	0.000072	0.0070	97.765	1.716	0.000028	0.000159
100	0.000061	0.0059	97.360	1.649	0.000028	0.000132

We compute the ratios between the observed errors and their upper bounds in the fourth column, $r_1 = \varepsilon^{\text{up}}/\varepsilon^{\text{ave}}$. The fifth column displays how r_1 is related to the possibly unnecessary factor in the theoretical bound,

$$r_2 = 0.103 r_1 / \ln^2 (n(k-1)^4).$$

The last two columns show the minimal $\varepsilon^{\min}$ and maximal $\varepsilon^{\max}$ relative errors over the set of initial vectors at each step $k-1$, respectively.

The fifth column of the table seems to suggest that the error of the Lanczos algorithm for the matrix with Chebyshevian distribution of eigenvalues behaves like k^{-2} and the factor $0.103 \ln^2 (n(k-1)^4)$ is probably an overestimate of the upper bound.

In Table 6.2 we indicate how many steps were needed to achieve relative error no greater than ε for six different values of ε. The values of ε are displayed in the first row of the table. The second row of the table shows the average number k^{ave} of performed steps with $k^{\text{ave}} = \sum_{i=1}^{30} k(A, b_i)/30$, where $k(A, b_i)$ was the number of steps needed for the pseudorandom vector b_i. The third row gives the minimal $k = k^{\text{up}}$ such that

$$0.103\left(\frac{\ln (n(k-1)^4)}{k-1}\right)^2 \leqq \varepsilon,$$

which is one of the two theoretical bounds for the Lanczos algorithm (see Theorem 3.2(a)). The fourth row presents the ratios between these two numbers, $r = k^{\text{up}}/k^{\text{ave}}$.

As we see, the theoretical bound exceeds the actual value by a factor of at most 15. This indicates once more that the factor $\ln^2 (n(k-1)^4)$ may be an overestimate in the theoretical bound. Observe also that all k^{up}'s are greater than the dimension $n = 250$ and the second bound of Theorem 3.2(a) gives a better estimate.

We complete this section by reporting an interesting property of the computed sequences $\xi_k = \xi^{\text{Lan}}(A, b, k)$ of the Lanczos algorithm. In some cases they have a "misconvergence" phenomenon (see Parlett, Simon, and Stringer [15]), that is, before reaching the largest eigenvalue λ_1, the sequence ξ_k remained constant (to a machine accuracy)

TABLE 6.2

ε	$5.0_{10}-4$	$2.5_{10}-4$	$2.0_{10}-4$	$1.5_{10}-4$	$1.0_{10}-4$	$5.0_{10}-5$
k^{ave}	35.27	48.03	53.13	62.27	76.33	110.67
k^{up}	428	638	724	853	1075	1590
r	12.13	13.28	13.63	13.70	14.08	14.38

TABLE 6.3

ε	$2.5_{10}-4$	$2_{10}-4$	$1.5_{10}-4$	$1.0_{10}-4$	$5.0_{10}-5$
p	0	6.67	46.67	80	100

for some consecutive steps, $\xi_k = \xi_{k+1} = \cdots = \xi_{k+t}$ and the value of t was sometimes quite large. The misconvergence phenomenon occurred when the sequence ξ_k approached the second largest eigenvalue λ_2 and sometimes even when ξ_k passed the third largest eigenvalue λ_3. For instance, for some vectors b the sequence ξ_k stabilized close to λ_2 for 28 consecutive steps. Table 6.3 shows the percentage (p) of the vectors b for which the misconvergence phenomenon occurred before the relative error reached ε.

7. Remarks.

Remark 7.1. As we know from § 2 it is impossible to compute an ε-approximation to the largest eigenvalue by algorithms using Krylov information with a deterministically chosen vector b. We may interpret this by saying that Krylov information is poor and hope that more general information may lead to a positive result. Indeed, using matrix-vector multiplications, we may compute $[Az_1, Az_2, \cdots, Az_k]$, where $z_1 = b$ and z_i can be an arbitrary function of the already computed $Az_1, Az_2, \cdots, Az_{i-1}$. Is it possible to define vectors z_i such that $\phi(Az_1, Az_2, \cdots, Az_k)$, for some ϕ, yields an ε-approximation to the largest eigenvalue of any symmetric positive definite matrix A? The answer is still *no* as long as $k \leqq n-1$ (see Traub, Wasilkowski, and Woźniakowski [19, pp. 183–186] for this and related results). Thus Krylov information as well as any other deterministic information with $k \leqq n-1$ does not supply enough knowledge of A to compute an ε-approximation to the largest eigenvalue.

On the other hand, if we are willing to settle for an ε-approximation to any eigenvalue, which is not necessarily the largest, then it can be done by using $\min\{n, \lceil \varepsilon^{-1} \rceil\}$ matrix-vector multiplications. This can be achieved by using deterministic Krylov information and the generalized minimal residual algorithm (see Kuczyński [8]). The number $\min\{n, \lceil \varepsilon^{-1} \rceil\}$ is within a factor of at most 2 of being minimal, as shown by Chou [2], whose analysis is based on Nemirovsky and Yudin [9].

Remark 7.2. As before, B_n is the unit ball of $\mathbf{R}^n$. Let $f: B_n \to \mathbf{R}$ be a measurable function such that f does not depend on the norm of b, $f(b) = f(\alpha b)$ for all $\alpha > 0$, and f does not depend on signs of b_i, $f(s_1 b_1, s_2 b_2, \cdots, s_n b_n) = f(b_1, b_2, \cdots, b_n)$ for all $s_i \in \{-1, 1\}$. The error of the power or Lanczos algorithm as a function of b satisfies these properties.

For such functions f, the average value of f over the unit sphere is the same as the average value over the unit ball, i.e.,

$$\int_{\|b\|=1} f(b)\mu(db) = \frac{1}{c_n} \int_{B_n} f(b)\, db,$$

where c_n is the measure of the unit ball in $\mathbf{R}^n$.

Indeed, using the polar coordinates $b = \phi(t) = [\phi_1(t), \cdots, \phi_n(t)]$ with $t = [r, t_1, \cdots, t_{n-1}] \in [0, 1] \times [0, \pi]^{n-1}$ and

$$\phi_1(t) = r \cos t_1 \cos t_2 \cdots \cos t_{n-1},$$

$$\phi_{i+1} = r \sin t_i \cos t_{i+1} \cdots \cos t_{n-1}, \qquad i = 1, 2, \cdots, n-1,$$

we have $|\det(\phi')| = r^{n-1}|\cos t_2 \cos^2 t_3 \cdots \cos^{n-2} t_{n-1}| = r^{n-1}g(t)$ and

$$ac_n := \int_{B_n} f(b)\,db = \int_{[0,1]\times[0,\pi]^{n-1}} f(\phi(t))r^{n-1}g(t)\,dr\,dt_{(n)},$$

where $dt_{(n)}$ stands for $dt_1 \cdots dt_{n-1}$. Since $f(\phi(t))$ does not depend on r, we can integrate over r to get

$$a = \frac{1}{nc_n}\int_{[0,\pi]^{n-1}} f(\phi(t))g(t)\,dt_{(n)}.$$

Change the variables once more by setting $b_i = \phi_i(t)/r$ for $i = 2, 3, \cdots, n$. Then for

$$b_1 = \sqrt{1 - \sum_{i=2}^{n} b_i^2}$$

we have

$$f(\phi(t)) = f(\phi(t)/r) = f(\pm b_1, b_2, \cdots, b_n) = f(b_1, b_2, \cdots, b_n)$$

and

$$db_{(n)} = db_2 \cdots db_n = |\cos t_1 \cos^2 t_2 \cdots \cos^{n-1} t_{n-1}|\,dt_{(n)}.$$

Therefore, $g(t)dt_{(n)} = |\cos t_1 \cdots \cos t_{n-1}|^{-1} db_{(n)} = (1 - \sum_{i=2}^{n} b_i^2)^{-1/2} db_{(n)}$ and

$$a = \frac{2}{nc_n}\int_{b_2^2+\cdots+b_n^2 \le 1} \frac{f(\sqrt{1 - \sum_{i=2}^{n} b_i^2}, b_2, \cdots, b_n)}{\sqrt{1 - \sum_{i=2}^{n} b_i^2}}\,db_{(n)} = \int_{\|b\|=1} f(b)\mu(db),$$

as claimed

Remark 7.3. The modified power algorithm is defined by

$$\xi^{\text{mpow}}(A, b, k) = (A^k b, b)^{1/k}, \qquad \|b\| = 1.$$

We show that

$$\sup_{A=A^T>0} e^{\text{avg}}(\xi^{\text{mpow}}, A, b) = e^{\text{avg}}(\xi^{\text{mpow}}, A^*, k) = 1 - \frac{\Gamma(n/2)\Gamma(0.5 + 1/k)}{\Gamma(n/2 + 1/k)\Gamma(0.5)} \simeq \frac{\ln n}{k},$$

where A^* is a symmetric matrix with eigenvalues $\lambda_1 > 0$, and $\lambda_i = 0$ for $i \ge 2$.

Indeed, for any $A = A^T > 0$ with $x_i = \lambda_i/\lambda_1$, we have

$$e^{\text{avg}}(\xi^{\text{mpow}}, A, k) = 1 - \int_{\|b\|=1} \left(\sum_{i=1}^{n} b_i^2 x_i^k\right)^{1/k} \mu(db) \le e^{\text{avg}}(\xi^{\text{mpow}}, A^*, k)$$

$$= 1 - \int_{\|b\|=1} b_1^{2/k}\mu(db)$$

$$= 1 - \frac{2}{nc_n}\int_{B_{n-1}} (1 - b_2^2 - \cdots - b_n^2)^{(1/k)-(1/2)}\,db_2 \cdots db_n$$

$$= 1 - 2\alpha\int_0^1 t^{n-2}(1-t^2)^{(1/k)-(1/2)}\,dt = 1 - \alpha B\left(\frac{n-1}{2}, \frac{1}{k} + \frac{1}{2}\right)$$

$$= 1 - \frac{\Gamma(n/2)\Gamma(0.5 + 1/k)}{\Gamma(n/2 + 1/k)\Gamma(0.5)},$$

where $\alpha = (n-1)c_{n-1}/(nc_n)$. For large k and any a we have

$$\frac{\Gamma(a+1/k)}{\Gamma(a)} = 1 + \frac{\Gamma'(a)}{\Gamma(a)}\frac{1}{k}(1+o(1)).$$

For $a = n/2$ and $a = \frac{1}{2}$ we have from Gradshteyn and Ryzhik [4, 8.360, 8.362, and 8.366]

$$\frac{\Gamma'(n/2)}{\Gamma(n/2)} \simeq \ln n/2, \qquad \frac{\Gamma'(1/2)}{\Gamma(1/2)} = -C - 2\ln 2 = -1.9365\cdots,$$

where C is the Euler constant. This implies the error behavior $\ln(n)/k$, as claimed.

Comparing this bound with parts (a) and (b) of Theorem 3.1, we see that the power algorithm has an error bound roughly 1.8 times smaller.

We can also compare the algorithms ξ^{pow} and ξ^{mpow} asymptotically. Assume for simplicity that the largest eigenvalue is of multiplicity $p = 1$. Then part (c) of Theorem 3.1 yields that the rate of convergence of the power algorithm is exponential and proportional to $(\lambda_2/\lambda_1)^{k-1}$. For the modified power algorithm, it is easy to show that the rate is only linear and roughly equal to $\ln(n)/k$. Thus the power algorithm is far superior asymptotically in the average case to the modified power algorithm.

We now turn to the probabilistic case. The modified power algorithm was analyzed in this case by Dixon [3], who proved that

$$\sup_{A=A^T>0} f^{\text{prob}}(\xi^{\text{mpow}}, A, k, \varepsilon) = f^{\text{prob}}(\xi^{\text{mpow}}, A^*, k, \varepsilon) \leqq 0.8\sqrt{n}(1-\varepsilon)^{k/2}.$$

For large n and k we have

$$f^{\text{prob}}(\xi^{\text{mpow}}, A^*, k, \varepsilon) = \sqrt{2/\pi}\sqrt{n}\,(1-\varepsilon)^{k/2}(1+o(1))$$

and $\sqrt{2/\pi} = 0.7978\cdots$.

This should be compared with the power algorithm, whose rate of convergence is roughly the square of the rate of the modified power algorithm.

It is easy to check that the asymptotic behavior of ξ^{mpow} does not depend on the distribution of eigenvalues but on the multiplicity p of the largest eigenvalue

$$f^{\text{prob}}(\xi^{\text{mpow}}, A, k, \varepsilon) = \frac{n-p}{n}\frac{\Gamma(1+n/2)}{\Gamma(1+p/2)\Gamma(1+(n-p)/2)}(1-\varepsilon)^{pk/2}(1+o(1)).$$

For the power algorithm with $\lambda_{p+1}/\lambda_1 < 1-\varepsilon$, the asymptotic rate of convergence is proportional to $(\lambda_{p+1}/\lambda_1)^{p(k-1)}$, which obviously tends to 0 faster.

Remark 7.4. For β close to 1, it is easy to find the exact value $w(\beta)$; see (17). Namely, we have

$$w(\beta) = \frac{1}{k^2}\frac{\sin^2(\pi/(2k))}{(1+\cos(\pi/(2k)))^2} \simeq \frac{\pi^2}{16k^4}$$

for $\beta \in [\cos^2(\pi/(2k))/\cos^2(\pi/(4k)), 1]$.

Indeed, let $\zeta_k = \cos(\pi/(2k))$ denote the largest zero of T_k. Take

$$Q(x) = \frac{1}{x-1}\frac{T_k((\zeta_k+1)x-1)}{(\zeta_k+1)T_k'(\zeta_k)}.$$

Note that Q is a polynomial of degree $\leqq k-1$ and $Q(1) = 1$. For $i = 1, 2, \cdots, k$, let $x_i = (1+\cos(i\pi/k))/(1+\zeta_k) = \cos^2(i\pi/(2k))/\cos^2(\pi/(4k))$. Then $x_i \in [0, \beta]$ and

$$(x_i - 1)Q(x_i) = \frac{T_k(\cos(i\pi/k))}{(\zeta_k+1)T_k'(\zeta_k)} = \frac{(-1)^i}{1+\cos(\pi/(2k))}\frac{\sin(\pi/(2k))}{k}.$$

Suppose there exists $P \in \mathscr{P}_k(1)$ such that

$$\max_{x\in[0,\beta]} P^2(x)(1-x^2) < \max_{x\in[0,\beta]} Q^2(x)(1-x)^2.$$

Then $h(x) = (1-x)(Q(x) - P(x))$ has a double zero at one. Since sign $h(x_i) = (-1)^i$, h has at least $k-1$ zeros in $[0, \beta)$. Thus $h \equiv 0$, which is a contradiction. Hence,

$$w(\beta) = \max_{x\in[0,\beta]} Q^2(x)(1-x)^2 = \frac{\sin^2(\pi/(2k))}{(1+\cos(\pi/(2k)))^2}\frac{1}{k^2},$$

as claimed.

Remark 7.5. We now consider an error relative to spread (see Parlett [14]) instead of the relative error as the error criterion, that is, we wish to compute ξ such that

$$|\lambda_1(A) - \xi| \leqq \varepsilon(\lambda_1(A) - \lambda_n(A)),$$

where, as before, $\lambda_1(A)$ and $\lambda_n(A)$ denote the largest and the smallest eigenvalues of A.

This error is a natural error criterion for the Lanczos algorithm since $\xi^{\mathrm{Lan}}(A + \alpha I, b, k) = \xi^{\mathrm{Lan}}(A, b, k) + \alpha$ and

$$\frac{\lambda_1(A+\alpha I) - \xi^{\mathrm{Lan}}(A+\alpha I, b, k)}{\lambda_1(A+\alpha I) - \lambda_n(A+\alpha I)} = \frac{\lambda_1(A) - \xi^{\mathrm{Lan}}(A, b, k)}{\lambda_1(A) - \lambda_n(A)}.$$

Thus, for the Lanczos algorithm the error relative to spread is shift invariant.

It is easy to see that the bounds for the Lanczos algorithm presented in Theorems 3.2 and 4.2 also hold for the error relative to spread. This follows by noting that

$$\frac{\lambda_1(A) - \xi^{\mathrm{Lan}}(A, b, k)}{\lambda_1(A) - \lambda_n(A)} = \frac{\lambda_1(B) - \xi^{\mathrm{Lan}}(B, b, k)}{\lambda_1(B)},$$

where $B = A - \lambda_n I$ and $B = B^* \geqq 0$.

Although B is not positive definite, a continuity argument yields that we can use estimates of Theorems 3.2 and 4.2 for the matrix B. Part (a) of both theorems will give estimates independent of eigenvalue distributions of B (or A). Part (b) of both theorems presents estimates that are shift invariant and therefore are the same for the matrix B as for the matrix A. Observe also that for the error relative to spread we need only to assume that A is symmetric but not necessarily positive definite.

For the power algorithm, we have different results since, in general, $\xi^{\mathrm{pow}}(A + \alpha I, b, k) \neq \xi^{\mathrm{pow}}(A, b, k) + \alpha$. To derive bounds for the power algorithm under the error relative to spread, consider the average case and the matrix A from part (b) of Theorem 3.1, that is, A has exactly two distinct eigenvalues λ_1 and $\lambda_n = \lambda_1(1 - \ln(n/\ln n)/(2(k-1)))$. Then the estimate of Theorem 3.2(b) yields for large k and n,

$$\int_{\|b\|=1} \frac{\lambda_1 - \xi^{\mathrm{pow}}(A, b, k)}{\lambda_1 - \lambda_n}\,\mu(db) = \frac{e^{\mathrm{avg}}(\xi^{\mathrm{pow}}, A, k)}{1 - \lambda_n/\lambda_1} = 1 + o(1).$$

Thus, no matter how many matrix-vector multiplications are performed, there exists a matrix A for which the average error of the power algorithm under the error relative to spread is about 1.

Similarly we can check that in the probabilistic case, the failure of the power algorithm under the error relative to spread for the matrix A with the two distinct eigenvalues λ_1 and $\lambda_1(1 - 1/(2k-1))$ is equal to $1 + o(1)$.

Obviously, the asymptotic bounds for the power algorithm under the error relative to spread can be easily obtained from part (c) of Theorems 3.1 and 4.1. For the average case, the only difference is to multiply the asymptotic constants by $1 - \lambda_n/\lambda_1$, whereas for the probabilistic case, ε should be replaced by $\varepsilon(1 - \lambda_n/\lambda_1)$.

Acknowledgments. We appreciate comments from J. F. Traub, G. W. Wasilkowski, and A. G. Werschulz.

Our special thanks are to B. N. Parlett for raising the issue of using error relative to spread and for many valuable suggestions.

We also thank the anonymous referee for carefully checking the paper and numerical tests.

REFERENCES

[1] G. E. P. BOX AND M. E. MULLER, *A note on the generation of random normal deviates*, Ann. Math. Statist., 29 (1958), pp. 610–611.

[2] A. W. CHOW, *On the optimality of Krylov information*, J. Complexity, 3 (1987), pp. 26–40.

[3] J. D. DIXON, *Estimating extremal eigenvalues and condition numbers of matrices*, SIAM J. Numer. Anal., 20 (1983), pp. 812–814.

[4] I. S. GRADSHTEYN AND I. W. RYZHIK, *Table of Integrals, Series, and Products*, Academic Press, New York, 1980.

[5] W. KAHAN AND B. N. PARLETT, *How far should we go with the Lanczos process?* In Sparse Matrix Computations, J. Bunch and D. Rose, eds., Academic Press, New York, 1976, pp. 131–144.

[6] S. KANIEL, *Estimates for some computational techniques in linear algebra*, Math. Comp., 20 (1966), pp. 369–378.

[7] D. E. KNUTH, *The Art of Computer Science, Vol. 2: Seminumerical Algorithms*, Second Edition, Addison-Wesley, Reading, MA, 1981.

[8] J. KUCZYŃSKI, *On the optimal solution of large eigenpair problems*, J. Complexity, 2 (1986), pp. 131–162.

[9] A. S. NEMIROVSKY AND D. B. YUDIN, *Problem Complexity and Method Efficiency in Optimization*, Wiley-Interscience, New York, 1983.

[10] D. P. O'LEARY, G. W. STEWART, AND J. S. VANDERGRAFT, *Estimating the largest eigenvalue of a positive definite matrix*, Math. Comp., 33 (1979), pp. 1289–1292.

[11] C. C. PAIGE, *The computation of eigenvalues and eigenvectors of very large sparse matrices*, Ph.D. thesis, University of London, London, U.K., 1971.

[12] ———, *Computational variants of the Lanczos method for the eigenproblem*, IMA J. Appl. Math., 10 (1972), pp. 373–381.

[13] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[14] ———, private communication, 1989.

[15] B. N. PARLETT, H. SIMON, AND L. M. STRINGER, *On estimating the largest eigenvalue with the Lanczos algorithm*, Math. Comp., 28 (1982), pp. 153–165.

[16] Y. SAAD, *On the rates of convergence of the Lanczos and the block Lanczos methods*, SIAM J. Numer. Anal., 17 (1980), pp. 687–706.

[17] D. S. SCOTT, *Analysis of the symmetric Lanczos process*, Ph.D. thesis and Memorandum NCB/ERLM 78/40, University of California, Berkeley, CA, 1978.

[18] B. T. SMITH, J. M. BOYLE, B. S. GARBOV, Y. IKEBE, V. C. KLEMA, AND C. B. MOLER, *Matrix Eigensystem Routines–EISPACK Guide*, Springer-Verlag, Berlin, 1974.

[19] J. F. TRAUB, G. W. WASILKOWSKI, AND H. WOŹNIAKOWSKI, *Information-Based Complexity*, Academic Press, New York, 1988.

[20] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Oxford University Press, London, New York, 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1123–1130, October 1992

008

EXPLICIT SOLUTIONS OF THE MATRIX EQUATION $\sum A^iXD_i = C^*$

HARALD K. WIMMER†

Abstract. A module theoretic approach is developed to study the linear matrix equation $\sum A^iXD_i = C$. If the solution X is unique it can be expressed in the form $X = \sum A^kCG_k$. The matrices G_k can be determined from an auxiliary equation which involves the companion matrix of the characteristic polynomial of A. A connection is made with realizations of the inverse of the associated polynomial matrix $D(x) = \sum D_ix^i$. The more general equation $\sum A_iXD_i \doteq C$ is discussed under the assumption that the matrices A_i are pairwise commutative.

Key words. linear matrix equations, Sylvester equations, realizations, simultaneous triangular form

AMS(MOS) subject classification. 15A24

1. Introduction. Linear matrix equations have been studied since the time of Sylvester [10]. They are important in numerous applications (see, e.g., [5] with the regulator problem as a more recent example). An algebraic approach that originated with Kalman [7], [8] was elaborated by Djaferis and Mitter [1] to yield explicit solutions of equations of the form $\sum f_{ik}A^iXB^k = C$ (see also [12] and [13]). In this note, we adapt the module theoretic point of view of those papers and consider the more general equation

$$\sum_{i=0}^{p-1} A^iXD_i = C. \tag{1.1}$$

We first review some known facts concerning the equation

$$\sum_{i=0}^{p-1}\sum_{k=0}^{q-1} f_{ik}A^iXB^k = C, \tag{1.2}$$

which should direct us towards corresponding results on (1.1). All matrices in (1.1) and (1.2) are assumed to be complex, in particular, $A \in \mathbb{C}^{p\times p}$, B, $D_i \in \mathbb{C}^{q\times q}$, $C \in \mathbb{C}^{p\times q}$. Furthermore, the polynomial

$$f(x, y) = \sum\sum f_{ik}x^iy^k$$

is in $\mathbb{C}[x, y]$. For the results of this section, we refer to [1], [12], and [15].

Let $\lambda_1, \cdots, \lambda_p$ be the eigenvalue of A and let

$$a(x) = a_0 + \cdots + a_{p-1}x^{p-1} + x^p$$

be the characteristic polynomial of A. Define a companion matrix of $a(x)$ by

$$F_a = \begin{pmatrix} 0 & & -a_0 \\ 1 & \ddots & \vdots \\ & 1 & -a_{p-1} \end{pmatrix}.$$

Let B have eigenvalues $\mu_1, \cdots, \mu_q$ and characteristic polynomial $b(x)$. The ideal in $\mathbb{C}[x, y]$ generated by $a(x)$ and $b(y)$ will be denoted by Ψ, i.e.,

$$\Psi = (a(x), b(y)).$$

* Received by the editors October 31, 1988; accepted for publication (in revised form) December 13, 1990.
† Mathematisches Institut, Universität Würzburg, D-8700 Würzburg, Germany.

Equation (1.2) is called *universally solvable* if it has a solution for every C (which is necessarily unique).

For $h = \sum h_{ik}x^i y^k \in \mathbb{C}[x, y]$ and $M \in \mathbb{C}^{p\times q}$, let a product $h * M$ be defined by

$$h * M = \sum_{i,k} h_{ik}A^i M B^k.$$

The Cayley–Hamilton theorem implies that $d * M = 0$ for all $d \in \Psi$. Hence it is meaningful to define

$$(H + \Psi) * M = h * M$$

such that $\mathbb{C}^{p\times q}$ becomes a module over the ring $\mathbb{C}[x, y]/\Psi$. Then (1.2) can be written as $(f + \Psi) * X = C$.

THEOREM 1.1. *The following statements are equivalent*:

(i) *Equation* (1.2) *is universally solvable.*

(ii) *The following holds*:

$$f(\lambda_i, \mu_j) \neq 0 \quad \text{for } i = 1, \cdots, p, \qquad j = 1, \cdots, q. \tag{1.3}$$

(iii) *The element* $f + \Psi$ *is a unit of the ring* $\mathbb{C}[x, y]/\Psi$.

(iv) *There exists a unique polynomial* $g \in \mathbb{C}[x, y]$ *such that*

$$fg \equiv 1 (\mathrm{mod}\ \Psi) \tag{1.4}$$

and the degree of g *in* x *is less than* p *and the degree in* y *is less than* q.

(v) *The equation*

$$\sum_{i=0}^{p-1}\sum_{k=0}^{q-1} f_{ik}F_a^i Y (F_b^T)^k = (1, 0, \cdots, 0)_{1\times p}^T (1, 0, \cdots, 0)_{q\times 1} \tag{1.5}$$

is consistent.

Under condition (iii) above, the solution X can be expressed as $X = (f + \Psi)^{-1} * C$ and we have the following representation of X.

THEOREM 1.2. *If* (1.2) *is universally solvable and* $g(x, y) = \sum g_{rs}x^r y^s$ *is the polynomial given by* (1.4), *then*

$$X = \sum_{r=0}^{p-1}\sum_{s=0}^{q-1} g_{rs}A^r C B^s \tag{1.6}$$

is the solution of (1.2).

The polynomial g in (1.4) can be obtained from the auxiliary equation (1.5).

THEOREM 1.3. *Let* $Y = G \in \mathbb{C}^{p\times q}$ *be the solution of* (1.5). *Then*

$$g(x, y) = (1, x, \cdots, x^{p-1})G(1, y, \cdots, y^{p-1})^T$$

is the polynomial with the properties of (iv) *in Theorem* 1.1.

The representation of X in (1.6) yields a bound for the rank of X. For $L \in \mathbb{C}^{p\times r}$ and $R \in \mathbb{C}^{r\times q}$, put

$$K(A, L) = (L, AL, \cdots, A^{p-1}L) \tag{1.7}$$

and

$$D(B, R) = \begin{pmatrix} R \\ RB \\ \vdots \\ RB^{q-1} \end{pmatrix}.$$

THEOREM 1.4. *If* (1.2) *is universally solvable and if* C *is factorized as* $C = LR$, *then*

$$\operatorname{rank} X \leqq \min \{\operatorname{rank} K(A, L), \operatorname{rank} D(B, R)\}.$$

2. An algebraic approach to explicit solutions. We now consider our target equation

$$\sum_{i=0}^{p-1} A^i X D_i = C. \tag{1.1}$$

Whether (1.1) is universally solvable or not depends on the eigenvalues λ_i of A and on a polynomial matrix $D \in \mathbb{C}^{q\times q}[x]$,

$$D(x) = \sum_{i=0}^{p-1} D_i x^i.$$

THEOREM 2.1 (see [2], [4], or [14]). *Equation* (1.1) *is solvable for all* C *if and only if*

$$\det D(\lambda_i) \neq 0 \quad \textit{for } i = 1, \cdots, p. \tag{2.1}$$

The left-hand side of (1.1) will be regarded as a product of X and D. For that purpose, let us define for $F = \sum F_i x^i \in \mathbb{C}^{q\times q}[x]$ and for $M \in \mathbb{C}^{p\times q}$ the operation

$$M * F = \sum A^i M F_i. \tag{2.2}$$

Then $\mathbb{C}^{p\times q}$ becomes a right module over the ring $\mathbb{C}^{q\times q}[x]$. Recall that $a(x)$ is the characteristic polynomial of A and put

$$\Phi = a(x)\mathbb{C}^{q\times q}[x]$$

such that Φ is the ideal in $\mathbb{C}^{q\times q}[x]$ generated by $a(x)I$. By the Cayley–Hamilton theorem, we have $M * H = 0$ for all $H \in \Phi$. Hence it makes sense to define

$$M * (F + \Phi) = M * F, \tag{2.3}$$

and it is easy to verify that $\mathbb{C}^{p\times q}$ together with the operation (2.3) is a unitary right module over the ring $\mathbb{C}^{q\times q}[x]/\Phi$. For a nonzero polynomial matrix $H = \sum_{i=0}^{m} H_i x^i$ with $H_m \neq 0$ we put $\deg H = m$. In the case $H = 0$ we set $\deg H = -\infty$. Obviously, in each coset $F + \Phi$ there is a unique representative of degree less than p, $p = \deg a$.

Let us write (1.1) as

$$X * (D + \Phi) = C. \tag{2.4}$$

It is clear that (2.4) can be solved if $D + \Phi$ is invertible in $\mathbb{C}^{q\times q}[x]/\Phi$.

THEOREM 2.2. *The following statements* (i) *and* (ii) *are equivalent to condition* (2.1) *for the universal solvability of* (1.1):

(i) $D + \Phi$ *is a unit in the ring* $\mathbb{C}^{q\times q}[x]/\Phi$.

(ii) *There exists a unique matrix* $G \in \mathbb{C}^{q\times q}[x]$ *such that*

$$DG = GD \equiv I (\operatorname{mod} \Phi), \qquad \deg G < p. \tag{2.5}$$

Proof. We assume first that (2.1) holds. Then $d(x) = \det D$ and $a(x)$ have no zeros in common. Hence we have $a\gamma + d\delta = 1$ for some polynomials γ and δ. If $Q \in \mathbb{C}^{q\times q}[x]$ is the adjoint matrix of D, then $QD = DQ = dI$. Division of δQ by aI yields polynomial matrices B and G such that $\delta Q = aB + G$, $\deg G < \deg a = p$. From $\delta QD = D\delta Q = I - a\gamma I$ we see that G satisfies (2.5). The condition $\det G < p$ together with (2.5) assures that the matrix G in (2.5) is uniquely determined. Obviously, (ii) implies (i). Finally, if

(i) holds, then (1.1) is solvable and

$$X = C*(D + \Phi)^{-1} \tag{2.6}$$

is the unique solution. □

From (2.6) we obtain the solution X in the following closed form.

THEOREM 2.3. *If* (1.1) *is universally solvable and*

$$G(x) = \sum_{i=0}^{p-1} G_i x^i \tag{2.7}$$

is the polynomial given by (2.5), *then*

$$X = \sum_{i=0}^{p-1} A^i C G_i \tag{2.8}$$

is the unique solution of (1.1).

The coefficient matrices G_i of (2.7) can be determined from an auxiliary equation which involves the companion matrix F_a of $a(x)$. For two matrices, $P = (p_{ij})$ and S, the Kronecker product is defined as $P \otimes S = (p_{ij}S)$. The identity matrix in (2.9) and (2.10) below is of size $q \times q$.

THEOREM 2.4. (i) *Let* $G = \sum_{i=0}^{p-1} G_i x^i$ *be given and put*

$$\tilde{G} = \begin{pmatrix} G_0 \\ G_1 \\ \vdots \\ G_{p-1} \end{pmatrix}, \qquad E = \begin{pmatrix} I \\ 0 \\ \vdots \\ 0 \end{pmatrix}. \tag{2.9}$$

Then G *satisfies* $DG = GD \equiv I (\mathrm{mod}\ \Phi)$ *if and only if*

$$\sum_{i=0}^{p-1} (F_a \otimes I)^i \tilde{G} D_i = E \tag{2.10}$$

holds.

(ii) *The equation*

$$\sum_{i=0}^{p-1} (F_a \otimes I)^i Y D_i = E$$

is consistent if and only if (2.1) *holds.*

Proof. (i) Because of $\deg a = p$, there exists a unique matrix $U = (I, Ix, \cdots, Ix^{p-1})\tilde{U} \in \mathbb{C}^{q \times q}[x]$, which satisfies $GD \equiv U (\mathrm{mod}\ \Phi)$. We want to show that $\tilde{U}$ is the matrix on the left-hand side of (2.10). Put $\tilde{x} = (1, x, \cdots, x^{p-1})$. From $x^p = -a_0 - \cdots - a_{p-1}x^{p-1} + a(x)$, we obtain

$$\tilde{x}x = (x, \cdots, x^p) = \tilde{x}F_a + a(x)(0, \cdots, 0, 1)$$

and $\tilde{x}x^i = \tilde{x}F_a^i + av_i$ for some $v_i \in \mathbb{C}^{1 \times p}[x]$. Because of $G = (\tilde{x} \otimes I)\tilde{G}$, we have

$$GD = \sum_{i=0}^{p-1} (\tilde{x} \otimes I)x^i \tilde{G} D_i = \sum_{i=0}^{p-1} [(\tilde{x}F_a^i + av_i) \otimes I]\tilde{G} D_i$$

$$= (\tilde{x} \otimes I) \sum_{i=0}^{p-1} (F_a \otimes I)^i \tilde{G} D_i + a \sum_{i=0}^{p-1} (v_i \otimes I)\tilde{G} D_i.$$

Hence

$$\tilde{U} = \sum_{i=0}^{p-1} (F_a \otimes I)^i \tilde{G} D_i .$$

Clearly, $I = (\tilde{x} \otimes I)E$ and $U = (\tilde{x} \otimes I)\tilde{U}$. Therefore, $U = I$ is equivalent to (2.10). Theorem 2.4(ii) follows immediately from Theorem 2.2. □

We consider now a particular case where $D(x)$ has only one invariant factor (different from 1). Put $e = (1, 0, \cdots, 0)^T$, $e \in \mathbb{C}^p$.

THEOREM 2.5. *Assume that there exists a vector* $v \in \mathbb{C}^{1\times q}$ *such that*

$$\operatorname{rank} \begin{pmatrix} D(z) \\ v \end{pmatrix} = q \quad \text{for all } z \in \mathbb{C}. \tag{2.11}$$

Then the equation

$$\sum_{i=0}^{p-1} F_a^i Z D_i = ev \tag{2.12}$$

is consistent if and only if

$$\det D(\lambda_i) \neq 0 \tag{2.1'}$$

for all eigenvalues λ_i *of* F_a.

Proof. Clearly, (2.1) implies the solvability of (2.12). Suppose now that $\det D(\lambda) = 0$ for some eigenvalue λ of F_a and that (2.12) has a solution Z. Then there exists a vector $w = (w_1, \cdots, w_q)^T$, $w \neq 0$, such that $D(\lambda)w = 0$ and a vector $u = (u_1, \cdots, u_p)$, $u \neq 0$, such that

$$uF_a = \lambda u. \tag{2.13}$$

From

$$\begin{pmatrix} D(\lambda) \\ v \end{pmatrix} w = \begin{pmatrix} 0 \\ vw \end{pmatrix}$$

and (2.11) follows $vw \neq 0$, and (2.13) yields $u_1 \neq 0$. We multiply both sides of (2.12) by u and w. On the left-hand side we obtain

$$u(\textstyle\sum F_a^i Z D_i)w = uZ(\sum \lambda^i D_i)w = uZD(\lambda)w = 0,$$

whereas on the right-hand side we have $u_1(vw) \neq 0$, which is a contradiction. Hence under the assumption (2.11) the condition (2.1) is also necessary for the consistency of (2.12). □

To estimate the rank of the solution X of (1.1), we use the representation (2.8).

THEOREM 2.6. *Assume that* (2.1) *holds and let* C *be factorized as* $C = LR$. *Then*

$$\operatorname{rank} X \leqq \min \{\operatorname{rank} K(A, L), \operatorname{rank} (I \otimes R)\tilde{G}\},$$

where $K(A, L)$ *is defined by* (1.7) *and* $\tilde{G}$ *is given by* (2.9).

Proof. Write X in (2.8) as

$$X = (L, AL, \cdots, A^{p-1}L) \begin{pmatrix} RG_0 \\ \vdots \\ RG_{p-1} \end{pmatrix} = K(A, L)(I \otimes R)\tilde{G}.$$

3. Realizations of D^{-1}. As its elements are rational functions, the matrix D^{-1} can be split into a proper rational and a polynomial part and thus be written as

$$D^{-1}(x) = K(Ix - F)^{-1}M + S(I - Nx)^{-1}T, \tag{3.1}$$

where N is nilpotent and the eigenvalues of F are zeros of $\det D$ (see, e.g., [9] or [6]). By a slight abuse of standard terminology, we will call a decomposition of the form (3.1) a *realization* of D^{-1}. Whenever a realization of D^{-1} is available, (1.1) can be reduced to a pair of equations of simpler type.

It will be convenient to express the solution of (1.1) by a contour integral.

THEOREM 3.1 [14]. *If the condition*

$$\det\left(\sum \lambda_\nu^i D^i\right) = \det D(\lambda_\nu) \neq 0, \qquad \nu = 1, \cdots, p \tag{2.1″}$$

holds, then the solution of (1.1) *is given by*

$$X = \frac{1}{2\pi i}\oint_\Gamma (Iz - A)^{-1}CD^{-1}(z)dz, \tag{3.2}$$

where Γ *is a positively orientated simple closed curve such that all eigenvalues of* A *are interior to* Γ *and all zeros of* $\det D(z)$ *are exterior to* Γ.

We single out two special equations.

COROLLARY 3.2. *Let* $\lambda_1, \cdots, \lambda_p$ *and* $\mu_1, \cdots, \mu_q$ *be the eigenvalues of* A *and* B, *respectively.*

(i) *If* $\lambda_i - \mu_j \neq 0$ *for* $i = 1, \cdots, p, j = 1, \cdots, q$, *then*

$$AX - XB = C \tag{3.3}$$

has the unique solution

$$X = \frac{1}{2\pi i}\oint_\Gamma (Iz - A)^{-1}C(Iz - B)^{-1}dz,$$

where the λ_i*'s are in the interior of* Γ *and the* μ_j*'s are outside of* Γ.

(ii) *If* $1 - \lambda_i\mu_j \neq 0$ *for* $i = 1, \cdots, p, j = 1, \cdots, q$, *then*

$$X - AXB = C \tag{3.4}$$

has the unique solution

$$X = \frac{1}{2\pi i}\oint_\Gamma (Iz - A)^{-1}C(I - zB)^{-1}dz,$$

where all λ_i*'s are inside of* Γ *and for* $\mu_j \neq 0$ *all the numbers* μ_j^{-1} *are outside of* Γ.

If we replace D^{-1} in the integrand of (3.2) by the realization (3.1) and apply Corollary 3.2, we are led to equations of type (3.3) and (3.4).

THEOREM 3.3. *Assume that* (2.1) *holds and let* (3.1) *be a realization of* D^{-1}. *Then the solution of* (1.1) *is given by*

$$X = X_1M + X_2T$$

where X_1 *and* X_2 *are the solutions of*

$$AX_1 - X_1F = CK$$

and

$$X_2 - AX_2N = CS,$$

respectively.

4. A more general equation with commuting matrices. Let $A_1, \cdots, A_n$ be complex $p \times p$ matrices which are pairwise commutative. In this section, we outline an algebraic approach for the equation

$$\sum A_1^{i_1}, \cdots, A_n^{i_n} X D_{i_1, \cdots, i_n} = C. \tag{4.1}$$

We assume that the i_ν's are nonnegative integers and that the sum in (4.1) is finite. The matrices $D_{i_1, \cdots, i_n}$ are of size $q \times q$ and $C \in \mathbb{C}^{p \times q}$.

Pairwise commutativity implies that the matrices $A_1, \cdots, A_n$ can be simultaneously triangularized [3]. Hence for some nonsingular S we have

$$S^{-1} A S = \operatorname{diag}(\lambda_{1i}, \cdots, \lambda_{pi}) + N_i, \qquad i = 1, \cdots, n,$$

where the matrices N_i are upper triangular and nilpotent. The vectors $\lambda_j = (\lambda_{j1}, \cdots, \lambda_{jn})$, $j = 1, \cdots, p$, are called *joint eigentuples* of $A_1, \cdots, A_n$. We associate (4.1) with a polynomial matrix $D \in \mathbb{C}^{q \times q}[y] = \mathbb{C}^{q \times q}[y_1, \cdots, y_n]$, where D is given by

$$D = \sum D_{i_1, \cdots, i_n} y_1^{i_1}, \cdots, y_n^{i_n}.$$

It is well known (see, e.g., [11]) that (4.1) is universally solvable if and only if

$$\det D(\lambda_j) \neq 0 \tag{4.2}$$

for all joint eigentuples λ_j of $A_1, \cdots, A_n$.

Let γ be the ideal of those polynomials in $\mathbb{C}[y]$ that vanish on $\{\lambda_1, \cdots, \lambda_p\}$,

$$\gamma = \bigcap_{j=1}^{p} (y_1 - \lambda_{j1}, \cdots, y_n - \lambda_{jn}),$$

and let $\varphi \subseteq \mathbb{C}[y]$ be the ideal given by

$$\varphi = \{f \mid f \in \mathbb{C}[y], f(A_1, \cdots, A_n) = 0\}.$$

Note that γ is determined by the semisimple part of the matrices A_i, whereas φ also involves the nilpotent parts. We have

$$\gamma^q \subseteq \varphi \subseteq \gamma,$$

which will allow us to work with γ^q instead of φ in what follows. We extend γ^q to an ideal Ψ in $\mathbb{C}^{q \times q}[y]$,

$$\Psi = \{hU \mid h \in \gamma^q, U \in \mathbb{C}^{q \times q}[y]\}.$$

For $F = \sum F_{k_1, \cdots, k_n} y_1^{k_1}, \cdots, y_n^{k_n} \in \mathbb{C}^{q \times q}[y]$ and $M \in \mathbb{C}^{p \times q}$, we define

$$M * (F + \Psi) = \sum A_1^{k_1}, \cdots, A_n^{k_n} M F_{k_1, \cdots, k_n}. \tag{4.3}$$

Then $\mathbb{C}^{p \times q}$ becomes a unitary right module over the ring $\mathbb{C}^{q \times q}[y]/\Psi$. Using Hilbert's Nullstellensatz, it can be shown that (4.2) holds if and only if $D + \Psi$ is a unit in $\mathbb{C}^{q \times q}[y]/\Psi$ or, equivalently, if there exists a matrix $G \in \mathbb{C}^{q \times q}[y]$ such that

$$DG = GD \equiv I (\operatorname{mod} \Psi). \tag{4.4}$$

If $G = \sum G_{k_1, \cdots, k_n} y_1^{k_1}, \cdots, y_n^{k_n}$ satisfies (4.4), then

$$X = \sum A_1^{k_1}, \cdots, A_n^{k_n} C G_{k_1, \cdots, k_n}$$

is the solution of (4.1).

REFERENCES

[1] T. E. DJAFERIS AND S. K. MITTER, *Algebraic methods for the study of some linear matrix equations*, Linear Algebra Appl., 44 (1982), pp. 125–142.

[2] G. S. DATUASHIVILI, *On the spectrum of a generalized matrix polynomial*, Bull. Acad. Sci. Georgian SSR, 44 (1966), pp. 7–9. (In Russian.)

[3] F. R. GANTMACHER, *Matrizentheorie*, Springer-Verlag, Berlin, 1986.

[4] H. L. J. HAUTUS, *On the solvability of linear matrix equations*, Memorandum 1982-07, Department of Mathematics, Eindhoven University of Technology, the Netherlands, 1982.

[5] H. L. J. HAUTUS, *Linear matrix equations with applications to the regulator problem*, in Mathematical Tools and Models for Control, Systems Analysis and Signal Processing, Vol. 3 (Toulouse/Paris 1981/82), Centre National de la Recherche Scientifique, Paris, 1983, pp. 399–412.

[6] T. KAILATH, *Linear Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[7] R. E. KALMAN, *On the Hermite–Fujiwara theorem in stability theory*, Quart. Appl. Math., 23 (1965), pp. 279–282.

[8] ———, *Algebraic characterization of polynomials whose zeroes lie in certain domains*, Proc. Nat. Acad. Sci. Math., 64 (1969), pp. 818–823.

[9] H. H. ROSENBROCK, *Structural properties of linear dynamical systems*, Internat. J. Control, 20 (1974), pp. 191–202.

[10] J. J. SYLVESTER, *Sur la solution du cas le plus général des équations linéaires en quantités binaires c'est-à-dire en quaternions ou en matrices du second ordre*, C.R. Acad. Sci. Paris, 22 (1884), pp. 117–118.

[11] J. H. M. WEDERBURN, *Lectures on Matrices*, AMS Colloquium Publications XVII, American Mathematical Society, New York, 1934.

[12] H. K. WIMMER, *Linear matrix equations, controllability and observability, and the rank of solutions*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 570–578.

[13] ———, *Linear matrix equations: The module theoretic approach*, Linear Algebra Appl., 120 (1989), pp. 149–164.

[14] H. K. WIMMER AND A. D. ZIEBUR, *Blockmatrizen und lineare Matrizengleichungen*, Math. Nachr., 59 (1974), pp. 213–219.

[15] N. J. YOUNG, *Formulae for the solution of Lyapunov matrix equations*, Internat. J. Control, 31 (1980), pp. 159–179.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1131–1141, October 1992

009

LEAST-INDEX RESOLUTION OF DEGENERACY IN LINEAR COMPLEMENTARITY PROBLEMS WITH SUFFICIENT MATRICES*

RICHARD W. COTTLE† AND YOW-YIEH CHANG‡

Abstract. This paper deals with the principal pivoting method (PPM) for the linear complementarity problem (LCP). It is shown here that when the matrix M of the LCP (q, M) is (row and column) sufficient, the incorporation of a least-index pivot selection rule in the PPM makes it a finite algorithm even when the LCP is degenerate.

Key words. linear complementarity problem, degeneracy resolution, least-index rule, sufficient matrices

AMS(MOS) subject classification. 90C33

1. Introduction. The principal pivoting method (PPM) for the linear complementarity problem (LCP)

$$w = q + Mz, \tag{1}$$

$$w \geqq 0, \qquad z \geqq 0, \tag{2}$$

$$z^T w = 0 \tag{3}$$

was originally established for the cases where the matrix $M \in R^{n \times n}$ is either a **P**-matrix (all principal submatrices have positive determinants) or else is **PSD** (positive semidefinite). See [2], [3], [7], and [5]. Furthermore, in all these papers it is assumed that the problem at hand is *nondegenerate*, meaning that for each solution of (1), at most n of the $2n$ variables $w_1, \cdots, w_n, z_1, \cdots, z_n$ are zero. Degeneracy was handled by allusion to perturbation and lexicographic techniques. For LCPs with **P**-matrices, a different approach was proposed by Murty [10] who developed a Bard-type algorithm with a least-index pivot selection rule. The present paper extends a technical report by Chang [1] in which degeneracy problems arising in the PPM and Lemke's method [9] are resolved by means of least-index pivot selection criteria. It should be noted that the results of [1] are confined to the two aforementioned matrix classes.

Recently, Cottle, Pang, and Venkateswaran [6] introduced the class of *sufficient* matrices that contains all **P**-matrices, all **PSD**-matrices, and other matrices as well. Actually, the class of sufficient matrices is the intersection of two other classes: the *row sufficient* matrices and the *column sufficient* matrices. As shown in [6], the latter two classes are intrinsically related to the LCP. Row sufficient matrices are linked to the *existence* of solutions, whereas column sufficient matrices are associated with the *convexity* of the solution set. (See [6] for details.)

More recently still, Cottle [4] showed that the PPM applies to nondegenerate LCPs with row sufficient matrices. This left open the question of whether a least-index degeneracy resolution scheme could be developed for this extension of the algorithm. In the present paper, we show that when the matrix M of the problem (q, M) is (both row and column) sufficient and the least-index pivot selection rule introduced by Chang [1] is used, the PPM will process the LCP (q, M) in a finite number of steps.

* Received by the editors August 10, 1990; accepted for publication December 10, 1990. This research was partially supported by Department of Energy grant DE-FG03-87ER25028, National Science Foundation grant DMS-8913089, and Office of Naval Research grant N00014-89J-1659.

† Department of Operations Research, Stanford University, Stanford, California 94305-4022 (cottle@sierra.stanford.edu).

‡ Advanced Hi-Tech Corporation, El Segundo, California 90245.

Section 2 of this paper contains a quick review of some background needed for an appreciation of the main result. This involves the basic definitions of row and column sufficient matrices, a review of principal pivoting, and some results on principal pivoting in systems with row and/or column sufficient matrices. In § 3, we state the PPM with least-index pivot selection criterion, and in § 4 we prove that it processes the LCP (q, M) whenever M is a sufficient matrix.

2. Background. Since the matrix classes with which we deal are not well known, we recall what is meant by row (and column) sufficient matrices.

DEFINITION. The matrix $M \in R^{n\times n}$ is

(i) *row sufficient* if

$$(4)\quad x_i(M^Tx)_i \leqq 0 \quad \text{for all } i = 1, \cdots, n \Rightarrow x_i(M^Tx)_i = 0 \quad \text{for all } i = 1, \cdots, n;$$

(ii) *column sufficient* if

$$(5)\quad x_i(Mx)_i \leqq 0 \quad \text{for all } i = 1, \cdots, n \Rightarrow x_i(Mx)_i = 0 \quad \text{for all } i = 1, \cdots, n;$$

(iii) *sufficient* if it is row and column sufficient.

Another way to express the conditions above is through the notion of the Hadamard product of vectors (or matrices). If $u \in R^n$ and $v \in R^n$, their *Hadamard product* is the vector $u*v \in R^n$ defined by

$$(u*v)_i = u_i \cdot v_i, \qquad i = 1, \cdots, n.$$

To apply this notion to the definition of a column sufficient matrix, we let $u = x$ and $v = Mx$. Then the defining condition is

$$x*(Mx) \leqq 0 \Rightarrow x*(Mx) = 0.$$

In the case of a row sufficient matrix, the defining condtion is

$$x*(M^Tx) \leqq 0 \Rightarrow x*(M^Tx) = 0.$$

It is demonstrated in [4] that sufficient matrices are different from **P**-matrices, adequate matrices (see Ingleton [8]), and **PSD**-matrices; moreover, they are not necessarily *positively scaled* versions of such matrices.

As preparation for the brief summary to follow and for the statement of the least-index PPM, it will be helpful at this point to review the notion of principal pivoting. In equation (1) we have an affine transformation of n-space into itself given by $z \mapsto w = q + Mz$ where $M \in R^{n\times n}$ and $q \in R^n$. For the present, let M be an *arbitrary* square matrix with the property that for some index set $\alpha \subset \{1, \cdots, n\}$ the principal submatrix $M_{\alpha\alpha}$ is nonsingular. We may assume that the corresponding principal submatrix of M, namely $M_{\alpha\alpha}$, is a *leading* principal submatrix of M. This is not a restrictive assumption, as it can be brought about by the process called *principal rearrangement*, the simultaneous permutation of row and column indices. Now consider the equation $w = q + Mz$ in partitioned form:

$$(6)\quad \begin{aligned} w_\alpha &= q_\alpha + M_{\alpha\alpha}z_\alpha + M_{\alpha\bar{\alpha}}z_{\bar{\alpha}}, \\ w_{\bar{\alpha}} &= q_{\bar{\alpha}} + M_{\bar{\alpha}\alpha}z_\alpha + M_{\bar{\alpha}\bar{\alpha}}z_{\bar{\alpha}}. \end{aligned}$$

In this representation, the z-variables are *nonbasic* (independent) and the w-variables are *basic* (dependent).

Since $M_{\alpha\alpha}$ is nonsingular by hypothesis, we may exchange the roles of w_α and z_α, thereby obtaining a system of the form

$$(7)\quad \begin{aligned} z_\alpha &= q'_\alpha + M'_{\alpha\alpha}w_\alpha + M'_{\alpha\bar{\alpha}}z_{\bar{\alpha}}, \\ w_{\bar{\alpha}} &= q'_{\bar{\alpha}} + M'_{\bar{\alpha}\alpha}w_\alpha + M'_{\bar{\alpha}\bar{\alpha}}z_{\bar{\alpha}}, \end{aligned}$$

where

$$
\begin{aligned}
&q'_\alpha = -M_{\alpha\alpha}^{-1} q_\alpha, \quad M'_{\alpha\alpha} = M_{\alpha\alpha}^{-1}, \quad M'_{\alpha\bar{\alpha}} = -M_{\alpha\alpha}^{-1} M_{\alpha\bar{\alpha}}, \\
&q'_{\bar{\alpha}} = q_{\bar{\alpha}} - M_{\bar{\alpha}\alpha} M_{\alpha\alpha}^{-1} q_\alpha, \quad M'_{\bar{\alpha}\alpha} = M_{\bar{\alpha}\alpha} M_{\alpha\alpha}^{-1}, \quad M'_{\bar{\alpha}\bar{\alpha}} = M_{\bar{\alpha}\bar{\alpha}} - M_{\bar{\alpha}\alpha} M_{\alpha\alpha}^{-1} M_{\alpha\bar{\alpha}}.
\end{aligned} \tag{8}
$$

DEFINITION. The system (7) is said to be obtained from (6) by a *principal pivotal transformation* on the matrix $M_{\alpha\alpha}$. In this process, the matrix $M_{\alpha\alpha}$ is called the *pivot block*.

The following facts are noteworthy.

1. Every row (column) sufficient matrix has nonnegative principal minors. See [6].
2. Every principal rearrangement of a row (column) sufficient matrix is row (column) sufficient. See [4].
3. Every principal submatrix of a row (column) sufficient matrix is row (column) sufficient. See [4].
4. The class of row sufficient matrices and the class of column sufficient matrices are invariant under principal pivoting. See [4].

3. The least-index PPM. In this section we tersely state the (symmetric) PPM least-index pivot selection rule. This algorithm (without the least-index, tie-breaking rules) has previously been extended to *nondegenerate* LCPs (q, M) in which the matrix M is *row sufficient*. The proof is given in [4], which may also be consulted for further details and references.

The PPM works with pivotal transforms of the system

$$w = q + Mz. \tag{9}$$

In the development below, we use the superscript ν as an iteration counter. The initial value of ν will be 0, and the system shown in (9) will be written as

$$w^0 = q^0 + M^0 z^0. \tag{10}$$

In general, after ν principal pivots, the system will be

$$w^\nu = q^\nu + M^\nu z^\nu. \tag{11}$$

Generically, the vectors w^ν and z^ν, which represent the system's basic and nonbasic variables, respectively, may each be composed of w and z variables. Principal rearrangements can be used to make $\{w_i^\nu, z_i^\nu\} = \{w_i, z_i\}$, $i = 1, \cdots, n$.

Systems like (11) are traditionally represented by a tableau (or schema)

	1	z_1^ν	$\cdots$	z_n^ν
w_1^ν	q_1^ν	m_{11}^ν	$\cdots$	m_{1n}^ν
$\vdots$	$\vdots$	$\vdots$		$\vdots$
w_n^ν	q_n^ν	m_{n1}^ν	$\cdots$	m_{nn}^ν

The symmetric version of the PPM executes principal pivotal transformations (with pivot blocks of order 1 or 2) in order to achieve one of two possible terminal sign configurations in the tableau. The first is a nonnegative "constant column," that is, $q_i^\nu \geqq 0$ for all $i = 1, \cdots, n$. The other is a row of the form

$$q_r^\nu < 0 \quad \text{and} \quad m_{rj}^\nu \leqq 0, \quad j = 1, \cdots, n.$$

The first sign configuration signals that $(\bar{w}^\nu, \bar{z}^\nu) = (q^\nu, 0)$ solves (q, M). The second sign configuration indicates that the problem has no feasible solution. The PPM (as originally conceived) does not actually check for this condition. It cannot occur when $m_{rr}^\nu > 0$ as

in the case of a **P**-matrix. In the more general *row sufficient* case, it can be inferred from the condition

$$q_r^\nu < 0, \quad m_{rr}^\nu = 0, \quad \text{and} \quad m_{ir}^\nu \geqq 0 \quad \forall i \neq r,$$

which would be detected during the "minimum ratio test."

The PPM consists of a sequence of *major cycles*, each of which begins with the selection of a *distinguished* variable whose value is currently negative. That variable remains the one and only distinguished variable throughout the major cycle. The object during the major cycle is to make the value of the distinguished variable increase to zero, if possible. Each iteration involves the increase of a nonbasic variable in an effort to drive the distinguished variable up to zero. This increasing nonbasic variable is called the *driving variable*. According to the rules of the method, all variables whose values are currently nonnegative must remain so. The initial trial solution is $(w^0, z^0) = (q^0, 0)$, hence at least n of the variables must be nonnegative. For those variables w_i^0 whose initial value is $q_i^0 < 0$, we impose[1] a *negative lower bound* λ where

$$\lambda < \min_{1 \leqq i \leqq n} \{q_i^0\}.$$

Then, in addition to requiring all variables with currently nonnegative values to remain so, the PPM also demands that the variables currently having a negative value remain at least as large as λ. This broadens the notion of *basic solution*; nonbasic variables are now allowed to have the value 0 or λ. (See [1]–[3].)

To distinguish between the names of variables and their particular values, we use bars over the generic variable names w_i^ν and z_i^ν. At the beginning of a major cycle in which negative lower bounds λ are in use, we will have $\bar{z}_i^\nu = 0$ or $\bar{z}_i^\nu = \lambda$, $i = 1, \cdots, n$. Next, we use the notation

$$W^\nu(z^\nu) = q^\nu + M^\nu z^\nu.$$

The definition of the mapping W^ν is the same as that of w^ν, but it emphasizes the argument z^ν.

If, at the outset of a major cycle, the selected distinguished variable is basic, the first driving variable is the *complement* of the distinguished variable. Thus, if w_r^ν is the distinguished variable for the current major cycle, then z_r^ν is the first driving variable. The distinguished variable need not be a basic variable, however. With the broader definition of basic solution (given above), the current solution $(\bar{w}^\nu, \bar{z}^\nu)$ may have $\bar{z}_r^\nu = \lambda < 0$ at the beginning of a major cycle. In such circumstances, z_r^ν can be the distinguished variable as well as the driving variable. In this event, the increase of the driving variable will always be blocked, either when a basic variable blocks the driving variable, i.e., reaches its (current) lower bound (0 or λ) or when the distinguished variable increases to zero (in which case the major cycle ends). The point of the least-index degeneracy resolution scheme presented here is that *ties* in the choice of the blocking variable can be broken so as to insure the finiteness of the PPM.

THE LEAST-INDEX RULE. In applying the PPM to the linear complementarity problem (q, M), break ties among the blocking variables as follows:

(A) If the distinguished variable is among the tied blocking variables, choose it as the blocking variable (and terminate the major cycle).

(B) Otherwise, choose the (basic) blocking variable with the smallest index as the exiting variable.

[1] This artifice is not needed when it is known that $M \in \mathbf{P}$.

SYMMETRIC PPM WITH LEAST-INDEX PIVOT SELECTION RULE.

Step 0. Set $\nu = 0$; define $(\bar{w}^0, \bar{z}^0) = (q^0, 0)$. Let λ be any number less than $\min_i q_i^0$.

Step 1. If $q^\nu \geqq 0$ or if $(\bar{w}^\nu, \bar{z}^\nu) \geqq (0, 0)$, stop; $(\bar{w}^\nu, \bar{z}^\nu) := (q^\nu, 0)$ is a solution. Otherwise,[2] determine an index r such that $\bar{z}_r^\nu = \lambda$ or (if none such exist) an index r such that $\bar{w}_r^\nu < 0$.

Step 2. Let ζ_r^ν be the largest value of $z_r^\nu \geqq \bar{z}_r^\nu$ satisfying the following conditions:

(i) $z_r^\nu \leqq 0$ if $\bar{z}_r^\nu = \lambda$.

(ii) $W_r^\nu(\bar{z}_1^\nu, \cdots, \bar{z}_{r-1}^\nu, z_r^\nu, \bar{z}_{r+1}^\nu, \cdots, \bar{z}_n^\nu) \leqq 0$ if $\bar{w}_r^\nu < 0$.

(iii) $W_i^\nu(\bar{z}_1^\nu, \cdots, \bar{z}_{r-1}^\nu, z_r^\nu, \bar{z}_{r+1}^\nu, \cdots, \bar{z}_n^\nu) \geqq 0$ if $\bar{w}_i^\nu > 0$.

(iv) $W_i^\nu(\bar{z}_1^\nu, \cdots, \bar{z}_{r-1}^\nu, z_r^\nu, \bar{z}_{r+1}^\nu, \cdots, \bar{z}_n^\nu) \geqq \lambda$ if $\bar{w}_i^\nu < 0$.

Step 3. If $\zeta_r^\nu = +\infty$, stop. No feasible solution exists. If $\zeta_r^\nu = 0$, let $\bar{z}_r^{\nu+1} = 0$, $\bar{z}_i^{\nu+1} = \bar{z}_i^\nu$ for all $i \neq r$, and let

$$\bar{w}^{\nu+1} = W^{\nu+1}(\bar{z}^{\nu+1}) = W^\nu(\bar{z}^{\nu+1}).$$

Return to Step 1 with ν replaced by $\nu + 1$. If $0 < \zeta_r^\nu < +\infty$, let s be the unique index determined in Step 2 by the conditions (ii), (iii), (iv), and the least-index rule.

Step 4. If $m_{ss}^\nu > 0$, perform the principal pivot $\langle w_s^\nu, z_s^\nu \rangle$. Put $\bar{z}_r^{\nu+1} = \zeta_r^\nu$ and let

$$\bar{z}_s^{\nu+1} = W_s^\nu(\bar{z}_1^\nu, \cdots, \bar{z}_{r-1}^\nu, \zeta_r^\nu, \bar{z}_{r+1}^\nu, \cdots, \bar{z}_n^\nu) \quad \text{and} \quad \bar{w}^{\nu+1} = W^{\nu+1}(\bar{z}^{\nu+1}).$$

If $s = r$, return to Step 1 with ν replaced by $\nu + 1$. If $s \neq r$, return to Step 2 with ν replaced by $\nu + 1$. If $m_{ss}^\nu = 0$, perform the principal pivot $\{\langle w_s^\nu, z_r^\nu \rangle, \langle w_r^\nu, z_s^\nu \rangle\}$. Put

$$\bar{w}_r^{\nu+1} = \bar{z}_s^\nu, \quad \bar{w}_s^{\nu+1} = \zeta_r^\nu, \quad \bar{z}_i^{\nu+1} = \bar{z}_i^\nu \quad \text{for all } i \notin \{r, s\},$$

$$\bar{z}_r^{\nu+1} = W_s^\nu(\bar{z}_1^\nu, \cdots, \bar{z}_{r-1}^\nu, \zeta_r^\nu, \bar{z}_{r+1}^\nu, \cdots, \bar{z}_n^\nu),$$

$$\bar{z}_s^{\nu+1} = W_r^\nu(\bar{z}_1^\nu, \cdots, \bar{z}_{r-1}^\nu, \zeta_r^\nu, \bar{z}_{r+1}^\nu, \cdots, \bar{z}_n^\nu),$$

and then

$$\bar{w}_i^{\nu+1} = W_i^{\nu+1}(\bar{z}^{\nu+1}) \quad \text{for all } i \notin \{r, s\}.$$

Return to Step 2 with ν replaced by $\nu + 1$ and r replaced by s.

4. The finiteness argument. In the following section, we show that the PPM with the least-index rule (stated above) will process any LCP (q, M) in which M is sufficient. It is interesting to observe that the mechanics of the algorithm itself appears to require only the *row sufficiency* property. The finite termination of the algorithm (with or without the least-index rule) is assured if each major cycle is finite, for the total number of negative variables is nonincreasing during each major cycle and decreases strictly at the end of the major cycle. Such finiteness is realized when the problem is nondegenerate. We show here that, even for degenerate problems, the major cycles of the PPM are finite provided the least-index rule is enforced in the pivot selection criterion. As will be seen below, the finiteness of the PPM with the least-index rule hinges on the *column sufficiency* property. This is why we assume that the matrix is both row and column sufficient.

If cycling occurs in a major cycle it is not restrictive to assume that it is one in which w_1 is the distinguished variable. It follows from [4, Thm. 4] that since w_1 and z_1

[2] At the beginning of a major cycle, for each index r, at most one of w_r^ν, z_r^ν can be negative.

are monotonically increasing, both w_1 and z_1 are fixed during cycling. However, the algorithm tries to increase w_1 or z_1 in this major cycle. Hence stalling occurs during these steps. Accordingly, if we delete all the variables that are not involved during cycling, the PPM with the least-index rule merely looks for the index i such that

$$s = \min \{ i : m_{i1}^{\nu} < 0 \}$$

and then pivots on m_{ss}^{ν} (if $m_{ss}^{\nu} \neq 0$) or it pivots on

$$\begin{pmatrix} m_{11}^{\nu} & m_{1s}^{\nu} \\ m_{s1}^{\nu} & m_{ss}^{\nu} \end{pmatrix} \quad \text{if } m_{ss}^{\nu} = 0.$$

Without loss of generality, we may assume that all the variables are involved in the pivoting during cycling. Then, during cycling, the PPM with the least-index rule performs the same pivoting sequence as the following scheme does.

SCHEME.

Step 0. Start with the system $w^{\nu} = q^{\nu} + M^{\nu} z^{\nu}$, $\nu = 0$, where $w^0 = q^0 + M^0 z^0$ is the initial system. (In the following, $M_{\cdot i}^{\nu}$ represents the column of M^{ν} corresponding to the nonbasic variable z_i^{ν} at iteration ν. Similarly, $M_{i\cdot}^{\nu}$ represents the row of M^{ν} corresponding to the basic variable w_i^{ν}.)

Step 1. If $M_{\cdot 1}^{\nu} \geqq 0$, stop. The driving variable z_1^{ν} can be increased strictly. Otherwise, let $s = \min \{ i : m_{i1}^{\nu} < 0 \}$.

Step 2. If $m_{ss}^{\nu} > 0$, perform a pivot on m_{ss}^{ν} and return to Step 1 with ν replaced by $\nu + 1$. Otherwise, perform a block pivot of order 2 on the principal submatrix

$$\begin{pmatrix} m_{11}^{\nu} & m_{1s}^{\nu} \\ m_{s1}^{\nu} & m_{ss}^{\nu} \end{pmatrix}$$

and return to Step 1 with ν replaced by $\nu + 1$.

If we can show that $M_{\cdot 1}^{\nu} \geqq 0$ after a finite number of pivots in the above scheme, then, since the driving variable z_1^{ν} can be increased strictly at this step, we obtain a contradiction to the assumption that cycling occurs in a major cycle (in which w_1 is the distinguished variable) of the PPM with the least-index rule.

Before proceeding, we present a small result on sufficient matrices.

LEMMA 1. *Let $M \in R^{n \times n}$ be column (row) sufficient. Then for any real numbers a, b, c such that $ab < 0 \leqq c$, the matrix*

$$\hat{M} = \begin{pmatrix} m_{11} & m_{12} & \cdots & m_{1n} & a \\ m_{21} & m_{22} & \cdots & m_{2n} & 0 \\ \vdots & \vdots & & \vdots & \vdots \\ m_{n1} & m_{n2} & \cdots & m_{nn} & 0 \\ b & 0 & \cdots & 0 & c \end{pmatrix}$$

is also column (row) sufficient.

Proof. It suffices to prove the assertion for column sufficient matrices. Assume that the vector $\hat{x} = (x_1, x_2, \cdots, x_n, x_{n+1})^T$ satisfies the inequalities $x_i(\hat{M}\hat{x})_i \leqq 0$ for $i = 1, \cdots, n+1$. Then, in particular,

$$x_1(m_{11}x_1 + \cdots + m_{1n}x_n) \leqq -ax_1x_{n+1}$$

and

$$bx_1x_{n+1} \leqq -cx_{n+1}^2 \leqq 0.$$

Since $ab < 0$ it follows that $-ax_1x_{n+1} \leqq 0$. Thus

$$x_i\left(\sum_{j=1}^{n} m_{ij}x_j\right) \leqq 0, \qquad i = 1, \cdots, n.$$

Since M is column sufficient,

$$x_i\left(\sum_{j=1}^{n} m_{ij}x_j\right) = 0, \qquad i = 1, \cdots, n.$$

In particular, it follows that $x_1x_{n+1} = 0$. This, in turn, implies that $\hat{x}*(\hat{M}\hat{x}) = 0$. □

Notice that Lemma 1 provides a mechanism for generating sufficient matrices of arbitrarily large order.

LEMMA 2. *In the above scheme, a pivot in row s, where $s \geqq 2$, must be followed by a pivot in some row with a larger index before another pivot in row s can occur.*

Proof. The proof is by induction. If the matrix M is of order 1 or 2, the lemma is trivial. Suppose the lemma holds when the order of M is less than n and now consider the case when M is of order n.

We shall examine the situation where two pivots occur in row s and

$$2 \leqq s \leqq n - 1.$$

If, between these two pivots, there is no pivot in some row with a larger index, then by deleting $M_{\cdot n}$ and $M_{n\cdot}$, a contradiction to the inductive hypothesis can be derived. Therefore, it suffices to show that there is at most one pivot in row n.

Suppose a pivot occurs in row n at iteration ν_1. Let (T1) denote the corresponding tableau at this iteration.[3]

	1	z_1	$\cdots$	z_n
w_1	q_1	m_{11}	$\cdots$	m_{1n}
$\vdots$	$\vdots$	$\vdots$		$\vdots$
w_n	q_n	m_{n1}	$\cdots$	m_{nn}

TABLEAU (T1)

By the choice of the pivot row, we have $m_{i1} \geqq 0$ for all $i \leqq n - 1$ and $m_{n1} < 0$ in (T1).

Suppose the next occurrence of a pivot in row n is at iteration ν_2. When this occurs, z_n must be the exiting basic variable and w_1 is either basic (Case I) or nonbasic (Case II).

Case I. (w_1 is a basic variable at iteration ν_2.) Let σ be the set of indices i such that w_i is nonbasic at iteration ν_2. Note that $1 \notin \sigma$. Let $\bar{M}$ denote the principal transform of M at this iteration. Clearly, $\bar{M}$ can be obtained from M by performing a block pivot on the principal submatrix $M_{\sigma\sigma}$. Thus

$$\bar{M}_{\sigma 1} = -M_{\sigma\sigma}^{-1}M_{\sigma 1}. \tag{12}$$

Now

$$\bar{M}_{\sigma 1} \simeq \begin{pmatrix} \oplus \\ \oplus \\ \vdots \\ \oplus \\ - \end{pmatrix} \quad \text{and} \quad M_{\sigma 1} \simeq \begin{pmatrix} \oplus \\ \oplus \\ \vdots \\ \oplus \\ - \end{pmatrix}. \tag{13}$$

[3] For simplicity, we represent (T1) without using superscripts.

Being a (nonsingular) principal submatrix of a sufficient matrix, $M_{\sigma\sigma}$ is also a sufficient matrix (see [4]). From (12) and (13), we have

$$M_{\sigma 1} * (M_{\sigma\sigma}^{-1} M_{\sigma 1}) \simeq \begin{pmatrix} \oplus \\ \oplus \\ \vdots \\ \oplus \\ - \end{pmatrix} * \begin{pmatrix} \ominus \\ \ominus \\ \vdots \\ \ominus \\ + \end{pmatrix} \simeq \begin{pmatrix} \ominus \\ \ominus \\ \vdots \\ \ominus \\ - \end{pmatrix},$$

which is impossible since $M_{\sigma\sigma}^{-1}$ is column sufficient.

Case II. (w_1 is a nonbasic variable at iteration ν_2.) Let the definition of σ be as in Case I, but note that now we have $1 \in \sigma$. Since $\bar{M}_{\sigma\sigma}$ is sufficient, the diagonal entry $\bar{m}_{11}$ is nonnegative. There are two cases.

Case II.1 ($\bar{m}_{11} > 0$.) The pivot on $\bar{m}_{11}$ would not change the sign configuration of $\bar{M}_{\sigma 1}$, namely,

$$\bar{M}_{\sigma 1} \simeq \begin{pmatrix} \oplus \\ \oplus \\ \vdots \\ \oplus \\ - \end{pmatrix}.$$

Once this pivot is performed, we have Case I (with a different index set σ).

Case II.2. ($\bar{m}_{11} = 0$.) Here there are two more cases.

Case II.2.1. ($m_{11} > 0$.) By performing a pivot on m_{11} in schema (T1), the variable w_1 becomes nonbasic and the sign configuration of $M_{\cdot 1}$ is unchanged. Therefore, as in Case I, a contradiction can be derived.

Case II.2.2. ($m_{11} = 0$.) Let (T2) denote the tableau at iteration ν_2.

	1	w_σ	$z_{\bar{\sigma}}$
z_σ	$\bar{q}_\sigma$	$\bar{M}_{\sigma\sigma}$	$\bar{M}_{\sigma\bar{\sigma}}$
$w_{\bar{\sigma}}$	$\bar{q}_{\bar{\sigma}}$	$\bar{M}_{\bar{\sigma}\sigma}$	$\bar{M}_{\bar{\sigma}\bar{\sigma}}$

TABLEAU (T2)

In this tableau, $\bar{\sigma} = \{1, \cdots, n\} \setminus \sigma$.

Now let $q_{n+1} \in R$ be arbitrary and enlarge (T1) to (T1*) as follows.

	1	z_1	z_2	$\cdots$	z_n	z_{n+1}
w_1	q_1	m_{11}	m_{12}	$\cdots$	m_{1n}	-1
w_2	q_2	m_{21}	m_{22}	$\cdots$	m_{2n}	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
w_n	q_n	m_{n1}	m_{n2}	$\cdots$	m_{nn}	0
w_{n+1}	q_{n+1}	1	0	$\cdots$	0	1

TABLEAU (T1*)

By Lemma 1, the bordered matrix of (T1*) is sufficient. The block pivot on the principal

submatrix $M_{\sigma\sigma}$ in (T1*) produces (T2*) having (T2) as a subtableau.

	1	w_σ	$z_{\bar\sigma}$	z_{n+1}
z_σ	$\bar q_\sigma$	$\bar M_{\sigma\sigma}$	$\bar M_{\sigma\bar\sigma}$	$\bar M_{\sigma,n+1}$
$w_{\bar\sigma}$	$\bar q_{\bar\sigma}$	$\bar M_{\bar\sigma\sigma}$	$\bar M_{\bar\sigma\bar\sigma}$	$\bar M_{\bar\sigma,n+1}$
w_{n+1}	$\bar q_{n+1}$	$\bar M_{n+1,\sigma}$	$\bar M_{n+1,\bar\sigma}$	$\bar M_{n+1,n+1}$

TABLEAU (T2*)

Note that (T2*) has the same basic z-variables as (T2), hence (T2*) is the corresponding enlargement of (T2).

$$\begin{array}{ccc} & \text{Pivot on } M_{\sigma\sigma} & \\ (\text{T1}) & \rightarrow & (\text{T2}) \\ \text{Enlargement} \downarrow & & \downarrow \text{ Enlargement} \\ (\text{T1}^*) & \rightarrow & (\text{T2}^*) \\ & \text{Pivot on } M_{\sigma\sigma} & \end{array}$$

By pivotal algebra, we have

$$\bar m_{n+1,1} = (M_{n+1,\sigma}\bar M_{\sigma\sigma})_1 = M_{n+1,\sigma}\bar M_{\sigma 1} = (1, 0, \cdots, 0)\bar M_{\sigma 1} = \bar m_{11} = 0.$$

Now the matrix

$$\begin{pmatrix} m_{11} & m_{1,n+1} \\ m_{n+1,1} & m_{n+1,n+1} \end{pmatrix} = \begin{pmatrix} 0 & -1 \\ 1 & 1 \end{pmatrix}$$

is nonsingular and can be used as a pivot block in (T1*). Denote the resulting tableau by (T2**). In it, w_1 and w_{n+1} are nonbasic while all the other w_i are basic.

	1	w_1	z_2	$\cdots$	z_n	w_{n+1}
z_1	$\bar{\bar q}_1$	$\bar{\bar m}_{11}$	$\bar{\bar m}_{12}$	$\cdots$	$\bar{\bar m}_{1n}$	$\bar{\bar m}_{1,n+1}$
w_2	$\bar{\bar q}_2$	$\bar{\bar m}_{21}$	$\bar{\bar m}_{22}$	$\cdots$	$\bar{\bar m}_{2n}$	$\bar{\bar m}_{2,n+1}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$
w_n	$\bar{\bar q}_n$	$\bar{\bar m}_{n1}$	$\bar{\bar m}_{n2}$	$\cdots$	$\bar{\bar m}_{nn}$	$\bar{\bar m}_{n,n+1}$
z_{n+1}	$\bar{\bar q}_{n+1}$	$\bar{\bar m}_{n+1,1}$	$\bar{\bar m}_{n+1,2}$	$\cdots$	$\bar{\bar m}_{n+1,n}$	$\bar{\bar m}_{n+1,n+1}$

TABLEAU (T2**)

In (T1*), we have

$$m_{11} = 0, \quad m_{i1} \geqq 0, \quad i = 2, \cdots, n-1, \quad m_{n1} < 0, \quad m_{n+1,1} = 1.$$

Hence

$$\bar{\bar m}_{11} = 1, \quad \bar{\bar m}_{i1} \geqq 0, \quad i = 2, \cdots, n-1, \quad \bar{\bar m}_{n1} < 0, \quad \bar{\bar m}_{n+1,1} = -1.$$

Now, since both (T2*) and (T2**) are principal transforms of (T1*), it follows that (T2**) is a principal transform of (T2*). In fact, if we define the index set $\rho = (\sigma \setminus \{1\}) \cup \{n+1\}$, then (T2**) can be obtained by performing a block pivot on the principal submatrix $\bar M_{\rho\rho}$ in (T2*). Therefore $\bar{\bar M}_{\rho 1} = -\bar M_{\rho\rho}^{-1}\bar M_{\rho 1}$. The indices n and $n+1$ belong to ρ and

$$\bar m_{n1} < 0, \quad \bar m_{n+1,1} = 0, \quad \bar{\bar m}_{n1} < 0, \quad \bar{\bar m}_{n+1,1} = -1,$$

while $\bar{m}_{i1} \geqq 0$ and $\bar{\bar{m}}_{i1} \geqq 0$ for all other $i \in \rho$. Accordingly, we obtain

$$\bar{M}_{\rho 1} * (\bar{M}_{\rho\rho}^{-1} \bar{M}_{\rho 1}) \simeq \begin{pmatrix} \oplus \\ \vdots \\ \oplus \\ - \\ 0 \end{pmatrix} * \begin{pmatrix} \ominus \\ \vdots \\ \ominus \\ + \\ + \end{pmatrix} \simeq \begin{pmatrix} \ominus \\ \vdots \\ \ominus \\ - \\ 0 \end{pmatrix}.$$

But this is impossible since $\bar{M}_{\rho\rho}^{-1}$ is (column) sufficient. □

LEMMA 3. *In* (T1), $M_{\cdot 1} \geqq 0$ *after a finite number of iterations.*

Proof. For $j > 1$, let $\mu(j)$ be the number of pivots that occur in row j. In the proof of Lemma 2, we have shown that $\mu(n) \leqq 1$. Furthermore, it follows from Lemma 2 that

$$\mu(j) \leqq 1 + \sum_{i=j+1}^{n} \mu(i).$$

In other words,

$$\begin{aligned} &\mu(n-1) \leqq \mu(n) + 1 \leqq 2, \\ &\mu(n-2) \leqq 2^2, \\ &\quad\vdots \\ &\mu(n-i) \leqq 1 + 2^{i-1} + 2^{i-2} + \cdots + 2 + 2^0 = 2^i. \end{aligned}$$

Therefore, the scheme will terminate after a finite number of iterations. □

THEOREM. *In the case of an LCP* (q, M) *with a sufficient matrix* M, *every major cycle of the PPM with the least-index rule consists of a finite number of pivots.*

Proof. Suppose cycling occurs in a major cycle in which w_1 is the distinguished variable. Then, since w_1 and z_1 are monotonically increasing, both w_1 and z_1 are fixed during cycling. However, it follows from Lemma 3 that $M_{\cdot 1} \geqq 0$ after a finite number of steps. Therefore either w_1 or z_1 can be strictly increased after a finite number of steps, in contradiction to the assumption that cycling occurs. □

COROLLARY. *In the sufficient matrix case, the PPM with least-index rule will process the LCP* (q, M) *in a finite number of steps.*

Proof. Each major cycle of the algorithm reduces the number of negative components in (w, z) by at least one. The assertion now follows from the theorem. □

Remark. In implementing the least-index rule it is important to obey statement (A), which says that if the distinguished variable is among the tied blocking variables, then it is to be chosen as the blocking variable. Failure to do so can lead to the false impression that the problem is infeasible.

REFERENCES

[1] Y.-Y. CHANG, *Least-index resolution of degeneracy in linear complementarity problems*, Tech. Report 79-14, Department of Operations Research, Stanford University, Stanford, CA, 1979.

[2] R. W. COTTLE, *Nonlinear programs with positively bounded Jacobians*, Ph.D. Thesis, Department of Mathematics, University of California, Berkeley, CA, 1964.

[3] ———, *The principal pivoting method of quadratic programming*, in Mathematics of the Decision Sciences, Part 1, American Mathematical Society, G. B. Dantzig and A. F. Veinott, Jr., eds., Providence, RI, 1968, pp. 144–162.

[4] ———, *The principal pivoting method revisited*, Math. Programming, Ser. B, 48 (1990), pp. 369–385.
[5] R. W. COTTLE AND G. B. DANTZIG, *Complementary pivot theory of mathematical programming*, in Mathematics of the Decision Sciences, Part 1, G. B. Dantzig and A. F. Veinott, Jr., eds., American Mathematical Society, Providence, RI, 1968, pp. 115–136.
[6] R. W. COTTLE, J.-S. PANG, AND V. VENKATESWARAN, *Sufficient matrices and the linear complementarity problem*, Linear Algebra Appl., 114/115 (1989), pp. 231–249.
[7] G. B. DANTZIG AND R. W. COTTLE, *Positive (semi-)definite programming*, in Nonlinear Programming, J. Abadie, ed., North-Holland, Amsterdam, 1967, pp. 55–73.
[8] A. W. INGLETON, *A problem in linear inequalities*, Proc. London Math. Soc., 16 (1966), pp. 519–536.
[9] C. E. LEMKE, *Bimatrix equilibrium points and mathematical programming*, Management Sci., 11 (1965), pp. 681–689.
[10] K. G. MURTY, *Note on a Bard-type scheme for solving the complementarity problems*, Opsearch, 11 (1974), pp. 123–130.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1142–1153, October 1992

010

INVERSE EIGENVALUE PROBLEMS FOR SYMMETRIC TOEPLITZ MATRICES*

SHMUEL FRIEDLAND†

Dedicated to Louise Hay, whose untimely death has saddened us all.

Abstract. The inverse eigenvalue problems for symmetric Toeplitz matrices with complex-valued (IEPSCTM) and real-valued (IEPSRTM) entries are studied. The main tools are complex and real algebraic geometry. In the complex case it is shown that the IEPSCTM is solvable for most spectra and always solvable for $n \leqq 4$. In the real case the natural decomposition of the space of all $n \times n$ real symmetric Toeplitz matrices to a finite number of connected components of matrices with a simple spectrum is given. It is then shown that the solvability of the IEPSRTM for all spectra can be deduced if the corresponding map to the IEPSRTM has a nonzero degree for at least one component. This is the case for $n \leqq 4$, which gives an alternative proof to Delsarte and Genin's results. The IEPSRTM for odd Toeplitz matrices with real-valued entries is also considered.

Key words. Toeplitz matrices, inverse eigenvalue problems

AMS(MOS) subject classifications. 15A18, 15A57

1. Introduction. Let $t = (t_0, \cdots, t_{n-1}) \in \mathbf{C}^n$. Then the associated symmetric Toeplitz matrix $T = T(t) = (a_{ij})_1^n \in M_n(\mathbf{C})$ is given by the equalities $a_{ij} = t_{|i-j|}$, $i, j = 1, \cdots, n$. T is called a real symmetric Toeplitz matrix if $t \in \mathbf{R}^n$. Denote by $\mathscr{T}_c^n \equiv \mathbf{C}^n$ and $\mathscr{T}_r^n \equiv \mathbf{R}^n$ the linear space of complex symmetric and real symmetric Toeplitz matrices, respectively. The inverse eigenvalue problem for symmetric complex-valued Toeplitz matrices (IEPSCTM) consists of finding $T \in \mathscr{T}_c^n$ with prescribed n complex eigenvalues $\{\sigma_1, \cdots, \sigma_n\}$. The inverse eigenvalue problem for symmetric real-valued Toeplitz matrices (IEPSRTM) consists of finding $T \in \mathscr{T}_r^n$ with prescribed n real eigenvalues $\{\sigma_1, \cdots, \sigma_n\}$. It is conjectured (e.g., [D-G]) that the IEPSRTM is always solvable. This conjecture is known to be true for $n \leqq 4$ [D-G]. In this paper, we outline an approach to the IEPSRTM using the degree theory. For $T \in \mathscr{T}_r^n$ let

$$\sigma_1 = \sigma_1(T) \leqq \sigma_2 = \sigma_2(T) \leqq \cdots \leqq \sigma_n = \sigma_n(T), \qquad \sigma = \sigma(T) = (\sigma_1, \cdots, \sigma_n)$$

be the eigenvalues of T arranged in an increasing order. Denote by $\mathscr{T}_{r,o}^n \subset \mathscr{T}_r^n$ all matrices with pairwise distinct eigenvalues. Let K_i^n, $i = 1, \cdots, \kappa_n$ be the connected components of $\mathscr{T}_{r,o}^n$. We then have the map $\sigma : K_i^n \to \Lambda_o^n$ where

$$\Lambda_o^n = \{x, x = (x_1, \cdots, x_n), x_1 < x_2 \cdots < x_n\}.$$

The topological degree d_i^n of this map is well defined. The continuity principle implies that IEPSRTM is solvable if $d_i^n \neq 0$ for some i. We show that this is the case for $n = 2$, 3, 4.

We now list briefly the contents of the paper. In § 2, we study the IEPSCTM. Using Bezout's theorem as in [Fri1], we show that for $n \leqq 4$ it is always solvable and the number of distinct Toeplitz matrices with a prescribed spectrum is $n!$ counted with multiplicities. We also show that the map σ is dominating. That is, $\sigma(\mathscr{T}_{r,o}^n)$ contains an open set for all n. Section 3 is devoted to basic properties of real symmetric Toeplitz matrices. We recall the well-known result (e.g., see [D-G]) that the spectrum of T splits to the even and the odd spectrum. It then follows that each component K_i^n induces a partition of

* Received by the editors October 1, 1990; accepted for publication (in revised form) June 19, 1991.

† Department of Mathematics, Statistics, and Computer Science, University of Illinois, Chicago, Illinois 60680 (U12735@UICVM.BITNET).

$\{1, \cdots, n\}$ to two (almost equal) sets which correspond to the even and the odd spectrum. In § 4, we show that the number of such partitions is at least $2^{\lfloor n/2 \rfloor}$. In § 5, we consider the odd Toeplitz matrices $\mathscr{T}_r^{n,od} \subset \mathscr{T}_r^n$ that are given by the condition $t_{2m} = 0$, $m = 0, \cdots$. The spectrum of odd Toeplitz matrices is odd. We then have a natural inverse eigenvalue conjecture for the odd Toeplitz matrices. *Any real odd set x is a spectrum of some real odd Toeplitz matrix* (?). This is true for $n = 2, 3, 4$. We conclude our paper by identifying all 10 components of $\mathscr{T}_{r,o}^4$ and those components on which the degree of the map is equal to zero.

2. Complex symmetric Toeplitz matrices. Let

$$\det(\lambda I - T) = \lambda^n + \sum_1^n (-1)^i s_i(T)\lambda^{n-i}, \quad T \in \mathscr{T}_c^n,$$
$$s(T) = (s_1(T), \cdots, s_n(T)). \tag{2.1}$$

We thus have a polynomial map $s : \mathscr{T}_c^n \to \mathbf{C}^n$. Note that each $s_i(T)$ is a homogeneous polynomial of degree i. IEPSCTM consists of finding the inverse image $s^{-1}(z)$, $z \in \mathbf{C}^n$. We now recall a few known facts about the spectral properties of the symmetric complex Toeplitz matrices, which can be found in [D-G]. Let $P \in M_n(\mathbf{C})$ be the following permutation matrix: $P(x_1, x_2, \cdots, x_{n-1}, x_n)^T = (x_n, x_{n-1}, \cdots, x_2, x_1)^T$. Thus P is an involution: $P^{-1} = P^T = P$ and P has $\lceil \frac{n}{2} \rceil$ eigenvalues equal to 1 and $\lfloor \frac{n}{2} \rfloor$ eigenvalues equal to -1. If we consider P as a linear transformation $P : \mathbf{C}^n \to \mathbf{C}^n$, $x \mapsto Px$, we then get the following eigenspace decomposition

$$\mathbf{C}^n = E_+ \oplus E_-, \quad \dim(E_+) = \left\lceil \frac{n}{2} \right\rceil, \quad \dim(E_-) = \left\lfloor \frac{n}{2} \right\rfloor,$$
$$Px = x, \quad x \in E_+ Px = -x, \quad x \in E_-. \tag{2.2}$$

Observe next that $PT = TP$, $T \in \mathscr{T}_c^n$. Thus, if we view each $T \in \mathscr{T}_c^n$ as a linear operator $T : \mathbf{C}^n \to \mathbf{C}^n$, we deduce that

$$T = T_+ \oplus T_-, \quad T_+ : E_+ \to E_+, \quad T_- : E_- \to E_-. \tag{2.3}$$

Hence the characteristic polynomial of any complex symmetric Toeplitz matrix splits to two factors:

(2.4)

$$p_n(\lambda) = p_n^+(\lambda)p_n^-(\lambda), \quad p_n(\lambda) = \det(\lambda I - T((t_0, \cdots, t_{n-1}))),$$
$$p_n^+(\lambda) = \det(\lambda I - T_+((t_0, \cdots, t_{n-1}))), \quad p_n^-(\lambda) = \det(\lambda I - T_-((t_0, \cdots, t_{n-1}))).$$

There is a remarkable recurrence relation among the above polynomials proved in [D-G]:

$$p_{n-1}(\lambda) = \tfrac{1}{2}[p_n^+(\lambda)p_{n-2}^-(\lambda) + p_n^-(\lambda)p_{n-2}^+(\lambda)]. \tag{2.5}$$

Let $F : \mathbf{C}^n \to \mathbf{C}^n$ be a polynomial map. Denote by $J(F)$ and $\det(J(F))$ the Jacobian matrix and the Jacobian of the map F. Recall that F is called dominating if $\det(J(F)) \not\equiv 0$. It is a standard fact in basic algebraic geometry that the range of a dominating map misses a thin set. Namely, $\mathbf{C}^n \backslash F(\mathbf{C}^n)$ is a quasi variety, i.e., an algebraic variety minus a subvariety. Consult, for example, [Mum].

THEOREM 2.6. *The map $s : \mathscr{T}_c^n \to \mathbf{C}^n$ is a dominating map for $n = 1, \cdots$. Assume that $1 \leqq n \leqq 4$. Then s is surjective. Furthermore, for any $z \in \mathbf{C}^n$, $2 \leqq n \leqq 4$ the set $s^{-1}(z)$*

consists of at most n! Toeplitz matrices. More precisely, for almost all z (outside the critical complex variety of codimension 1), the cardinality of the set $s^{-1}(z)$ is equal to n!.

Proof. Clearly, we may assume that $n \geqq 2$. Suppose, to the contrary, that s is not dominating. That is, $s(\mathscr{T}_c^n)$ is a quasi variety. In that case, it is known that each fiber $s^{-1}(s(T))$ is a finite union of irreducible varieties, each one of complex dimension one at least (see [Mum]). Let $J = T((0, 1, 0, \cdots, 0))$. Then J is the classical Jacobian matrix. It is well known that J has n pairwise distinct eigenvalues (see, e.g., [Gan]). (Actually, the eigenvalues of J are equal to $2\cos[(s+1)\pi/(n+1)]$, $s = 0, \cdots, n-1$.) Let orb $(J) \subset M_n(\mathbf{C})$ be the complex variety of all complex symmetric matrices having the same eigenvalues as J. Assume that $X \in$ orb (J). As X is symmetric with pairwise distinct eigenvalues, the standard argument shows that X is (complex) orthogonally similar to a diagonal matrix (see, e.g., [Gan]). Let $\mathscr{O}_n$ be the algebraic group of all $n \times n$ complex-valued orthogonal matrices. Then

$$\text{orb}\,(J) = \{X, X = QJQ^T, Q \in \mathscr{O}_n\}.$$

Recall that $\mathscr{O}_n$ is a complex manifold and a complex Lie group with the Lie algebra $\mathscr{A}_n \subset M_n(\mathbf{C})$ of complex skew symmetric matrices. As the stabilizer of any diagonal matrix $D \in$ orb (J) is a finite group, it then easily follows that orb (J) is a complex manifold of dimension $n(n-1)/2$. Next note that $s^{-1}(s(J)) =$ orb $(J) \cap \mathscr{T}_c^n$. Thus to get the contradiction, it is enough to show that the tangent space to orb (J) at J and $\mathscr{T}_c^n$ intersect transversally at J (see, e.g., [A-R]). From the arguments above it follows that the tangent space of orb (J) at J is of the form $Y = ZJ - JZ$, $Z \in \mathscr{A}_n$. That is, it is enough to show that if $ZJ - JZ = T(t) \in \mathscr{T}_c^n$, then $t = 0$. Clearly, trace $(T(t)) = 0$. Hence $t_0 = 0$. More generally, trace $(J^k T(t)) = 0$, $k = 1, \cdots, n-1$. It now follows that the condition trace $(JT(t)) = 0 \Rightarrow t_1 = 0$. Assume that we already showed that $t_i = 0$, $i = 0, \cdots, k-1$. Use the condition trace $(J^k T(t)) = 0$ to deduce that $t_k = 0$. The above arguments prove that s is a dominating map.

Since each s_i is a homogeneous function, the nonlinear alternative (essentially, the precise version of Bezout's theorem (e.g., [Fri1])) yields that s is surjective whenever $s^{-1}(s(0)) = 0$. That is, s is surjective whenever we can prove that the only $n \times n$ complex symmetric Toeplitz nilpotent matrix is the zero matrix. For $n = 1, 2$ this is straightforward. Consider the case in which $n = 3$. Suppose that $T((t_0, t_1, t_2))$ is nilpotent. Then $t_0 = 0$. As $-t_2$ is an eigenvalue of any $T((0, t_1, t_2))$, it follows that $t_2 = 0$. Hence the nilpotent T is $t_1 J$. Therefore $t_1 = 0$. We now consider the case in which $n = 4$. A straightforward calculation shows

$$\begin{aligned} p_4^+(\lambda) &= \lambda^2 - (2t_0 + t_1 + t_3)\lambda + (t_0 + t_3)(t_0 + t_1) - (t_1 + t_2)^2, \\ p_4^-(\lambda) &= \lambda^2 - (2t_0 - t_1 - t_3)\lambda + (t_0 - t_3)(t_0 - t_1) - (t_1 - t_2)^2. \end{aligned} \tag{2.7}$$

Suppose that $T((t_0, t_1, t_2, t_3))$ is nilpotent. Then $p_4^+(\lambda) = p_4^-(\lambda) = \lambda^2$. Using the fact that the coefficients of λ in these two polynomials are equal to zero, we get that $t_0 = 0$ and $t_3 = -t_1$. The other two conditions boil down to

$$(t_1 + t_2)^2 = (t_1 - t_2)^2 = -t_1^2.$$

Clearly, the only solution to the above two equations is $t_1 = t_2 = 0$. Hence $T = 0$. □

The arguments of the proof of Theorem 2.6 show that the map s is surjective whenever we can show that the only nilpotent complex symmetric Toeplitz matrix is the zero matrix. It may be the case that this statement is not valid for $n \geqq 5$. See [Fri3, Thm. 7.15], where a similar problem exhibits such behavior.

3. Real symmetric Toeplitz matrices. In what follows we shall need some basic concepts and results in real algebraic geometry. For simple results and notions on real

algebraic sets, see [Mil2]. For the basic notions of semialgebraic sets and Tarski–Seidenberg's theorem, see, for example, [A-R, App. B]. See [Bru] for a thorough account and references on semialgebraic sets. We now recall briefly the concepts and the results on semialgebraic sets used here.

A set $X \subset \mathbf{R}^n$ is called (real) algebraic if it is a zero set of a finite number of real polynomials. It then follows that any finite intersection or union of algebraic sets is algebraic. A semialgebraic set is a finite union of sets $Y \subset \mathbf{R}^n$ such that either Y is an algebraic set or $Y = \{x, x \in X, p_i(x) > 0, i = 1, \cdots, k\}$. Here, X is an algebraic set and $p_1, \cdots, p_k$ are real polynomials (depending on Y). Note that an algebraic set is semialgebraic. It easily follows that a finite intersection or union of semialgebraic sets is semialgebraic. Let $F : \mathbf{R}^m \to \mathbf{R}^n$ be a polynomial map. Assume that $X \subset \mathbf{R}^n$ is semialgebraic (algebraic). It then follows that $F^{-1}(X)$ is semialgebraic (algebraic). Assume that $Y \subset \mathbf{R}^m$ is semialgebraic. Then the fundamental theorem of Tarski and Seidenberg states that $F(Y)$ is semialgebraic. It can happen that Y is algebraic, but $F(Y)$ is semialgebraic and not algebraic. The simplest example is served by the quadratic map $F : \mathbf{R} \to \mathbf{R}$, $x \mapsto x^2$. Here the real line $\mathbf{R}$ is an algebraic set but the half-line $\mathbf{R}_+ = F(\mathbf{R})$ is obviously semialgebraic and not algebraic.

Let Λ^n be the set of n ordered real numbers in increasing order and let Λ^n_o be its interior:

$$\Lambda^n = \{x, x = (x_1, \cdots, x_n), x_1 \leqq x_2 \cdots \leqq x_n\},$$

$$\Lambda^n_o = \{x, x = (x_1, \cdots, x_n), x_1 < x_2 \cdots < x_{n1}\}.$$

Clearly, Λ^n, Λ^n_o, $\partial\Lambda^n = \Lambda^n \backslash \Lambda^n_o$ are semialgebraic sets. By noting that $(x_1, \cdots, x_n) \in \Lambda^n$ if and only if $y_i = x_{i+1} - x_i \geqq 0$, $i = 1, \cdots, n-1$, we deduce the isomorphism $\Lambda^n \equiv \mathbf{R} \times \mathbf{R}^{n-1}_+$. For $x = (x_1, \cdots, x_n) \in \mathbf{R}^n$, let $\omega_i(x)$ be the ith elementary symmetric polynomial for $i = 1, \cdots, n$. Set $\omega(x) = (\omega_1(x), \cdots, \omega_n(x))$. Thus we have a polynomial map $\omega : \mathbf{R}^n \to \mathbf{R}^n$. Clearly, $\Omega^n = \omega(\mathbf{R}^n)$ is a closed semialgebraic set which is not algebraic for $n > 1$. (Ω^n has an interior and is not equal to $\mathbf{R}^n$.) Furthermore, $\omega : \mathbf{R}^n \to \Omega^n$ is $n! \to 1$ (counted with multiplicities). Finally, $\omega : \Lambda^n \to \Omega^n$ is a homeomorphism. We shall sometimes identify Ω^n with Λ^n and no ambiguity will arise.

Let $T \in \mathscr{T}^n_r$. Denote by

$$\sigma_1 = \sigma_1(T) \leqq \sigma_2 = \sigma_2(T) \leqq \cdots \leqq \sigma_n = \sigma_n(T), \qquad \sigma = \sigma(T) = (\sigma_1, \cdots, \sigma_n)$$

the eigenvalues of T arranged in increasing order. Thus we have a continuous algebraic map $\sigma : \mathscr{T}^n_r \to \Lambda^n$. Note that $\omega \circ \sigma = s$, where s is the polynomial map defined in (2.1). As $\omega : \Lambda^n \to \Omega^n$ is a homeomorphism, for our purposes, we may consider σ as a polynomial map. Recall that

$$\text{trace}\,(T^2) = \sum_1^n \sigma_i(T)^2,$$

$$\text{trace}\,((T-Q)^2) \geqq \sum_1^n (\sigma_i(T) - \sigma_i(Q))^2, \quad T, Q \in \mathscr{T}^n_r. \tag{3.1}$$

The trace identity is obvious, while the inequality is due to Hoffman and Wielandt [H-W] (see also [Fri1]). Recall that $\Lambda^n \subset \mathbf{R}^n$ has a natural Euclidean metric while $\mathscr{T}^n_r$ has a natural metric induced by the Frobenius inner product $(T, Q) = \text{trace}\,(TQ)$. Thus (3.1) is equivalent to the assertion that the map σ is norm preserving and contracting. Note that Theorem 2.6 implies that σ is a dominating map. That is, $\sigma(\mathscr{T}^n_r)$ has an interior. The inverse eigenvalue problem for real symmetric Toeplitz matrices consists of determining $\sigma^{-1}(x)$ for a given $x \in \Lambda^n$.

Let

$$\text{Disc}(T) = \prod_{1 \leq i < j \leq n} (\sigma_i(T) - \sigma_j(T))^2, \qquad T \in \mathcal{T}_r^n$$

be the discriminant of the polynomial $\det(\lambda I - T)$. It is well known that $\text{Disc}(T)$ is a polynomial in the entries of T. (Arguments in [Fri3] yield that $\deg(\text{Disc}(T)) = n(n-1)$.) Set

$$\mathcal{T}_{r,o}^n = \sigma^{-1}(\Lambda_o^n) = \{T, T \in \mathcal{T}_r^n, \text{Disc}(T) > 0\},$$

$$\partial\mathcal{T}_{r,o}^n = \sigma^{-1}(\partial\Lambda^n) = \{T, T \in \mathcal{T}_r^n, \text{Disc}(T) = 0\}.$$

Thus $\mathcal{T}_{r,o}^n$ is an open semialgebraic set and its boundary is a real algebraic set. It is known that an open semialgebraic set consists of a finite number of connected components. See [Mil1] for an estimate on the number of connected components. Let κ_n be the number of connected components in $\mathcal{T}_{r,o}^n$. We show that $\kappa_n \geq 2^{\lfloor n/2 \rfloor}$. Denote by K_i^n, $i = 1, \cdots, \kappa_n$, the connected components of $\mathcal{T}_{r,o}^n$. By the construction of K_i^n we have the relation

$$\sigma^{-1}(\partial\Lambda^n) \cap \text{closure}(K_i^n) = \partial K_i^n, \qquad i = 1, \cdots, \kappa_n.$$

It then follows that we can define d_i^n as the degree of the map $\sigma : K_i^n \to \Lambda_o^n$, $i = 1, \cdots, \kappa_n$ (see, e.g., [Nir]). To compute d_i^n, we consider $x \in \Lambda_o^n$ so that $\sigma^{-1}(x) \cap K_i^n$ consists of a finite number of Toeplitz matrices $T_1, \cdots, T_k$ such that $\det(J(\sigma)(T_i)) \neq 0$, $i = 1, \cdots, k$. Then

$$d_i^n = \sum_1^k \text{signum}(\det(J(\sigma)(T_i))). \tag{3.2}$$

In particular, $d_i^n = 0$ if $\sigma^{-1}(x) = \varnothing$ for some $x \in \Lambda_o^n$. In topological terms, d_i^n is the value of the induced map $\sigma^* : H_n(\text{closure}(K_i^n), \partial K_i^n) \to H_n(\Lambda^n, \partial\Lambda^n)$ on the relative $\mathbf{Z}$ top homology. The continuity argument yields the conjecture $\sigma(\mathcal{T}_r^n) = \Lambda^n$ if $d_i^n \neq 0$ for some $1 \leq i \leq \kappa_n$. This is the case for $n = 2, 3, 4$. (The case $n = 4$ is treated in § 5.)

Observe that

$$DT((t_0, t_1, \cdots, t_{n-1}))D = T((t_0, -t_1, \cdots, (-1)^{n-1}t_{n-1})) \Rightarrow$$

$$\sigma(T((t_0, t_1, \cdots, t_{n-1}))) = \sigma(T((t_0, -t_1, \cdots, (-1)^{n-1}t_{n-1}))),$$

$$p_{2m}^+(\lambda, DTD) = p_{2m}^-(\lambda, T), \qquad p_{2m}^-(\lambda, DTD) = p_{2m}^+(\lambda, T), \tag{3.3}$$

$$p_{2m-1}^+(\lambda, DTD) = p_{2m-1}^+(\lambda, T), \qquad p_{2m-1}^-(\lambda, DTD) = p_{2m-1}^-(\lambda, T),$$

$$D = \text{diag}\{1, -1, \cdots, (-1)^{n-1}\}.$$

Hence

$$\det(J(\sigma)(DTD)) = (-1)^{\lfloor n/2 \rfloor} \det(J(\sigma)(T)). \tag{3.4}$$

In particular, $DK_i^n D = K_j^n$, $j = j(i)$, $i = 1, \cdots, \kappa_n$. That is, the action of D induces a permutation of the components of $\mathcal{T}_{r,o}^n$. Use (3.2) and the above formula to deduce

$$d_j^n = (-1)^{\lfloor n/2 \rfloor} d_i^n, \qquad DK_i^n D = K_j^n. \tag{3.5}$$

Thus

$$d_i^n = 0, \quad \text{if } DK_i^n D = K_i^n \quad \text{and} \quad \left\lfloor \frac{n}{2} \right\rfloor \text{ is odd.} \tag{3.6}$$

For $n = 2$ it is immediate that $\mathscr{T}^2_{r,o}$ consists of two components given by the inequality $t_1^2 > 0$. Clearly, the set $\sigma^{-1}(x)$, $x \in \Lambda^2_o$ consists of two distinct Toeplitz matrices, each of them belonging to a different component. Hence $|d_i^2| = 1$, $i = 1, 2$. The case $n = 3$ is slightly more complex.

THEOREM 3.7. *The set $\mathscr{T}^3_{r,o}$ decomposes to four connected components*:

$$
\begin{aligned}
K_1^3 &= \{T((t_0, t_1, t_2, t_3)),\ |t_2| < t_1\}, \qquad K_2^3 = -K_1^3,\\
K_3^3 &= \{T((t_0, t_1, t_2, t_3)),\ |t_1| < t_2\}, \qquad K_4^3 = -K_3^3,
\end{aligned} \tag{3.8}
$$

For each $x \in \Lambda^3_o$ the set $\sigma^{-1}(x)$ consists of at least three points and at most four points with the following distribution. The sets $\sigma^{-1}(x) \cap K_i^3$, $i = 1, 2$ consist of exactly one point. Hence $|d_i^3| = 1$, $i = 1, 2$. There exists $3 \leqq i = i(x) \leqq 4$ so that $\sigma^{-1}(x) \cap K_i^3 = \varnothing$. Hence $d_i^3 = 0$, $i = 3, 4$. Finally, if $\sigma(T(t_0, 0, t_2)) \neq x \in \Lambda^3_o$, for all $(t_0, t_2) \in \mathbf{R}^2$ then $\sigma^{-1}(x)$ consists of four distinct Toeplitz matrices.

Proof. Note that

$$
\sigma(T((t_0, \cdots, t_{n-1}))) = \sigma(T((0, t_1, \cdots, t_{n-1}))) + (t_0, \cdots, t_0). \tag{3.9}
$$

Hence, it is enough to study the restriction of $\mathscr{T}^n_{r,o}$ to the hyperplane $t_0 = 0$. In that case, we easily deduce that

$$
p_3^+(\lambda) = \lambda^2 - t_2\lambda - 2t_1^2, \qquad p_3^-(\lambda) = \lambda + t_2.
$$

Thus $p_3^+(\lambda)$ has a multiple root if and only if $t_1 = t_2 = 0$. Furthermore, $p_3^+(-t_2) = 0$ if and only if $t_1^2 = t_2^2$. It then follows that the four components of $\mathscr{T}^3_{r,o}$ are given by (3.8). Other claims can be verified by a straightforward computation. □

4. Partitions of $\mathscr{N}^n$. Let $\mathscr{N}^n = \{1, \cdots, n\}$. Assume that K_i^n is a connected component of $\mathscr{T}^n_{r,o}$. Then K_i^n induces the following partition of $\mathscr{N}^n$:

$$
\begin{aligned}
&\mathscr{N}^n = \mathscr{N}_i^{n,+} \cup \mathscr{N}_i^{n,-}, \quad \operatorname{card}(\mathscr{N}_i^{n,+}) = \left\lceil \frac{n}{2} \right\rceil, \quad \operatorname{card}(\mathscr{N}_i^{n,-}) = \left\lfloor \frac{n}{2} \right\rfloor,\\
&\sigma(T_+) = \{\sigma_i(T),\ i \in \mathscr{N}_i^{n,+}\}, \quad \sigma(T_-) = \{\sigma_i(T),\ i \in \mathscr{N}_i^{n,-}\}, \quad \forall T \in K_i^n.
\end{aligned} \tag{4.1}
$$

For $n = 2$, we have the immediate identities:

$$
\begin{aligned}
&\mathscr{N}_1^{2,+} = \{2\}, \quad \mathscr{N}_1^{2,-} = \{1\}, \quad K_1^2 = \{T((t_0, t_1)),\ t_1 > 0\},\\
&\mathscr{N}_2^{2,+} = \{1\}, \quad \mathscr{N}_2^{2,-} = \{2\}, \quad K_2^2 = \{T((t_0, t_1)),\ t_1 < 0\}.
\end{aligned} \tag{4.2}
$$

Using the results of Theorem 3.7 and its proof we deduce that

$$
\begin{aligned}
&\mathscr{N}_1^{3,+} = \{1, 3\}, \quad \mathscr{N}_1^{3,-} = \{2\}, \quad \mathscr{N}_2^{3,+} = \{1, 3\}, \quad \mathscr{N}_2^{3,-} = \{2\},\\
&\mathscr{N}_3^{3,+} = \{2, 3\}, \quad \mathscr{N}_1^{3,-} = \{1\}, \quad \mathscr{N}_4^{3,+} = \{1, 2\}, \quad \mathscr{N}_4^{3,-} = \{3\}.
\end{aligned} \tag{4.3}
$$

THEOREM 4.4. *Let Π^n be the following subgroup of permutations of order n*:

$$
\Pi^n = \left\{\pi,\ \pi : \{i, n-i\} \to \{i, n-i\},\ i = 1, \cdots, \left\lfloor \frac{n}{2} \right\rfloor\right\}. \tag{4.5}
$$

Then for each $\pi \in \Pi^n$ there exists at least one component K_i^n such that

$$
\mathscr{N}_i^{n,-} = \left\{\pi(1), \cdots, \pi\left(\left\lfloor \frac{n}{2} \right\rfloor\right)\right\}, \quad i = i(\pi), \quad \pi \in \Pi^n. \tag{4.6}
$$

In particular, $\mathscr{T}^n_{r,o}$ has at least $2^{\lfloor n/2 \rfloor}$ connected components ($\kappa_n \geqq 2^{\lfloor n/2 \rfloor}$).

Proof. From (4.2) and (4.3) we deduce that our theorem holds for $n = 2, 3$. For $n \geqq 4$ we prove the theorem by induction. First note that $T((0, \cdots, 0, a))$, $a \neq 0$ is a rank-two matrix with zero trace. A straightforward calculation shows that

$$\begin{aligned} &\sigma(T((0, \cdots, 0, a))) = (-|a|, 0, \cdots, 0, |a|), \\ &T((0, \cdots, 0, a))(1, 0, \cdots, 0, 1)^T = a(1, 0, \cdots, 0, 1)^T, \\ &T((0, \cdots, 0, a))(1, 0, \cdots, 0, -1)^T = -a(1, 0, \cdots, 0, -1)^T. \end{aligned} \tag{4.7}$$

Thus all the eigenvectors corresponding to the 0 eigenvalue are of the form $(0, v_2, \cdots, v_{n-1}, 0)^T$. Fix $(t_0, \cdots, t_{n-1})$, $t_{n-1} \neq 0$ and consider a Toeplitz matrix $T(\varepsilon)$ depending analytically on a real parameter ε, that is, $T(\varepsilon) = T((\varepsilon t_0, \varepsilon t_1, \cdots, \varepsilon t_{n-2}, t_{n-1}))$. Rellich's theorem implies that all the eigenvalues and the eigenvectors of $T(\varepsilon)$ are analytic functions of a complex variable ε in the neighborhood of the real line. Consider first the n eigenvalues as analytic functions in the neighborhood of the origin. Restrict first the attention to $n - 2$ eigenvalues $\{\sigma_2(\varepsilon), \cdots, \sigma_{n-1}(\varepsilon)\}$, which are equal to 0 for $\varepsilon = 0$. Then the classical variation formula (e.g., [Kat] or [Fri2]) claims that $\{\sigma_2'(0), \cdots, \sigma_{n-1}'(0)\}$ are the eigenvalues of $T((t_0, \cdots, t_{n-3}))$. (This also follows from [Tre] when we consider the matrix $\frac{T}{t}$ in his notation.) Assume that $T((t_0, \cdots, t_{n-3})) \in \mathscr{T}_{r,o}^{n-2}$. The analyticity of the eigenvectors $v^i(\varepsilon) = (v_1^i(\varepsilon), \cdots, v_n^i(\varepsilon))^T$ implies that $v^i(0) = (0, v_2^i(0), \cdots, v_{n-1}^i(0), 0)^T$, $i = 2, \cdots, n-1$. Furthermore, the vectors $(v_2^i(0), \cdots, v_{n-2}^i(0))^T$, $i = 2, \cdots, n-1$ are the eigenvectors of $T((t_0, \cdots, t_{n-3}))$. Hence (up to $O(\varepsilon^2)$),

$$\begin{aligned} &\sigma(T_+((\varepsilon t_0, \cdots, \varepsilon t_{n-2}, t_{n-1}))) = \{\sigma_n(\varepsilon)\} \cup \varepsilon\sigma(T_+((t_0, \cdots, t_{n-3}))) \\ &\qquad\qquad \text{for } 0 < \varepsilon \ll 1,\ t_{n-1} > 0, \\ &\sigma(T_+((\varepsilon t_0, \cdots, \varepsilon t_{n-2}, t_{n-1}))) = \{\sigma_1(\varepsilon)\} \cup \varepsilon\sigma(T_+((t_0, \cdots, t_{n-3}))) \\ &\qquad\qquad \text{for } 0 < \varepsilon \ll 1,\ t_{n-1} < 0, \\ &T((t_0, \cdots, t_{n-3})) \in \mathscr{T}_{r,o}^{n-2}. \end{aligned} \tag{4.8}$$

The above equality and the induction hypothesis prove the theorem. □

LEMMA 4.9. *For $n = 4$ any partition of $\mathscr{N}^4$ into two distinct sets having two elements each is a partition corresponding to some K_i^4.*

Proof. According to Theorem 4.4, the following partitions of $\mathscr{N}^4 = \mathscr{N}_+^4 \cup \mathscr{N}_-^4$ are induced by the components of $\mathscr{T}_{r,o}^4$:

$$\{1, 2\} \cup \{3, 4\}, \quad \{1, 3\} \cup \{2, 4\}, \quad \{2, 4\} \cup \{1, 3\}, \quad \{3, 4\} \cup \{1, 2\}.$$

We now show that the partition $\{2, 3\} \cup \{1, 4\}$ is an induced partition. Let $T = T((0, 1, -1, 1))$. We then deduce that the even spectrum of T consists of $\sigma_2 = \sigma_3 = 1$ and the odd spectrum of T consists of $\sigma_1 = -3$, $\sigma_4 = 1$. Consider the close by matrices $T((0, 1, -1, 1 - \varepsilon))$ with $\varepsilon > 0$. Use (2.7) to deduce that $\sigma_4(\varepsilon)$ belongs to the odd spectrum. Finally, the fact that the last partition $\{1, 4\} \cup \{2, 3\}$ is an induced partition follows from (3.3). □

The following lemma can be useful to study the sets $\sigma^{-1}(x)$ in the neighborhood of special points $x \in \Lambda^n$. To make it slightly more general we stated the lemma in the complex context.

LEMMA 4.10. *A complex Toeplitz matrix $T((t_0, \cdots, t_{n-1}))$ is a rank-one matrix if and only if either $t_0 = t_1 = \cdots = t_{n-1} \neq 0$ or $t_0 = -t_1 = \cdots = (-1)^{n-1} t_{n-1} \neq 0$. A complex Toeplitz matrix $T((0, t_1, \cdots, t_{n-1}))$ is a rank-two matrix if either $t_0 = t_1 = \cdots = t_{n-2} = 0$, $t_{n-1} \neq 0$ or $t_0 = t_2 = t_4 = \cdots = 0$, $t_1 = t_3 = \cdots \neq 0$.*

Proof. For $n = 2$ the lemma is immediate. Assume that $n \geqq 3$ and we prove the lemma by induction. Suppose that $T((t_0, \cdots, t_{n-1}))$ is a rank-one matrix. Hence, $T((t_0, \cdots, t_{n-2}))$ is either a zero matrix or a rank-one matrix. The first possibility implies that $T(t)$ is either a zero matrix ($t_{n-1} = 0$) or a rank-two matrix. Thus the first possibility is ruled out. Therefore, $T((t_0, \cdots, t_{n-2}))$ is a rank-one matrix. Assume for the simplicity of the argument that $t_0 = t_1 = \cdots = t_{n-2} \neq 0$. By considering the 2×2 minor of $T(t)$ based on $\{1, n\}$ rows and columns, we deduce that either $t_{n-1} = t_0$ or $t_{n-1} = -t_0$. As $n \geqq 3$ it follows that in the second case the first and the last rows of $T(t)$ are linearly independent. Hence, the second case is ruled out and we proved the lemma for the rank-one matrices.

Assume that $T((0, t_1, \cdots, t_{n-1}))$ is a rank-two matrix. Hence, the matrix $T((0, t_1, \cdots, t_{n-2}))$ is either a zero matrix, a rank-one matrix, or a rank-two matrix. In the first case, we deduce that $t_0 = t_1 = \cdots = t_{n-2} = 0$. Hence $T(t)$ is a rank-two matrix if and only if $t_{n-1} \neq 0$, as we claimed. In view of our result for a rank-one matrix, and the assumption that $t_0 = 0$, the second possibility is ruled out. We are left with the case where $T((0, t_1, \cdots, t_{n-2}))$ is a rank-two matrix. By induction we have two possibilities. Assume first that $t_0 = t_1 = \cdots = t_{n-3} = 0$, $t_{n-2} \neq 0$. For $n \geqq 4$ the rows $1, 2, n-1, n$ are linearly independent, which contradicts our assumption. For $n = 3$ the matrix $T(t)$ is rank-two if and only if $t_2 = 0$. In that case, we get one of our possibilities. Assume finally that $t_0 = t_2 = \cdots = 0$, $t_1 = t_3 = \cdots \neq 0$. Assume first that $n \geqq 4$ is even. Hence, the second and the third rows of $T(t)$ are linearly independent. It then follows that the first row of $T(t)$ is a linear combination of the second and the third rows if and only if the first row is equal to the third row. That is, $t_{n-1} = t_{n-3}$. Assume now that n is odd. Since we already discussed $n = 3$, we may assume that $n \geqq 5$. As for the even case, it follows that the first row must be equal to the third one. Hence $t_{n-1} = 0$. □

COROLLARY 4.11. *Assume that $T(t) \in \mathcal{T}_r^n$ has exactly one nonzero eigenvalue λ. Then either $t = (\frac{\lambda}{n}, \frac{\lambda}{n}, \cdots, \frac{\lambda}{n})$ or $t = (\frac{\lambda}{n}, -\frac{\lambda}{n}, \cdots, (-1)^{n-1}\frac{\lambda}{n})$. Let $T(t) \in \mathcal{T}_r^n$ have the spectrum $(-\lambda, 0, \cdots, 0, \lambda)$, $\lambda > 0$. Then t has one of the following forms*:

$$t = \pm(0, \cdots, 0, \lambda),$$

$$t = \pm \frac{\lambda}{m}(0, 1, 0, 1, \cdots, 0, 1), \qquad n = 2m, \tag{4.12}$$

$$t = \pm \frac{\lambda}{\sqrt{m(m-1)}}(0, 1, 0, 1, \cdots, 1, 0), \qquad n = 2m - 1 > 1.$$

Note that in the first two cases of (4.12), the two nonzero eigenvalues split evenly between the even and the odd spectrum of $T(t)$. In the last case, the two nonzero eigenvalues belong to the even spectrum.

5. Odd Toeplitz matrices. Denote by the odd Toeplitz matrices the following subspaces of the Toeplitz matrices:

$$\begin{aligned} &\mathcal{T}_c^{n,od} = \{T,\ T = T((t_0, t_1, \cdots, t_{n-1})) \in \mathcal{T}_c^n,\ t_{2m} = 0,\ m = 0, 1, \cdots\}, \\ &\mathcal{T}_r^{n,od} = \mathcal{T}_c^{n,od} \cap \mathcal{T}_r^n, \qquad \mathcal{T}_{r,o}^{n,od} = \mathcal{T}_r^{n,od} \cap \mathcal{T}_{r,o}^n. \end{aligned} \tag{5.1}$$

Clearly

$$DTD = -T, \quad T = T((0, t_1, 0, t_3, \cdots)), \quad D = \operatorname{diag}\{1, -1, \cdots, (-1)^{n-1}\}. \tag{5.2}$$

Combine (3.3) and the above equality to deduce

$$\sigma(T_+) = \sigma(-T_-), \quad T \in \mathscr{T}_r^{2m,od}, \quad \sigma(T_+) = \sigma(-T_+),$$

$$\sigma(T_-) = \sigma(-T_-), \quad T \in \mathscr{T}_r^{2m-1,od} \tag{5.3}$$

$$\Rightarrow \sigma(T) = \sigma(-T), \quad T \in \mathscr{T}_r^{n,od}.$$

Let

$$\tilde{\Lambda}^n = \{x, x = (x_1, \cdots, x_n) \in \Lambda^n, x = (-x_n, \cdots, -x_1)\}, \qquad \tilde{\Lambda}_o^n = \tilde{\Lambda}^n \cap \Lambda_o^n. \tag{5.4}$$

It then follows that $\tilde{\Lambda}^n$ and $\tilde{\Lambda}_o^n$ are isomorphic in a trivial way to the sets $\Theta^{\lfloor n/2 \rfloor}$ and $\Theta_o^{\lfloor n/2 \rfloor}$, respectively, where

$$\Theta^k = \{x, x = (x_1, \cdots, x_k), x \in \Lambda^k, 0 \leqq x_1\},$$
$$\Theta_o^k = \{x, x = (x_1, \cdots, x_k), x \in \Lambda_o^k, 0 < x_1\}. \tag{5.5}$$

IEPSROTM (the inverse eigenvalue problem for symmetric real-valued odd Toeplitz matrices). Given $x \in \tilde{\Lambda}^n$, does there exist $T \in \mathscr{T}_r^{n,od}$ with $\sigma(T) = x$?

The arguments of Theorem 4.4 yield that the map $\sigma : \mathscr{T}_r^{2m,od} \to \tilde{\Lambda}^{2m}$ is dominating and $\mathscr{T}_{r,o}^{2m,od}$ has at least 2^m components. For $n = 2, 3, 4$ we identify all the components of $\mathscr{T}_{r,o}^{n,od}$.

THEOREM 5.6. *For $n = 2, 3, 4$ the inverse eigenvalue problem on $\mathscr{T}_{r,o}^{n,od}$ is always solvable. Moreover,*

$$C_1^2 = C_1^3 = \{t_1, t_1 > 0\}, \qquad C_2^2 = C_2^3 = -C_1^2,$$

$$C_1^4 = \{(t_1, t_3), |t_3| < t_1\}, \qquad C_2^4 = \{(t_1, t_3), 0 < t_1 < t_3\}, \tag{5.7}$$

$$C_3^4 = \{(t_1, t_3), 0 < t_1 < -t_3\}, \quad C_4^4 = -C_1^4, \quad C_5^4 = -C_2^4, \quad C_6^4 = -C_3^4.$$

The map $\sigma : C_i^n \to \tilde{\Lambda}_o^n$ is a homeomorphism for $n = 2, 3, 4$ except in the cases $n = 4$, $i = 2, 5$. In these cases, σ is not surjective. (Hence the degree of σ is equal to zero in these cases.) Furthermore, in these two cases, the map $\sigma : C_i^n \to \sigma(C_i^n)$ is $2 \to 1$ counted with the multiplicities.

Proof. For $n = 2, 3$ the theorem is immediate. Assume that $n = 4$. Substitute the values $t_0 = t_3 = 0$ in (2.7). It then follows that $p_4^+(\lambda)$ ($p_4^-(\lambda)$) have a double root if and only if $t_1 = t_3 = 0$. Hence the boundary of $\mathscr{T}_{r,o}^{4,od}$ is formed by all $T \in \mathscr{T}_r^{4,od}$ such that these two polynomials have a common root. By considering $p_4^+(\lambda) - p_4^-(\lambda)$, we deduce that either $t_1 + t_3 = 0$ or $(t_3 - t_1)t_1 = 0$. Moreover, if one of these equalities holds, then p_4^+ and p_4^- have a common zero. These arguments show that $\mathscr{T}_{r,o}^{4,od}$ are partitioned into the six components given by (5.7).

Let $a < b$ be two real numbers such that $a + b \neq 0$. We then pose the problem when a, b are the two roots of p_4^+, that is, when the system

$$t_1 + t_3 = a + b, \qquad (t_3 - t_1)t_1 = ab$$

is solvable over the reals. Clearly, the above two equations are reduced to one quadratic equation

$$2t_1^2 - (a + b)t_1 + ab = 0. \tag{5.8}$$

This equation has two real roots if and only if

$$(a - b)^2 \geqq 4ab. \tag{5.9}$$

Suppose first that $ab < 0$. Then (5.8) has two real roots with the opposite sides. We next

observe for $T \in C_i^4$, $i \neq 2, 5$, that p_4^+ has two real roots of opposite signs. Since $t_1 = 0$ is one of the boundaries of $\mathscr{T}_{r,o}^{4,od}$ we deduce that the map $\sigma : C_i^4 \to \tilde{\Lambda}_o^4$, $i \neq 2, 5$ is a homeomorphism. For $T \in C_2^4$ the two roots of p_4^+ are positive. In view of (5.9) the pair $a = 1$, $b = 2$ are not the roots of any p_4^+. Hence $\sigma : C_2^4 \to \tilde{\Lambda}_o^4$ is not surjective and the degree of this map is 0. On the other hand, if we have two positive distinct numbers a and b which satisfy the strict inequality (5.9), then we have two positive solutions for t_1 which correspond to two Toeplitz matrices in the component C_2^4. That is, $\sigma : C_2^4 \to \sigma(C_2^4)$ is $2 \to 1$ map. The same arguments hold for the component C_5^4. □

We close our paper with some results concerning the full map $\sigma : \mathscr{T}_r^4 \to \Lambda^4$.

THEOREM 5.10. *Let $x = (-b, -a, a, b)$, $0 < a < b$. If a, b satisfy the strict inequality in* (5.9), *then $\sigma^{-1}(x) \subset \mathscr{T}_{r,o}^4$ consists of* 12 *Toeplitz matrices,* 8 *of which are odd. If a, b do not satisfy* (5.9), *then $\sigma^{-1}(x) \in \mathscr{T}_{r,o}^4$ consist of* 8 *Toeplitz matrices,* 4 *of which are odd. In that case, there is no $T \in \mathscr{T}_r^4$ so that either $\sigma(T_+) = \{a, b\}$, $\sigma(T^-) = \{-b, -a\}$, or $\sigma(T_+) = \{-b, -a\}$, $\sigma(T_-) = \{a, b\}$. If the equality sign holds in* (5.9), *then $\sigma^{-1}(x) \in \mathscr{T}_{r,o}^4$ consists of* 10 *Toeplitz matrices,* 6 *of which are odd.*

Proof. Assume that $\sigma(T) = x$. Then trace $(T) = 4t_0 = 0$. As the last coefficients of p_4^+ and p_4^- given by (2.7) must coincide, we deduce that either $t_1 = 0$ or $t_2 = 0$. In the first case, the two real roots of p_4^+ must have opposite signs. That is, the two roots of p_4^+ are either $-a$, b or $-b$, a. Thus we have four Toeplitz matrices:

$$t_1 = 0, \quad t_2 = \pm\sqrt{ab}, \quad t_3 = \pm(b - a). \tag{5.11}$$

In the second case we deduce that T is odd. We then use Theorem 5.6 to conclude the proof of this theorem. □

In Theorem 5.10 choose $a = \varepsilon^2$, $b = 1$ where ε is very small and positive. In that case (5.9) is satisfied. Suppose that $\sigma(T(\varepsilon)) = (-1, -\varepsilon^2, \varepsilon^2, 1)$. If $T(\varepsilon) \in \mathscr{T}_{r,o}^{4,od}$ then (5.8) yields that $T(\varepsilon)$ depends analytically on ε^2. Otherwise, (5.11) shows that $t_2(\varepsilon)$ depends analytically on ε while t_1, t_3 depend analytically on ε^2. Using the variation formula pointed out in the proof of Theorem 4.4, we can show that such behavior is exhibited for general n around the point $\sigma^{-1}((-1, 0, \cdots, 0, 1))$.

THEOREM 5.12. *The boundary of the set $\mathscr{T}_{r,o}^4$ is given by the equation*

$$(t_1 - t_3)[t_2^2(3t_1 + t_3) - t_1(t_1 + t_3)^2] = 0. \tag{5.13}$$

The set $\mathscr{T}_{r,o}^4$ decomposes to 10 *connected components K_i^4, $i = 1, \cdots, 10$. The six components K_i^4, $i = 1, \cdots, 6$ are the only extensions of the components C_i^4, $i = 1, \cdots, 6$ of $\mathscr{T}_{r,o}^{4,od}$. Moreover, the degree of the map $\sigma : K_i^4 \to \Lambda_o^4$ is nonzero for $i = 1, 3, 4, 6$. (Hence σ is surjective in these cases.) For other values of i, the map σ is not surjective. (Hence its degree is equal to zero in these cases.)*

Proof. We may assume that $t_0 = 0$, and it is enough to consider the partition of the three-dimensional space (t_1, t_2, t_3) into the connected components. Then p_4^+ (p_4^-) has a double root only on the line $t_1 = -t_2 = t_3$ ($t_1 = t_2 = t_3$). As a line in three-dimensional space does not separate the space, it follows that $B = \partial\mathscr{T}_{r,o}^4$ is obtained when p_4^+ and p_4^- have a common root. Assume first that $t_1 + t_3 \neq 0$. By considering $p_4^+ - p_4^-$, we deduce that this common root is $\lambda = -2t_1t_2/(t_1 + t_3)$. Substituting this value of λ to p_4^+ and multiplying by $(t_1 + t_3)^2$, we deduce that the intersection of B with $t_1 + t_3 \neq 0$ is given by (5.13). Since B is closed, all the points satisfying (5.13) must be boundary points. Assume that $t_1 + t_3 = 0$. Then p_4^+ and p_4^- have a common root if and only if $t_1t_2 = 0$. It then follows that these two lines satisfy (5.13). From the form (5.13), we deduce that any two Toeplitz matrices $T((0, a_1, a_2, a_3))$, $T((0, b_1, b_2, b_3))$ such that $(a_1 - a_3)(b_1 - b_3) < 0$ cannot lie in the same component. That is, the hyperplane $t_1 - t_3 = 0$ lies entirely on the boundary of $\mathscr{T}_{r,o}^4$.

Discarding the factor $(t_1 - t_3)$, which partitions $\mathcal{T}_r^4$ into two components, we are left with the boundary

$$t_2^2(3t_1 + t_3) - t_1(t_1 + t_3)^2 = 0. \tag{5.14}$$

It is then convenient to view this boundary as the following two surfaces in (t_1, t_2, t_3):

$$t_2 = \pm|t_1 + t_3| \sqrt{\frac{t_1}{(3t_1 + t_3)}}. \tag{5.15}$$

Note that in the region

$$[t_1(t_1 + t_3)]^2 > 0, \qquad t_1(3t_1 + t_3) \leqq 0, \tag{5.16}$$

we do not have either of the two surfaces given by (5.15). We now show that the number of K_i^4 for which $t_1 > t_3$ is five. First note that the region

$$t_1 > 0, \qquad 3t_1 + t_3 \leqq 0 \tag{5.17}$$

belongs to one component. Next, the hypersurfaces (5.15) decompose each of the three-dimensional pieces over the two-dimensional regions: C_1^4, C_5^4, and $C_3^4 \cap \{3t_1 + t_3 > 0\}$ into three parts. The upper and lower parts above C_5^4 are connected in the region (5.17). We call K_3^4 the connected component of $\mathcal{T}_{r,o}^4$, which includes C_3^4. It then follows that on K_3^4 the left-hand side of (5.13) is negative. On the other hand, on the connected component of $\mathcal{T}_{r,o}^4$, which includes C_5^4, the left-hand side is positive. That is, the component $K_5^4 \supset C_5^4$ is bounded by $t_1 - t_3 = 0$ and the inner boundary of K_3^4. Thus the top and the bottom part regions over C_3^4 and C_3^4, $3t_1 + t_3 > 0$ created by the hypersurfaces (5.15) give rise to two new components K_7^4, K_8^4 for which the left-hand side of (5.14) is positive. C_1^4 extends to the unique component K_1^4 whose boundary is formed by $t_1 - t_3 = 0$ and the inner boundaries of K_7^4, K_8^4. The same argument applies to the part $t_1 - t_3 < 0$. This shows that $\mathcal{T}_{r,o}^4$ decomposes into 10 connected components.

We now discuss the map $\sigma : K_i^4 \to \Lambda_o^4$. Let x be as in Theorem 5.10. Suppose that (5.9) does not hold. The results of Theorems 5.6 and 5.10 yield that $\sigma^{-1}(x) \cap K_i^4 = \varnothing$, $i = 2, 5, 7, 8, 9, 10$. Hence, for these values of i, σ is not surjective. On the other hand, for $i = 1, 3, 4, 6$, the set $\sigma^{-1}(x) \cap K_i^4$ contains an odd number of points. We claim that most $x = (-b, -a, a, b)$, $0 < a < b$ are regular values. (x is regular if, at each point of $\sigma^{-1}(x)$, the Jacobian determinant of σ is not zero.) This claim is proved by direct computation. Indeed, it is enough to consider the Jacobian of the equivalent map

$$(t_1, t_2, t_3) \mapsto (t_1 + t_3, t_1t_2, t_1t_3 - (t_1 + t_2)^2)$$

at $\sigma^{-1}(x)$. As for these points, either $t_2 = 0$ or $t_1 = 0$, this claim is easy to verify. Compute the degree of the map $\sigma : K_i^4 \to \Lambda_o^4$ by using the points $\sigma^{-1}(x) \cap K_i^4$. As the number of these points is odd, the degree of σ cannot be equal to zero. □

REFERENCES

[A-R] R. ABRAHAM AND J. W. ROBBIN, *Transversal Mappings and Flows*, Benjamin, New York, 1967.

[Bru] G. W. BRUMFIEL, *Partial Ordered Rings and Semialgebraic Geometry*, Cambridge University Press, London, 1979.

[D-G] P. DELSARTE AND Y. GENIN, *Spectral properties of finite Toeplitz matrices*, in Lecture Notes in Control and Information Sciences 58, Springer-Verlag, Berlin, New York, 1983, pp. 194–213.

[Fri1] S. FRIEDLAND, *Inverse eigenvalue problems*, Linear Algebra Appl., 17 (1977), pp. 15–51.

[Fri2] ———, *Extremal eigenvalue problems*, Bull. Brazilian Math. Soc., 9 (1978), pp. 13–40.

[Fri3] ———, *Simultaneous similarity of matrices*, Adv. in Math, 50 (1983), pp. 189–265.
[Gan] F. R. GANTMACHER, *The Theory of Matrices*, II, Chelsea, New York, 1959.
[H-W] A. J. HOFFMAN AND H. W. WIELANDT, *The variation of the spectrum of a normal matrix*, Duke Math. J., 20 (1953), pp. 37–39.
[Kat] T. KATO, *A Short Introduction to Perturbation Theory for Linear Operators*, Springer-Verlag, Berlin, New York, 1982.
[Mil1] J. MILNOR, *On the Betti numbers of real varieties*, Proc. Amer. Math. Soc., 15 (1964), pp. 275–280.
[Mil2] ———, *Singular Points of Complex Hypersurfaces*, Princeton University Press, Princeton, NJ, 1968.
[Mum] D. MUMFORD, *Algebraic Geometry* I: *Complex Projective Varieties*, Springer-Verlag, Berlin, New York, 1976.
[Nir] L. NIRENBERG, *Topics in Nonlinear Functional Analysis*, NYU–Courant Lectures, Courant Institute of Mathematical Sciences, New York, 1974.
[Tre] W. F. TRENCH, *Spectral evolution of a one-parameter extension of a real symmetric Toeplitz matrix*, SIAM J. Matrix Anal. Appl. 11 (1990), pp. 601–611.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1154–1158, October 1992

011

EQUALITY CASES IN MATRIX EXPONENTIAL INEQUALITIES*

WASIN SO†

Abstract. The Golden–Thompson inequality states that for any Hermitian matrices A and B, $\operatorname{tr} e^A e^B \geqq \operatorname{tr} e^{A+B}$ and the Bernstein inequality states that for any matrix A, $\operatorname{tr} e^{A^*+A} \geqq \operatorname{tr} e^{A^*} e^A$. In this paper, $\{\operatorname{tr} (e^{X/2^k} e^{Y/2^k})^{2^k}\}$ is shown to be a monotonic sequence when X and Y are Hermitian matrices or when $X = Y^*$. Then we prove that (i) equality holds in the Golden–Thompson inequality if and only if A and B commute, and (ii) equality holds in the Bernstein inequality if and only if A is normal.

Key words. trace inequality, matrix exponential

AMS(MOS) subject classifications. 15A18, 15A42, 15A45

1. Introduction. In studying statistical mechanics, Golden [5] and Thompson [8] proved independently that

$$\operatorname{tr} e^A e^B \geqq \operatorname{tr} e^{A+B},$$

where A and B are Hermitian matrices. And, motivated by problems in optimal feedback control, Bernstein [1] proved that

$$\operatorname{tr} e^{A^*} e^A \leqq \operatorname{tr} e^{A^*+A},$$

where A is any matrix and A^* denotes its conjugate transpose. Like other inequalities, it is interesting to know when the equality holds in these inequalities. In both cases, the obvious sufficient condition turns out to be necessary as well. The answers, as stated in the abstract, are contained in §§ 3 and 4. The equality case for the first inequality was mentioned by Lenard in [6] and, in turn was quoted by Marshall and Olkin [7], but no proof was found in either case.

2. Preliminaries. In this section, we collect some facts for later reference. The first fact is the following product exponential formula: For any $n \times n$ matrices X and Y, $\lim_{m \to \infty} (e^{X/m} e^{Y/m})^m = e^{X+Y}$.

For the proof of this formula, we refer to the interesting discussion in [2]. Since trace is a continuous function on matrices, we have the following lemma.

LEMMA 2.1. *For any matrices X and Y,*

$$\lim_{m \to \infty} \operatorname{tr} (e^{X/m} e^{Y/m})^m = \operatorname{tr} e^{X+Y}.$$

Next is the local injectivity of the exponential function for matrices. To be precise, we let Ω be the set $\{L\colon |\operatorname{Im} \lambda| < \pi$ for each eigenvalue λ of $L\}$; then the last statement can be stated as: For X, $Y \in \Omega$, $e^X = e^Y$ implies $X = Y$. A proof of this fact can be found in [9, p. 111]. A useful consequence is presented in Lemma 2.2.

LEMMA 2.2. *For $X \in \Omega$ and invertible matrix T, $T^{-1} e^X T$ is diagonal implies $T^{-1} X T$ is also diagonal.*

Note that Hermitian matrices form a subset of Ω. Hence, we have the following corollary.

* Received by the editors May 21, 1990; accepted for publication (in revised form) March 4, 1991. This research was supported in part by National Science Foundation grant DMS89-01059.

† Department of Mathematics, University of California, Santa Barbara, California 93106. Present address, Institute for Mathematics and Its Applications, University of Minnesota, Minneapolis, Minnesota 55455 (so% csfsa@cs.umn.edu).

COROLLARY 2.3. *For Hermitian matrices X and Y,*

$$e^X e^Y = e^Y e^X \quad \text{iff } XY = YX.$$

Proof. The sufficiency is obvious. For the necessity, we proceed as follows. Since e^X and e^Y commute, there exists a unitary matrix U such that $U^* e^X U$ and $U^* e^Y U$ are both diagonal. Then, by Lemma 2.2, $U^* XU$ and $U^* YU$ are also both diagonal. Hence, X and Y commute.

Since all matrices with spectral radius $\rho(X) < \pi$ form a subset of Ω, by means of a similar proof as that above, we obtain Corollary 2.4.

COROLLARY 2.4. *For matrix X with spectral radius $\rho(X) < \pi$, X is normal if and only if e^X is normal.*

3. Equality case in the Golden–Thompson inequality. We begin with a special case of Weyl's majorant theorem [4, pp. 39–41].

LEMMA 3.1. *Suppose that $\{\lambda_i\}$ and $\{s_i\}$ are the eigenvalues and singular values of a matrix X. Then $\sum_{i=1}^n |\lambda_i|^{2r} \leqq \sum_{i=1}^n s_i^{2r}$, where r is a positive integer.*

LEMMA 3.2. *For positive semidefinite Hermitian matrices X and Y, $\operatorname{tr}(X^p Y^p)^{2q} \leqq \operatorname{tr}(X^{2p} Y^{2p})^q$, where p and q are positive integers.*

Proof. Since $X^p Y^p$ has nonnegative eigenvalues, by Lemma 3.1, we have

$$\begin{aligned} \operatorname{tr}(X^p Y^p)^{2q} &\leqq \operatorname{tr}((X^p Y^p)^* (X^p Y^p))^q \\ &= \operatorname{tr}(Y^p X^p X^p Y^p)^q \\ &= \operatorname{tr}(X^{2p} Y^{2p})^q. \end{aligned}$$

The last equality is due to the cyclic-invariant property of the trace function.

THEOREM 3.3. *If A and B are Hermitian matrices, then $\{\operatorname{tr}(e^{A/2^k} e^{B/2^k})^{2^k}\}$ is monotonically decreasing to $\operatorname{tr} e^{A+B}$.*

Proof. By Lemma 3.2, we have $\operatorname{tr} e^A e^B \geqq \operatorname{tr}(e^{A/2} e^{B/2})^2 \geqq \operatorname{tr}(e^{A/4} e^{B/4})^4 \geqq \cdots$. Hence, the result follows from Lemma 2.1.

THEOREM 3.4. *If A and B are Hermitian matrices, then $\operatorname{tr} e^A e^B = \operatorname{tr} e^{A+B}$ if and only if $AB = BA$.*

Proof. We only need to prove the necessity part. Now suppose that $\operatorname{tr} e^A e^B = \operatorname{tr} e^{A+B}$. Then, by Theorem 3.3, $\operatorname{tr} e^A e^B = \operatorname{tr}(e^{A/2} e^{B/2})^2$ and a direct computation gives $\operatorname{tr}(e^{A/2} e^{B/2} - e^{B/2} e^{A/2})(e^{A/2} e^{B/2} - e^{B/2} e^{A/2})^* = 0$, which in turn implies that $e^{A/2} e^{B/2} = e^{B/2} e^{A/2}$. Finally, Corollary 2.3 gives the result $AB = BA$.

COROLLARY 3.5. *If A and B are Hermitian matrices, then*

$$e^A e^B = e^{A+B} \quad \text{iff } AB = BA.$$

Example 3.6. This example shows that the above corollary is not true in general. Take

$$A = \begin{pmatrix} \pi i & 0 \\ 0 & -\pi i \end{pmatrix} \quad \text{and} \quad B = \begin{pmatrix} 0 & 1 \\ 0 & -2\pi i \end{pmatrix};$$

then

$$A + B = \begin{pmatrix} \pi i & 1 \\ 0 & -3\pi i \end{pmatrix}.$$

Moreover,

$$e^A = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}, \quad e^B = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \text{and} \quad e^{A+B} = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Hence, $e^A e^B = e^{A+B}$, but

$$AB = \begin{pmatrix} 0 & \pi i \\ 0 & -2\pi^2 \end{pmatrix} \neq \begin{pmatrix} 0 & -2\pi i \\ 0 & -2\pi^2 \end{pmatrix} = BA.$$

4. Equality case in the Bernstein inequality. Fan [3] compared the singular values of an $n \times n$ matrix X and those of its power X^p. Let tr_i be the sum of the first i eigenvalues with largest absolute values. He obtained the following theorem.

THEOREM 4.1. *For any matrix X and positive integer p,*

$$\mathrm{tr}_i\, X^{*p}X^p \leqq \mathrm{tr}_i\, (X^*X)^p.$$

The following consequence is essential.

THEOREM 4.2. *For any matrix X and positive integers p, q,*

$$\mathrm{tr}\,(X^{*p}X^p)^q \leqq \mathrm{tr}\,(X^*X)^{pq}.$$

Moreover, equality implies that $\mathrm{tr}\, X^{*p}X^p = \mathrm{tr}\,(X^*X)^p$.

Proof. Let $a = (a_1, \cdots, a_n)$ be the eigenvalues of $X^{*p}X^p$ in decreasing order and $b = (b_1, \cdots, b_n)$ be the eigenvalues of $(X^*X)^p$ in decreasing order. Then Theorem 4.1 states that a is weakly majorized by b. Hence, there exists $d = (d_1, \cdots, d_n)$ [7, p. 123], such that $a_i \leqq d_i$ and d is majorized by b. Since $g(x) = |x|^q$ is a convex function, by [7, p. 115], we have

$$\mathrm{tr}\,(X^{*p}X^p)^q = f(a) \leqq f(d) \leqq f(b) = \mathrm{tr}\,(X^*X)^{pq},$$

where $f((x_1, \cdots, x_n)) = \sum_{i=1}^n |x_i|^q$.

Now suppose that $\mathrm{tr}\,(X^{*p}X^p)^q = \mathrm{tr}\,(X^*X)^{pq}$, i.e., $f(a) = f(b)$. Then $f(a) = f(d)$ and so $a = d$, since $a_i \leqq d_i$. Consequently, a is majorized by b; hence $\mathrm{tr}\, X^{*p}X^p = \mathrm{tr}\,(X^*X)^p$. □

It is clear that if X is normal, then $\mathrm{tr}\,(X^{*p}X^p)^q = \mathrm{tr}\,(X^*X)^{pq}$. The converse is also true for $p \geqq 2$ and $q \geqq 1$ (see Corollary 4.5). We start with the basic lemma.

LEMMA 4.3. *If* $\mathrm{tr}\, X^{*2}X^2 = \mathrm{tr}\,(X^*X)^2$, *then X is normal.*

Proof. Note that $\mathrm{tr}\,(X^*X - XX^*)(X^*X - XX^*)^* = 2[\mathrm{tr}\,(X^{*2}X^2) - \mathrm{tr}\,(X^*X)^2] = 0$. Hence $X^*X - XX^* = 0$ and so X is normal. □

The idea in [3] helps to finish the crucial theorem.

THEOREM 4.4. *If* $\mathrm{tr}\, X^{*p}X^p = \mathrm{tr}\,(X^*X)^p$ *for some $p \geqq 2$, then X is normal.*

Proof. Suppose that $X = UH$ is the polar decomposition of X. Then U is unitary and H is a positive semidefinite Hermitian matrix with eigenvalues $\lambda_1 \geqq \lambda_2 \geqq \cdots \geqq \lambda_n \geqq 0$ and the corresponding orthonormal eigenvectors $v_1, \cdots, v_n$. We only need to consider the case when X is nonzero, i.e., when there exists $1 \leqq k \leqq n$ such that $\lambda_k > \lambda_{k+1} = \cdots = \lambda_n = 0$. Consider

$$\mathrm{tr}\,(X^*X)^p - \mathrm{tr}\, X^{*p}X^p = \sum_{i=1}^k \lambda_i^{2p} - \sum_{i=1}^k \|(UH)^p v_i\|^2$$

(where $\|\ \|$ is the usual Euclidean norm)

$$= \sum_{i=1}^k \lambda_i^{2p} - \sum_{i=1}^k \lambda_i^2 \|(UH)^{p-1}Uv_i\|^2$$

$$= \lambda_k^2 \sum_{i=1}^k \{\lambda_i^{2p-2} - \|(UH)^{p-1}Uv_i\|^2\}$$

$$+ \sum_{i=1}^{k} \{\lambda_i^2 - \lambda_k^2\}\{\lambda_i^{2p-2} - \|(UH)^{p-1}Uv_i\|^2\}$$

$$\geqq \lambda_k^2 \left\{ \sum_{i=1}^{k} \lambda_i^{2p-2} - \sum_{i=1}^{n} \|(UH)^{p-1}Uv_i\|^2 \right\} + D(k, p-1)$$

$$= \lambda_k^2 \{\operatorname{tr}(X^*X)^{p-1} - \operatorname{tr} X^{*p-1}X^{p-1}\} + D(k, p-1)$$

where

$$D(t, s) = \sum_{i=1}^{t} \{\lambda_i^2 - \lambda_t^2\}\{\lambda_i^{2s} - \|(UH)^s Uv_i\|^2\}.$$

Note that [3, Thm. 2]

$$D(t+1, s) - D(t, s) = \{\lambda_t^2 - \lambda_{t+1}^2\} \sum_{i=1}^{t} \{\lambda_i^{2s} - \|(UH)^2 Uv_i\|^2\} \geqq 0.$$

Therefore, $D(k, p-1) \geqq D(k-1, p-1) \geqq \cdots \geqq D(1, p-1) = 0$. Consequently, $\operatorname{tr} X^{*p}X^p = \operatorname{tr}(X^*X)^p$ implies that $\operatorname{tr}(X^*X)^{p-1} = \operatorname{tr} X^{*p-1}X^{p-1}$ since λ_k is nonzero. Repeat the above argument until we have $\operatorname{tr}(X^*X)^2 = \operatorname{tr} X^{*2}X^2$ and so, by Lemma 4.3, X is normal. □

Although we do not need the following result in the remainder of this paper, we include it because it is an extension of Theorem 4.4.

COROLLARY 4.5. *If* $\operatorname{tr}(X^{*p}X^p)^q = \operatorname{tr}(X^*X)^{pq}$ *for some* $p \geqq 2$ *and* $q \geqq 1$, *then* X *is normal.*

Proof. Combine the results of Theorems 4.2 and 4.4 for the proof. □

THEOREM 4.6. $\{\operatorname{tr}(e^{A^*/2^k}e^{A/2^k})^{2^k}\}$ *is monotonically increasing to* $\operatorname{tr} e^{A^*+A}$.

Proof. By Theorem 4.2, we have $\operatorname{tr} e^{A^*}e^A \leqq \operatorname{tr}(e^{A^*/2}e^{A/2})^2 \leqq \operatorname{tr}(e^{A^*/4}e^{A/4})^4 \leqq \cdots$. Then the result follows from Lemma 2.1. □

THEOREM 4.7. $\operatorname{tr} e^{A^*}e^A = \operatorname{tr} e^{A^*+A}$ *if and only if* A *is normal.*

Proof. Suppose that $\operatorname{tr} e^{A^*}e^A = \operatorname{tr} e^{A^*+A}$. Then, by Theorem 4.6, $\operatorname{tr} e^{A^*}e^A = \operatorname{tr}(\exp(A^*/2^k)\exp(A/2^k))^{2^k}$ for all k. Hence, by Theorem 4.4, $\exp(A/2^k)$ is normal for all k. Choose k_0 large enough such that $\rho(A/2_0^k) < \pi$. Consequently, A is normal by Corollary 2.4. The converse is obvious. □

COROLLARY 4.8. *If* $e^{A^*}e^A = e^{A^*+A}$, *then* $e^{A^*}e^A = e^Ae^{A^*}$.

Remark 4.9. The converse of Corollary 4.8 is false, as the example

$$A = \begin{pmatrix} \pi i & 1 \\ 0 & -\pi i \end{pmatrix}$$

shows. Now

$$e^A = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \quad \text{and so} \quad e^{A^*}e^A = e^Ae^{A^*} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

But

$$A^* + A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \text{and so} \quad e^{A^*+A} = \frac{1}{2}\begin{pmatrix} e + e^{-1} & e - e^{-1} \\ e - e^{-1} & e + e^{-1} \end{pmatrix}.$$

Acknowledgment. The author would like to thank Professor R. C. Thompson for his support and for his suggestions on improving the initial draft of this paper. Moreover, the author would like to thank the referee for constructive comments.

REFERENCES

[1] D. S. BERNSTEIN, *Inequalities for the trace of matrix exponentials*, SIAM J. Matrix Anal. Appl., 9 (1988), pp. 156–158.

[2] J. COHEN, S. FRIEDLAND, T. KATO, AND F. P. KELLY, *Eigenvalue inequalities for products of matrix exponentials*, Linear Algebra Appl., 45 (1982), pp. 55–95.

[3] K. FAN, *On a theorem of Weyl concerning eigenvalues of linear transformations* I, Proc. Nat. Acad. Sci. U.S.A., 35 (1949), pp. 652–655.

[4] I. C. GOHBERG AND M. G. KREIN, *Introduction to the theory of nonself-adjoint operators*, Trans. Math. Monographs 18, American Mathematical Society, Providence, RI, 1969.

[5] S. GOLDEN, *Lower bounds for the Helmholtz function*, Phys. Rev., 137 (1965), pp. B1127–1128.

[6] A. LENARD, *Generalization of the Golden–Thompson inequality* $\operatorname{Tr}(e^A e^B) \geqq \operatorname{Tr}(e^{A+B})$, Indiana Univ. Math. J., 21 (1971), pp. 457–467.

[7] A. W. MARSHALL AND I. OLKIN, *Inequalities: Theory of Majorization and its Applications*, Academic Press, New York, 1979.

[8] C. J. THOMPSON, *Inequality with applications in statistical mechanics*, J. Math. Phys., 6 (1965), pp. 1812–1813.

[9] V. S. VARADARAJAN, *Lie Group, Lie Algebras, and Their Representations*, Prentice-Hall, Englewood Cliffs, NJ, 1974.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1159–1171, October 1992

012

BACKWARD ERROR ANALYSIS FOR A POLE ASSIGNMENT ALGORITHM II: THE COMPLEX CASE*

CHRISTOPHER L. COX† AND WILLIAM F. MOSS†

Abstract. In a previous paper [*SIAM J. Matrix Anal. Appl.*, 10 (1989), pp. 446–456], Cox and Moss proved that the pole assignment algorithm of Petkov, Christov, and Konstantinov [*IEEE Trans. Automat. Control*, AC-29 (1984), pp. 1045–1048] is numerically stable for the real case. In this paper, a modified version of the algorithm of Petkov, Christov, and Konstantinov for the complex case is analyzed and the full algorithm (real and complex) is shown to be numerically stable.

Key words. backward error analysis, complex pole assignment, numerical stability

AMS(MOS) subject classifications. 65G05, 93B55, 93D15

1. Introduction. The pole assignment problem for a single-input, time-invariant, linear control system can be posed in the following way. Determine $\mathbf{k} \in R^n$ so that for a given $A \in R^{n\times n}$ and $\mathbf{b} \in R^n$, $A - \mathbf{b}\mathbf{k}^T$ has specified eigenvalues that are real and/or complex conjugate pairs. The name "pole" comes from the fact that these eigenvalues are the poles of a related transfer function. It is well known that if the pair $(A, \mathbf{b})$ is completely controllable, a unique $\mathbf{k}$ can be found [K], [R].

A crucial question in evaluating any pole assignment algorithm is the question of stability with regard to the effect of round-off errors [vD]. In a previous paper [CM], we showed that the algorithm of Petkov, Christov, and Konstantinov [PCK], when restricted to the assignment of real eigenvalues, is numerically stable. This algorithm also provides a method for assigning complex conjugate pairs of eigenvalues using real arithmetic.

The goal of this paper is to extend our backward error analysis to the full algorithm. However, to obtain a satisfactory result, we have found it necessary to modify that part of the algorithm dealing with the assignment of complex conjugate pairs of eigenvalues. In this paper we refer to our modification as the PCK algorithm.

The first step of the PCK algorithm is the reduction of the pair $(A, \mathbf{b})$ to a canonical form $(A^{(0)}, \mathbf{b}^{(0)})$ where the matrix $A^{(0)}$ is unreduced upper Hessenberg, and the vector $\mathbf{b}^{(0)}$ is a nonzero multiple of the first standard basis vector.

In the second step, as the eigenvalues are assigned, the components of an orthogonally transformed gain vector are found. One component of the gain vector is found when a real eigenvalue is assigned. In [CM] we referred to this procedure as PCK deflation. Two components are found when a complex conjugate pair is assigned. We call this process C-PCK deflation.

The third and final step of the PCK algorithm is the transformation of the resulting gain vector back to the original coordinate system. The main result of this paper is the following theorem.

THEOREM 1.1. *Let $A \in R^{n\times n}$, $\mathbf{b} \in R^n$, and let the pair $(A, \mathbf{b})$ be completely controllable. Let $\lambda_1, \cdots, \lambda_{n_r}, p_1 \pm iq_1, \cdots, p_{n_c} \pm iq_{n_c}$ denote the eigenvalues to be assigned, with $n_r + 2n_c = n$. Let $\mathbf{k}$ denote the gain vector computed by the* PCK *algorithm using floating point arithmetic, and denote the unit rounding error by u.*

* Received by the editors October 16, 1989; accepted for publication (in revised form) March 15, 1991.

† Department of Mathematical Sciences, Clemson University, Clemson, South Carolina 29634-1907 (clcox@math.clemson.edu and bmoss@math.clemson.edu).

Then there exist $\Delta A \in R^{n\times n}$ and $\Delta \mathbf{b} \in R^n$ so that $A + \Delta A - (\mathbf{b} + \Delta \mathbf{b})\mathbf{k}^T$ has the desired eigenvalues where

$$\frac{\|\Delta A\|_F}{\|A\|_F} = O\left[n_r\left(2nn_c + \frac{n_r^2}{3}\right)u\right] + O\left[\max_{1\le i\le n_c}\left\{\frac{\|A - p_iI\|_F}{\|A\|_F}\right\}n_c^3 u\right] + O(n_c^3 u),$$

and

$$\frac{\|\Delta \mathbf{b}\|_2}{\|\mathbf{b}\|_2} = O[n_r(n + 2n_c)u] + O(n_c^2 u).$$

The operation count for the PCK algorithm is $10n^3/3$ flops, as stated in [CM].

In § 2, the PCK algorithm for assigning real or complex conjugate poles is outlined. A description of C-PCK deflation is given in § 3, and a backward error analysis is given in § 4.

2. The PCK algorithm. In this section we describe the three steps in the PCK Algorithm.

Step 1. The PCK algorithm first uses a product of Householder transformations P to reduce the pair $(A, \mathbf{b})$ to the canonical form $(A^{(0)}, \mathbf{b}^{(0)})$ where $A^{(0)} := P^TAP$ is unreduced upper Hessenberg, and $\mathbf{b}^{(0)} := P\mathbf{b}$ is a nonzero multiple of the first standard basis vector, $\mathbf{e}_1$ (see [CM]).

Step 2. Let the desired eigenvalues $\lambda_1, \cdots, \lambda_{n_r}, p_1 + iq_1, \cdots, p_{n_c} + iq_{n_c}$ be given with $n_r + 2n_c = n$.

(a) Real case. If $n_r \neq 0$, set

$$\text{iend} = \begin{cases} n_r, & \text{if } n_c \neq 0, \\ n - 2, & \text{if } n_c = 0. \end{cases}$$

For $i = 1, \cdots,$ iend, apply PCK deflation to the pair $(A^{(i-1)}, \mathbf{b}^{(i-1)})$ and the eigenvalue λ_i to find the scalar α_i and an $n - i + 1$ by $n - i + 1$ orthogonal matrix $Q^{(i-1)}$ so that for any $n - i$ vector $\mathbf{k}^{(i)}$,

$$Q^{(i-1)T}A^{(i-1)}Q^{(i-1)} - [Q^{(i-1)T}\mathbf{b}^{(i-1)}][\alpha_i, \mathbf{k}^{(i)T}]$$

has the form

$$\left[\begin{array}{c|c} \lambda_i & \\ \hline 0 & \\ \hline \mathbf{O} & \end{array}\right] = \left[\begin{array}{c|c} \gamma_{11}^{(i)} & \\ \hline \gamma_{21}^{(i)} & \\ \hline \mathbf{O} & A^{(i)} \end{array}\right] - \begin{bmatrix} \beta_i \\ \\ \mathbf{b}^{(i)} \end{bmatrix}[\alpha_i, \mathbf{k}^{(i)T}],$$

where $A^{(i)}$ is an $n - i$ by $n - i$ unreduced upper Hessenberg matrix, $\mathbf{b}^{(i)} = b_1^{(i)}\mathbf{e}_1$ is an $n - i$ vector with $b_1^{(i)} \neq 0$, and β_i, $\gamma_{11}^{(i)}$ and $\gamma_{21}^{(i)}$ are scalars. See [CM] for more details about PCK deflation.

If $n_c = 0$, apply PCK deflation to the pair $(A^{(n-2)}, b^{(n-2)})$ and the eigenvalue λ_{n-1} to find the scalar α_{n-1} and a 2×2 orthogonal matrix Q_{n-2} so that for any scalar α_n,

$$Q^{(n-2)T}A^{(n-2)}Q^{(n-2)} - [Q^{(n-2)T}\mathbf{b}^{(n-2)}][\alpha_{n-1}, \alpha_n]$$

has the form

$$\left[\begin{array}{c|c} \lambda_{n-1} & \\ \hline 0 & \end{array}\right] = \left[\begin{array}{c|c} \gamma_{11}^{(n-1)} & \\ \hline \gamma_{21}^{(n-1)} & \gamma_{22}^{(n-1)} \end{array}\right] - \begin{bmatrix} \beta_{n-1} \\ \beta_n \end{bmatrix}[\alpha_{n-1}, \alpha_n].$$

Find the scalar α_n from $\alpha_n := (\gamma_{22}^{(n-1)} - \lambda_n)/\beta_n$.

(b) Complex case. If $n_c \neq 0$, for $j = n_r + 1, \cdots, n_r + n_c$, apply C-PCK (complex PCK) deflation to the pair $(A^{(j-1)}, \mathbf{b}^{(j-1)})$ and the eigenvalue pair $p_{j-n_r} \pm iq_{j-n_r}$ to find the scalars α_{2j-n_r-1}, α_{2j-n_r} and a $2(n_r + n_c - j + 1)$ by $2(n_r + n_c - j + 1)$ matrix Q_{j-1} so that

$$Q_{j-1}^T A^{(j-1)} Q_{j-1} - [Q_{j-1}^T \mathbf{b}^{(j-1)}][Q_{j-1}^T \mathbf{k}^{(j-1)}]^T$$

has the form

$$\left[\begin{array}{c|c} \hat{S}_j & \\ \hline \mathbf{O} & \end{array}\right] = \left[\begin{array}{cc|c} * & * & \\ * & * & \\ \hline * & * & \\ \mathbf{O} & & A^{(j)} \end{array}\right] - \begin{bmatrix} \beta_{2j-n_r-1} \\ \beta_{2j-n_r} \\ \mathbf{b}^{(j)} \end{bmatrix} [\alpha_{2j-n_r-1}, \alpha_{2j-n_r}, \mathbf{k}^{(j)T}],$$

where $\hat{S}_j$ is a 2×2 matrix defined in § 3, and (unless $j = n_r + n_c$) $A^{(j)}$ is a $2(n_r + j - n_c)$ by $2(n_r + j - n_c)$ unreduced upper Hessenberg matrix and $\mathbf{b}^{(j)} = b_1^{(j)} \mathbf{e}_1$ with $b_1^{(j)} \neq 0$. If $j = n_r + n_c$, the resulting system consists of the upper left 2×2 blocks of each part.

Step 3. Transform back to obtain the gain vector $\mathbf{k}^{(0)}$.

If $n_c \neq 0$, set

$$\mathbf{k}^{(n_c+n_r-1)} = Q_{n_c+n_r-1} \begin{bmatrix} \alpha_{2n_c+n_r-1} \\ \alpha_{2n_c+n_r} \end{bmatrix}.$$

If $n_c > 1$, For $i = n_r + n_c - 2, \cdots, n_r$

$$\mathbf{k}^{(i)} = Q_i \begin{bmatrix} \alpha_{2i-n_r+1} \\ \alpha_{2i-n_r+2} \\ \mathbf{k}^{(i+1)} \end{bmatrix}.$$

If $n_r \neq 0$, set

$$\mathbf{k}^{(n_r-1)} = Q_{n_r-1} \begin{bmatrix} \alpha_{n_r} \\ \mathbf{k}^{(n_r)} \end{bmatrix}.$$

If $n_r > 1$, For $i = n_r - 2, \cdots, 0$,

$$\mathbf{k}^{(i)} = Q_i \begin{bmatrix} \alpha_{i+1} \\ \mathbf{k}^{(i+1)} \end{bmatrix} \in R^{n-i}.$$

Finally, set $\mathbf{k} = P\mathbf{k}^{(0)} \in R^n$.

3. C-PCK deflation. Let $D^{(0)}$ be an $m \times m$ unreduced upper Hessenberg matrix and let $\mathbf{b}^{(0)} = b_1^{(0)} \mathbf{e}_1$ be an m vector with $b_1^{(0)} \neq 0$. We describe C-PCK deflation for the pair $(D^{(0)}, \mathbf{b}^{(0)})$ and the eigenvalue pair $p \pm iq$. C-PCK deflation finds scalars α_1 and α_2 and an $m \times m$ orthogonal matrix Q so that for any $m - 2$ vector α

$$C^{(m-1)} := Q^T D^{(0)} Q - Q^T \mathbf{b}^{(0)} [\alpha_1, \alpha_2, \alpha^T] \tag{3.1}$$

is block 2×2 upper triangular with an upper left-hand 2×2 block $\hat{S}$, which is defined below. Define $D^{(m-1)} := Q^T D^{(0)} Q$. In addition, if $m > 2$, let $C_{22}^{(m-1)}$ and $D_{22}^{(m-1)}$ denote the lower right-hand $(m-2) \times (m-2)$ blocks of $C^{(m-1)}$ and $D^{(m-1)}$, respectively. Then $C_{22}^{(m-1)}$ and $D_{22}^{(m-1)}$ are unreduced upper Hessenberg.

As in the real case, C-PCK deflation is based on computing an eigenvector. The key facts are that $D^{(0)}$ and $C^{(0)} := D^{(0)} - \mathbf{b}^{(0)} \mathbf{k}^{(0)T}$ are identical except for row 1. Thus for any $\mathbf{k}^{(0)}$, $C^{(0)}$ is unreduced upper Hessenberg. Suppose that $\mathbf{v}^{(0)}$ is an eigenvector of $C^{(0)}$

corresponding to the eigenvalue $p + iq$. Then $v_m^{(0)} \neq 0$, and once $v_m^{(0)}$ has been chosen, $\mathbf{v}^{(0)}$ can be found by backward substitution without knowing $\mathbf{k}^{(0)}$. If $q \neq 0$, then $\mathbf{v}^{(0)}$ is complex and $\mathbf{v}^{(0)} = \mathbf{x}^{(0)} + i\mathbf{y}^{(0)}$ with $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ linearly independent. Using only real arithmetic, the vectors $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ could be found by solving, via backward substitution, the system $C^{(0)}\mathbf{x}^{(0)} = p\mathbf{x}^{(0)} - q\mathbf{y}^{(0)}$ and $C^{(0)}\mathbf{y}^{(0)} = q\mathbf{x}^{(0)} + p\mathbf{y}^{(0)}$. However, for q near zero $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ could be nearly linearly dependent, which would result in a numerically unstable algorithm. Suppose, instead, that the system $C^{(0)}\mathbf{x}^{(0)} = p\mathbf{x}^{(0)} - q^2\mathbf{y}^{(0)}$ and $C^{(0)}\mathbf{y}^{(0)} = \mathbf{x}^{(0)} + p\mathbf{y}^{(0)}$ is solved. Then for $q \neq 0$, $\mathbf{x}^{(0)}/q + i\mathbf{y}^{(0)}$ is an eigenvector of $C^{(0)}$ corresponding to the eigenvalue $p + iq$, and, for $q = 0$, $\mathbf{x}^{(0)}$ is an eigenvector and $\mathbf{y}^{(0)}$ a generalized eigenvector of $C^{(0)}$ corresponding to the eigenvalue p. Furthermore, it is possible to make $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ orthogonal by proper choice of $x_m^{(0)}$ and $y_m^{(0)}$.

After $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$ have been found, C-PCK deflation computes an orthogonal matrix Q as the product of Givens rotations so that $\mathbf{x}^{(m-1)} := Q^T\mathbf{x}^{(0)} = x_1^{(m-1)}\mathbf{e}_1$ with $x_1^{(m-1)} \neq 0$, $\mathbf{y}^{(m-1)} := Q^T\mathbf{y}^{(0)} = y_1^{(m-1)}\mathbf{e}_1 + y_2^{(m-1)}\mathbf{e}_2$ with $y_2^{(m-1)} \neq 0$, and if $m > 2$, $[\beta_1, \beta_2, \beta_3, 0, \cdots, 0]^T := Q^T\mathbf{b}^{(0)}$ with $\beta_3 \neq 0$, while if $m = 2$, $[\beta_1, \beta_2]^T := Q^T\mathbf{b}^{(0)}$ with $\beta_2 \neq 0$.

Now define

$$S := \begin{bmatrix} p & 1 \\ -q^2 & p \end{bmatrix}, \qquad R := \begin{bmatrix} x_1^{(m-1)} & y_1^{(m-1)} \\ 0 & y_2^{(m-1)} \end{bmatrix},$$

and

$$\hat{S} := \begin{bmatrix} p - \dfrac{q^2 y_1^{(m-1)}}{x_1^{(m-1)}} & \dfrac{x_1^{(m-1)}}{y_2^{(m-1)}} + \dfrac{q^2 (y_1^{(m-1)})^2}{y_2^{(m-1)} x_1^{(m-1)}} \\ -\dfrac{q^2 y_2^{(m-1)}}{x_1^{(m-1)}} & p + \dfrac{q^2 y_1^{(m-1)}}{x_1^{(m-1)}} \end{bmatrix},$$

and note that $\hat{S} = RSR^{-1}$. Since

$$C^{(0)}[\mathbf{x}^{(0)}, \mathbf{y}^{(0)}] - [\mathbf{x}^{(0)}, \mathbf{y}^{(0)}]S = [\mathbf{0}, \mathbf{0}],$$

it follows that

$$[\mathbf{e}_2, \cdots, \mathbf{e}_m]^T\{D^{(0)}[\mathbf{x}^{(0)}, \mathbf{y}^{(0)}] - [\mathbf{x}^{(0)}, \mathbf{y}^{(0)}]S\} = [\mathbf{0}, \mathbf{0}].$$

Therefore, there exists unique scalars α_1, α_2 so that

$$D^{(0)}[\mathbf{x}^{(0)}, \mathbf{y}^{(0)}] - [\mathbf{x}^{(0)}, \mathbf{y}^{(0)}]S = \mathbf{b}^{(0)}[\alpha_1, \alpha_2]R.$$

Multiplying on the left by Q^T and on the right by R^{-1}, we find that

$$D^{(m-1)}[\mathbf{e}_1, \mathbf{e}_2] - [\mathbf{e}_1, \mathbf{e}_2]\hat{S} = Q^T\mathbf{b}^{(0)}[\alpha_1, \alpha_2], \tag{3.2}$$

from which it follows that the matrix $C^{(m-1)}$ in equation (3.1) is block 2×2 upper triangular with an upper left-hand 2×2 block given by $\hat{S}$.

If $m > 2$, (3.2) yields three equations for determining α_1 and three equations for determining α_2. We use the equations involving the largest value of $|\beta_i|$, $i = 1, 2, 3$. A similar strategy is used in the $m = 2$ case. This is the first of two ways in which our version of the PCK algorithm differs from that in [PCK]. In [PCK] only the equations involving $|\beta_i|$, $i = 2, 3$, were used. Our approach is required by the error analysis of § 4.

Now $Q^T := (V_{m-2}U_{m-1}\cdots V_1U_2)U_1$ where the U_i and V_i are Givens rotations. C-PCK deflation computes the following transformations of $\mathbf{x}^{(0)}$ and $\mathbf{y}^{(0)}$: $\mathbf{x}^{(1)} := U_1\mathbf{x}^{(0)}$, $\mathbf{y}^{(1)} := U_1\mathbf{y}^{(0)}$, $\mathbf{x}^{(2)} := V_1U_2\mathbf{x}^{(1)}$, $\mathbf{y}^{(2)} := V_1U_2\mathbf{y}^{(1)}$, $\cdots$, $\mathbf{x}^{(m-1)} := V_{m-2}U_{m-1}\mathbf{x}^{(m-2)}$, $\mathbf{y}^{(m-1)} := V_{m-2}U_{m-1}\mathbf{y}^{(m-2)}$. As was pointed out in [PCK], by computing only those

components of $\mathbf{x}^{(0)}, \mathbf{y}^{(0)}, \cdots, \mathbf{x}^{(m-1)}, \mathbf{y}^{(m-1)}$ which are required, the operation count and round-off error can be reduced. We now present the details of C-PCK deflation.

ALGORITHM (C-PCK DEFLATION). Given $D^{(0)} \in R^{m\times m}$ unreduced upper Hessenberg, $\mathbf{b}^{(0)} = b_1^{(0)}\mathbf{e}_1 \in R^m$ with $b_1^{(0)} \neq 0$ and a pair of complex conjugate eigenvalues $p \pm iq$, compute $\alpha_1, \alpha_2, \beta_1, \beta_2, \beta_3$, and Q as follows:

Comment: Begin C-PCK deflation.

$$x_m^{(0)} := 1,\ y_m^{(0)} := 1$$

$$x_{m-1}^{(0)} := ((p - d_{mm}^{(0)})x_m^{(0)} - q^2 y_m^{(0)})/d_{m,m-1}^{(0)} \tag{3.3}$$

$$y_{m-1}^{(0)} := (x_m^{(0)} + (p - d_{mm}^{(0)})y_m^{(0)})/d_{m,m-1}^{(0)}$$

For $i = 1, \cdots, m-1$

if $i \neq m-1$

(3.4)

$$x_{m-i-1}^{(i-1)} := [(p - d_{m-i,m-i}^{(i-1)})x_{m-i}^{(i-1)} - q^2 y_{m-i}^{(i-1)} - d_{m-i,m-i+1}^{(i-1)}x_{m-i+1}^{(i-1)}]/d_{m-i,m-i-1}^{(i-1)}$$

$$y_{m-i-1}^{(i-1)} := [x_{m-i}^{(i-1)} + (p - d_{m-i,m-i}^{(i-1)})y_{m-i}^{(i-1)} - d_{m-i,m-i+1}^{(i-1)}y_{m-i+1}^{(i-1)}]/d_{m-i,m-i-1}^{(i-1)}$$

endif

Construct a Givens rotation $U_i = U_i(m-i, m-i+1) \in R^{m\times m}$ so that

$$\mathbf{e}_{m-i}^T U_i \mathbf{x}^{(i-1)} =: x_{m-i}^{(i)} > 0$$

$$\mathbf{e}_{m-i+1}^T U_i \mathbf{x}^{(i-1)} =: x_{m-i+1}^{(i)} = 0$$

For $k = m-i+2, \cdots, m$ (do not execute if $m-i+2 > m$)

$$x_k^{(i)} := 0$$

repeat

Set

$$\tilde{D}^{(i-1)} := U_i D^{(i-1)} U_i^T$$

$$\mathbf{e}_{m-i}^T U_i \mathbf{y}^{(i-1)} =: \tilde{y}_{m-i}^{(i-1)}$$

$$\mathbf{e}_{m-i+1}^T U_i \mathbf{y}^{(i-1)} =: \tilde{y}_{m-i+1}^{(i-1)}$$

and note that the $(m-i+1, m-i-1)$ element of $\tilde{D}^{(i-1)}$ may not be zero.

If $i \neq 1$

Construct a Givens rotation $V_{i-1} = V_{i-1}(m-i+1, m-i+2) \in R^{m\times m}$ so that

$$\mathbf{e}_{m-i+1}^T V_{i-1} \tilde{\mathbf{y}}^{(i-1)} =: y_{m-i+1}^{(i)} > 0$$

$$\mathbf{e}_{m-i+2}^T V_{i-1} \tilde{\mathbf{y}}^{(i-1)} =: y_{m-i+2}^{(i)} = 0$$

Note that $\mathbf{x}^{(i)}$ is unaffected when multiplied by V_{i-1}.

For $k = m-i+3, \cdots, m$ (do not execute if $m-i+3 > m$)

$$y_k^{(i)} := 0$$

repeat

Set

$$D^{(i)} := V_{i-1}\tilde{D}^{(i-1)}V_{i-1}^T$$

$$y_{m-i}^{(i)} := \tilde{y}_{m-i}^{(i-1)}$$

and note that the $(m-i+2, m-i-1)$ element of $D^{(i)}$ may not be zero.

else

Set

$$D^{(i)} := \tilde{D}^{(i-1)}$$

$$y^{(i)} := \tilde{\mathbf{y}}^{(i-1)}$$

endif

repeat

Comment: all components of $\mathbf{x}^{(m-1)}$ and $\mathbf{y}^{(m-1)}$ have been computed.

Set $Q^T := V_{m-2}U_{m-1}\cdots V_1U_2U_1$ (if $m = 2$, $Q^T := U_1$),

$$[\beta_1, \beta_2, \beta_3, 0, \cdots, 0]^T := V_{m-2}U_{m-1}\mathbf{b}_0$$

Comment: $D^{(m-1)} = Q^TD^{(0)}Q$, $C^{(m-1)} = Q^TD^{(0)}Q - Q^T\mathbf{b}_0[Q^T\mathbf{k}^{(0)}]^T$

Comment: $V_{m-2}U_{m-1}\mathbf{b}_0 = Q^T\mathbf{b}_0$

If $m = 2$, set $\beta_3 = 0$.

If $(|\beta_1| \geqq \max\{|\beta_2|, |\beta_3|\})$

$$\alpha_1 := (d_{11}^{(m-1)} - \hat{s}_{11})/\beta_1 \qquad (3.5)$$

$$\alpha_2 := (d_{12}^{(m-1)} - \hat{s}_{12})/\beta_1$$

else

If $(|\beta_2| \geqq |\beta_3|)$

$$\alpha_1 := (d_{21}^{(m-1)} - \hat{s}_{21})/\beta_2 \qquad (3.6)$$

$$\alpha_2 := (d_{22}^{(m-1)} - \hat{s}_{22})/\beta_2$$

else

$$\alpha_1 := d_{31}^{(m-1)}/\beta_3 \qquad (3.7)$$

$$\alpha_2 := d_{32}^{(m-1)}/\beta_3$$

endif

end if

Comment: End of C-PCK deflation.

We conclude this section by showing that if $m > 2$, $\beta_3 \neq 0$ and that $C_{22}^{(m-1)}$ and $D_{22}^{(m-1)}$ are unreduced upper Hessenberg.

First note that for $i = 1, \cdots, m-1$,

$$x_{m-i}^{(i)} = \sqrt{[x_{m-i}^{(i-1)}]^2 + [x_{m-i+1}^{(i-1)}]^2} \geqq |x_{m-i+1}^{(i-1)}| = |x_{m-(i-1)}^{(i-1)}| \qquad (3.8)$$

$$\geqq |x_{m-(i-2)}^{(i-2)}| \geqq \cdots \geqq |x_m^{(0)}| = 1.$$

Similarly,

$$y_{m-i+1}^{(i)} \geqq |y_{m-i+2}^{(i-1)}| \geqq \cdots \geqq |y_m^{(1)}|. \qquad (3.9)$$

By direct computation,

$$y_m^{(1)} = \frac{-|d_{m,m-1}^{(0)}|}{d_{m,m-1}^{(0)}} \cdot \frac{[x_m^{(0)}]^2 + q^2[y_m^{(0)}]^2}{\sqrt{[(p - d_{mm}^{(0)})x_m^{(0)} - q^2y_m^{(0)}]^2 + [d_{m,m-1}^{(0)}x_m^{(0)}]^2}} \neq 0. \qquad (3.10)$$

Since $Q^T\mathbf{b}^{(0)} = V_{m-2}U_{m-1}\mathbf{b}^{(0)}$, $\beta_i = 0$, $i = 4, \cdots, m$ and we have

$$\beta_3 = \frac{\tilde{y}_3^{(m-2)}x_2^{(m-2)}}{\sqrt{[\tilde{y}_2^{(m-2)}]^2 + [\tilde{y}_3^{(m-2)}]^2}\sqrt{[x_1^{(m-2)}]^2 + [x_2^{(m-2)}]^2}}\, b_1^{(0)}.$$

Thus, because $\tilde{y}_3^{(m-2)} = y_3^{(m-2)}$, $\beta_3 \neq 0$.

Finally, we show that $C_{22}^{(m-1)}$ and $D_{22}^{(m-1)}$ are unreduced upper Hessenberg. Following an argument similar to the one in [CM], it can be shown, using the zero, nonzero structure of $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$, $i = 0, \cdots, m-1$, that $D^{(m-1)}$ is upper Hessenberg except for fill-in in the (3, 1) entry. Thus $D_{22}^{(m-1)}$ is upper Hessenberg. Since $C^{(m-1)}$ and $D^{(m-1)}$ differ only in their first three rows, $C_{22}^{(m-1)}$ and $D_{22}^{(m-1)}$ differ only in their first row, so that $C_{22}^{(m-1)}$ must also be upper Hessenberg.

Now for any $m-2$ vector α, $C^{(0)} = D^{(0)} - \mathbf{b}^{(0)}[\alpha_1, \alpha_2, \alpha^T]Q^T$ is unreduced upper Hessenberg. Therefore, the pair $(C^{(0)}, \mathbf{b}^{(0)})$ is completely controllable. It follows that $M := [Q^T\mathbf{b}^{(0)}, C^{(m-1)}Q^T\mathbf{b}^{(0)}, \cdots, \{C^{(m-1)}\}^{m-1}Q^T\mathbf{b}^{(0)}]$ has rank m. Define $\mathbf{b}^{(1)} := [\beta_3, 0 \cdots, 0]^T$. Due to the block structure of $C^{(m-1)}$, the $(m-3) \times m$ matrix consisting of rows 3, $\cdots$, m of M must have the form $[\mathbf{b}^{(1)}, C_{22}^{(m-1)}\mathbf{b}^{(1)}, \cdots, \{C_{22}^{(m-1)}\}^{m-1}\mathbf{b}^{(1)}]$. Consequently, if $C_{22}^{(m-1)}$ is not unreduced, then one or more of the last $m-3$ rows of M must be trivial, which is a contradiction. Thus $C_{22}^{(m-1)}$ is unreduced, and so $D_{22}^{(m-1)}$ must be also.

4. Backward error analysis. As in [CM], we use the notation of Wilkinson [W] in the analysis of floating-point errors. All symbols representing computed quantities denote floating point results, u represents the unit rounding error, and d denotes a constant of order one. We have tried to include enough details in the proofs so that a reader knowledgeable in the subject could fill in what is left out. To shorten this analysis, we assume that all Givens rotations are computed exactly. At the end of this section, we point out the changes that must be made to take into account the errors in computing these rotations.

We need Propositions 4.1, 4.2, and 4.3, together with the result of [CM], to prove Theorem 1.1. In these propositions, we use the notation of § 3 with one exception. Let $\mathbf{x}^{(m-1)} := \alpha\mathbf{e}_1$ and $\mathbf{y}^{(m-1)} := \gamma\mathbf{e}_1 + \beta\mathbf{e}_2$. Then

$$R = \begin{bmatrix} \alpha & \gamma \\ 0 & \beta \end{bmatrix} \quad \text{and} \quad \hat{S} = \begin{bmatrix} p - \dfrac{q^2\gamma}{\alpha} & \dfrac{\alpha}{\beta} + \dfrac{q^2\gamma^2}{\alpha\beta} \\ -q^2\dfrac{\beta}{\alpha} & p + \dfrac{q^2\gamma}{\alpha} \end{bmatrix}.$$

We begin with an analysis of C-PCK deflation.

PROPOSITION 4.1. *Suppose that* C-PCK *deflation is applied using floating point arithmetic. Then there exists matrices* $G, F \in R^{m\times m}$ *and a vector* $\mathbf{f} \in R^m$ *so that*

$$D^{(0)} + G + F - (\mathbf{b}^{(0)} + \mathbf{f})[\alpha_1, \alpha_2, \alpha^T] = Q\left[\begin{array}{c|c} \hat{S} & \\ \hline \mathbf{O} & \end{array}\right]Q^T,$$

where

$$\frac{\|G\|_F}{\|D^{(0)}\|_F} \leqq 10(m-1)\text{Factor}_1(1 + \text{Factor}_2)du,$$

$$\frac{\|F\|_F}{\|D^{(0)}\|_F} \leqq 11\sqrt{3}\text{Factor}_1 du + 10\sqrt{3}(m-1)\text{Factor}_1(1 + \text{Factor}_2)du$$

$$+12\sqrt{3}\text{Factor}_3 + 26\text{Factor}_4 du,$$

$$\frac{\|\mathbf{f}\|_2}{\|\mathbf{b}^{(0)}\|_2} \leqq 4du,$$

and

$$\text{Factor}_1 = \frac{\|D^{(0)} - pI\|_F}{\|D^{(0)}\|_F}, \qquad \text{Factor}_2 = \frac{|\gamma|}{|\beta|},$$

$$\text{Factor}_3 = \frac{q^2\gamma^2}{|\alpha\beta|\,\|D^{(0)}\|_F}, \qquad \text{Factor}_4 = \frac{|\gamma|q^2}{|\alpha|\,\|D^{(0)}\|_F}.$$

Proof. We use a generic floating point error factor $(1 + \varepsilon)$ where the ε may be different each time it appears. Rearranging the result of applying floating point arithmetic to the C-PCK equations, we have

$$(1+\varepsilon)d^{(0)}_{m,m-1}x^{(0)}_{m-1} + (1+\varepsilon)(d^{(0)}_{m,m} - p)x^{(0)}_m = -q^2 y^{(0)}_m,$$
$$(1+\varepsilon)d^{(0)}_{m,m-1}y^{(0)}_{m-1} + (1+\varepsilon)(d^{(0)}_{m,m} - p)y^{(0)}_m = x^{(0)}_m,$$

and for $i = 1, \cdots, m-2$,

$$(1+\varepsilon)d^{(i-1)}_{m-i,m-i-1}x^{(i-1)}_{m-i-1} + (1+\varepsilon)(d^{(i-1)}_{m-i,m-i} - p)x^{(i-1)}_{m-i} + (1+\varepsilon)d^{(i-1)}_{m-i,m-i+1}x^{(i-1)}_{m-i+1} = -q^2 y^{(i-1)}_{m-i},$$
$$(1+\varepsilon)d^{(i-1)}_{m-i,m-i-1}y^{(i-1)}_{m-i-1} + (1+\varepsilon)(d^{(i-1)}_{m-i,m-i} - p)y^{(i-1)}_{m-i} + (1+\varepsilon)d^{(i-1)}_{m-i,m-i+1}y^{(i-1)}_{m-i+1} = x^{(i-1)}_{m-i},$$

where $|\varepsilon| \leqq 5du$. Now define

$$-\eta^{(0)}_m := \varepsilon d^{(0)}_{m,m-1}x^{(0)}_{m-1} + \varepsilon(d^{(0)}_{m,\mathrm{m}} - p)x^{(0)}_m,$$
$$-\zeta^{(0)}_m := \varepsilon d^{(0)}_{m,m-1}y^{(0)}_{m-1} + \varepsilon(d^{(0)}_{m,m} - p)y^{(0)}_m,$$

and for $i = 1, \cdots, m-2$,

$$-\eta^{(i-1)}_{m-i} := \varepsilon d^{(i-1)}_{m-i,m-i-1}x^{(i-1)}_{m-i-1} + \varepsilon(d^{(i-1)}_{m-i,m-i} - p)x^{(i-1)}_{m-i} + \varepsilon d^{(i-1)}_{m-i,m-i+1}x^{(i-1)}_{m-i+1},$$
$$-\zeta^{(i-1)}_{m-i} := \varepsilon d^{(i-1)}_{m-i,m-i-1}y^{(i-1)}_{m-i-1} + \varepsilon(d^{(i-1)}_{m-i,m-i} - p)y^{(i-1)}_{m-i} + \varepsilon d^{(i-1)}_{m-i,m-i+1}y^{(i-1)}_{m-i+1}.$$

Also, set $\eta^{(m-2)}_1 := 0$ and $\zeta^{(m-2)}_1 := 0$, and note that $[\eta^{(1)}, \zeta^{(1)}] = U_1[\eta^{(0)}, \zeta^{(0)}]$ and for $i = 2, \cdots, m-1$, $[\eta^{(i)}, \zeta^{(i)}] = V_{i-1}U_i[\eta^{(i-1)}, \zeta^{(i-1)}]$.

Then

$$[\mathbf{e}_2^T, \cdots, \mathbf{e}_m^T]\{D^{(m-2)}[\mathbf{x}^{(m-2)}, \mathbf{y}^{(m-2)}] - [\mathbf{x}^{(m-2)}, \mathbf{y}^{(m-2)}]S\} = [\mathbf{e}_2^T, \cdots, \mathbf{e}_m^T][\eta^{(m-2)}, \zeta^{(m-2)}], \tag{4.1}$$

$$\|\eta^{(m-2)}\|_2 \leqq \sqrt{m-1}\,|\varepsilon|\,\|D^{(0)} - pI\|_F\|\mathbf{x}^{(m-1)}\|_2 \leqq 5(m-1)du\|D^{(0)} - pI\|_F|\alpha|,$$

and

$$\|\zeta^{(m-2)}\|_2 \leqq \sqrt{m-1}\,|\varepsilon|\,\|D^{(0)} - pI\|_F\|\mathbf{y}^{(m-1)}\|_2 \leqq 5(m-1)du\|D^{(0)} - pI\|_F\sqrt{\gamma^2+\beta^2}.$$

Set $\mathbf{b}^{(1)} := Q^T\mathbf{b}^{(0)} = [\beta_1, \beta_2, \beta_3, 0, \cdots, 0]^T$. We find $F^{(1)} \in R^{m\times m}$, $\mathbf{f}^{(1)} \in R^m$, and scalars α_1 and α_2 so that for any vector $\alpha \in R^{m-2}$,

$$\{D^{(m-1)} + F^{(1)} - (\mathbf{b}^{(1)} + \mathbf{f}^{(1)})[\alpha_1, \alpha_2, \alpha^T]\}[\mathbf{x}^{(m-1)}, \mathbf{y}^{(m-1)}] - [\mathbf{x}^{(m-1)}, \mathbf{y}^{(m-1)}]S = [\eta^{(m-1)}, \zeta^{(m-1)}]. \tag{4.2}$$

Define

$$G^{(1)} := \left[-\frac{\eta^{(m-1)}}{\alpha}, \frac{\gamma\eta^{(m-1)}}{\alpha\beta} - \frac{\zeta^{(m-1)}}{\beta}, \mathbf{0}, \cdots, \mathbf{0} \right].$$

Then $[\eta^{(m-1)}, \zeta^{(m-1)}] = -G^{(1)}[\mathbf{x}^{(m-1)}, \mathbf{y}^{(m-1)}]$ and

$$\|G^{(1)}\|_F \leqq \frac{1}{|\alpha|} + \frac{|\gamma|\,\|\eta^{(m-1)}\|_2}{|\alpha\beta|} + \frac{\|\zeta^{(m-1)}\|_2}{|\beta|}.$$

Setting $G := QG^{(1)}Q^T$, we get the desired inequality for G.

Next, multiplying (4.2) on the right by R^{-1}, we obtain

$$\{D^{(m-1)} + G^{(1)} + F^{(1)} - (\mathbf{b}^{(1)} + \mathbf{f}^{(1)})[\alpha_1, \alpha_2, \alpha^T]\}[\mathbf{e}_1, \mathbf{e}_2] = [\mathbf{e}_1, \mathbf{e}_2]\hat{S}. \tag{4.3}$$

C-PCK deflation solves for α_1 and α_2 using (4.3) in one of three ways, depending on the magnitudes of β_1, β_2, and β_3 relative to each other.

Case 1 ($|\beta_1| \geqq \max\{|\beta_2|, |\beta_3|\}$). The floating point result for α_2 can be written as

$$\alpha_2 = [(1 + \delta_1)d_{12}^{(m-1)} - \alpha/\beta - (1 + \delta_2)q^2\gamma^2/(\alpha\beta)]/[(1 + \delta_3)\beta_1], \tag{4.4}$$

where $|\delta_1| \leqq 2du$, $|\delta_2| \leqq 6du$, $|\delta_3| \leqq 4du$. For α_1 we have

$$\alpha_1 = [(1 + \delta_4)(d_{11}^{(m-1)} - p) + (1 + \delta_5)q^2\gamma/\alpha]/[(1 + \delta_3)\beta_1], \tag{4.5}$$

where $|\delta_4| \leqq 7du$, $|\delta_5| \leqq 9du$. Combining (4.5) and the (1, 1) part of (4.3) leads us to set $f_1^{(1)} := \delta_3\beta_1$ and $f_{11}^{(1)} := -g_{11}^{(1)} + \delta_4(d_{11}^{(m-1)} - p) + \delta_5 q^2\gamma/\alpha$. Using (4.4) and the (1, 2) part of (4.3) leads to $f_{12}^{(1)} := -g_{12}^{(1)} + \delta_1 d_{12}^{(m-1)} + \delta_2 q^2\gamma^2/(\alpha\beta)$. It suffices to set all other entries of $F^{(1)}$ and $\mathbf{f}^{(1)}$ to zero, except for $f_{21}^{(1)}$, $f_{22}^{(1)}$, $f_{31}^{(1)}$, and $f_{32}^{(1)}$, which we find in the following way. From (4.1), we find that there exists unique scalars $\tilde{\alpha}_1$ and $\tilde{\alpha}_2$ so that

$$D^{(m-2)}[\mathbf{x}^{(m-2)}, \mathbf{y}^{(m-2)}] - [\eta^{(m-2)}, \zeta^{(m-2)}] - [\mathbf{x}^{(m-2)}, \mathbf{y}^{(m-2)}]S = \mathbf{b}^{(0)}[\tilde{\alpha}_1, \tilde{\alpha}_2]R.$$

Multiplying on the left by $V_{m-2}U_{m-1}$ and on the right by R^{-1}, we have

$$(D^{(m-1)} + G^{(1)})[\mathbf{e}_1, \mathbf{e}_2] - \mathbf{b}^{(1)}[\tilde{\alpha}_1, \tilde{\alpha}_2] = [\mathbf{e}_1, \mathbf{e}_2]\hat{S},$$

from which we find that

$$\begin{aligned}
&\text{(a)}\ d_{11}^{(m-1)} + g_{11}^{(1)} - \beta_1\tilde{\alpha}_1 = \hat{s}_{11}, && \text{(d)}\ d_{12}^{(m-1)} + g_{12}^{(1)} - \beta_1\tilde{\alpha}_2 = \hat{s}_{12},\\
&\text{(b)}\ d_{21}^{(m-1)} + g_{21}^{(1)} - \beta_2\tilde{\alpha}_1 = \hat{s}_{21}, && \text{(e)}\ d_{22}^{(m-1)} + g_{22}^{(1)} - \beta_2\tilde{\alpha}_2 = \hat{s}_{22},\\
&\text{(c)}\ d_{31}^{(m-1)} + g_{31}^{(1)} - \beta_3\tilde{\alpha}_1 = 0, && \text{(f)}\ d_{32}^{(m-1)} + g_{32}^{(1)} - \beta_3\tilde{\alpha}_2 = 0.
\end{aligned} \tag{4.6}$$

From (4.6) (a) and (b) we have

$$d_{21}^{(m-1)} + g_{21}^{(1)} - (\beta_2/\beta_1)(d_{11}^{(m-1)} + g_{11}^{(1)} - \hat{s}_{11}) - \hat{s}_{21} = 0.$$

Comparing this with the (2, 1) part of (4.3) leads us to define $f_{21}^{(1)}$ so that

$$f_{21}^{(1)} - \beta_2\alpha_1 + (\beta_2/\beta_1)(d_{11}^{(m-1)} + g_{11}^{(1)} - \hat{s}_{11}) = 0.$$

Using formula (4.5) for α_1, the definition of $\hat{s}_{11}$, and the assumption that $|\beta_1| \geqq |\beta_2|$, we find that

$$|f_{21}^{(1)}| \leqq |\delta_6(d_{11}^{(m-1)} - p) - g_{11}^{(1)} + \delta_7 q^2\gamma/\alpha|,$$

where $|\delta_6| \leqq 11du$, $|\delta_7| \leqq 13du$. Similar considerations lead to

$$|f_{22}^{(1)}| \leqq |\delta_3/(1+\delta_3)|\,|d_{22}^{(m-1)} - p + g_{22}^{(1)} - q^2\gamma/\alpha| + |1/(1+\delta_3)|\,|\delta_1 d_{12}^{(m-1)} - g_{12}^{(1)} - \delta_2 q^2\gamma^2/(\alpha\beta)|,$$

$$|f_{31}^{(1)}| \leqq |\delta_8(d_{11}^{(m-1)} - p) - g_{11}^{(1)} + \delta_9 q^2\gamma/\alpha|,$$

where $|\delta_8| \leqq 11du$, $|\delta_9| \leqq 13du$, and

$$|f_{32}^{(1)}| \leqq |\delta_3/(1+\delta_3)|\,|d_{32}^{(m-1)} + g_{32}^{(1)}| + |1/(1+\delta_3)|\,|\delta_1 d_{12}^{(m-1)} - g_{12}^{(1)} - \delta_2 q^2\gamma^2/(\alpha\beta)|.$$

With these bounds on the entries of $F^{(1)}$, we have

$$\|F^{(1)}\|_F \leqq 11du\sqrt{3}\,\|D^{(m-1)} - pI\|_F + \sqrt{3}\|G^{(1)}\|_F + 12du\sqrt{3}q^2\gamma^2/|\alpha\beta| + 26duq^2|\gamma/\alpha|. \tag{4.7}$$

Case 2 ($|\beta_2| \geqq \max\{|\beta_1|, |\beta_3|\}$). The floating point result for α_1 can be written as

$$\alpha_1 = [(1+\delta_1)d_{21}^{(m-1)} + q^2\beta/\alpha]/[(1+\delta_2)\beta_2], \tag{4.8}$$

where $|\delta_1| \leqq 4du$, $|\delta_2| \leqq 4du$. For α_2 we have

$$\alpha_2 = [(1+\delta_3)(d_{22}^{(m-1)} - p) + (1+\delta_4)q^2\gamma/\alpha]/[(1+\delta_2)\beta_2], \tag{4.9}$$

where $|\delta_3| \leqq 7du$, $|\delta_4| \leqq 9du$. Combining (4.8) and the (2, 1) part of (4.3) leads us to set $f_2^{(1)} := \delta_2\beta_2$ and $f_{21}^{(1)} := -g_{21}^{(1)} + \delta_1 d_{21}^{(m-1)}$. Using (4.9) and the (2, 2) part of (4.3) leads to $f_{22}^{(1)} := \delta_3(d_{22}^{(m-1)} - p) - g_{22}^{(1)} - \delta_4 q^2\gamma/\alpha$. It suffices to set all other entries of $F^{(1)}$ and $\mathbf{f}^{(1)}$ to zero, except for $f_{11}^{(1)}$, $f_{12}^{(1)}$, $f_{31}^{(1)}$, and $f_{32}^{(1)}$. Using the same technique as in the previous case, we find that

$$|f_{11}^{(1)}| \leqq |\delta_2/(1+\delta_2)|\,|d_{11}^{(m-1)} - p + g_{11}^{(1)} + q^2\gamma/\alpha| + |1/(1+\delta_2)|\,|\delta_1 d_{21}^{(m-1)} + g_{21}^{(1)}|,$$

$$|f_{12}^{(1)}| \leqq |\delta_5(d_{22}^{(m-1)} - p) - g_{22}^{(1)} + \delta_4 q^2\gamma/\alpha|,$$

where $|\delta_5| \leqq 11du$, and

$$|f_{31}^{(1)}| \leqq |\delta_2/(1+\delta_2)|\,|d_{31}^{(m-1)} + g_{31}^{(1)}| + |1/|(1+\delta_2)|\,|\delta_1 d_{21}^{(m-1)} - g_{21}^{(1)}|,$$

$$|f_{32}^{(1)}| \leqq |\delta_6(d_{22}^{(m-1)} - p) - g_{22}^{(1)} - \delta_7 q^2\gamma/\alpha|,$$

where $|\delta_6| \leqq 11du$, $|\delta_7| \leqq 13du$. With these bounds on the magnitudes of the entries of $F^{(1)}$, we have

$$\|F^{(1)}\|_F \leqq 11du\sqrt{3}\,\|D^{(m-1)} - pI\|_F + \sqrt{3}\|G^{(1)}\|_F + 26duq^2|\gamma/\alpha|. \tag{4.10}$$

Case 3 ($|\beta_3| \geqq \max\{|\beta_1|, |\beta_2|\}$). The floating point representations for α_1 and α_2 can be written as

$$\alpha_1 = (1+\varepsilon_1)d_{31}^{(m-1)}/\beta_3, \tag{4.11}$$

$$\alpha_2 = (1+\varepsilon_2)d_{32}^{(m-1)}/\beta_3, \tag{4.12}$$

where $|\varepsilon_1|, |\varepsilon_2| \leqq du$. It suffices to set all entries of $F^{(1)}$ and $\mathbf{f}^{(1)}$ to zero, except for $f_{11}^{(1)}$, $f_{12}^{(1)}$, $f_{21}^{(1)}$, $f_{22}^{(1)}$, $f_{31}^{(1)}$, and $f_{32}^{(1)}$. Combining (4.11) and the (3, 1) part of (4.3) leads us to set $f_{31}^{(1)} := \varepsilon_1 d_{31}^{(m-1)} - g_{31}^{(1)}$. Similarly, $f_{32}^{(1)} := \varepsilon_2 d_{32}^{(m-1)} - g_{32}^{(1)}$. Proceeding

as in the other cases, we find that

$$|f_{11}^{(1)}| \leqq |\varepsilon_1 d_{31}^{(m-1)} - g_{31}^{(1)}|, \qquad |f_{12}^{(1)}| \leqq |\varepsilon_2 d_{32}^{(m-1)} - g_{32}^{(1)}|,$$
$$|f_{21}^{(1)}| \leqq |\varepsilon_1 d_{31}^{(m-1)} - g_{31}^{(1)}|, \qquad |f_{22}^{(1)}| \leqq |\varepsilon_2 d_{32}^{(m-1)} - g_{32}^{(1)}|.$$

With these bounds on the magnitudes of the entries of $F^{(1)}$, we have

$$\|F^{(1)}\|_F \leqq du\sqrt{3}\|D^{(m-1)} - pI\|_F + \sqrt{3}\|G^{(1)}\|_F. \tag{4.13}$$

Setting $F := QF^{(1)}Q^T$, the desired inequality for F follows from (4.7), (4.10), and (4.13). Setting $\mathbf{f} := Q\mathbf{f}^{(1)}$, the inequality for $\mathbf{f}$ follows from the above definitions. □

Proposition 4.2 verifies that through a simple restart procedure, the amplification factors Factor_2, Factor_3, and Factor_4 can be made negligible.

PROPOSITION 4.2. *Suppose that* C-PCK *deflation is repeated with starting values* $\tilde{x}_m^{(0)} = 1 - bq^2$ *and* $\tilde{y}_m^{(0)} = 1 + b$, *where* b *is the root of*

$$-\alpha\gamma q^2 b^2 + (\alpha^2 - \gamma^2 q^2 - \beta^2 q^2)b + \alpha\gamma = 0,$$

which maximizes $|\tilde{x}_m^{(0)}|$. *Then* $\tilde{\mathbf{x}}^{(m-1)} := \tilde{\alpha}\mathbf{e}_1$ *and* $\tilde{\mathbf{y}}^{(m-1)} := \tilde{\gamma}\mathbf{e}_1 + \tilde{\beta}\mathbf{e}_2$, *with* $\tilde{\gamma} = O(u)$.

Proof. First, suppose that the computation is done in exact arithmetic. Define T, a 2×2 matrix, by $t_{11} = 1$, $t_{12} = b$, $t_{21} = -bq^2$, and $t_{22} = 1$. From $[\tilde{\mathbf{x}}^{(m-1)}, \tilde{\mathbf{y}}^{(m-1)}] = [\mathbf{x}^{(m-1)}, \mathbf{y}^{(m-1)}]T$ and $\text{TST}^{-1} = S$, it follows that $\tilde{\mathbf{x}}^{(m-1)T}\tilde{\mathbf{y}}^{(m-1)} = 0$. Since $\tilde{\mathbf{x}}^{(m-1)} = \tilde{\alpha}\mathbf{e}_1$ and $\tilde{\mathbf{y}}^{(m-1)} = \tilde{\gamma}\mathbf{e}_1 + \tilde{\beta}\mathbf{e}_2$, $\tilde{\gamma} = 0$. Using floating point arithmetic, we find that $\tilde{\alpha}\tilde{\gamma} = \tilde{\mathbf{x}}^{(m-1)T}\tilde{\mathbf{y}}^{(m-1)} = O(u)$, and hence $|\tilde{\gamma}| \leqq O(u)/|\tilde{\alpha}| \leqq O(u)/|\tilde{x}_m^{(0)}|$. □

The conclusion of Proposition 4.1 is simplified if C-PCK deflation is repeated in the event that Factor_2, or Factor_3, or Factor_4 is too large. In our implementation of the PCK algorithm, C-PCK deflation is repeated if

$$\max\{\text{Factor}_2, \text{Factor}_3, \text{Factor}_4\} > 10,$$

in which case the conclusion of Proposition 4.1 may be modified so that

$$\frac{\|G + F\|_F}{\|D^{(0)}\|_F} \leqq [468 + \text{Factor}_1(330m - 310)]du \leqq 330(\text{Factor}_1 m + 2)u. \tag{4.14}$$

If we did not include repetition of C-PCK deflation in the PCK algorithm, our estimate for $\|\Delta A\|_F$ in Theorem 1.1 would contain Factor_2, Factor_3, and Factor_4 and our result would be degraded. We point out, however, that in our implementation of the PCK algorithm, we have observed that C-PCK deflation is repeated only for extremely ill conditioned problems. For such problems, we have yet to observe a case where repeating C-PCK deflation changed the computed gain vector significantly. Next, we analyze Steps 2 and 3 of the PCK algorithm.

PROPOSITION 4.3. *Suppose that the* PCK *algorithm is applied using floating point arithmetic to the pair* $(D^{(0)}, \mathbf{b}^{(0)})$ *where* $D^{(0)} \in R^{n\times n}$ *is unreduced upper Hessenberg and* $\mathbf{b}^{(0)} = b_1^{(0)}\mathbf{e}_1$ *with* $b_1^{(0)} \neq 0$. *Let* $\lambda_1, \cdots, \lambda_{n_r}, p_1 \pm iq_1, \cdots, p_{n_c} \pm iq_{n_c}$ *denote the eigenvalues to be assigned.*

Then there exists positive constants C_1, C_2, *and* C_3 *and matrices* Z, $\Delta D^{(0)} \in R^{n\times n}$ *and* $\Delta\mathbf{b}^{(0)} \in R^n$ *with* Z *orthogonal so that*

$$D^{(0)} + \Delta D^{(0)} + (\mathbf{b}^{(0)} + \Delta\mathbf{b}^{(0)})\mathbf{k}^{(0)T} = Z \begin{bmatrix} \lambda_1 & & & & \\ & \ddots & & & \\ & & \lambda_{n_r} & & \\ & & & \hat{S}_1 & \\ & & & & \ddots \\ \mathbf{0} & & & & \hat{S}_{n_c} \end{bmatrix} Z^T, \tag{4.15}$$

where

$$\frac{\|\Delta D^{(0)}\|_F}{\|D^{(0)}\|_F} \leqq C_1 n_r(n + 2n_c + 1)u + C_2 \max_{1 \leqq i \leqq n_c} \left\{ \frac{\|D^{(0)} - p_i I\|_F}{\|D^{(0)}\|_F} \right\} n_c(n_c + 1)u + C_2 2n_c u + C_3 n_c(n_c - 1)u$$

and

$$\|\Delta \mathbf{b}^{(0)}\|_2 \leqq \|\mathbf{b}^{(0)}\|_2 C_3 n u.$$

Proof by induction on n. Set $C_3 := 2du$. If $n_r \neq 0$, we apply PCK deflation with $m = n$ and $\lambda = \lambda_1$, and following the proof of Proposition 3.2 in [CM], we obtain our result. As an alternative, the method used in the proof of Proposition 4.1 can be applied to the real case.

If $n_r = 0$, we apply C-PCK deflation with $m = n = 2n_c$ and $p = p_1$, $q = q_1$. We find that

$$D^{(n-1)} + F^{(1)} + G^{(1)} + \left\{ \begin{bmatrix} \beta_1 \\ \beta_2 \\ \mathbf{b}^{(1)} \end{bmatrix} + \mathbf{f}^{(1)} \right\} [\alpha_1, \alpha_2, \alpha^T] = \left[\begin{array}{c|c} \hat{S}_1 & \\ \hline 0 & D_{22}^{(n-1)} - \mathbf{b}^{(1)} \alpha^T \end{array} \right], \tag{4.16}$$

where

$$\|F^{(1)} + G^{(1)}\|_F \leqq \|D^{(0)}\|_F C_2 \left(\frac{\|D^{(0)} - p_1 I\|_F}{\|D^{(0)}\|_F} 2n_c + 2 \right) u,$$

$$\|\mathbf{f}^{(1)}\|_2 \leqq \|\mathbf{b}^{(0)}\|_2 C_3 2u, \quad \mathbf{b}^{(1)} = \beta_3 \mathbf{e}_1.$$

The const C_2 is equal to 330 according to (4.14). The PCK algorithm proceeds by assigning the remaining eigenvalues to $(D_{22}^{(n-1)}, \mathbf{b}^{(1)})$. Denote the resulting gain vector by $\mathbf{k}^{(1)}$. From the induction hypothesis, we have matrices $Z^{(1)}$, $\Delta D_{22}^{(n-1)} \in R^{(n-2)\times(n-2)}$, with $Z^{(1)}$ orthogonal, and an $n - 2$ vector $\Delta \mathbf{b}^{(1)}$ so that

$$\frac{\|\Delta D_{22}^{(n-1)}\|_F}{\|D_{22}^{(n-1)}\|_F} \leqq C_2 \max_{2 \leqq i \leqq n_c} \left\{ \frac{\|D^{(0)} - p_i I\|_F}{\|D^{(0)}\|_F} \right\} (n_c - 1)n_c u + C_2 2(n_c - 1)u + C_3(n_c - 1)(n_c - 2)u,$$

$$\|\Delta \mathbf{b}^{(1)}\|_2 \leqq \|\mathbf{b}^{(1)}\|_2 C_3 (n - 2)u,$$

and

$$D_{22}^{(n-1)} + \Delta D_{22}^{(n-1)} + (\mathbf{b}^{(1)} + \Delta \mathbf{b}^{(1)})\mathbf{k}^{(1)T} = Z^{(1)} \begin{bmatrix} \hat{S}_2 & & \\ & \ddots & \\ \mathbf{0} & & \hat{S}_{n_c} \end{bmatrix} Z^{(1)\mathrm{T}}. \tag{4.17}$$

Now set $W := \operatorname{diag}(I_2, Z^{(1)})$, $H := \operatorname{diag}(\Theta_2, \Delta D_{22}^{(n-1)})$, where I_2 denotes the 2×2 identity matrix and Θ_2 the 2×2 zero matrix, and set $J := [0, 0, \Delta \mathbf{b}^{(1)T}]^T [\alpha_1, \alpha_2, \mathbf{0}]$. Since $f_{ij}^{(1)} \neq 0$ only if $1 \leqq i \leqq 3$ and $1 \leqq j \leqq 2$, and $f_i^{(1)} \neq 0$ only if $i = 1, 2$, we find using (4.16) and (4.17) that

$$D^{(n-1)} + F^{(1)} + G^{(1)} + H + J - \left\{ \begin{bmatrix} \beta_1 \\ \beta_2 \\ \mathbf{b}^{(1)} \end{bmatrix} + \mathbf{f}^{(1)} + \begin{bmatrix} 0 \\ 0 \\ \Delta \mathbf{b}^{(1)} \end{bmatrix} \right\} [\alpha_1, \alpha_2, \mathbf{k}^{(1)\mathrm{T}}] = W \begin{bmatrix} \hat{S}_1 & & \\ & \ddots & \\ \mathbf{0} & & \hat{S}_{n_c} \end{bmatrix} W^T.$$

Using (4.3), we obtain the estimate

$$\|J\|_F \leqq |\alpha_1| \|\Delta \mathbf{b}^{(1)}\|_2 + |\alpha_2| \|\Delta \mathbf{b}^{(1)}\|_2 \leqq [|\alpha_1 \beta_3| + |\alpha_2 \beta_3|] C_3(n-2)u$$
$$= \|D^{(0)}\|_F C_3(n-2)u + O(u^2).$$

Also, $\|H\|_F = \|\Delta D_{22}^{(n-1)}\|_F$, $\|D_{22}^{(n-1)}\|_F \leqq \|D^{(0)}\|_F$, and $\|\mathbf{b}^{(1)}\|_2 \leqq \|\mathbf{b}^{(0)}\|_2$. Setting $Z := QW$,

$$\Delta D^{(0)} := Q[F^{(1)} + G^{(1)} + H + J]Q^T \quad \text{and} \quad \Delta \mathbf{b}^{(0)} := Q\{\mathbf{f}^{(1)} + [0, 0, \Delta \mathbf{b}^{(1)T}]^T\}Q^T,$$

and using the above estimates, we obtain the bounds on $\|D^{(0)}\|_F$ and $\|\Delta \mathbf{b}^{(0)}\|_2$. □

In conclusion, we prove our main result.

Proof of Theorem 1.1. Theorem 1.1 is a corollary of Proposition 4.3. The effect of round-off error on the orthogonal transformations in Proposition 4.3 can be analyzed as in [CM]. Keeping only the highest-order terms in n, n_r, and n_c, we find that Z must be replaced by $Z + \Delta Z$ with $\|\Delta Z\|_F = O[n_r(n + 2n_c)u]$ and that the estimates in the conclusion of this proposition must be changed to

$$\frac{\|\Delta D^{(0)}\|_F}{\|D^{(0)}\|_F} = O\left[n_r\left(2nn_c + \frac{n_r^2}{3}\right)u\right] + O\left[\max_{1 \leqq i \leqq n_c}\left\{\frac{\|D^{(0)} - p_i I\|_F}{\|D^{(0)}\|_F}\right\} n_c^3 u\right] + O(n_c^3 u),$$

$$\frac{\|\Delta \mathbf{b}^{(0)}\|_2}{\|\mathbf{b}^{(0)}\|_2} = O[n_r(n + 2n_c)u] + O(n_c^2 u).$$

The floating point analysis of the transformation back to the original coordinate system appears in the proof of Theorem 1.1 in [CM]. Finally, we have

$$A + \Delta A + (\mathbf{b} + \Delta \mathbf{b})\mathbf{k}^T = (T + \Delta T)\begin{bmatrix} \lambda_1 & & & & \\ & \ddots & & & \\ & & \lambda_{n_r} & & \\ & & & \hat{S}_1 & \\ & & & & \ddots \\ \mathbf{0} & & & & & \hat{S}_{n_c} \end{bmatrix}(T + \Delta T)^{-1},$$

with T orthogonal and with the same estimates for $\|\Delta A\|_F / \|A\|_F$, $\|\Delta \mathbf{b}\|_2 / \|\mathbf{b}\|_2$, and $\|\Delta T\|_F$ as for $\|\Delta D^{(0)}\|_F / \|D^{(0)}\|_F$, $\|\Delta \mathbf{b}^{(0)}\|_2 / \|\mathbf{b}^{(0)}\|_2$, and $\|\Delta Z\|_F$. □

REFERENCES

[CM] C. L. Cox and W. F. Moss, *Backward error analysis for a pole assignment algorithm*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 446–456.

[K] T. Kailath, *Linear Systems*, Prentice-Hall, Englewood Cliffs, NJ, 1980.

[PCK] P. Hr. Petkov, N. D. Christov, and M. M. Konstantinov, *A computational algorithm for pole assignment of linear single-input systems*, IEEE Trans. Automat. Control, AC-29 (1984), pp. 1045–1048.

[R] D. L. Russell, *Mathematics of Finite-Dimensional Control Systems*, Marcel Dekker, New York, 1979.

[vD] P. M. van Dooren, *The generalized eigenstructure problem in linear system theory*, IEEE Trans. Automat. Control, 26 (1981), pp. 111–129.

[W] J. H. Wilkinson, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, U.K., 1965.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1172–1188, October 1992

013

CIRCULATIVE MATRICES OF DEGREE θ*

HSIN-CHU CHEN†

Abstract. In this paper a special class of matrices W in $\mathcal{C}^{n\times n}$ that are a generalization of reflexive and antireflexive matrices are introduced, their fundamental properties are developed, and a decomposition method associated with W is presented. The matrices W have the relation $W = e^{i\theta}P^*WP$, $i = \sqrt{-1}$, $\theta \in \mathcal{R}$, where e is the exponential function and P an $n \times n$ unitary matrix with the property $P^k = I$, $k \geq 1$. The superscript $*$ denotes the conjugate transpose and I is the identity matrix. It is assumed that k is finite and is the smallest positive integer for which the relation holds. The matrices W are referred to as circulative matrices of degree θ with respect to P. Embedded in this class of matrices are two special types of matrices U and V, $U = P^*UP$ and $V = -P^*VP$, which bear a great resemblance to reflexive matrices and antireflexive matrices, respectively. The matrices U and V are simply called circulative matrices and anticirculative matrices, respectively, without referring to their degrees, for the sake of brevity. These matrices are introduced and general theories associated with them are developed. Then their special cases are discussed, and, in particular, two more special classes of matrices are defined, which will be referred to as rotative and antirotative matrices. These are a special case of the circulative/anticirculative matrices on one hand and a generalization of (block) circulant/anticirculant matrices on the other. Numerical examples are presented.

Key words. centrosymmetric matrices, circulant matrices, circulation matrices, circulative matrices of degree θ, circulative (anticirculative) matrices, circulative decomposition method, dihedral matrices, group representations, rotation matrices, rotative matrices, reflection matrices, reflexive matrices

AMS(MOS) subject classifications. 15A03, 15A18, 15A59, 20C30

1. Introduction.

In [ChSa87] and [Chen88] we presented two special classes of matrices A and B in $\mathcal{C}^{n\times n}$ with the relations

$$A = PAP \quad \text{and} \quad B = -PBP, \tag{1.1}$$

where P is some reflection (symmetric signed permutation) matrix. The matrices A and B are referred to as reflexive and antireflexive matrices, respectively. The special properties of these matrices and their applications to the numerical solution of elasticity problems are further exploited in [ChSa89a] and [ChSa89b]. In this paper, we first introduce a generalization of these two special classes of matrices and develop basic theories associated with this generalization. The new matrices, for example, W in $\mathcal{C}^{n\times n}$ have the relation

$$W = e^{i\theta}P^*WP, \quad i = \sqrt{-1}, \quad \theta = \frac{2\pi}{k}j, \tag{1.2}$$

where j and k are integers, e the exponential function, and P an $n \times n$ unitary matrix with the following property:

$$P^k = I, \qquad k \geq 1. \tag{1.3}$$

The superscript $*$ denotes the conjugate transpose; I is the identity matrix. We assume that k is finite and is the smallest positive integer for which (1.3) holds.

* Received by the editors April 9, 1990; accepted for publication (in revised form) January 31, 1991.

† Center for Supercomputing Research and Development, University of Illinois, 305 Talbot Laboratory, 104 South Wright St., Urbana, Illinois 61801-2932 (hchen@csrd.uiuc.edu). This work was supported by Department of Energy grant DOE DE-FG02-85ER25001 and National Science Foundation grant NSF CCR-8717942.

We call the matrix P a circulation matrix in this paper for two reasons: (1) all the eigenvalues of P lie on the unit circle of the complex plane, and (2) the powers of P circulate around I, P, $P^2, \cdots$, and P^{k-1}. The matrices W are, therefore, referred to as circulative matrices of degree θ.

Embedded in the class of matrices W are two special types of matrices U and V that bear a great resemblance to the matrices A and B in (1.1). In other words, the matrices U and V have the relations

$$U = P^* U P \quad \text{and} \quad V = -P^* V P \,, \tag{1.4}$$

which correspond to the matrices W with $\theta = 0$ and $\theta = \pi$, respectively. Evidently, the matrices U are circulative of degree 0 and the matrices V circulative of degree π. For the sake of brevity in this paper, however, we call a matrix that is circulative of degree 0 a circulative matrix and a matrix that is circulative of degree π an anticirculative matrix, without referring to their degrees. After introducing these special classes of matrices, we then discuss some of their special cases. In particular, we define another two special classes of matrices, to be referred to as rotative and antirotative matrices, which are a generalization of (block) circulant/anticirculant matrices [Davi79].

The matrices U possess several special properties that are not only theoretically interesting but computationally useful. In addition to including reflexive matrices as a special case, this class of matrices also contains rotative matrices as another important special one. Reflexive matrices and rotative matrices frequently arise from a very wide class of scientific and engineering applications. Numerical examples derived from the discretization of certain partial differential equations are presented for demonstration purposes. The matrices V, which do not seem to appear naturally from physical problems, are the counterpart of the matrices U and include antirotative matrices and antireflexive matrices as special cases.

Since the two classes of matrices U and V can be embedded in the class of matrices W, in what follows we shall mainly address the special properties possessed by W and show the necessary and sufficient conditions to split an arbitrary matrix A in $\mathcal{C}^{n\times n}$ into matrices W_1, W_2, $\cdots$, W_m, $1 \le m \le k$ such that they are circulative of different degrees with respect to the same circulation matrix P. In addition, a decomposition method referred to as the circulative decomposition method will also be presented for the solution of nonsingular linear systems $Ax = b$ with A being circulative of degree θ with respect to some circulation matrix P. The special properties and decompositions associated with both U and V can easily be deduced from those of W and are, therefore, only briefly stated.

2. The special matrices and vectors. Unless otherwise noted, matrices, vectors, and scalars are represented by the uppercase roman, the lowercase roman, and the lowercase Greek letters, respectively, throughout the paper. We use the superscripts T, $*$, -1, and $+$ to denote the transpose, conjugate transpose, inverse, and generalized inverse of matrices (or vectors), respectively. All matrix-matrix and matrix-vector multiplications are assumed to be conformable. The subscripts and superscripts i, j, k, $\cdots$, n are considered to be positive integers except that i associated with the exponential function e will be used exclusively for $\sqrt{-1}$ in this section. The symbol θ is assumed to be a real number. Before presenting the special properties of the two classes of matrices mentioned previously, we give some basic definitions.

DEFINITION 2.1. Circulation matrices. Any matrix $P \in \mathcal{C}^{n\times n}$ is defined to be a circulation matrix if (1) P is unitary and (2) $P^k = I$ for some integer $k \ge 1$.

Note that the matrix P can be considered to be a matrix representation of the cyclic group $G = \{1, r, r^2, \cdots, r^{k-1}\}$ generated by some element r of order k, frequently encountered in the area of group representation theory [CuRe62], [Hill75] and its applications [Satt79], [MuIk89].

DEFINITION 2.2. Circulative vectors of degree θ. Let P be some circulation matrix of dimension n. A vector $x \in \mathcal{C}^n$ is said to be circulative of degree θ with respect to P if $Px = e^{i\theta}x$.

DEFINITION 2.3. Circulative matrices of degree θ. Let P be some circulation matrix of dimension n. A matrix $A \in \mathcal{C}^{n\times n}$ is said to be circulative of degree θ with respect to P if $A = e^{i\theta}P^*AP$.

It should be noted that if $e^{i\theta}$ is an eigenvalue of P with $P^k = I$, then $e^{ik\theta} = 1$. The converse, however, is not true in general. Therefore, in the definition of circulative matrices of degree θ, we do not assume $e^{i\theta}$ to be an eigenvalue of P although $e^{ik\theta}$ must be unity for $A \neq 0$.

DEFINITION 2.4. Circulative subspaces of degree θ. Let P be some circulation matrix of dimension n. A subspace $\mathcal{V} \subset \mathcal{C}^n$ is said to be circulative of degree θ with respect to P if $Px = e^{i\theta}x$ for every $x \in \mathcal{V}$. A subspace $\mathcal{M} \subset \mathcal{C}^{n\times n}$ is also said to be circulative of degree θ with respect to P if $A = e^{i\theta}P^*AP$ for every $A \in \mathcal{M}$.

In the following, we define $\mathcal{C}_\theta^n(P)$ and $\mathcal{C}_\theta^{n\times n}(P)$ to be, respectively, the subspaces of vectors and matrices that are circulative of degree θ with respect to P. In other words,

$$\mathcal{C}_\theta^n(P) = \{x \mid x \in \mathcal{C}^n \text{ and } Px = e^{i\theta}x\}$$

and

$$\mathcal{C}_\theta^{n\times n}(P) = \{A \mid A \in \mathcal{C}^{n\times n} \text{ and } A = e^{i\theta}P^*AP\},$$

where P is some circulation matrix of dimension n. Note that $\mathcal{C}_\theta^n(P)$ is the eigenspace of P corresponding to the eigenvalue $e^{i\theta}$. Therefore, from the spectral theory of normal matrices [HoJo85], it can easily be shown that $\mathcal{C}_{\theta_1}^n(P)$ and $\mathcal{C}_{\theta_2}^n(P)$ are mutually orthogonal if $e^{i\theta_1} \neq e^{i\theta_2}$, since P is normal. Although it is not the purpose of this paper to discuss group representation theory, it is worth noting that the set of all nonsingular matrices U that are circulative (of degree 0) with respect to the same matrix P form a matrix group. The same, however, cannot be said for matrices circulative of degree that is not a multiple of 2π, including anticirculative matrices V. We are now in a position to introduce the special properties related to these vectors and matrices.

LEMMA 2.5. *A subspace of matrices or vectors is circulative of degree θ with respect to some circulation matrix P if and only if it is a subspace circulative of degree $-\theta$ with respect to P^*; i.e., $\mathcal{C}_\theta^n(P) = \mathcal{C}_{-\theta}^n(P^*)$ and $\mathcal{C}_\theta^{n\times n}(P) = \mathcal{C}_{-\theta}^{n\times n}(P^*)$.*

Proof. Since P is unitary, we have $P^* = P^{-1}$, and therefore $Px = e^{i\theta}x$ if and only if $P^*x = e^{-i\theta}x$. Likewise, $A = e^{i\theta}P^*AP$ if and only if $A = e^{-i\theta}PAP^* = e^{-i\theta}(P^*)^*AP^*$. Thus the proof is trivial.

LEMMA 2.6. *Let α and $\beta \in \mathcal{C}$.*

(i) *If A and $B \in \mathcal{C}_\theta^{n\times n}(P)$, then $(\alpha A + \beta B) \in \mathcal{C}_\theta^{n\times n}(P)$, $(\alpha A^* + \beta B^*) \in \mathcal{C}_{-\theta}^{n\times n}(P)$, and $(\alpha A^+ + \beta B^+) \in \mathcal{C}_{-\theta}^{n\times n}(P)$.*

(ii) *If $A \in \mathcal{C}_{\theta_1}^{n\times n}(P)$ and $B \in \mathcal{C}_{\theta_2}^{n\times n}(P)$, then $(\alpha A^*A + \beta B^*B)$ and $(\alpha AA^* + \beta BB^*) \in \mathcal{C}_0^{n\times n}(P)$; but $AB \in \mathcal{C}_\omega^{n\times n}(P)$ where $\omega = \theta_1 + \theta_2$.*

Proof. The proof for $(\alpha A + \beta B)$ and $(\alpha A^* + \beta B^*)$ in (i) and for (ii) is trivial. Therefore, we present only the proof for $(\alpha A^+ + \beta B^+)$. To prove $(\alpha A^+ + \beta B^+) \in \mathcal{C}_{-\theta}^{n\times n}(P)$ in (i), where A and B are assumed to be circulative of degree θ with respect to P, it suffices to prove that A^+ is circulative of degree $-\theta$ with respect to P. The generalized inverse of a matrix A is typically defined to be the unique matrix X that satisfies the following four Moore–Penrose conditions [Davi79]:

(a) $AXA = A$,
(b) $XAX = X$,
(c) $(AX)^* = AX$, and
(d) $(XA)^* = XA$.

Premultiplying and postmultiplying both sides of (a) by P^* and P, respectively, and using the fact that P is unitary, we obtain

$$P^*APP^*A^+PP^*AP = P^*AP, \tag{2.1}$$

where we have replaced X by A^+ since A^+ is the generalized inverse of A. Substitution of $A = e^{i\theta}P^*AP$ into (2.1) yields

$$AYA = A$$

where $Y = e^{-i\theta}P^*A^+P$. Note that Y satisfies the first Moore–Penrose condition. Applying the same procedures to (b), (c), and (d) shows that Y also satisfies all three of these Moore–Penrose conditions. Therefore, Y is a generalized inverse of A. Since the Moore–Penrose inverse is known to be unique, we conclude that $A^+ = Y$ and, therefore,

$$A^+ = e^{-i\theta}P^*A^+P \in \mathcal{C}_{-\theta}^{n\times n}(P). \tag{2.2}$$

Likewise,

$$B^+ = e^{-i\theta}P^*B^+P \in \mathcal{C}_{-\theta}^{n\times n}(P). \tag{2.3}$$

Hence, from (2.2) and (2.3) we conclude $(\alpha A^+ + \beta B^+) \in \mathcal{C}_{-\theta}^{n\times n}(P)$. □

From Lemma 2.6, it is clear that the sum of two matrices that are both circulative of degree θ with respect to a given P is itself circulative of degree θ with respect to the same P. The product of two matrices of which one is circulative of degree θ_1 and the other circulative of degree θ_2 with respect to a given P is circulative of degree $\theta_1 + \theta_2$ with respect to the same P.

In the following, we use P^{l*} to denote $(P^l)^*$, the conjugate transpose of the lth power of P. It is worth noting that if $A \in \mathcal{C}_{\theta}^{n\times n}(P)$, then $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$ for any integer m. The converse, however, is not true in general.

THEOREM 2.7. *Circulative decomposition of matrices. Let P be an $n \times n$ circulation matrix with $P^k = I$. A matrix $A \in \mathcal{C}^{n\times n}$ can be decomposed into $W_0, W_1, \cdots, W_{m-1}$, $A = \sum_{j=0}^{m-1} W_j$, such that $W_j \in \mathcal{C}_{\omega_j}^{n\times n}(P)$ for $0 \le j \le m-1$ if and only if $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$ where $\theta \in \mathcal{R}$ with $e^{ik\theta} = 1$ and $\omega_j = \theta + (2\pi j/m)$, $1 \le m \le k$.*

Proof. (a) If $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$, then $A = e^{im\theta}P^{m*}AP^m$ by definition. By taking

$$W_j = \frac{1}{m}\sum_{l=0}^{m-1} e^{il\omega_j}P^{l*}AP^l, \qquad 0 \le j \le m-1, \tag{2.4}$$

we immediately obtain

$$\sum_{j=0}^{m-1} W_j = \frac{1}{m} \sum_{j=0}^{m-1} \sum_{l=0}^{m-1} e^{il\omega_j} P^{l*} A P^l$$
$$= \frac{1}{m} \sum_{l=0}^{m-1} \left(\sum_{j=0}^{m-1} e^{il\omega_j} \right) P^{l*} A P^l = A,$$

where we have interchanged the order of summations and used the geometric series identity [Davi79]

$$\sum_{j=0}^{m-1} e^{il\omega_j} = \begin{cases} m & \text{if } l = 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

Now, premultiplying W_j in (2.4) by $e^{i\omega_j} P^*$ and postmultiplying W_j by P yields

$$e^{i\omega_j} P^* W_j P = \frac{1}{m} \sum_{l=0}^{m-1} e^{i(l+1)\omega_j} P^{(l+1)*} A P^{(l+1)}$$
$$= \frac{1}{m} \left[\left(\sum_{l=1}^{m-1} e^{il\omega_j} P^{l*} A P^l \right) + e^{im\omega_j} P^{m*} A P^m \right]$$
$$= \frac{1}{m} \left[\left(\sum_{l=1}^{m-1} e^{il\omega_j} P^{l*} A P^l \right) + e^{im\theta} P^{m*} A P^m \right]$$
$$= \frac{1}{m} \left[\left(\sum_{l=1}^{m-1} e^{il\omega_j} P^{l*} A P^l \right) + A \right]$$
$$= \frac{1}{m} \sum_{l=0}^{m-1} e^{il\omega_j} P^{l*} A P^l$$
$$= W_j.$$

Note that $e^{im\omega_j} = e^{im(\theta+(2\pi j/m))} = e^{im\theta} e^{2\pi j} = e^{im\theta}$. It is clear that $W_j \in \mathcal{C}_{\omega_j}^{n\times n}(P)$, $0 \le j \le m-1$. Therefore, the decomposition is completed if $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$.

(b) Now, assume that A can be decomposed into $W_0, W_1, \cdots, W_{m-1}$ such that $W_j \in \mathcal{C}_{\omega_j}^{n\times n}(P)$, $0 \le j \le m-1$. We want to show that $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$.

From $W_j \in \mathcal{C}_{\omega_j}^{n\times n}(P)$ and $e^{im\omega_j} = e^{im\theta}$, we have

$$W_j = e^{i\omega_j} P^* W_j P = e^{i2\omega_j} P^{2*} W_j P^2 = \cdots$$
$$= e^{im\omega_j} P^{m*} W_j P^m = e^{im\theta} P^{m*} W_j P^m. \tag{2.6}$$

Since A is decomposed into W_j, $0 \le j \le m-1$, we obtain from (2.6)

$$A = \sum_{j=0}^{m-1} W_j = \sum_{j=0}^{m-1} e^{im\theta} P^{m*} W_j P^m$$
$$= e^{im\theta} P^{m*} \left(\sum_{j=0}^{m-1} W_j \right) P^m$$
$$= e^{im\theta} P^{m*} A P^m.$$

Therefore, $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$. This completes the proof. □

In the following, we show that the decomposition in Theorem 2.7 is unique (apart from θ) if it exists. In other words, if the matrix A can be decomposed into W_0, W_1, ..., W_{m-1}, $A = \sum_{j=0}^{m-1} W_j$, such that $W_j \in \mathcal{C}_{\omega_j}^{n\times n}(P)$ for $0 \le j \le m-1$, then (2.4) holds. Observe that

$$\begin{aligned}\sum_{l=0}^{m-1} e^{il\omega_j} P^{l*} A P^l &= \sum_{l=0}^{m-1} e^{il\omega_j} P^{l*} \sum_{r=0}^{m-1} W_r P^l = \sum_{l=0}^{m-1}\sum_{r=0}^{m-1} e^{il\omega_j} P^{l*} W_r P^l \\ &= \sum_{r=0}^{m-1}\sum_{l=0}^{m-1} e^{il\omega_j} P^{l*} W_r P^l = \sum_{r=0}^{m-1}\sum_{l=0}^{m-1} e^{il(\omega_j-\omega_r)} e^{il\omega_r} P^{l*} W_r P^l \\ &= \sum_{r=0}^{m-1} \left(\sum_{l=0}^{m-1} e^{il(\omega_j-\omega_r)} \right) W_r = mW_j, \quad 0 \le j \le m-1.\end{aligned}$$

Here we have again used the relations in (2.6) and the geometric series identity (2.5). Obviously, we have

$$W_j = \frac{1}{m} \sum_{l=0}^{m-1} e^{il\omega_j} P^{l*} A P^l, \qquad 0 \le j \le m-1.$$

The decomposition is, therefore, unique if it exists. Note also that the above decomposition is always possible when m is equal to k since $P^k = I$ implies $A \in \mathcal{C}_{k\theta}^{n\times n}(I) = \mathcal{C}_{2\pi}^{n\times n}(I) = \mathcal{C}_0^{n\times n}(I)$. For the particular case of circulative matrices U and anticirculative matrices V, we have the following results.

COROLLARY 2.8. (i) *A matrix $A \in \mathcal{C}^{n\times n}$ can be decomposed into U and V, $A = U + V$, such that $U \in \mathcal{C}_0^{n\times n}(P)$ and $V \in \mathcal{C}_\pi^{n\times n}(P)$ if and only if $A \in \mathcal{C}_0^{n\times n}(P^2)$.*

(ii) *Any matrix $A \in \mathcal{C}^{n\times n}$ can be decomposed into U and V, $A = U + V$, such that $U \in \mathcal{C}_0^{n\times n}(P)$ and $V \in \mathcal{C}_\pi^{n\times n}(P)$ if $P^2 = I$.*

(iii) *If $P^{2i+1} = I$ for some integer i, then $\mathcal{C}_\pi^{n\times n}(P) = \{\mathcal{O}_{n\times n}\}$.*

Proof. Applying Theorem 2.7 with $m = 2$ and $\theta = 0$ to the matrix A yields the result of (i). The statement (i) immediately implies (ii). To prove (iii), we use the fact that if $A \in \mathcal{C}_\theta^{n\times n}(P)$, then $A \in \mathcal{C}_{m\theta}^{n\times n}(P^m)$ for any integer m. For any matrix $V \in \mathcal{C}_\pi^{n\times n}(P)$, we have

$$V = -P^*VP = P^{2*}VP^2 = -P^{3*}VP^3 = \cdots = (-1)^m P^{m*} V P^m.$$

Therefore,

$$V = -V = 0 \quad \text{if } P^{2j+1} = I$$

for some integer j. □

Analogous to circulative matrices of degree θ, vectors that are circulative of degree θ also possess the same circulating nature. In other words, if $b \in \mathcal{C}_\theta^n(P)$, then $b \in \mathcal{C}_{m\theta}^n(P^m)$ for any integer m. The converse is not true either. In particular, for the circulative vectors u and anticirculative vectors v, we have

$$u = Pu = P^2u = \cdots = P^m u$$

and

$$v = -Pv = P^2v = -P^3v = \cdots = (-1)^m P^m v$$

for any integer m.

THEOREM 2.9. *Circulative decomposition of vectors. Let P be an $n \times n$ circulation matrix with $P^k = I$. A vector $b \in \mathcal{C}^n$ can be decomposed into $w_0, w_1, \cdots, w_{m-1}$, $b = \sum_{j=0}^{m-1} w_j$, such that $w_j \in \mathcal{C}^n_{\omega_j}(P)$ for $0 \le j \le m-1$ if and only if $b \in \mathcal{C}^n_{m\theta}(P^m)$ where $\theta \in \mathcal{R}$ with $e^{ik\theta} = 1$ and $\omega_j = \theta + (2\pi j/m)$, $1 \le m \le k$.*

Proof. Take

$$w_j = \frac{1}{m} \sum_{l=0}^{m-1} e^{-il\omega_j} P^l b, \qquad 0 \le j \le m-1. \tag{2.7}$$

The proof is then analogous to that shown in Theorem 2.7 and is therefore omitted. □

COROLLARY 2.10. (i) *A vector $b \in \mathcal{C}^n$ can be decomposed into u and v, $b = u + v$, such that $u \in \mathcal{C}^n_0(P)$ and $v \in \mathcal{C}^n_\pi(P)$ if and only if $b \in \mathcal{C}^n_0(P^2)$.*

(ii) *Any vector $b \in \mathcal{C}^n$ can be decomposed into u and v, $b = u + v$, such that $u \in \mathcal{C}^n_0(P)$ and $v \in \mathcal{C}^n_\pi(P)$ if $P^2 = I$.*

(iii) *If $P^{2j+1} = I$ for some integer j, then $\mathcal{C}^n_\pi(P) = \{\mathcal{O}_n\}$.*

THEOREM 2.11. *Given a linear system $Ax = b$, $A \in \mathcal{C}^{n\times n}$, and b, $x \in \mathcal{C}^n$, if A is nonsingular and circulative of degree θ_1 with respect to some circulation matrix P, then $x \in \mathcal{C}^n_{\theta_2}(P)$ if and only if $b \in \mathcal{C}^n_\omega(P)$ where $\omega = \theta_1 + \theta_2$.*

Theorems 2.9 and 2.11 immediately suggest a decomposition method for the solution of nonsingular linear systems $Ax = b$ with the coefficient matrix A being circulative of degree θ with respect to some circulation matrix P. The right-hand side b can be arbitrary. From Theorem 2.9, it is clear that if $P^k = I$,then any right-hand side b can be decomposed, using (2.7), into k parts, $w_0, w_1, \cdots, w_{k-1}$, such that each part is circulative of some degree with respect to P. It can also be shown that, apart from θ, the circulative decomposition of vectors is unique. Let

$$x_j = A^{-1} w_j, \qquad 0 \le j \le k-1.$$

We have

$$x = A^{-1}b = A^{-1} \sum_{j=0}^{k-1} w_j = \sum_{j=0}^{k-1} A^{-1} w_j = \sum_{j=0}^{k-1} x_j.$$

Therefore, solving such a single system is equivalent to solving the following k independent systems:

$$Ax_j = w_j, \qquad 0 \le j \le k-1. \tag{2.8}$$

To solve the k independent systems, we can use Theorem 2.11 to take advantage of the known information available among the components of the unknowns so long as P is given. This approach therefore is referred to as the circulative decomposition method.

To serve as an example of this special decomposition method, we assume in the following that the matrix A is of dimension $6m$ and is circulative (of degree 0) with respect to the following matrix P:

$$P = \begin{bmatrix} 0 & T & 0 \\ 0 & 0 & T \\ T & 0 & 0 \end{bmatrix} \quad \text{with } T = I_m \otimes \begin{bmatrix} \cos\frac{2\pi}{3} & \sin\frac{2\pi}{3} \\ -\sin\frac{2\pi}{3} & \cos\frac{2\pi}{3} \end{bmatrix},$$

where I_m is the identity matrix of dimension m and $\otimes$ denotes the Kronecker product. Note that the matrix P is a circulation matrix with $P^3 = I$. From Theorem 2.9, the circulative decomposition of the vector b into w_0, w_1, and w_2 yields

$$w_j = \tfrac{1}{3}\left[b + e^{-i\omega_j} Pb + e^{-2i\omega_j} P^2 b\right], \qquad 0 \le j \le 2,$$

where $\omega_j = (2\pi j/3)$. Note that $w_j \in \mathcal{C}^n_{\omega_j}(P)$. We now show how to take advantage of the special properties possessed by the k ($k = 3$ in this example) independent linear systems (2.8). To begin with, let the linear system $Ax_j = w_j$ be uniformly partitioned, in accordance with P, as

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \begin{bmatrix} x_{1j} \\ x_{2j} \\ x_{3j} \end{bmatrix} = \begin{bmatrix} w_{1j} \\ w_{2j} \\ w_{3j} \end{bmatrix}. \tag{2.9}$$

Since $w_j \in \mathcal{C}^n_{\omega_j}(P)$ and $A \in \mathcal{C}_0^{n\times n}(P)$, we know that $x_j \in \mathcal{C}^n_{\omega_j}(P)$ from Theorem 2.11. Therefore, from the circulation matrix P shown above, we have the following relations:

$$x_{1j} = e^{-i\omega_j} T x_{2j}, \quad x_{2j} = e^{-i\omega_j} T x_{3j}, \quad \text{and} \quad x_{3j} = e^{-i\omega_j} T x_{1j}. \tag{2.10}$$

Apparently, the three blocks of components x_{1j}, x_{2j}, and x_{3j} are interrelated. Knowing the results of any one of them immediately yields the solution for the other two. It is, therefore, not necessary to solve the whole system $Ax_j = w_j$ for x_j. In fact, by expressing both x_{2j} and x_{3j} in terms of x_{1j} and substituting them into the first block of equations in (2.9), $A_{11}x_{1j} + A_{12}x_{2j} + A_{13}x_{3j} = w_{1j}$, we easily obtain the linear subsystem

$$\tilde{A}_j x_{1j} = w_{1j} \tag{2.11}$$

where

$$\tilde{A}_j = \left[A_{11} + e^{-2i\omega_j} A_{12} T^2 + e^{-i\omega_j} A_{13} T\right].$$

Now we can solve for x_{1j} much more efficiently from (2.11) since this subsystem is of dimension $2m$, only one third of the dimension of the whole system. Once x_{1j} is known, x_{2j} and x_{3j} can then be computed from (2.10), yielding the complete solution of x_j. Note that (2.11) is valid for $0 \le j \le 2$. This clearly indicates that this decomposition method yields, in this example, three independent linear subsystems that not only are smaller in size but also can be solved in parallel on multiprocessor computers. The final solution of x is simply the sum of x_0, x_1, and x_2. This concludes our demonstration.

3. Special cases. Depending on the circulation matrix P, the matrices (vectors) presented in §2 consist of several other important special classes of matrices (vectors) as special cases. For example, a matrix circulative with respect to P is also a reflexive matrix if the matrix P is a reflection matrix (that is, a symmetric signed permutation matrix) [Chen88], [ChSa89a]. On the other hand, a matrix reflexive with respect to some reflection matrix P is always a circulative matrix since any reflection matrix is necessarily a circulation matrix. The class of centrosymmetric matrices [Aitk49], [Andr73a], [Andr73b], [CaBu76], [Good70] is of course a special class of circulative matrices because it is a special class of reflexive matrices. In this section, we give another important special case.

Let $U_1(x) = 1$ and $U_2(x)$ and $U_3(x)$, $x = 2\pi/k$ for some positive integer k be the following two orthogonal matrices obtained from coordinate transformations:

$$U_2(x) = \begin{bmatrix} \cos x & \sin x \\ -\sin x & \cos x \end{bmatrix}$$

and

$$U_3(x) = \begin{bmatrix} \cos x & \sin x & 0 \\ -\sin x & \cos x & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

Also let $V_1 = \cos x$ and

$$V_2 = \begin{bmatrix} \cos x & 0 \\ 0 & 1 \end{bmatrix}.$$

Now we define

$$T_l(x) = I_l \otimes U(x), \quad U(x) \in \begin{cases} \{U_1, U_2, U_3\} & \text{if } k \neq 2, \\ \{U_1, U_2, U_3, V_1, V_2\} & \text{if } k = 2, \end{cases} \tag{3.1}$$

where I_l is the identity matrix of dimension $l > 0$. It should be mentioned that the dimension of T_l depends on U, whose dimension can be 1, 2, or 3, depending on which element is taken. Once U is decided, the dimension of T_l is fixed.

Before going any further, let us define a block circulant matrix C [Davi79],

$$C = \begin{bmatrix} C_0 & C_1 & C_2 & \cdot & \cdot & \cdot & C_{k-1} \\ C_{k-1} & C_0 & C_1 & \cdot & & & \cdot \\ \cdot & C_{k-1} & \cdot & \cdot & \cdot & & \cdot \\ \cdot & & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & & & \cdot & \cdot & \cdot & C_2 \\ C_2 & & & & \cdot & C_0 & C_1 \\ C_1 & C_2 & \cdot & \cdot & \cdot & C_{k-1} & C_0 \end{bmatrix}, \tag{3.2}$$

by

$$C = \text{bcirc } (C_0, C_1, C_2, \cdots, C_{k-1} \mid k),$$

where the integer k after the vertical rule $\mid$ is used to indicate that the matrix has block order k. Note that

$$\text{bcirc } (C_0, 0, \cdots, 0 \mid k)$$

reduces to a block diagonal matrix with identical diagonal blocks C_0. In other words,

$$\text{bcirc } (C_0, 0, \cdots, 0 \mid k) = I_k \otimes C_0.$$

We are now in a position to discuss three special types of circulation matrices. Let P_c, P_o, and P_t be square matrices given by

$$P_c = \text{bcirc } (0, T_l(x), 0, \cdots, 0 \mid k), \quad l > 0, \quad k > 1, \tag{3.3}$$

$$P_o = T_m(x), \qquad m > 0, \tag{3.4}$$

and

$$P_t = P_c \oplus P_o, \tag{3.5}$$

where $T_i(x)$, $i > 0$, is as defined in (3.1) and the symbol $\oplus$ denotes the direct sum. Note that the subscript and superscript i will, hereafter, be used as a positive integer instead of $\sqrt{-1}$. It is not difficult to see that P_c, P_o, and P_t are all circulation matrices. Indeed, from (3.1), we immediately have

$$T_m^i(x) = T_m(ix), \qquad 1 \le i \le k$$

and

$$T_m^k(x) = T_m(2\pi) = I.$$

The matrix T_m is obviously orthogonal since

$$T_m^T(x)T_m(x) = I_m \otimes (U^T(x)U(x)) = I.$$

Therefore, P_o is a circulation matrix for any integer $m > 0$. Also from (3.1) and (3.3), we can easily show the matrix P_c has the following properties:

$$\begin{aligned} P_c^T P_c &= I_k \otimes (T_l^T(x)T_l(x)) = I, \\ P_c^2 &= \text{bcirc } (0,\ 0,\ T_l(2x),\ 0, \cdots,\ 0 \mid k), \\ &\vdots \\ P_c^{k-1} &= \text{bcirc } (0,\ 0, \cdots,\ 0,\ T_l((k-1)x) \mid k), \end{aligned}$$

and, therefore,

$$\begin{aligned} P_c^k &= \text{bcirc } (T_l(kx),\ 0,\ \cdots,\ 0 \mid k) \\ &= I_k \otimes T_l(kx) \\ &= I, \end{aligned}$$

where we have used the fact that $T_l(ix)T_l(jx) = T_l((i+j)x)$ and $T_l(kx) = T_l(2\pi) = I$. The matrix P_c is, thus, a circulation matrix for any integer $l > 0$. The matrix P_t is, of course, a circulation matrix since P_c and P_o are.

These special circulation matrices serve two different types of operations: (block) cyclic permutations and/or coordinate transformations of unknowns through rotations about principal axes in the three-dimensional real space. They can be used in many discretized scientific and engineering problems to identify, from the matrix point of view, whether rotational symmetry is present when the unknowns of the problem are properly ordered. Therefore, circulation matrices of this special type are referred to as rotation matrices for the sake of emphasizing the role they play. In addition, any matrix resulting from any symmetric permutation of a rotation matrix will also be considered as a rotation matrix since it does not change its physical nature. For example, the matrix R,

$$R = \text{bcirc } (W(x),\ 0,\ \cdots,\ 0 \mid l)$$

with

$$W(x) = \text{bcirc } (0,\ U(x),\ 0,\ \cdots,\ 0 \mid k),$$

is also a rotation matrix. Based on this reasoning, we give the following definitions.

DEFINITION 3.1. Rotation matrices. Let $\mathcal{M}_p = \{P_c,\ P_o,\ P_t\}$ where P_c, P_o, and P_t are defined in (3.3), (3.4), and (3.5). A matrix P is said to be a rotation matrix if either $P \in \mathcal{M}_p$ or there exists some permutation matrix Π such that $\Pi^T P \Pi \in \mathcal{M}_p$.

DEFINITION 3.2. Rotative and antirotative vectors, matrices, and subspaces. Let R be a rotation matrix. Any vector, matrix, or subspace that is circulative (or anticirculative) with respect to R is also said to be rotative (or antirotative) with respect to R.

Now suppose P is a rotation matrix taking the form of P_c. Define S and Q to be

$$S = \text{bcirc}\ (0,\ I,\ 0,\ \cdots,\ 0 \mid k)$$

and

$$Q = \text{bcirc}\ (T_l(x),\ 0,\ 0,\ \cdots,\ 0 \mid k).$$

Then the matrix P can be rewritten as

$$P = SQ = QS. \tag{3.6}$$

For a matrix A with the relation $A = P^T A P$, we have from (3.6)

$$A = Q^T S^T A S Q = S^T Q^T A Q S$$

or, equivalently,

$$Q A Q^T = S^T A S \quad \text{and} \quad S A S^T = Q^T A Q.$$

Since any (block) circulant matrix C has the relation $C = S^T C S$ [Davi79], and since the rotation matrix P reduces to S if Q is taken to be the identity matrix, it is obvious that the class of circulant matrices is a special case of that of rotative matrices and, accordingly, a special case of that of circulative matrices. To close this section, we define another interesting special case of the class of rotative (antirotative) matrices.

DEFINITION 3.3. Dihedral and antidihedral matrices. A matrix $A \in \mathcal{C}^{n \times n}$ is said to be dihedral (or antidihedral) with respect to R and S if it satisfies the following three conditions:

- A is reflexive (or antireflexive) with respect to R,
- A is rotative (or antirotative) with respect to S, and
- $RSR = S^{k-1}$ for some k, $1 \leq k \leq n$

where R is a reflection matrix and S a rotation matrix with $S^k = I$.

It deserves mentioning that the main idea of this definition originates from the concept of dihedral groups in group presentation theory. Dihedral and antidihedral vectors and subspaces of vectors and matrices can be defined in a similar way. Since a dihedral matrix is both reflexive and rotative, the special properties possessed by these two classes of matrices can be applied to it directly. Examples of reflexive, rotative, and dihedral matrices are provided in the next section.

4. Examples. Numerical discretizations of physical problems very often give rise to circulative matrices when the problems exhibit some sort of symmetry and are properly discretized. Reflexive symmetry usually yields reflexive matrices and rotational symmetry results in rotative matrices, which all belong to the class of circulative matrices. In this section, we present three such matrices obtained from

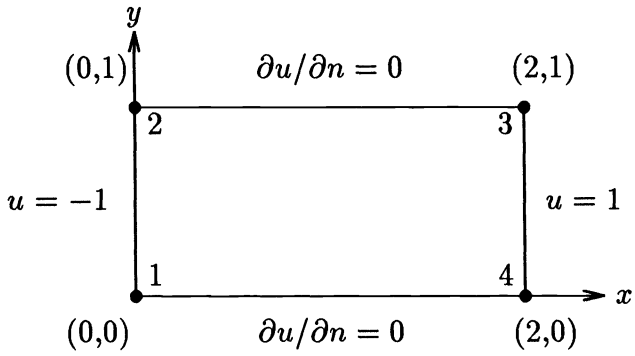


FIG. 1. *Boundary conditions and ordering of unknowns of Example* 1.

physical problems modeled by partial differential equations. The physical problems are also briefly described.

Example 1. The matrix A given below is obtained from a numerical approximation to a potential flow in a rectangular region discretized by four linear boundary elements [LiLi83]:

$$A = \begin{bmatrix} 0.750 & 0.250 & -0.376 & -0.357 \\ 0.250 & 0.750 & -0.357 & -0.376 \\ -0.376 & -0.357 & 0.750 & 0.250 \\ -0.357 & -0.376 & 0.250 & 0.750 \end{bmatrix}. \tag{4.1}$$

Figure 1 shows the boundary conditions of the flow and the ordering of the unknowns $(\partial u/\partial n)_i$, $i = 1, \cdots, 4$, where u is the potential. The integral equations used to yield such a matrix can be expressed as

$$\alpha\, u(x_i, y_i) = \int_\Gamma \left(\frac{u}{r} \frac{\partial r}{\partial n} - \ln r \frac{\partial u}{\partial n} \right) ds,$$

where α is the interior angle between the boundary elements at the singular point (x_i, y_i), r the distance between the point (x_i, y_i) and another point (x_j, y_j), Γ the boundary, and n the unit outward normal on Γ. Since it is not our purpose to discuss how to apply the boundary element method to obtain such a matrix, we omit all derivations. Interested readers should refer to [LiLi83] for details. Let

$$P_1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}, \quad P_2 = \begin{bmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{bmatrix}, \quad \text{and} \quad P_3 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}.$$

It is easy to see that

$$P_1^2 = P_2^2 = P_3^2 = I$$

and

$$A = P_1^T A P_1 = P_2^T A P_2 = P_3^T A P_3.$$

In other words, the matrix A is circulative with respect to P_1, P_2, and P_3. Note that P_1, P_2, and P_3 are all reflection matrices. P_3 is clearly a rotation matrix. In addition, P_1 and P_2 are rotation matrices as well. The reason for P_1 and P_2 to be rotation matrices is that they can be symmetrically permuted to yield P_3. Therefore, A is both reflexive and rotative with respect to all of them. This particular matrix is of course centrosymmetric because of P_2. It is also interesting to note that

$$P_1 = P_2P_3 = P_3P_2, \quad P_2 = P_3P_1 = P_1P_3 \quad \text{and} \quad P_3 = P_1P_2 = P_2P_1.$$

Accordingly, the matrix A is dihedral with respect to P_1 and P_2, to P_2 and P_3, and to P_3 and P_1.

Example 2. We now show a matrix derived from the finite-difference approximation to Laplace's equation

$$\frac{\partial^2 u}{\partial r^2} + \frac{1}{r}\frac{\partial u}{\partial r} + \frac{1}{r^2}\frac{\partial^2 u}{\partial \theta^2} = 0$$

in a circular annulus subject to Dirichlet boundary conditions u_a along the inner boundary and u_b along the outer boundary, as shown in Fig. 2. The matrix under consideration is one that, via proper discretization and ordering, has the following block tridiagonal form [Smit78], [Gera78]:

$$B = \begin{bmatrix} B_1 & \alpha_1 I & & & & & \\ \beta_1 I & B_2 & \alpha_2 I & & & & \\ & \beta_2 I & . & . & & & \\ & & . & . & . & & \\ & & & . & . & . & \\ & & & & . & B_{k-1} & \alpha_{k-1} I \\ & & & & & \beta_{k-1} I & B_k \end{bmatrix}, \tag{4.2}$$

where B_i, $i = 1, \cdots, k$, is a circulant matrix of dimension m given by

$$\begin{aligned} B_i &= \text{circ } (a_i,\ b_i,\ 0,\ \cdots,\ 0,\ b_i \mid m) \\ &= \begin{bmatrix} a_i & b_i & & & & & b_i \\ b_i & a_i & b_i & & & & \\ & b_i & . & . & & & \\ & & . & . & . & & \\ & & & . & . & . & \\ & & & & . & a_i & b_i \\ b_i & & & & & b_i & a_i \end{bmatrix}, \end{aligned}$$

where a_i and b_i are some constants and the notation circ(.) is defined in the same way as bcirc(.) except that now each block is just a scalar.

Let J_1, J_2, and J_3 be matrices of dimension m and be defined as

$$J_1 = \begin{bmatrix} & & & & 1 \\ & & & . & \\ & & . & & \\ & . & & & \\ 1 & & & & \end{bmatrix},$$

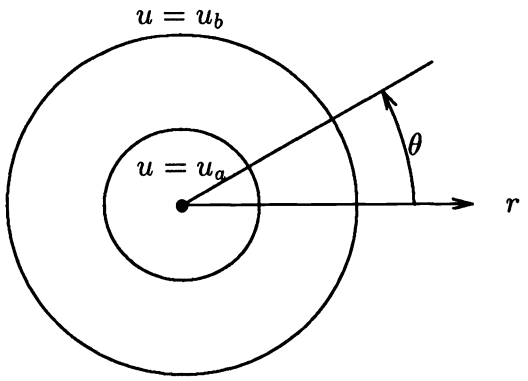


FIG. 2. *Domain and boundary conditions of Example* 2.

$$J_2 = \begin{cases} \begin{bmatrix} 0 & I_p \\ I_p & 0 \end{bmatrix} & \text{if } m = 2p, \\ \begin{bmatrix} 0 & 0 & I_p \\ 0 & 1 & 0 \\ I_p & 0 & 0 \end{bmatrix} & \text{if } m = 2p+1, \end{cases}$$

where p is a positive integer and

$$J_3 = \text{circ } (0,\ 1,\ 0,\ \cdots,\ 0 \mid m).$$

It is apparent that

$$B_i = J^T B_i J, \qquad i = 1,\ \cdots,\ k$$

for any $J \in \{J_1,\ J_2,\ J_3\}$. Now let

$$P_4 = I_k \otimes J_1, \quad P_5 = I_k \otimes J_2, \quad \text{and} \quad P_6 = I_k \otimes J_3.$$

We have

$$P_4^2 = P_5^2 = P_6^m = I,$$
$$P_4^T P_4 = P_5^T P_5 = P_6^T P_6 = I,$$

and

$$B = P^T B P$$

for any $P \in \{P_4,\ P_5,\ P_6\}$. Note that P_4 and P_5 are reflection matrices and P_6 is a rotation matrix. Therefore, the matrix B is reflexive with respect to P_4 and P_5, rotative with respect to P_6, and accordingly, circulative with respect to all of them. Furthermore, B is dihedral with respect to P_4 and P_6 since

$$P_4 P_6 P_4 = P_6^{m-1}.$$

The matrix B, however, is not dihedral with respect to P_5 and P_6 in general.

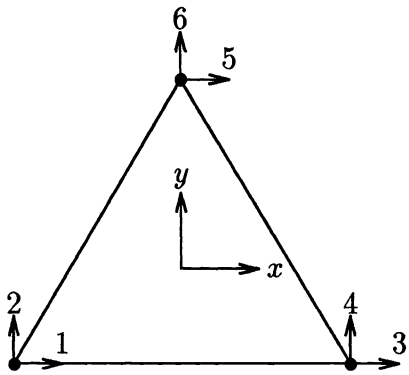


FIG. 3. *An equilateral triangular element for Example* 3.

Example 3. The final example we consider is a stiffness matrix obtained from the plane stress analysis of an isotropic elasticity problem using only one equilateral triangular element shown in Fig. 3. The two arrows at each node in Fig. 3 denote the displacements in the direction they point to. The governing differential equations for such an analysis can be written as

$$\mathcal{L}^T D\mathcal{L}u + b = 0,$$

where u and b are, respectively, the displacements and body forces,

$$\mathcal{L} = \begin{bmatrix} \partial/\partial x & 0 \\ 0 & \partial/\partial y \\ \partial/\partial x & \partial/\partial y \end{bmatrix} \quad \text{and} \quad D = \frac{E}{1-\mu^2} \begin{bmatrix} 1 & \mu & 0 \\ \mu & 1 & 0 \\ 0 & 0 & (1-\mu)/2 \end{bmatrix},$$

in which E is the elastic modulus and μ the Poisson's ratio. Following the procedures of the finite-element process [Sege76], we can obtain the following stiffness matrix:

(4.3)

$$C = \alpha \begin{bmatrix} 3+\nu & \sqrt{3}(\mu+\nu) & -3+\nu & \sqrt{3}(\mu-\nu) & -2\nu & -2\sqrt{3}\mu \\ \sqrt{3}(\mu+\nu) & 1+3\nu & -\sqrt{3}(\mu-\nu) & 1-3\nu & -2\sqrt{3}\nu & -2 \\ -3+\nu & -\sqrt{3}(\mu-\nu) & 3+\nu & -\sqrt{3}(\mu+\nu) & -2\nu & 2\sqrt{3}\mu \\ \sqrt{3}(\mu-\nu) & 1-3\nu & -\sqrt{3}(\mu+\nu) & 1+3\nu & 2\sqrt{3}\nu & -2 \\ -2\nu & -2\sqrt{3}\nu & -2\nu & 2\sqrt{3}\nu & 4\nu & 0 \\ -2\sqrt{3}\mu & -2 & 2\sqrt{3}\mu & -2 & 0 & 4 \end{bmatrix},$$

where

$$\nu = (1-\mu)/2 \quad \text{and} \quad \alpha = \left(\frac{t}{12a^2}\right)\left(\frac{E}{1-\mu^2}\right),$$

in which t is the thickness and $2a$ the lateral length.

Let

$$P_7 = \begin{bmatrix} 0 & J & 0 \\ J & 0 & 0 \\ 0 & 0 & J \end{bmatrix} \quad \text{and} \quad P_8 = \begin{bmatrix} 0 & T & 0 \\ 0 & 0 & T \\ T & 0 & 0 \end{bmatrix},$$

where

$$J = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad T = \begin{bmatrix} \cos 2\pi/3 & \sin 2\pi/3 \\ -\sin 2\pi/3 & \cos 2\pi/3 \end{bmatrix}.$$

We have

$$C = P_7^T C P_7 \quad \text{and} \quad C = P_8^T C P_8.$$

Although it is not difficult to show by inspection that the matrix C is both reflexive and rotative with respect to P_7 (P_7 is both a reflection and a rotation matrix), it is almost unlikely to see, from the entries of the matrix C, that C is rotative with respect to P_8. For a problem like this, an understanding of the material property, the geometry, and the boundary conditions of the physical problem plays an essential role in helping determine whether the derived matrix is rotative. Finally, we observe that the matrix C is also dihedral with respect to P_7 and P_8.

5. Conclusions. A special class of matrices, referred to as circulative matrices of degree θ, and a special decomposition method, referred to as the circulative decomposition method, have been introduced in this paper. The important special properties associated with this class of matrices have also been exploited. Matrices circulative of degree θ are simply called circulative matrices when $\theta = 0$ and anticirculative matrices when $\theta = \pi$, without mentioning their degrees for brevity. A new special case of (anti)circulative matrices, referred to as (anti)rotative matrices, has also been presented. Rotative (antirotative) matrices are themselves a generalization of circulant (anticirculant) matrices.

Circulative matrices arise very often in many scientific and engineering applications. They include centrosymmetric matrices, reflexive matrices, circulant matrices, and rotative matrices as important special cases. Numerical examples obtained from discretized physical problems, modeled by partial differential equations, have been provided to demonstrate their frequent occurrences. Anticirculative matrices, which do not occur naturally, are the counterpart of circulative matrices. Analogous to circulative matrices, these matrices contain the counterparts of centrosymmetric matrices, reflexive matrices, circulant matrices, and rotative matrices as special cases.

Acknowledgments. The author is grateful to Professor Roger A. Horn and the referees for their constructive comments and suggestions, especially to the referee who suggested using the term "circulative matrices of degree θ" to embed both circulative matrices and anticirculative matrices. The author thanks Dr. Hsin-Fong Chen, Dr. Craig Douglass, and Professor Ahmed Sameh for pointing him toward the field of group representation theory. He would also like to thank Professors Paul Saylor and Robert Skeel for their encouragement during the early stage of his research in this area.

REFERENCES

[Aitk49] A. C. AITKEN, *Determinants and Matrices*, Sixth Edition, Wiley–Interscience, New York, 1949.

[Andr73a] A. L. ANDREW, *Solution of equations involving centrosymmetric matrices*, Technometrics, 15 (1973), pp. 405–407.

[Andr73b] ———, *Eigenvectors of certain matrices*, Linear Algebra Appl., 7 (1973), pp. 151–162.

[CaBu76] A. CANTONI AND P. BUTLER, *Eigenvalues and eigenvectors of symmetric centrosymmetric matrices*, Linear Algebra Appl., 13 (1976), pp. 275–288.

[Chen88] H-C. CHEN, *The* SAS *Domain Decomposition Method for Structural Analysis*, CSRD Tech. Report 754, Center for Supercomputing Research and Development, University of Illinois, Urbana, IL, 1988.

[ChSa87] H-C. CHEN AND A. SAMEH, *Numerical linear algebra algorithms on the cedar system*, in Parallel Computations and Their Impact on Mechanics, A. K. Noor, ed., The American Society of Mechanical Engineers, AMD-Vol. 86, 1987, pp. 101–125.

[ChSa89a] ———, *A matrix decomposition method for orthotropic elasticity problems*, SIAM J. Matrix Anal. Appl., 10 (1989), pp. 39–64.

[ChSa89b] ———, *A domain decomposition method for 3D elasticity problems*, in Applications of Supercomputers in Engineering: Fluid Flow and Stress Analysis Applications, C. A. Brebbia and A. Peters, eds., Computational Mechanics Publications, Southampton University, Southampton, England, 1989, pp. 171–188.

[CuRe62] C. W. CURTIS AND I. REINER, *Representation Theory of Finite Groups and Associative Algebras*, Wiley–Interscience, New York, 1962.

[Davi79] P. J. DAVIES, *Circulant Matrices*, John Wiley, New York, 1979.

[Gera78] C. F. GERALD, *Applied Numerical Analysis*, Second Edition, Addison-Wesley, Reading, MA, 1978.

[Good70] I. J. GOOD, *The inverse of a centrosymmetric matrix*, Technometrics, 12 (1970), pp. 925–928.

[Hill75] V. E. HILL, *Groups, Representations, and Characters*, Hafner, New York, 1975.

[HoJo85] R. A. HORN AND C. A. JOHNSON, *Matrix Analysis*, Cambridge University Press, New York, 1985.

[LiLi83] J. A. LIGGET AND P. L.-F. LIU, *The Boundary Integral Equation for Porous Media Flow*, Allen & Unwin, London, 1983.

[MuIk89] K. MUROTA AND K. IKEDA, *Computational use of group theory in bifurcation analysis of symmetric structures*, SIAM J. Sci. Statist. Comput., 12 (1991), pp. 273–297.

[Satt79] D. H. SATTINGER, *Group Theoretic Methods in Bifurcation Theory*, Springer-Verlag, New York, 1979.

[Sege76] L. J. SEGERLIND, *Applied Finite Element Analysis*, John Wiley, New York, 1976.

[Smit78] G. D. SMITH, *Numerical Solution of Partial Differential Equations*, Second Edition, Clarendon Press, Oxford, 1978.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1189–1203, October 1992

014

PARALLEL SOLUTION OF LARGE LYAPUNOV EQUATIONS*

A. SCOTTEDWARD HODEL† AND KAMESHWAR POOLLA‡

Abstract. In this paper two algorithms for the solution of large-order ($100 \leq n \leq 1000$) Lyapunov equations $AX + XA^T + Q = 0$ are presented. First, a parallel version of the Hammarling algorithm for the solution of Lyapunov equations where the coefficient matrix A is *large* and *dense* is presented. Then a novel iterative parallel algorithm, called full-rank perturbed iteration (FRPI), is presented for the solution of Lyapunov equations where the matrix A is *large* and *banded*.

Key words. Lyapunov equation, parallel computation, direct methods, iterative methods, banded systems

AMS(MOS) subject classifications. 15A06, 65F05, 65F10

1. Introduction.

The Lyapunov equation

$$AX + XA^T + Q = 0, \qquad A,\ Q,\ X \in \mathbb{R}^{n\times n} \tag{1.1}$$

plays a significant role in numerous problems in control, communication systems theory, and power systems. These include the problems of $\mathcal{H}_\infty$ optimal control [7]; system balancing [22]; model reduction and reduced-order control of linear, time-invariant systems [19], [22]; stability analysis of dynamical systems [21]; etc. In this paper we address the parallel solution of *large* ($100 \leq n \leq 1000$) Lyapunov equations in the two cases where the coefficient matrix A is (i) *dense* and (ii) *banded*.

Standard solution methods for *small, dense* Lyapunov equations [2], [13] make use of the real Schur decomposition to transform the Lyapunov equation (1.1) into a form that is readily solved through forward substitution. More recently, Lu and Wachspress [23], [28] proposed the iterative numerical solution of the Lyapunov equation through the alternating direction implicit method. The ADI method requires the reduction of the matrix A to tridiagonal form through elementary transformations; stable reduction of an arbitrary matrix A to tridiagonal form is discussed in [8].

Large scale systems, such as those arising from finite element models of flexible structures [1], require the solution of Lyapunov equations of such large dimensions that parallel solution on commercially available multiprocessors is rendered practical. In this regard, O'Leary and Stewart [25] outline a parallel version of the Bartels–Stewart algorithm for the case where A has strictly real eigenvalues. (It is straightforward to generalize their method to the case where A has complex eigenvalues.) They also propose a parallel method for the computation of the Schur decomposition $A = USU^T$. (We are unaware of any numerical experiments involving this approach for the computation of the Schur decomposition.) The ADI iteration readily admits a parallel implementation.

In this paper, we present two new parallel algorithms for the solution of large Lyapunov equations. We first present in §2 a parallel implementation of the Hammarling algorithm [13]. Our algorithm takes advantage of vector pipelining through

* Received by the editors February 12, 1990; accepted for publication March 1, 1991. This work was supported in part by National Science Foundation grant ECS 87-09265-EIA, Joint Services Electronics Program contract N00014-84-C-0214, and the Center for Intelligent Control Systems, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139.

† Department of Electrical Engineering, Auburn University, Auburn, Alabama 36849-5201 (scotte@eng.auburn.edu).

‡ Coordinated Science Laboratory, University of Illinois, Urbana, Illinois 61801 (poolla@control.csl.uiuc.edu).

a vector-segment wavefront (coarse grain) approach. (We do *not* consider the parallel computation of the Schur decomposition $A = USU^T$ for large, dense Lyapunov equations. This is reasonable, since many algorithms [22], [26], [27] require the solution of a number of Lyapunov equations that require only a single common Schur decomposition.) Following this, in §3, we present the full-rank perturbed iteration (FRPI) algorithm for the solution of *large, banded* Lyapunov equations (i.e., A is a large, banded matrix). (A preliminary version of this algorithm may be found in [18].) By exploiting available problem structure, the FRPI algorithm provides for dramatic reductions in computation time relative to the Bartels–Stewart and Hammarling algorithms, even when run on a single processor machine. This algorithm admits efficient parallel implementation on shared-memory concurrent machines; hence we regard this algorithm as a powerful tool for the solution of banded Lyapunov equations. In §4 we detail our numerical experience with these parallel algorithms, and in §5 we summarize our results and make some concluding remarks.

2. The Hammarling algorithm. In this section, we discuss the parallel implementation of the Hammarling algorithm [13]. We first discuss in §2.1 the serial Hammarling algorithm for solution of the Lyapunov equation. We then present a vector segment parallel scheme in §2.2. (It is not advantageous to implement a block-parallel scheme of the Hammarling algorithm.) We conclude in §2.3.

2.1. The serial algorithm. Hammarling [13] introduces a method to directly compute the Cholesky factorization $GG^T = X$ of the solution of the Lyapunov equation

$$AX + XA^T + BB^T = 0 \tag{2.1}$$

without explicitly computing the product BB^T and without first computing the solution X. (The Hammarling algorithm requires that the eigenvalues of the matrix A lie in the open left half-plane.)

The Hammarling algorithm is motivated as follows. Suppose that the matrix A has strictly *real* eigenvalues and assume (without loss of generality) that A is in lower Schur form (lower triangular), and that B is lower triangular. Substitute the Cholesky factorization $X = GG^T$ into (2.1) and partition

$$G = \begin{bmatrix} g_{11} & 0 \\ \bar{g} & G_{22} \end{bmatrix}, \quad B = \begin{bmatrix} b_{11} & 0 \\ \bar{b} & B_{22} \end{bmatrix}, \quad \text{and} \quad A = \begin{bmatrix} a_{11} & 0 \\ \bar{a} & A_{22} \end{bmatrix},$$

where g_{11}, b_{11}, and a_{11} are scalars; $\bar{g}$, $\bar{b}$, and $\bar{a}$ are $(n-1)$-vectors; and G_{22}, B_{22}, and A_{22} are lower triangular matrices in $\mathbb{R}^{(n-1)\times(n-1)}$. Algebraic manipulation of these quantities reveals that

$$g_{11} = \frac{b_{11}}{\sqrt{-2a_{11}}}, \tag{2.2}$$

$$(A_{22} + a_{11}I)\bar{g} = -\left(g_{11}\bar{a} + \frac{b_{11}}{g_{11}}\bar{b}\right), \tag{2.3}$$

$$A_{22}(G_{22}G_{22}{}^T) + (G_{22}G_{22}{}^T)A_{22}{}^T = -(B_{22}B^T{}_{22} + \bar{y}\bar{y}^T) = -\hat{R}_{22}\hat{R}_{22}^T, \tag{2.4}$$

where $\bar{y} = \bar{b} - (b_{11}/g_{11})\bar{g}$. Equations (2.2)–(2.4) are readily solved. Furthermore, (2.4) is an order-$(n-1)$ Lyapunov equation that is in the same form as the original problem. Hence, since (2.4) is an order-$(n-1)$ Lyapunov equation of the form (2.1),

1. compute $Q^T B^T := B^T$ (QR factorization, B lower triangular, Q is not stored)
2. do $j = 1, n$
   (a) $g_{jj} = |b_{jj}| / [-2a_{jj}]^{1/2}$
   (b) if $j < n$ then /* solve for subdiagonal vector $\bar{g}$ */
      i. $\beta = b_{jj}/g_{jj}$
      ii. do $i = j+1, n$

$$g_{ij} = -\left(g_{jj}a_{ij} + \beta b_{ij} + \sum_{k=j+1}^{i-1} a_{ik}g_{kj} \right) / (a_{ii} + a_{jj})$$

      iii. end do
      iv. /* update the Cholesky factor B for the next iteration */
         do $i = j+1, n$
            $b_{ij} = b_{ij} - \beta g_{ij}$
      v. end do
      vi. do $i = j, m$
            compute Givens rotation $(c, s)_{ij}$ to cancel b_{ij} and apply to columns j and i.
      vii. end do
   (c) end if
3. end do

FIG. 2.1. *The serial Hammarling algorithm.*

we may recursively apply the Hammarling algorithm until all columns of G have been computed. Note that no portion of X is formed during the computation of g_{11} and $\bar{g}$.

We present a coded form of the serial Hammarling algorithm for the case where A has strictly *real* eigenvalues in Fig. 2.1. Vector pipelining may be applied in steps 2((b))ii and 2((b))iv, and in the application of the Givens rotations in step 2((b))vi. The serial Hammarling algorithm is easily modified for the case where A has complex eigenvalues; see [13] for further details.

2.2. Parallelism in the Hammarling algorithm. In this section we discuss parallelism in the Hammarling algorithm. We may attempt a parallel implementation of the Hammarling algorithm through the use of a parallel linear system solver (see, e.g., [6], [9], [12], [14], and [15]) to compute the vector $\bar{g}$ in (2.3), followed by parallel application of the associated Givens rotations to the matrix $\begin{bmatrix} B_{22} & \bar{y} \end{bmatrix}$ (see (2.4)). Further concurrency in the Hammarling algorithm is difficult without abandoning the column-major approach of the serial Hammarling algorithm. While the data dependencies in the Hammarling algorithm are more complex than those of the Bartels–Stewart algorithm [2], [25] because of the rank-one updates of B (see (2.4)), our strategy is to solve for the Cholesky factor G along antidiagonal wavefronts of vector segments $\gamma_{ij}^h = [g_{ij} \cdots g_{i+h-1,j}]^T$.

2.2.1. Static length vector level parallelism. We motivate our strategy as follows. Consider the case $h = 1$ and suppose g_{11} and g_{21} are computed as in (2.2) and (2.3). We may then update b_{22} completely, since the only Givens rotation applied to b_{21} and b_{22} is $(c, s)_{21}$ (see Fig. 2.1). Once b_{22} has been updated, we may compute g_{31} and g_{22} in parallel. In general, once an element g_{ij} has been computed, rotations $(c, s)_{j+1,j}$ through $(c, s)_{i-1,j}$ are applied to $b_{ij}, \ldots, b_{i-1,i}$, and then a new rotation $(c, s)_{ij}$ is computed to update b_{ij} and b_{ii}. This process is illustrated in Fig. 2.2. The

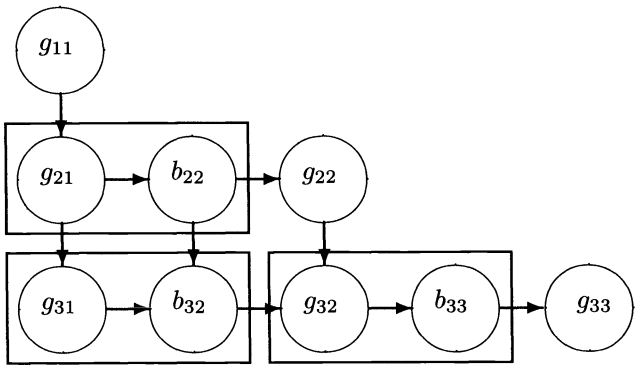


FIG. 2.2. *Computational nodes for parallelism.*

Givens rotations associated with g_{ij} perform a partial QR update of B so that we may immediately compute $g_{i+1,j}$ and $g_{i,j+1}$. This is in contrast to the serial Hammarling algorithm, where an entire column of G is computed before applying the corresponding Givens rotations to an entire column of B.

A precise description of a computational node for general values of h is presented in procedure *vsolve* (see Fig. 2.3). The kth antidiagonal wavefront of the matrix G is computed by concurrently calling $vsolve(k,1),\ldots,vsolve(1,k)$ for all applicable indices. (Since $i+h \leq j$ implies $\gamma_{ij}^h = 0$, the corresponding call to *vsolve* is not executed; similarly, wavefronts are truncated to remove elements g_{ij} that lie outside the range $1 \leq i,j \leq n$.)

The cost of parallelism is apparent in the additional storage required for the Givens rotations used in partial updates. These storage requirements may be reduced by concurrently solving for only p columns of G, which requires at most $2np$ floating point words of storage for the associated Givens rotations. More precisely, if there are p processors, then we concurrently solve for the first p columns of G along antidiagonal wavefronts of length p. (In order to increase efficiency, processors freed as the wavefront drops through the nth row of G immediately "wrap around" to the next available column of G.)

It is relatively straightforward to schedule calls to *vsolve* when the matrix A has strictly real eigenvalues since each wavefront is of the form

$$\left\{\gamma_{ih+1,j}^h,\ \gamma_{(i-1)h+1,j+1}^h,\cdots,\ \gamma_{(i-(p-1))h+1,j+p-1}^h\right\};$$

that is, row indices differ by exactly h between consecutive columns. However, scheduling is more complicated in the complex eigenvalue case due to varying block sizes in the partition of G ($h\times 1$, $h\times 2$, $(h+1)\times 1$, and $(h+1)\times 2$). For this case, scheduling is accomplished through the use of length p doubly linked lists i, j, and s. During each iteration of *vsched* the indices of the upper left element of the block associated with processor q are stored in i_q, j_q. s_q indicates whether column j_q corresponds to a Schur block in A. If so, then processor q simultaneously solves for columns j_q and j_q+1 by subdividing them into $h\times 2$ and $(h+1)\times 2$ blocks. Otherwise, processor q solves for column j_q only by subdividing it into $h\times 1$ and $(h+1)\times 1$ blocks. Once processor q has computed the block containing $g_{i_qj_q}$, it computes new indices i_q', j_q', s_q' for use during the next iteration. These may be computed concurrently and then copied into i, j, and s at the beginning of the next wavefront computation.

Procedure *vsolve*(i', j): compute vector segment $\gamma^h_{i'j}$

1. $\hat{i} = 1 + h(i' - 1)$
2. if $\hat{i} = j$ /*diagonal solve—no update */
   (a) $g_{jj} = |b_{jj}|/(-2a_{jj})^{-1/2}$ /* scalar arithmetic—no pipelining */
   (b) $\hat{i} = \hat{i} + 1$
3. end if
4. $\beta = b_{jj}/g_{jj}$
5. do $i = \hat{i}, hi'$ /* subdiagonal solve */
   /* pipelined inner product */

$$g_{ij} = -(g_{jj}a_{ij}) + \beta b_{ij} + (a_{ii} + a_{jj})^{-1} \sum_{k=j+1}^{i-1} a_{ik}g_{kj}$$

6. end do
7. do $i = \hat{i}, hi'$ /* update B; pipelined loop*/
   $b_{ij} = b_{ij} - \beta g_{ij}$
8. end do
9. do $k = j + 1, i - 1$ /* apply previous Givens rotations from column j */
   (a) do $i = \hat{i}, hi'$ /* pipelined loop */

$$\begin{bmatrix} b_{ik} & b_{ij} \end{bmatrix} = \begin{bmatrix} b_{ik} & b_{ij} \end{bmatrix} \begin{bmatrix} c_{kj} & -s_{kj} \\ s_{kj} & c_{kj} \end{bmatrix}$$

   (b) end do
10. end do
11. do $i = \hat{i}, hi'$ /* Compute and apply new Givens rotations */
   (a) compute Givens rotation $(c, s)_{ij}$ such that

$$\begin{bmatrix} b_{ii} & b_{ij} \end{bmatrix} \begin{bmatrix} c_{ij} & -s_{ij} \\ s_{ij} & c_{ij} \end{bmatrix} = \begin{bmatrix} * & 0 \end{bmatrix}$$

   (b) do $k = i, hi'$ /* pipelined loop */

$$\begin{bmatrix} b_{ik} & b_{ij} \end{bmatrix} = \begin{bmatrix} b_{ik} & b_{ij} \end{bmatrix} \begin{bmatrix} c_{ij} & -s_{ij} \\ s_{ij} & c_{ij} \end{bmatrix}$$

   (c) end do
12. end do

end *vsolve*

FIG. 2.3. *Parallel Hammarling solution procedure.*

2.2.2. Dynamic length vector level parallelism. The static vector length parallel algorithm loses efficiency as it progresses through the matrix G due to a forced loss of achievable parallelism induced by shortened antidiagonal wavefronts. In particular, the last h columns of G are computed *serially* since each vector segment γ^h_{lk} containing g_{nk} must be computed before work can begin on vector segment $\gamma^h_{l,k+1}$. That is, while a given static vector length h may be optimal for computation of the *entire* Cholesky factor G (see §2.3), we may increase efficiency by *dynamically* adjusting vector length during the course of the computation (see Fig. 2.4).

Such an approach requires only simple modifications to routine *vsolve*; specifically, we define procedure *dsolve*(i, j, h) identical to *vsolve* except that the vector length h is an input parameter. However, scheduling calls to *dsolve* is more complex than the static length scheduling problem, since the wavefront indices $(i_1, j_1), \ldots, (i_p, j_p)$ cannot be directly adjusted; even though we may freely change the vector length parameter h at any time, the distance $i_q - i_{q+1}$ remains unchanged until the

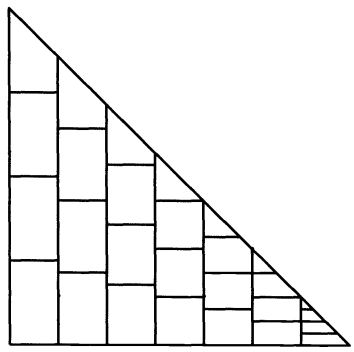


FIG. 2.4. *Dynamic vector-length partition of G.*

wavefront wraps around to the next vertical strip. That is, modifying the vector length h changes only the time required to execute $dsolve(i, j, h)$, and does not immediately affect the "slope" of the wavefront. Our approach is to update the vector length h each time a processor wraps around to a new vertical strip v_s. All processors corresponding to elements in strip v_s will use the same vector length h_s; a convenient heuristic is to choose h_s to minimize the time required to compute vertical strip v_s (see §2.3). Observe that if $s \geq 2$ then h_s may be chosen while neglecting the first $p-1$ wavefronts of v_s, since they are computed simultaneously with the last $p-1$ wavefronts of vertical strip v_{s-1}.

2.2.3. Block solution with accumulation of Givens rotations. Bischof and Van Loan [3] use accumulated Householder transforms as a way of reducing computation time requirements in a block QR factorization. Since the update step in procedure *vsolve* forms a portion of a QR factorization, one may attempt to use this method in our parallel algorithm. Unfortunately, such an approach will actually *increase* computational requirements.

Indeed, the product

$$T_{qj} = (c, s)_{j+1,j}(c, s)_{j+2,j} \cdots (c, s)_{j+q,j}$$

can be shown to be of the form

$$T_{qj} = \begin{bmatrix} u_{11} & \bar{u}_{12}^T \\ \bar{u}_{21} & U_{22} \end{bmatrix},$$

where $\bar{u}_{12}$ and $\bar{u}_{21}$ are dense q-vectors and U_{22} is upper triangular. That is, accumulating q Givens rotations results in a *dense* matrix $T_{qj} \in \mathbb{R}^{(q+1)\times(q+1)}$, requiring $O(q^2)$ storage, as opposed to $2q$ floating point words to store the original rotations. Furthermore, multiplication by the matrix T_{qj} requires $O(q^2)$ flops, while the application of q independent Givens rotations requires only $4q$ flops [12]. The reason that the accumulated transform performs so poorly in our problem is that our application (rank-one update) requires that we annihilate only $O(n)$ elements of B, not $O(n^2)$, as in [3]. We conclude that accumulation of Givens rotations is *not* desirable for parallelization of the Hammarling algorithm, in terms of both storage and computation time requirements.

2.3. Time analysis: Real eigenvalue case. We compare in this section the computational requirements of the above parallel implementation of the Hammarling algorithm with those of the pipelined serial Hammarling algorithm. We measure algorithm performance in terms of the problem size n and two positive constants α and δ that provide a measure of pipeline efficiency. More precisely, we say that the execution of n vector-pipelined floating point operations ("flops") requires time

$$t(n, \alpha, \delta) = \alpha + \delta n, \tag{2.5}$$

where α is the pipeline "startup" time, and δ is the maximum execution time of the individual pipeline stages. Our analysis indicates that, for a fixed number p of processors, our parallel algorithm will increasingly outperform the Hammarling serial algorithm as the dimension n of the Lyapunov equation (2.1) increases. This conclusion is consistent with the results of our numerical experiments, presented in §4. The details of the timing calculations in this section are in [17]; most of these results were obtained with the aid of *Mathematica*™ [29].

The serial Hammarling algorithm solves for the matrix G in (2.1) in column-major order. From Fig. 2.1, it can be seen that the computation of each column of G requires the computation of a diagonal element g_{jj}, the solution of a linear system of equations (2.3), and a rank-one update to the right-hand side matrix B. Summing the time for the computation of each column j yields $T_s(n, \alpha, \delta) = 5\delta n^3/6 + O(n^2)$ flops for the computation of the entire Cholesky factor G. Notice that the pipeline startup time α does not enter into the dominant term of T_s, since the jth column is computed in its entirety before proceeding to the $j+1$st column.

We may similarly analyze the time requirements of the static vector length concurrent algorithm as follows. Suppose the vector length h divides n and the number of processors p divides h. In order to aid in our analysis of the time requirements of the static vector length approach, we respectively partition G into *vertical strips* $v_1, \ldots, v_{n_v}$ of width p and *large vertical strips* $V_1, \ldots, V_{n_h}$, where $n_h = n/h$,

$$v_s = \begin{bmatrix} g_{1,1+(s-1)p} & \cdots & g_{1,sp} \\ \vdots & \ddots & \vdots \\ g_{n,1+(s-1)p} & \cdots & g_{n,sp} \end{bmatrix},$$

and

$$V_\sigma = \begin{bmatrix} g_{1,1+(\sigma-1)h} & \cdots & g_{1,\sigma h} \\ \vdots & \ddots & \vdots \\ g_{n,1+(\sigma-1)h} & \cdots & g_{n,\sigma h} \end{bmatrix}.$$

The time $t'_{v_\sigma}(n, \sigma, p, h, \alpha, \delta)$ required to compute all vertical strips v_s contained in the large vertical strip V_σ is

$$t'_{v_\sigma}(n, \sigma, p, h, \alpha, \delta) = \sigma^2 h^2 \left(\frac{\alpha}{2p} + \frac{5\delta h}{2p}\right) - \sigma h \left(\frac{5\delta h n}{p} + \frac{\alpha n}{p}\right) + \frac{5\delta h n^2}{2p} + \frac{\alpha n^2}{2p} + O(n).$$

Thus the total time $T_v(n, p, h, \alpha, \delta)$ required to compute G using pipelined static vector length parallelism is

$$T_v(n, p, h, \alpha, \delta) = \sum_{\sigma=1}^{n/h} t'_{v_\sigma}(n, \sigma, p, h, \alpha, \delta) = n^3 \left(\frac{5\delta}{6p} + \frac{\alpha}{6hp}\right) + O(n^2).$$

Notice that each column of G is partitioned into vector segments γ_{ij}^h; hence, the pipeline startup time *does* enter the n^3 term of T_v, in contrast to the time T_s associated with the serial Hammarling algorithm. This additional cost may be reduced as n increases by increasing the vector length h while preserving available parallelism. From the above, we expect that the static vector length concurrent algorithm will provide nearly linear speedup for large values of n. Both of these conclusions are consistent with the experimental data presented in §4.

3. Parallel solution of banded Lyapunov equations. An important class of problems in which large Lyapunov equations arise is that of control system design for large flexible structures [1]. Finite element approximations of such structures yield large-dimensional systems of linear differential equations such as

$$\dot{x} = Ax + Bu,$$

where $x \in \mathbb{R}^n$ is a state vector, $u \in \mathbb{R}^m$ is an input vector, and $A \in \mathbb{R}^{n \times n}$ is a banded matrix.

To approximate these large-dimensional systems for purposes of controller design and/or simulation it becomes necessary to solve several Lyapunov equations such as

$$AX + XA^T + Q = 0 \tag{3.1}$$

for various choices of the matrix Q (see, for example, [10], [22], and [26]). Neither the Bartels–Stewart algorithm nor the Hammarling algorithm is suitable for this purpose, since the required Schur decomposition will generally destroy the sparsity of A. Hence we adopt a new approach, similar in spirit to Jacobi and Gauss–Seidel iteration for solution of large, sparse, diagonally dominant systems of linear equations $Ax = b$ [12]. This algorithm, called full-rank perturbed iteration (FRPI), selects a block-diagonal matrix A_0 and searches for a *sparse* right-hand side perturbation matrix $\tilde{C}$ such that the solution X of (3.1) also satisfies the perturbed Lyapunov equation

$$A_0X + XA_0{}^T + Q + \tilde{C} = 0. \tag{3.2}$$

Since A_0 is block diagonal, its lower real Schur form matrix S_0 and the associated orthogonal transformation U_0 will also be block diagonal. That is, FRPI requires only that we compute the lower real Schur form of the diagonal blocks of A_0. Since the computation of the Schur decomposition of an $n \times n$ matrix requires $O(n^3)$ time, the Schur decomposition of the block-diagonal matrix A_0 may be computed much more rapidly than the Schur decomposition of the matrix A. We will show in §3.3 that this feature of our algorithm provides dramatic savings in computation time, even when FRPI is executed on a single processor.

Analysis of our algorithm requires that $A + A^T$ is negative definite ($A + A^T < 0$). This condition is reasonable since typically finite element modeling of physical systems yields stable system matrices A that are diagonally dominant. Notice that $A + A^T < 0$ implies that the system $(sI - A)^{-1}$ is passive, a condition that is typical for these large structures; see [20] for further discussion. We shall henceforth *assume* that $A + A^T$ is negative definite.

3.1. Algorithm description. We motivate our approach as follows. Suppose that the matrix A in (3.1) was not only banded, but also block diagonal, i.e., $A = \operatorname{diag}(A_{11}, \ldots, A_{NN})$. Then the individual *small* blocks of the Lyapunov equation solution X (partitioned conformably with A) satisfy small, dense, independent Sylvester

equations

$$A_{ii}X_{ij} + X_{ij}A_{jj}{}^T + Q_{ij} = 0, \tag{3.3}$$

whose solutions can be readily computed using the Bartels–Stewart algorithm [2] or the related Schur–Hessenberg algorithm [11]. That is, each block X_{ij} of X may be computed *concurrently* with *any* other block of X. Our strategy is to construct a perturbed block-diagonal Lyapunov equation "nearby" the original Lyapunov equation (3.1), which has the same solution X.

More precisely, for a general banded matrix A, we select a block size h such that $n = hN$ and decompose $A = A_0 + \tilde{A}$ where $A_0 = \text{block diag}(A_{11}, \ldots, A_{NN})$, $A_{ii} \in \mathbb{R}^{h\times h}$, and $\tilde{A}$ is low rank, block diagonal, and sparse. (We choose a block size h that divides n for simplicity in exposition; this choice is not essential to our algorithm.) (If the diagonal blocks of A_0 are drawn from the corresponding blocks of the matrix A, then the nonzero entries of $\tilde{A}$ appear in small "bow-tie–shaped" regions along the diagonal.)

For $Z \in \mathbb{R}^{n\times n}$, define the linear operators

$$\begin{aligned}
\phi(Z) &: \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n} \;:\; Z \to AZ + ZA^T, \\
\phi_0(Z) &: \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n} \;:\; Z \to A_0Z + ZA_0{}^T, \\
\tilde{\phi}(Z) &: \mathbb{R}^{n\times n} \to \mathbb{R}^{n\times n} \;:\; Z \to \tilde{A}Z + Z\tilde{A}^T.
\end{aligned}$$

Notice first that A is Hurwitz (all its eigenvalues lie in the left half-plane) since $A+A^T$ is assumed to be negative definite. As a consequence, the operator ϕ is invertible. Furthermore, $A + A^T < 0$ implies that all principle submatrices of A are Hurwitz, and consequently ϕ_0 is also invertible. However, $\tilde{\phi}$ is singular since $\tilde{A}$ is low rank, i.e., $\tilde{\phi}^{-1}$ does not exist.

Let X be the solution of the original Lyapunov equation (3.1). Now observe that

$$\phi(X) + Q = \phi_0(X) + \tilde{\phi}(X) + Q = 0.$$

Solving for X we obtain $X = -\phi_0{}^{-1}(Q+\tilde{\phi}(X))$, and so the perturbation matrix $\tilde{C}$ we desire in (3.2) is $\tilde{C} = \tilde{\phi}(X)$. We make this substitution and apply the linear operator $\tilde{\phi}$ to obtain

$$\tilde{C} = -\tilde{\phi}\phi_0{}^{-1}(Q + \tilde{C}). \tag{3.4}$$

That is, the desired perturbation matrix $\tilde{C}$ is a (fixed point) solution of (3.4). Furthermore, since $\tilde{C} = \tilde{\phi}(X) = \tilde{A}X + X\tilde{A}^T$, $\tilde{C}$ is a *sparse* matrix whose nonzero entries appear in horizontal and vertical strips centered every h rows and columns, respectively.

In general, (3.4) may have several fixed points. However, if all eigenvalues of $\tilde{\phi}\phi_0{}^{-1}$ lie inside the unit circle, then $\tilde{C}$ is the *unique* asymptotic limit of the solution of the linear time-invariant discrete time system

$$\tilde{C}_{k+1} = -\tilde{\phi}\phi_0{}^{-1}(\tilde{C}_k) - \tilde{\phi}\phi_0{}^{-1}(Q).$$

This result naturally suggests the FRPI algorithm shown in Fig. 3.1. Observe that the Schur decomposition of the block-diagonal matrix A_0 is computed only once.

1. $\tilde{C}_0 = 0$, $i = 0$
2. Select a block size h and partition the matrix A to obtain block-diagonal matrices A_0 and $\tilde{A} = A - A_0$.
3. Concurrently compute the lower Schur decompositions $U_{ii}S_{ii}{U_{ii}}^T = A_{ii}$.
4. Repeat
   (a) Compute in parallel $\tilde{C}_{i+1} = -\left[\tilde{\phi}{\phi_0}^{-1}(\tilde{C}_i) + \tilde{\phi}{\phi_0}^{-1}(Q)\right]$.
   (b) $i = i + 1$.
5. until $\|\tilde{C}_i - \tilde{C}_{i-1}\| \leq \epsilon$.

FIG. 3.1. *Full-rank perturbed iteration.*

3.2. Proof of convergence. In this section we establish sufficient conditions on the matrix A such that FRPI will converge to the correct solution X of the banded Lyapunov equation (3.1). In the previous section, we showed that if the spectrum of $\tilde{\phi}{\phi_0}^{-1}$ lies inside the unit circle, then FRPI will converge. Since it is difficult to directly determine the eigenvalues of the linear operator $\tilde{\phi}{\phi_0}^{-1}$, we will instead derive a diagonal dominance condition on A such that $\tilde{\phi}{\phi_0}^{-1}$ is a contraction, i.e.,

$$\left\|\tilde{\phi}{\phi_0}^{-1}\right\| = \sup_{X \neq 0} \frac{\left\|\tilde{\phi}{\phi_0}^{-1}(X)\right\|_F}{\|X\|_F} < 1,$$

where $\|\cdot\|_F$ is the Frobenius norm (see [12]). This condition is sufficient (but not necessary) to ensure that the spectrum of $\tilde{\phi}{\phi_0}^{-1}$ lies in the unit disc, and thus to guarantee that FRPI will converge.

In order to derive this diagonal dominance condition, we shall require the following simple result.

LEMMA 3.1. $\|AB\|_F \leq \min(\|A\|_F \|B\|_2, \|A\|_2 \|B\|_F)$.

Proof. Let $B = U\Sigma V^T$ (singular value decomposition) with $\Sigma = \operatorname{diag}(\sigma_1 \geq \cdots \geq \sigma_n)$. Then

$$\|AB\|_F^2 = \operatorname{tr}(ABB^TA^T) = \operatorname{tr}(AU\Sigma^2U^TA^T) = \operatorname{tr}(\Sigma^2U^TA^TAU).$$

Let $Q = U^TA^TAU$. Then

$$\|AB\|_F^2 = \operatorname{tr}(\Sigma^2Q) = \sum_{i=1}^{n} \sigma_i^2 q_{ii} \leq \sigma_1^2 \sum_{i=1}^{n} q_{ii} = \sigma_1^2 \operatorname{tr}(U^TA^TAU) = \|A\|_F^2 \|B\|_2^2,$$

from which we have $\|AB\|_F \leq \|A\|_F \|B\|_2$.

$\|AB\|_F \leq \|A\|_2 \|B\|_F$ can be shown in a similar fashion, which establishes the theorem. □

Since A_0 is Hurwitz, we can write

$$\phi_0^{-1}(X) = \int_0^\infty e^{A_0t} X e^{{A_0}^T t} dt. \tag{3.5}$$

We may bound the matrix exponential e^{A_0t} using the *log norm* $\mu(A_0)$ of A_0. The log norm $\mu(A)$ of a matrix $A \in \mathbb{R}^{n\times n}$ is defined as

$$\mu(A) = \tfrac{1}{2}\underline{\lambda}(-(A + A^T)).$$

Notice that $\mu(A)$ is not a norm in any sense since $\mu(A) < 0$ for $A+A^T$ positive definite. Nevertheless, the log norm provides a useful bound on the matrix exponential when $A + A^T < 0$.

THEOREM 3.2. $\|e^{At}\|_2 \leq e^{-\mu(A)t}$ *for all* $A \in \mathbb{R}^{n\times n}$, $t \geq 0$.

This bound is well known (see, e.g., [16] and [24]); hence the proof is omitted.

We are now in a position to establish our main convergence result.

THEOREM 3.3. *For* $A_0 = \text{block diag}(A_{11}, \ldots, A_{NN})$ *define* $a_i = \mu(A_{ii})$, $i = 1, \ldots, N$, *and let* $a = \max_i a_i$. *Then the operator* $\tilde{\phi}\phi_0^{-1}$ *is a contraction if*

$$\gamma \triangleq \frac{\|\tilde{A}\|_2}{a} < 1. \tag{3.6}$$

Consequently, (3.6) *is a sufficient condition for* FRPI *to converge to the correct solution* X *of the Lyapunov equation* (3.1).

Proof. It is readily observed from Lemma 3.1 that

$$\|\tilde{\phi}\| = \sup_{\|X\|_F=1} \|\tilde{A}X + X\tilde{A}^T\|_F \leq 2\|\tilde{A}\|_2.$$

Similarly, since the eigenvalues of A_0 lie in the open left half-plane, application of Lemma 3.1, Theorem 3.2, and equation (3.5) yields

$$\|\phi_0^{-1}\| = \sup_{\|X\|_F=1} \left\| \int_0^\infty e^{A_0 t} X e^{A_0^T t} dt \right\|_F \leq \int_0^\infty \|e^{A_0 t}\|_2 \, \|e^{A_0^T t}\|_2 dt \leq \frac{1}{2\mu(A_0)}.$$

Hence, if condition (3.6) holds, we have

$$\left\|\tilde{\phi}\phi_0^{-1}\right\| \leq \|\tilde{\phi}\| \, \|\phi_0^{-1}\| \leq 2\|\tilde{A}\|_2 \left(\frac{1}{2a}\right) < 1,$$

completing the proof. □

Remark 1. Theorem 3.3 states that FRPI will converge to the desired solution X of the Lyapunov equation (3.1) if the "coupling" blocks contained in $\tilde{A}$ are sufficiently small. The constant $\gamma = \|\tilde{A}\|_2/a$ in condition (3.6) provides a bound on the speed of convergence of FRPI, since

$$\|X - X_{i+1}\|_F \leq \|\tilde{\phi}\phi_0^{-1}\| \, \|X - X_i\|_F \leq \gamma \, \|X - X_i\|_F ;$$

The condition $\gamma < 1$ may be interpreted as a diagonal dominance condition, and is sufficient, but not necessary, for convergence of the FRPI algorithm.

3.3. Timing analysis. We now consider the computational requirements of the FRPI algorithm. The Schur decomposition of an $n \times n$ matrix requires roughly $\frac{15}{3}n^3 = O(n^3)$ time [12]. However, the *serial* computation of the Schur decomposition of N $(n/N) \times (n/N)$ blocks requires $O(N\,(n/N)^3) = O(n^3/N^2)$ time. If $p \leq N$ processors are used to compute these Schur decompositions in parallel, then step 3 in Fig. 3.1 requires only $O(n^3/(pN^2))$ time. It is in this step that the FRPI algorithm achieves its greatest advantage over standard techniques.

Since the solution X of (3.2) is symmetric, serial execution of step 4(a) of the FRPI algorithm requires the solution of $(1/2)N(N-1)$ Sylvester's equations in each iteration, or only $O\left(n^3/N\right)$ time. Since any block $X_{ij}^{(i)}$ of X_i may be computed in parallel with any other block of X_i, the algorithm is well suited for parallel implementation. In particular, if $p \leq N(N-1)/2$ processors are used, each iteration of

step 4(a) can be executed in $O\left(n^3/(pN)\right)$ time. Hence FRPI allows efficient solution of large, banded Lyapunov equations.

These timing comparisons do not guarantee that the FRPI algorithm will always be faster than the standard Bartels–Stewart or Hammarling algorithms. However, the designer may use the figure of merit $\gamma = \|\tilde{A}\|_2/\mu(A_0)$ in order to predict the speed of convergence through the use of Theorem 3.3; this data allows an intelligent choice of the algorithm to be used.

4. Results. In this section we present the results of our numerical experiments on an Alliant FX-8 vector-concurrent multiprocessor. We measure algorithm performance in terms of the *speedup* $s = t_s/t_p$, where t_s is the execution time for a single pipelined processor and t_p is the execution time when using all available processors ($p = 8$ in our experiments). We also use the *residue*

$$R(\hat{X}) = A\hat{X} + \hat{X}A^T + BB^T$$

of the computed solution of the Lyapunov equation as a measure of its accuracy. The error $\|X - \hat{X}\|_2$ can be bounded above and below by using the residue of the Lyapunov equation.

THEOREM 4.1 (see [18]). *Let A, $Q \in \mathbb{R}^{n\times n}$, $A + A^T < 0$, and let X satisfy the Lyapunov equation* (1.1). *If $\hat{X}$ is an estimate of Xm, then*

$$\frac{\|R(\hat{X})\|_2}{2\|A\|_2} \leq \left\|X - \hat{X}\right\|_2 \leq \frac{\|R(\hat{X})\|_2}{2\mu(A)},$$

where $\mu(A)$ is the log norm of A (see §3).

4.1. The Hammarling algorithm. We tested the static vector length parallel implementation of the Hammarling algorithm and compared our results with the pipelined serial Hammarling algorithm. Both of these approaches solve the Lyapunov equation (1.1) where A is stable with (possibly) complex eigenvalues. We implemented the pipelined serial Hammarling algorithm in routine *dlych*. In order to reduce memory requirements, *dlych* does *not* use vector pipelining when applying Givens rotations to the right-hand side Cholesky factor B. We implemented our parallel version of the Hammarling algorithm in two different routines, since the application of Givens rotations cannot be pipelined in steps 9 and 11 of Fig. 2.3 for the scalar case $h = 1$. We solved the Lyapunov equation with vector lengths $h = 8, 16, 24$, and 32.

We tested our routines on randomly generated diagonally dominant matrices $A \in \mathbb{R}^{n\times n}$ of dimension $n = 10, 20, \ldots, 400$; the corresponding Lyapunov equations (1.1) were transformed so that A was in lower real Schur form before applying the Hammarling algorithm to the transformed problem. The computed Cholesky factors were back-transformed and "squared-up" in order to compute the Frobenius norm of the residue of the Lyapunov equation. Differences between the solutions computed by individual routines were also computed. Computed residues were consistently small ($\|R(\hat{X})\|_F/\|\hat{X}\|_F \leq 10^{-11}$). Speedup results are presented in Fig. 4.1.

The chief computational bottleneck in the solution of the Lyapunov equation is the required Schur decomposition of the matrix A. In our numerical experiments the computation time for the Schur decomposition $A = USU^T$ exceeded the time required for the solution of the transformed Lyapunov equation

$$S\hat{G}\hat{G}^T + \hat{G}\hat{G}^T S^T + (U^T B)(B^T U) = 0$$

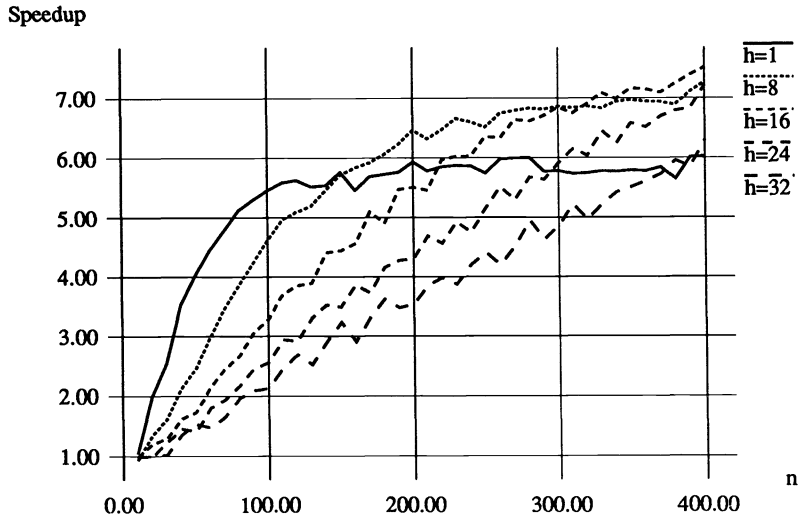


FIG. 4.1. *Speedup using the parallel Hammarling algorithm,* $p = 8$.

by a factor of 10–20. For example, the Alliant FX-8 requires roughly 270 seconds to compute the Schur decomposition of a 400×400 matrix, while the corresponding execution time for *dclys* is only 22 seconds. (This disparity is due, in part, to the short Householder vectors used in the double-Francis QR algorithm.) Hence the parallelism available in the actual solution of the transformed Lyapunov equation is of significant benefit only when several Lyapunov equations must be solved that use the same A matrix on the left-hand side, as in [26]. The parallel computation of the Schur decomposition remains an area of active study [4], [5].

4.2. Full-rank perturbed iteration. We implemented the full-rank perturbed iteration algorithm in our routine *dcfrpi*. *dcfrpi* was tested on randomly generated diagonally dominant banded systems of order $50, 100, \ldots, 400$ and bandwidth 3, 7, or 11. Each problem was perturbed into a block-diagonal Lyapunov equation (3.2) with A_0 having up to 16 blocks along the diagonal. The algorithm terminates either when the perturbation term $\tilde{C}$ converges, or after 50 iterations. In all tested examples, $\tilde{C}$ converged within 10–15 iterations. We compared execution times of the *dcfrpi* routine with the pipelined serial Hammarling algorithm: *dcfrpi* speedup values were consistently in the range of 2.0–5.0 for $n \geq 50$.

We also tested the algorithm on a single processor Sun workstation, where FRPI provided reductions in required computation time over the serial Hammarling algorithm. We wish to emphasize that this "speedup" on a single processor is due to the $n \times n$ Schur decomposition required by standard solution methods; hence FRPI is an effective method for solving banded Lyapunov equations even in a serial computing environment.

Solution accuracy was checked by comparing the Frobenius norm of the residue of *dcfrpi* with that of the standard Hammarling algorithm *dlych*. Corresponding relative errors were consistently within an order of magnitude of each other (order 10^{-11}), and neither algorithm was found to be consistently superior to the other in this measure.

5. Summary and conclusions. In this paper we have addressed the parallel numerical solution of *large, dense* and *large, banded* Lyapunov equations (1.1). For the *large, dense case*, we present a static length *vector level* parallel implementation of the Hammarling algorithm. This method solves for the Cholesky factor $GG^T = X$ of the solution of the Lyapunov equation by partitioning G into vertical strips and solving for G along antidiagonal wavefronts. In the *large, banded* case we present the full-rank perturbed iteration algorithm that solves a sequence of *block-diagonal* Lyapunov equations in order to compute the solution X of (1.1). Numerical experiments show that these algorithms efficiently utilize the processors on an Alliant FX-8 for problem sizes $n \geq 100$. We are in the process of acquiring physical plant models with which to further validate our results obtained with randomly generated plant models.

Acknowledgment. We are indebted to Dr. Ahmed Sameh at the Center for Supercomputing Research and Development, University of Illinois at Urbana-Champaign, for his helpful input and suggestions during the course of this work.

REFERENCES

[1] M. J. BALAS, *Trends in large space structure control theory: Fondest hopes, wildest dreams*, IEEE Trans. Automat. Control, AC-27 (1982), pp. 522–535.

[2] R. H. BARTELS AND G. W. STEWART, *Solution of the matrix equation $AX + XB = C$*, Comm. ACM, 15 (1972), pp. 820–826.

[3] C. BISCHOF AND C. VAN LOAN, *The WY representation for products of Householder matrices*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. s2–s13.

[4] G. J. DAVIS, R. E. FUNDERLIC, AND G. A. GEIST, *A hypercube implementation of the implicit double shift QR algorithm*, in Hypercube Multiprocessors 1987: Proceedings of the Second Annual Conference on Hypercube Multiprocessors, Knoxville, TN, September 29–October 1, 1987, Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 619–626.

[5] P. J. EBERLEIN, *On using the Jacobi method on the hypercube*, in Hypercube Multiprocessors 1987: Proceedings of the Second Annual Conference on Hypercube Multiprocessors, Knoxville, TN, September 29–October 1, 1987, Society for Industrial and Applied Mathematics, Philadelphia, PA, pp. 605–611.

[6] H. C. ELMAN, Y. SAAD, AND P. E. SAYLOR, *A hybrid Cheyshev Krylov subspace algorithm for solving nonsymmetric systems of linear equations*, SIAM J. Sci. Statist. Comput., 7 (1986), pp. 840–855.

[7] B. A. FRANCIS, *A course in H_∞ control theory*, Lecture Notes in Control and Information Sciences 88, Springer-Verlag, New York, 1987.

[8] G. A. GEIST, A. LU, AND E. L. WACHSPRESS, *Stabilized Gaussian reduction of an arbitrary matrix to tridiagonal form*, Tech. Report ORNL/TM-11089, Oak Ridge National Laboratory, Oak Ridge, TN, June 1989.

[9] A. GEORGE, M. T. HEATH, J. LIU, AND E. NG, *Sparse Cholesky factorization on a local-memory multiprocessor*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 327–340.

[10] K. GLOVER, *All optimal Hankel-norm approximations of linear multivariable systems and their L_∞-error bounds*, Internat. J. Control, 39 (1984), pp. 1115–1193.

[11] G. H. GOLUB, S. NASH, AND C. VAN LOAN, *A Hessenberg–Schur method for the problem $AX + XB = C$*, IEEE Trans. Automat. Control, AC-24 (1979), pp. 909–913.

[12] G. H. GOLUB AND C. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[13] S. J. HAMMARLING, *Numerical solution of the stable, non-negative definite Lyapunov equation*, IMA J. Numer. Anal., 2 (1982), pp. 303–323.

[14] M. T. HEATH AND C. H. ROMINE, *Parallel solution of triangular systems on distributed memory multiprocessors*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 558–588.

[15] D. HELLER, *A survey of parallel algorithms in numerical linear algebra*, SIAM Rev., 20 (1978), pp. 740–777.

[16] G. HEWER AND C. KENNEY, *The sensitivity of the stable Lyapunov equation*, SIAM J. Control Optim., 26 (1988), pp. 321–344.

[17] A. S. HODEL, *Numerical Methods for the Solution of Large and Very Large, Sparse Lyapunov Equations*, Ph.D. thesis, Department of Electrical and Computer Engineering, University

of Illinois at Urbana-Champaign, Urbana, IL, 1989.

[18] A. S. HODEL AND K. POOLLA, *Heuristic methods to the solution of very large, sparse Lyapunov and algebraic Riccati equations*, in Proc. of the 27th IEEE Conference Decision and Control, Austin, TX, Dec. 7–9, 1988, pp. 2217–2222.

[19] D. C. HYLAND AND D. S. BERNSTEIN, *The optimal projection equations for fixed-order dynamic compensation*, IEEE Trans. Automat. Control, AC-29 (1984), pp. 1034–1037.

[20] E. A. JONCKHEERE AND L. M. SILVERMAN, *A new set of invariants for linear systems–application to reduce order compensator design*, IEEE Trans. Automat. Control, AC-28 (1983), pp. 953–964.

[21] J. LASALLE AND S. LEFSCHETZ, *Stability of Liapunov's Direct Method*, Academic Press, New York, 1961.

[22] A. J. LAUB, M. T. HEATH, C. PAIGE, AND R. C. WARD, *Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms*, IEEE Trans. Automat. Control, AC-32 (1987), pp. 115–122.

[23] A. LU, *Alternating Direction Implicit iteration solution of Lyapunov equations*, Master's thesis, Department of Mathematics, University of Tennessee, Knoxville, TN, 1990.

[24] C. MOLER AND C. VAN LOAN, *Nineteen dubious ways to compute the exponential of a matrix*, SIAM Rev., 20 (1978), pp. 801–836.

[25] D. P. O'LEARY AND G. W. STEWART, *Data-flow algorithms for parallel matrix computations*, Comm. ACM, 28 (1985), pp. 840–853.

[26] S. RICHTER AND E. G. COLLINS JR., *A homotopy algorithm for reduced order compensator design using the optimal projection equations*, in Proc. 28th IEEE Conference on Decision and Control, Tampa, FL, Dec. 13–15, 1989, pp. 506–511.

[27] S. RICHTER AND A. S. HODEL, *Homotopy methods for the solution of general modified algebraic Riccati equations*, in Proc. 29th IEEE Conference on Decision and Control, Honolulu, HI, Dec. 5–7, 1990, pp. 971–976.

[28] E. L. WACHSPRESS, *Iterative solution of the Lyapunov matrix equation*, Appl. Math. Lett., 1 (1989), pp. 87–90.

[29] S. WOLFRAM, *Mathematica™: A System for Doing Mathematics by Computer*, Addison-Wesley, New York, 1988.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1204–1245, October 1992

015

JACOBI'S METHOD IS MORE ACCURATE THAN QR*

JAMES DEMMEL† AND KREŠIMIR VESELIĆ‡

Abstract. It is shown that Jacobi's method (with a proper stopping criterion) computes small eigenvalues of symmetric positive definite matrices with a uniformly better relative accuracy bound than QR, divide and conquer, traditional bisection, or any algorithm which first involves tridiagonalizing the matrix. Modulo an assumption based on extensive numerical tests, Jacobi's method is optimally accurate in the following sense: if the matrix is such that small relative errors in its entries cause small relative errors in its eigenvalues, Jacobi will compute them with nearly this accuracy. In other words, as long as the initial matrix has small relative errors in each component, even using infinite precision will not improve on Jacobi (modulo factors of dimensionality). It is also shown that the eigenvectors are computed more accurately by Jacobi than previously thought possible. Similar results are proved for using one-sided Jacobi for the singular value decomposition of a general matrix.

Key words. Jacobi, symmetric eigenproblem, singular value decomposition

AMS(MOS) subject classifications. 65F15, 65G05

1. Introduction.

Jacobi's method and QR iteration are two of the most common algorithms for solving eigenvalue and singular value problems. Both are backward stable, and so compute all eigenvalues and singular values with an absolute error bound equal to $p(n)\varepsilon \|H\|_2$, where $p(n)$ is a slowly growing function of the dimension n of the matrix H, ε is the machine precision, and $\|H\|_2$ is the spectral norm of the matrix. Thus large eigenvalues and singular values (those near $\|H\|_2$) are computed with high relative accuracy, but tiny ones may not have any relative accuracy at all. Indeed, it is easy to find symmetric positive definite matrices where QR returns negative eigenvalues. This error analysis does not distinguish Jacobi and QR, and so we might expect Jacobi to compute tiny values with as little relative accuracy as QR.

In this paper we show that Jacobi (with a proper stopping criterion) computes eigenvalues of positive definite symmetric matrices, and singular values of general matrices with a uniformly better relative error bound than QR, or any other method which initially tridiagonalizes (or bidiagonalizes) the matrix. This includes divide and conquer algorithms, traditional bisection, Rayleigh quotient iteration, and so on. We also show that Jacobi computes eigenvectors and singular vectors with better error bounds than QR.

In fact, for the symmetric positive definite eigenproblem, we show that Jacobi is optimally accurate in the following sense. Suppose the initial matrix entries have small relative uncertainties, perhaps from prior computations. The eigenvalues will then themselves have inherent uncertainties, independent of which algorithm is used to compute them. We show that the eigenvalues computed by Jacobi have error bounds which are nearly as small as these inherent uncertainties. In other words, as long as the initial data is slightly uncertain, even using infinite precision cannot

* Received by the editors October 15, 1989; accepted for publication (in revised form) December 20, 1990.

† Computer Science Division and Mathematics Department, University of California, Berkeley, California 94720. This research was performed while the author was visiting Fernuniversität Hagen, D-5800 Hagen, Germany. This research was supported by National Science Foundation grants DCR-8552474 and ASC-8715728 and by Defense Advanced Research Projects Agency grant F49620-87-C-0065. This author is also a Presidential Young Investigator.

‡ Lehrgebiet Mathematische Physik, Fernuniversität Hagen, D-5800 Hagen, Germany.

improve on Jacobi (modulo factors of n). For the singular value decomposition, we can prove a similar, but necessarily somewhat weaker, result.

These results depend on new perturbation theorems for eigenvalues and eigenvectors (or singular values and singular vectors) as well as a new error analysis of Jacobi, all of which are stronger than their classical counterparts. They also depend on an empirical observation for which we have overwhelming numerical evidence but somewhat weaker theoretical understanding.

First, we discuss the new perturbation theory for eigenvalues, contrasting the standard error bounds with the new ones. Let H be a positive definite symmetric matrix, and δH a small perturbation of H in the sense that $|\delta H_{ij}/H_{ij}| \le \eta/n$ for all i and j. Then $\|\delta H\|_2 \le \eta \|H\|_2$. Let λ_i and λ_i' be the ith eigenvalues of H and $H+\delta H$, respectively (numbered so that $\lambda_1 \le \cdots \le \lambda_n$). Then the standard perturbation theory [16] states that

$$\frac{|\lambda_i - \lambda_i'|}{\lambda_i} \le \frac{\eta \|H\|_2}{\lambda_i} \le \eta \|H\|_2 \cdot \|H^{-1}\|_2 = \eta\kappa(H), \tag{1.1}$$

where $\kappa(H) \equiv \|H\|_2 \cdot \|H^{-1}\|_2$ is the condition number of H. We prove the following stronger result: Write $H = DAD$, where $D = \operatorname{diag}(H_{ii}^{1/2})$ and $A_{ii} = 1$. By a theorem of van der Sluis [21], [6], $\kappa(A)$ is less than n times $\min_{\hat{D}} \kappa(\hat{D}H\hat{D})$, i.e., it nearly minimizes the condition number of H over all possible diagonal scalings. Then we show that

$$\frac{|\lambda_i - \lambda_i'|}{\lambda_i} \le \eta\kappa(A), \tag{1.2}$$

i.e., the error bound $\eta\kappa(H)$ is replaced by $\eta\kappa(A)$. Clearly, it is possible that $\kappa(A) \ll \kappa(H)$ (and it is always true that $\kappa(A) \le n\kappa(H)$), so the new bound is always at least about as good as, and can be much better than, the old bound.

In the case of the singular values of a general matrix G, we similarly replace the conventional relative error bound $\eta\kappa(G)$ with $\eta\kappa(B)$, where $G = BD$, D chosen diagonal so the columns of B have unit two-norm. This implies $\kappa(B) \le n^{1/2} \min_{\hat{D}} \kappa(G\hat{D})$, and, as before, it is possible that $\kappa(B) \ll \kappa(G)$.

The effects of rounding errors in Jacobi are bounded as follows. We can weaken the assumption of small componentwise relative error $|\delta H_{ij}/H_{ij}| \le \eta/n$ in the perturbation theory to $|\delta H_{ij}|/(H_{ii}H_{jj})^{1/2} \le \eta/n$ without weakening bound (1.2). This more general perturbation bounds the rounding errors introduced by applying *one* Jacobi rotation, so that one Jacobi rotation causes relative errors in the eigenvalues bounded by $O(\varepsilon)\kappa(A)$. (In contrast, QR, or any algorithm that first tridiagonalizes the matrix, only computes eigenvalues with relative error bound $O(\varepsilon)\kappa(H)$.)

To bound the errors from *all* the Jacobi rotations, we proceed as follows. Let $H_0 = D_0A_0D_0$ be the original matrix and let $H_m = D_mA_mD_m$ where H_m is obtained from H_{m-1} by applying a single Jacobi rotation, D_m is diagonal, and A_m has unit diagonal. The desired error bound is proportional to $\kappa(A_0)$, i.e., it depends only on the original matrix. But our analysis only says that at step m we get an error bounded by something proportional to $\kappa(A_m)$. Thus the error bound for all the Jacobi steps is proportional to $\max_m \kappa(A_m)$. So, for Jacobi to attain optimal accuracy, $\max_m \kappa(A_m)/\kappa(A_0)$ must be modest in size. In extensive random numerical tests, its maximum value was less than 1.82. Wang [23] has recently found isolated examples where it is almost 8. Our theoretical understanding of this behavior is incomplete and providing it remains an open problem.

We must finally bound the errors introduced by Jacobi's stopping criterion. To achieve accuracy proportional to $\kappa(A)$, we have had to modify the standard stopping criterion. Our modified stopping criterion has been suggested before [22], [5], [3], [20], but without our explanation of its benefits. The standard stopping criterion may be written thus:

$$\text{if } |H_{ij}| \leq \text{tol} \cdot \max_{kl} |H_{kl}|, \quad \text{set } H_{ij} = 0,$$

whereas the new one is

$$\text{if } |H_{ij}| \leq \text{tol} \cdot (H_{ii}H_{jj})^{1/2}, \quad \text{set } H_{ij} = 0$$

(here *tol* is a small threshold value, usually machine precision).

Now we consider the eigenvectors and singular vectors. Here and throughout the paper whenever we refer to an eigenvector, we assume its eigenvalue is simple. Again, let H be a positive definite symmetric matrix with eigenvalues λ_i and unit eigenvectors v_i. Let δH be a small componentwise relative perturbation as before, and let λ_i' and v_i' be the eigenvalues and eigenvectors of $H + \delta H$. Then the standard perturbation theory [16] says that v_i' can be chosen such that

$$\|v_i - v_i'\| \leq \frac{\eta}{\text{absgap}_{\lambda i}} + O(\eta^2), \tag{1.3}$$

where the *absolute gap for eigenvalues* is defined as

$$\text{absgap}_{\lambda i} \equiv \min_{j \neq i} \frac{|\lambda_i - \lambda_j|}{\|H\|_2}. \tag{1.4}$$

We prove a generally stronger result, which replaces this bound with

$$\|v_i - v_i'\| \leq \frac{(n-1)^{1/2}\kappa(A) \cdot \eta}{\text{relgap}_{\lambda i}} + O(\eta^2), \tag{1.5}$$

where the *relative gap for eigenvalues* is defined as

$$\text{relgap}_{\lambda i} \equiv \min_{j \neq i} \frac{|\lambda_i - \lambda_j|}{|\lambda_i \cdot \lambda_j|^{1/2}}. \tag{1.6}$$

The point is that if H has two or more tiny eigenvalues, their absolute gaps are necessarily small, but their relative gaps may be large, so that the corresponding eigenvectors are really well conditioned. We prove an analogous perturbation theorem for singular vectors of general matrices. We also prove a perturbation theorem which shows that even tiny components of eigenvectors and singular vectors may be well conditioned. Again, we show that Jacobi is capable of computing the eigenvectors and singular vectors to their inherent accuracies, but QR is not.

To illustrate, consider the symmetric positive definite matrix $H = DAD$, where

$$H = \begin{bmatrix} 10^{40} & 10^{29} & 10^{19} \\ 10^{29} & 10^{20} & 10^{9} \\ 10^{19} & 10^{9} & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & .1 & .1 \\ .1 & 1 & .1 \\ .1 & .1 & 1 \end{bmatrix}, \quad D = \text{diag}\,(10^{20}, 10^{10}, 1).$$

Here $\kappa(H) \approx 10^{40}$ and $\kappa(A) \approx 1.33$. Thus η relative perturbations in the matrix entries only cause 4η relative perturbations in the eigenvalues according to the new

theorem, and $3 \cdot 10^{40} \cdot \eta$ relative perturbations according to the conventional theorem. Also, the absolute gaps for the eigenvalues of H are $\text{absgap}_{\lambda 1,2,3} \approx 10^{-20}, 10^{-20}, 1$, whereas, the relative gaps $\text{relgap}_{\lambda 1,2,3}$ are all approximately 10^{10}. Thus the new theory predicts errors in v_1 and v_2 of norm $2 \cdot 10^{-10}\eta$, whereas the old theory predicts errors of $10^{20}\eta$. Jacobi attains these new error bounds, but QR generally does not. For this example, QR computes two out of the three eigenvalues as negative, whereas H is positive definite. In contrast, Jacobi computes all the eigenvalues to nearly full machine precision. In fact, for this example we can show that Jacobi computes all components of all eigenvectors to nearly full relative accuracy, even though they vary by 21 orders of magnitude; again, QR does not even get the signs of many small components correct.

One might object to this example on the grounds that by reversing the order of the rows and columns before tridiagonalizing and applying QR, we compute the correct eigenvalues. However, we can easily find similar matrices (see §7) where Jacobi gets accurate eigenvalues and QR gets at least one zero or negative eigenvalue, no matter how the rows and columns are ordered.

We also show that bisection and inverse iteration (with appropriate pivoting, and applied to the original positive definite symmetric matrix) are capable of attaining the same error bounds as Jacobi. Of course, bisection and inverse iteration on a dense matrix are not competitive in speed with Jacobi, unless only one or a few eigenvalues are desired and good starting guesses are available. We use these methods to verify our numerical tests.

This work is an extension of work in [2], where analogous results were proven for matrices that are called *scaled diagonally dominant* (s.d.d.). The positive definite matrix $H = DAD$ is s.d.d. if $\|A - I\|_2 < 1$. This work replaces the assumption that A is diagonally dominant with mere positive definiteness, extending the results of [2] to all positive definite symmetric matrices, as well as to the singular value decomposition of general matrices.

This work does not contradict the results of [8] and [2], where it was shown how a variation of QR could compute the singular values of a bidiagonal matrix or the eigenvalues of a symmetric positive definite tridiagonal matrix with high relative accuracy. This is because reducing a dense matrix to bidiagonal or tridiagonal form can cause large relative errors in its singular values or eigenvalues independent of the accuracy of the subsequent processing. In contrast, the results in this paper are for dense matrices.

We also discuss an accelerated version of Jacobi for the symmetric positive definite eigenproblem with an attractive speedup property: The more its accuracy exceeds that attainable by QR or other traditional methods, the faster it converges. See also [22] where earlier references for Jacobi methods on positive definite matrices, as well as for one-sided methods, can be found.

We use the following terminology to distinguish among different versions of Jacobi. "Two-sided Jacobi" refers to the original method applying Jacobi rotations to the left and right of a symmetric matrix. "One-sided Jacobi" refers to computing the SVD by applying Jacobi rotation from one side only. "Right-handed Jacobi" is one-sided Jacobi applying rotations on the right, and "left-handed Jacobi" is one-sided Jacobi applying rotations on the left.

The remainder of this paper is organized as follows. Section 2 presents the new perturbations theorems. Section 3 discusses two-sided Jacobi for the symmetric positive definite eigenproblem. Section 4 discusses one-sided Jacobi for the singular value

decomposition, and also presents the accelerated version of Jacobi just mentioned. Section 5 discusses bisection and inverse iteration. Section 6 discusses bounds on $\max_m \kappa(A_m)/\kappa(A_0)$. Section 7 contains numerical experiments. Section 8 presents our conclusions and discussion of open problems.

2. Perturbation theory. In this section, we prove new perturbation theorems for eigenvalues and eigenvectors of symmetric positive definite matrices, and for singular values and singular vectors of general matrices. In §2.1, we consider eigendecompositions of symmetric positive definite matrices. In §2.2, we discuss the optimality of these bounds. In §2.3, we consider the singular value decomposition of general matrices. In §2.4, we discuss the optimality of this second set of bounds.

2.1. Symmetric positive definite matrices. The next two lemmas were proved in [2].

LEMMA 2.1. *Let H and K be symmetric matrices with K positive definite. Let the pencil $H - \lambda K$ have eigenvalues λ_i. Let δH and δK be symmetric perturbations and let λ_i' be the (properly ordered) eigenvalues of $(H + \delta H) - \lambda(K + \delta K)$. Suppose that*

$$|x^T \delta H x| \leq \eta_H \cdot |x^T H x| \quad \text{and} \quad |x^T \delta K x| \leq \eta_K \cdot |x^T K x|$$

for all vectors x and some $\eta_H < 1$ and $\eta_K < 1$. Then either $\lambda_i = \lambda_i' = 0$ or

$$\frac{1 - \eta_H}{1 + \eta_K} \leq \frac{\lambda_i'}{\lambda_i} \leq \frac{1 + \eta_H}{1 - \eta_K}$$

for all i.

LEMMA 2.2. *Let $H = \Delta_H^T A_H \Delta_H$ and A_H be symmetric matrices. H and A_H need not have the same dimensions, and Δ_H may be an arbitrary full-rank conforming matrix. Similarly, let $K = \Delta_K^T A_K \Delta_K$ and A_K be symmetric positive definite matrices, where K and A_K need not have the same dimensions and Δ_K may be an arbitrary full-rank conforming matrix. Let $\delta H = \Delta_H^T \delta A_H \Delta_H$ be a perturbation of H such that $|x^T \delta A_H x| \leq \eta_H |x^T A_H x|$ for all x where $\eta_H < 1$. Similarly, let $\delta K = \Delta_K^T \delta A_K \Delta_K$ be a perturbation of K such that $|x^T \delta A_K x| \leq \eta_K |x^T A_K x|$ for all x where $\eta_K < 1$. Let λ_i be the ith eigenvalue of $H - \lambda K$ and λ_i' the ith eigenvalue of $(H + \delta H) - \lambda(K + \delta K)$. Then either $\lambda_i = \lambda_i' = 0$ or*

$$\frac{1 - \eta_H}{1 + \eta_K} \leq \frac{\lambda_i'}{\lambda_i} \leq \frac{1 + \eta_H}{1 - \eta_K}.$$

THEOREM 2.3. *Let $H = DAD$ be a symmetric positive definite matrix, and $D = \operatorname{diag}(H_{ii}^{1/2})$ so $A_{ii} = 1$. Let $\delta H = D \delta A D$ be a perturbation such that $\|\delta A\|_2 \equiv \eta < \lambda_{\min}(A)$. Let λ_i be the ith eigenvalue of H and λ_i' be the ith eigenvalue of $H + \delta H$. Then*

$$\left| \frac{\lambda_i - \lambda_i'}{\lambda_i} \right| \leq \frac{\eta}{\lambda_{\min}(A)} \leq \kappa(A) \cdot \eta. \tag{2.1}$$

In particular, if $|\delta H_{ij}/H_{ij}| \leq \eta/n$, then $\|\delta A\|_2 \leq \eta$ and the bound (2.1) applies.

Proof. Note that for all nonzero vectors x,

$$\left| \frac{x^T \delta H x}{x^T H x} \right| = \left| \frac{x^T \Delta^T \delta A \Delta x}{x^T \Delta^T A \Delta x} \right| = \left| \frac{y^T \delta A y}{y^T A y} \right| \leq \frac{\eta}{\lambda_{\min}(A)}.$$

Lemma 2.2 yields the desired bound, using $K = I$ and $\delta K = 0$. It remains to prove that $|\delta H_{ij}/H_{ij}| \le \eta/n$ implies $\|\delta A\|_2 \le \eta$. But $A_{ii} = 1$ and A positive definite imply that no entry of A is larger than 1 in absolute value. (Note that this means $\kappa(A)$ is at most n times larger than $1/\lambda_{\min}(A)$.) Therefore, $|\delta A_{ij}| = |\delta H_{ij}/H_{ij} \cdot A_{ij}| \le \eta/n$ and so $\|\delta A\|_2 \le \eta$, as desired. □

Proposition 2.10 in the next subsection shows that the bound of Theorem 2.3 is nearly attained for at least one eigenvalue. However, other eigenvalues may be much less sensitive than this most sensitive one. The next proposition provides individual eigenvalue bounds which may be much tighter.

PROPOSITION 2.4. *Let $H = DAD$ be as in Theorem 2.3, with eigenvalues λ_i and unit eigenvectors v_i. Let $H + \delta H = D(A + \delta A)D$ have eigenvalues λ_i'. Let $\|\delta A\|_2 \equiv \eta \ll \lambda_{\min}(A)$. Then the bound*

$$\frac{|\lambda_i - \lambda_i'|}{\lambda_i} \le \frac{\eta \|Dv_i\|_2^2}{\lambda_i} + O(\eta^2) \tag{2.2}$$

is attainable by the diagonal perturbation $\delta A_{jj} = \eta$.

Proof. Bound (2.2) is derived from the standard first-order perturbation theory, which says that $\lambda_i(H + \delta H) = \lambda_i(H) + v_i^T \delta H v_i + O(\|\delta H\|_2^2)$, and substituting $|v_i^T \delta H v_i| = |v_i^T D \delta A D v_i| \le \|Dv_i\|_2^2 \|\delta A\|_2$. The inequality $|v_i^T D \delta A D v_i| \le \|Dv_i\|_2^2 \|\delta A\|_2$ is clearly attained for the diagonal choice of δA in the statement of the proposition. □

We may also prove a version of Lemma 2.1 in an infinite-dimensional setting [14, §VI.3].

Now we turn to eigenvectors. A weaker version of the following theorem also appeared in [2].

THEOREM 2.5. *Let $H = DAD$ be as in Theorem 2.3. Define $H(\epsilon) = D(A + \epsilon E)D$, where E is any matrix with unit two-norm. Let $\lambda_i(\epsilon)$ be the ith eigenvalue of $H(\epsilon)$, and assume that $\lambda_i(0)$ is simple so that the corresponding unit eigenvector $v_i(\epsilon)$ is well defined for sufficiently small ϵ. Then*

$$\|v_i(\epsilon) - v_i(0)\|_2 \le \frac{(n-1)^{1/2}\epsilon}{\lambda_{\min}(A) \cdot \mathrm{relgap}_{\lambda i}} + O(\epsilon^2) \le \frac{(n-1)^{1/2}\kappa(A)\epsilon}{\mathrm{relgap}_{\lambda i}} + O(\epsilon^2).$$

Proof. Let $v_k(0)$ be abbreviated by v_k. From [11] we have

$$v_i(\epsilon) = v_i + \epsilon \sum_{k \ne i} \frac{v_k^T DED v_i}{\lambda_i - \lambda_k} \cdot v_k + O(\epsilon^2).$$

Let $y_k = Dv_k$, so that

$$v_i(\epsilon) = v_i + \epsilon \sum_{k \ne i} \frac{y_k^T E y_i}{\lambda_i - \lambda_k} \cdot v_k + O(\epsilon^2). \tag{2.3}$$

The pair (λ_i, y_i) is an eigenpair of the pencil $A - \lambda D^{-2}$. Thus

$$\lambda_k = \lambda_k y_k^T D^{-2} y_k = y_k^T A y_k \ge \lambda_{\min}(A) \|y_k\|_2^2,$$

and so $\|y_k\|_2 \le (\lambda_k/\lambda_{\min}(A))^{1/2}$. Letting $z_k = y_k/\|y_k\|_2$ lets us write

$$v_i(\epsilon) = v_i + \epsilon \sum_{k \ne i} \frac{\xi_{ik} \cdot z_k^T E z_i}{(\lambda_i - \lambda_k)/(\lambda_k \lambda_i)^{1/2}} \cdot v_k + O(\epsilon^2),$$

where $|\xi_{ik}| = \|y_k\|_2\|y_i\|_2/(\lambda_k\lambda_i)^{1/2} \le 1/\lambda_{\min}(A)$. Taking norms yields the result. □

Proposition 2.11 in the next subsection shows that the bound in Theorem 2.5 is nearly attainable for all v_i.

As in Corollary 3 in [2], it is possible to derive a nonasymptotic result from Theorem 2.5.

COROLLARY 2.6. *Let $H = DAD$ be as in Theorem 2.3. Suppose that $\delta \equiv \|\delta A\|_2/\lambda_{\min}(A)$ satisfies*

$$\delta < \frac{1}{4} \text{ and } \frac{3\cdot 2^{-1/2}\cdot\delta}{1-\delta} < \text{relgap}_{\lambda i}.$$

Let v_i be the ith unit eigenvector of $H = DAD$. Then the ith unit eigenvector v_i' of $H' = D(A+\delta A)D$ can be chosen so that

$$\|v_i - v_i'\|_2 \le \frac{(n-1)^{1/2}\delta}{(1-4\delta)((1-\delta)\text{relgap}_{\lambda i} - 3\cdot 2^{-1/2}\delta)}.$$

Proof. Let $H(\epsilon) = D(A + \epsilon\cdot\delta A/\|\delta A\|_2)D$. Let $\lambda_i(\epsilon)$ be the ith eigenvalue of $H(\epsilon)$, and abbreviate $\lambda_i(0)$ by λ_i. Let $\text{relgap}_{\lambda i}(\epsilon)$ denote the relative gap of the ith eigenvalue of $H(\epsilon)$, and $\text{relgap}_\lambda(a,b) \equiv |a-b|/(ab)^{1/2}$. The idea is that if ϵ is small, then $\lambda_i(\epsilon)$ can only change by a small relative amount, and so $\text{relgap}_{\lambda i}(\epsilon)$ can only change by a small absolute or relative amount. Note that $\lambda_{\min}(A)$ can decrease by as much as $\|\delta A\|_2$. Then by Theorem 2.3, we can bound $\text{relgap}_{\lambda i}(\epsilon)$ below by

$$\begin{aligned}\text{relgap}_{\lambda i}(\epsilon) &= \min_{k\ne i}\frac{|\lambda_i(\epsilon)-\lambda_k(\epsilon)|}{(\lambda_i(\epsilon)\lambda_k(\epsilon))^{1/2}} \ge \min_{k\ne i}\frac{|\lambda_i-\lambda_k| - \delta(1-\delta)^{-1}(\lambda_i+\lambda_k)}{(\lambda_i\lambda_k)^{1/2}(1+\delta(1-\delta)^{-1})}\\ &\ge (1-\delta)\min_{k\ne i}\left(\text{relgap}_\lambda(\lambda_i,\lambda_k) - \frac{\delta}{1-\delta}\cdot\frac{\lambda_i+\lambda_k}{(\lambda_i\lambda_k)^{1/2}}\right).\end{aligned}$$

We consider two cases, $\text{relgap}_\lambda(\lambda_i,\lambda_k) \ge 2^{-1/2}$ and $\text{relgap}_\lambda(\lambda_i,\lambda_k) < 2^{-1/2}$. The first case corresponds to λ_i and λ_k differing by at least a factor of 2, whence

$$\frac{\lambda_i+\lambda_k}{(\lambda_i\lambda_k)^{1/2}} \le 3\cdot\text{relgap}_\lambda(\lambda_i,\lambda_k).$$

The second case corresponds to λ_i and λ_k differing by at most a factor of 2, whence

$$\frac{\lambda_i+\lambda_k}{(\lambda_i\lambda_k)^{1/2}} \le 3\cdot 2^{-1/2}.$$

Altogether, we have

$$\text{relgap}_{\lambda i}(\epsilon) \ge (1-\delta)\left(1-\frac{3\delta}{1-\delta}\right)\left(\text{relgap}_{\lambda i} - \frac{3\cdot 2^{-1/2}\cdot\delta}{1-\delta}\right).$$

Now integrate the bound of Theorem 2.5 from $\epsilon = 0$ to $\epsilon = \|\delta A\|_2$ to get the desired result. □

In complete analogy to [2], we may also prove the following proposition.

PROPOSITION 2.7. *Let $\lambda_1 \le \cdots \le \lambda_n$ be the eigenvalues of H and $h_1 \le \cdots \le h_n$ be its diagonal entries in increasing order. Then*

$$\lambda_{\min}(A) \le \frac{\lambda_i}{h_i} \le \lambda_{\max}(A).$$

In other words, the diagonal entries of H can differ from the eigenvalues only by factors bounded by $\kappa(A)$.

Proof. See the proof of Proposition 2 in [2]. □

PROPOSITION 2.8. *Let $H = DAD$ with eigenvalues λ_i. Let d_i be the diagonal entries of D. Let v_i be the ith eigenvector of H normalized so that its ith component $v_i(i) = 1$. Then*

$$|v_i(j)| \le \overline{v}_i(j) \equiv (\kappa(A))^{3/2} \cdot \min\left(\left(\frac{\lambda_i}{\lambda_j}\right)^{1/2}, \left(\frac{\lambda_j}{\lambda_i}\right)^{1/2} \right).$$

We also have

$$|v_i(j)| \le (\kappa(A))^{3/2} \cdot \min\left(\frac{d_i}{d_j}, \frac{d_j}{d_i} \right).$$

In other words, the eigenvectors are scaled analogously to the diagonal of H.

Proof. See the proof of Proposition 6 in [2]. □

PROPOSITION 2.9. *Let $H(\epsilon)$ and $v_i(\epsilon)$ be as in Theorem 2.5, and $\overline{v}_i(j)$ be as in Proposition 2.8. Then*

$$|v_i(\epsilon)(j) - v_i(0)(j)| \le \frac{(2n-2)^{1/2}}{\lambda_{\min}(A) \cdot \min(\text{relgap}_{\lambda i}, 2^{-1/2})} \cdot \epsilon \cdot \overline{v}_i(j) + O(\epsilon^2).$$

In other words, each component of each eigenvector is perturbed by a small amount relative to its upper bound of $\overline{v}_i(j)$ of Proposition 2.8. Thus small components of eigenvectors may be determined with as much relative accuracy as large components. Note that $\text{relgap}_{\lambda i}$ *exceeds $2^{-1/2}$ only when λ_i differs from its nearest neighbor by at least a factor of 2.*

Proof. See the proof of Theorem 7 in [2]. □

We illustrate these results with two examples. First, we consider the matrix $H = DAD$ of the introduction:

$$H = \begin{bmatrix} 10^{40} & 10^{29} & 10^{19} \\ 10^{29} & 10^{20} & 10^{9} \\ 10^{19} & 10^{9} & 1 \end{bmatrix}, \quad A = \begin{bmatrix} 1 & .1 & .1 \\ .1 & 1 & .1 \\ .1 & .1 & 1 \end{bmatrix}, \quad D = \text{diag}\,(10^{20}, 10^{10}, 1).$$

To six correct figures, H's eigenvalue matrix Λ and eigenvector matrix V (normalized to have the largest entry of each eigenvector equal to 1) are

$$\Lambda = \text{diag}\,(1.00000 \cdot 10^{40}, 9.90000 \cdot 10^{19}, 9.81818 \cdot 10^{-1})$$

and

$$V = \begin{bmatrix} 1.00000 & -1.00000 \cdot 10^{-11} & -9.09091 \cdot 10^{-22} \\ 1.00000 \cdot 10^{-11} & 1.00000 & -9.09091 \cdot 10^{-12} \\ 1.00000 \cdot 10^{-21} & 9.09091 \cdot 10^{-12} & 1.00000 \end{bmatrix}.$$

We may compute that $\kappa(H) \approx 10^{40}$ and $\kappa(A) \approx 1.33$. Thus, according to Theorem 2.3, changing each entry of H in its seventh decimal place or beyond would not change Λ in the figures shown. The refined error bounds of Proposition 2.4 are essentially the same in this case. We can further verify the assertion of Proposition 2.7 that the ratios of the eigenvalues to the diagonal entries of H are bounded between $.9 = \lambda_{\min}(A)$ and

$1.2 = \lambda_{\max}(A)$. One may also compute that the relative gaps $\mathrm{relgap}_{\lambda i}$ for all three eigenvalues are approximately 10^{10}. Thus, according to Theorem 2.5, seventh-figure changes in H would not change its eigenvectors by more than 10^{-16} in norm. In fact, the eigenvectors are even more accurately determined than this. Let $\bar{V} = \{\bar{v}_i(j)\}$ be the matrix of upper bounds of entries of V as defined in Proposition 2.8:

$$\bar{V} \approx \begin{bmatrix} 1.5 & 1.5 \cdot 10^{-10} & 1.5 \cdot 10^{-20} \\ 1.5 \cdot 10^{-10} & 1.5 & 1.5 \cdot 10^{-10} \\ 1.5 \cdot 10^{-20} & 1.5 \cdot 10^{-10} & 1.5 \cdot 10^{-20} \end{bmatrix}.$$

Then, according to Proposition 2.9, seventh-figure changes in H cause changes in at most the fifth digits of all the entries of V. In other words, for this, examples of all the eigenvalues and all the components of all the eigenvectors, are determined to nearly full relative precision by the data. Later, we show that Jacobi can compute them with this accuracy. In contrast, QR does not even get the signs of the two small eigenvalues or many components of the eigenvectors correct.

The second example serves to illustrate the difference between Theorem 2.3 and the refined bounds of Proposition 2.4. Let $H = DAD$ where D is the same as before and

$$A = \begin{bmatrix} 1 & 1-\mu & 1-\mu \\ 1-\mu & 1 & 1-\mu \\ 1-\mu & 1-\mu & 1 \end{bmatrix},$$

where $\mu = 10^{-6}$. The eigenvalues of H are 10^{40}, $2 \cdot 10^{14}$, and $1.5 \cdot 10^{-6}$. Now $\kappa(A) \approx 10^6$, so according to Theorem 2.3, an η relative change in the matrix entries causes as much as a $10^6\eta$ relative change in the eigenvalues. In contrast, the refined bounds predict a relative change of η in 10^{40} and $10^6\eta$ in the two smaller eigenvalues. Thus the largest eigenvalue is just as insensitive as predicted by standard norm-based perturbation theory.

2.2. Optimality of the bounds for symmetric positive definite matrices. In this section, we show that the bounds of the last section are attainable. In other words, the only symmetric positive definite matrices whose eigenvalues are determined to high relative accuracy by the matrix entries are those $H = DAD$, where A is well conditioned.

In particular, we give explicit small, componentwise, relative perturbations, which attain the eigenvalue bounds; it suffices to choose a diagonal perturbation. We have (necessarily) slightly weaker results for the optimality of our eigenvector bounds.

We begin by showing that the assumption $\|\delta A\|_2 < \lambda_{\min}(A)$ of the last section is essential to having relative error bounds at all. If this bound were violated, $A + \delta A$ (and so $H + \delta H$) could become indefinite, implying that all relative accuracy in at least one eigenvalue is completely lost. In contrast to standard perturbation theory, however, which assumes a bound on $\|\delta H\|_2$ instead of $\|\delta A\|_2$, one cannot say which eigenvalue will lose relative accuracy first. In the conventional case, as $\|\delta H\|_2$ grows, it is the smallest eigenvalues that lose accuracy first, the larger ones remaining accurate. As $\|\delta A\|_2$ grows, however, *any* eigenvalue in the spectrum (except the very largest) may lose its relative accuracy first. The following example illustrates this:

$$H = \begin{bmatrix} 10^{20} & & & \\ & 1 & .99 & \\ & .99 & 1 & \\ & & & 10^{-20} \end{bmatrix}, \qquad A = \begin{bmatrix} 1 & & & \\ & 1 & .99 & \\ & .99 & 1 & \\ & & & 1 \end{bmatrix},$$

and $D = \text{diag}\,(10^{10}, 1, 1, 10^{-10})$. Note that $\lambda_{\min}(A) = .01$. As $\|\delta A\|_2$ approaches .01, the eigenvalues near 10^{20}, 1.99, and 10^{-20} retain their accuracy, but the one near .01 can lose all its relative accuracy.

We next show that the relative error bound of Theorem 2.1 can be nearly attained for at least one eigenvalue simply by making appropriate small relative perturbations to the diagonal of H.

PROPOSITION 2.10. *Let $H = DAD$ be symmetric positive definite, with $D = \text{diag}\,(H_{ii}^{1/2})$ diagonal and $A_{ii} = 1$. Let $\delta A = \eta I$, $0 < \eta < \lambda_{\min}(A)$, and $H + \delta H = D(A + \delta A)D$. Then for some i we have*

$$\frac{\lambda_i(H + \delta H)}{\lambda_i(H)} \geq \left(1 + \frac{\eta}{\lambda_{\min}(A)}\right)^{1/n} \approx 1 + \frac{\eta}{n\lambda_{\min}(A)}.$$

Proof. We have

$$\prod_i \lambda_i(H) = \det(DAD) = \det(D^2)\det(A) = \det(D^2)\prod_i \lambda_i(A)$$

and

$$\prod_i \lambda_i(H + \delta H) = \det(D(A + \eta I)D) = \det(D^2)\det(A + \eta I) = \det(D^2)\prod_i(\lambda_i(A) + \eta).$$

Therefore,

$$\prod_i \frac{\lambda_i(H + \delta H)}{\lambda_i(H)} = \prod_i \frac{\lambda_i(A) + \eta}{\lambda_i(A)} \geq 1 + \frac{\eta}{\lambda_{\min}(A)},$$

implying that at least one factor $\lambda_i(H + \delta H)/\lambda_i(H)$ must exceed $(1 + \eta/\lambda_{\min}(A))^{1/n}$. This last expression is approximately $1 + \eta/(n\lambda_{\min}(A))$, when $\eta \ll \lambda_{\min}(A)$. □

The example at the beginning of this section showed that the error bound of Theorem 2.3 and the last proposition may only be attained for one eigenvalue. Proposition 2.4 of §2.1 showed that for asymptotically small $\|\delta A\|_2$, the maximum perturbation in each eigenvalue may be attained only with small diagonal perturbations of A.

After we show that the rounding errors introduced by Jacobi are of the form $\|\delta A\|_2 = O(\varepsilon)$ in §2.3, Propositions 2.10 and 2.4 show that Jacobi (modulo the assumption on $\max_m \kappa(A_m)/\kappa(A_0)$) computes all the eigenvalues with optimal accuracy, provided that only the diagonal entries of H have small relative errors. The same optimality property is true of bisection.

Now we consider eigenvectors. Here our results are necessarily weaker, as the following example shows. Suppose H is diagonal with distinct eigenvalues. Then small relative perturbations to the matrix entries leave H diagonal and its eigenvalue matrix (the identity matrix) unchanged. Therefore, the only way we can hope to attain the bounds of Theorem 2.5 is to use perturbations δA, which are possibly dense, even if H is not. Furthermore, a block diagonal example like the first one in this section shows that the attainable eigenvector perturbations do not necessarily grow with $\kappa(A)$. Thus the following is the best we can prove.

PROPOSITION 2.11. *Let $H = DAD$, λ_i, v_i, δH, δA and let η be as in Proposition 2.4. Let v_i' be the unit eigenvectors of $H + \delta H$. Then we can choose δA, $\|\delta A\|_2 \equiv \eta \ll \lambda_{\min}(A)$, so that*

$$\|v_i - v_i'\|_2 \geq \frac{\eta}{\lambda_{\max}(A)\text{relgap}_{\lambda_i}} + O(\eta^2).$$

Proof. Consider expression (2.3) for $v_i - v_i'$ (there, δA is written ϵE). By using a Householder transformation, we can prove that there exists a symmetric δA such that $y_k^T \delta A y_i = \|y_k\|_2 \|y_i\|_2 \|\delta A\|_2$ for arbitrary y_k and y_i. Since $\lambda_k = y_k^T A y_k \leq \|y_k\|^2 \lambda_{\max}(A)$, we can find δA to make $y_k^T \delta A y_i \geq (\lambda_i \lambda_k)^{1/2} \|\delta A\|_2 / \lambda_{\max}(A)$. Choosing k so that λ_k is closest to λ_i completes the proof. □

2.3. Singular value decomposition. The results on singular values and singular vectors are analogous to the results for eigenvalues and eigenvectors in the first subsection, so we do not include the proofs. Just as we derived perturbation bounds for eigenvalues from a more general result for generalized eigenvalues of pencils, we start with a perturbation bound for generalized singular values and then specialize to standard singular values.

Let G_1 and G_2 be matrices with the same number of columns, G_2 of full column rank, and both arbitrary. We define the *ith generalized singular value* $\sigma_i(G_1, G_2)$ *of the pair* (G_1, G_2) as the square root of the ith eigenvalue of the definite pencil $G_1^T G_1 - \lambda G_2^T G_2$ [11]. If we let G_2 be the identity, $\sigma_i(G_1, G_2)$ is the same as the standard singular value $\sigma_i(G_1)$ of G_1.

LEMMA 2.12. *Let G_1 and G_2 be matrices with the same number of columns, G_2 of full column rank, and both arbitrary. Let δG_j be a perturbation of G_j such that*

$$\|\delta G_j x\|_2 \leq \eta_j \|G_j x\|_2$$

for all x and some $\eta_j < 1$. Let σ_i be the ith generalized singular value of (G_1, G_2) and σ_i' be the ith generalized singular value of $(G_1 + \delta G_1, G_2 + \delta G_2)$. Then either $\sigma_i = \sigma_i' = 0$ or

$$\frac{1 - \eta_1}{1 + \eta_2} \leq \frac{\sigma_i'}{\sigma_i} \leq \frac{1 + \eta_1}{1 - \eta_2}.$$

LEMMA 2.13. *Let G_1 and G_2 be as in Lemma 2.12. Let $G_j = B_j \Delta_j$, where Δ_j has full rank and is otherwise arbitrary. Let $\delta G_j = \delta B_j \Delta_j$ be a perturbation of G_j such that $\|\delta B_j x\|_2 \leq \eta_j \|B_j x\|_2$ for all x and some $\eta_j < 1$. Let σ_i and σ_i' be the ith generalized singular values of (G_1, G_2) and $(G_1 + \delta G_1, G_2 + \delta G_2)$, respectively. Then either $\sigma_i = \sigma_i' = 0$ or*

$$\frac{1 - \eta_1}{1 + \eta_2} \leq \frac{\sigma_i'}{\sigma_i} \leq \frac{1 + \eta_1}{1 - \eta_2}.$$

THEOREM 2.14. *Let $G = BD$ be a general full-rank matrix, and let D be chosen diagonal so that the columns of B have unit two-norm (i.e., D_{ii} equals the two-norm of the ith column of G). Let $\delta G = \delta B D$ be a perturbation of G such that $\|\delta B\|_2 \equiv \eta < \sigma_{\min}(B)$. Let σ_i and σ_i' be the ith singular values of G and $G + \delta G$, respectively. Then*

$$\frac{|\sigma_i - \sigma_i'|}{\sigma_i} \leq \frac{\eta}{\sigma_{\min}(B)} \leq \kappa(B) \cdot \eta, \tag{2.4}$$

where $\kappa(B) = \sigma_{\max}(B)/\sigma_{\min}(B) \leq n^{1/2}/\sigma_{\min}(B)$, and n is the number of columns of G. In particular, if $|\delta G_{ij}/G_{ij}| \leq \eta/n$, then $\|\delta B\|_2 \leq \eta$ and the bound (2.4) applies.

Just as the bounds of Theorem 2.3 were not attainable by all eigenvalues, neither are the bounds of Theorem 2.14 attainable for all singular values. Analogous to Proposition 2.4, we may derive tighter bounds for individual singular values.

PROPOSITION 2.15. *Let $G = BD$ be as in Theorem 2.14, with singular values σ_i, right unit singular vectors v_i, and left unit singular vectors u_i. Let $G+\delta G = (B+\delta B)D$ have singular values σ_i', where $\|\delta B\|_2 \equiv \eta \ll \sigma_{\min}(B)$. Then the bound*

$$\frac{|\sigma_i - \sigma_i'|}{\sigma_i} \leq \frac{\eta \|Dv_i\|_2}{\sigma_i} + O(\eta^2) \tag{2.5}$$

is attainable by the perturbation $\delta B = \eta u_i (Dv_i)^T / \|Dv_i\|_2$.

Now we consider the singular vectors. For simplicity, we assume that G is square. We use the fact that if $G = U\Sigma V^T$ is the singular value decomposition of G, then

$$2^{-1/2} \cdot \begin{bmatrix} V & V \\ U & -U \end{bmatrix}$$

is the eigenvector matrix of the symmetric matrix [11]

$$\begin{bmatrix} 0 & G^T \\ G & 0 \end{bmatrix}.$$

Therefore, we can use perturbation theory for eigenvectors of symmetric matrices to do perturbation theory for singular vectors of general matrices.

We also need to define the gaps for the singular vector problem. The *absolute gap for singular values* is

$$\text{absgap}_{\sigma i} \equiv \min_{k \neq i} \frac{|\sigma_i - \sigma_k|}{\|G\|_2},$$

i.e., essentially the same as the absolute gap for eigenvalues. However the *relative gap for singular values*,

$$\text{relgap}_{\sigma i} \equiv \min_{k \neq i} \frac{|\sigma_i - \sigma_k|}{\sigma_i + \sigma_k},$$

is somewhat different from the relative gap for eigenvalues.

The standard perturbation theorem for singular vectors is essentially the same as for eigenvectors. Let G have right (or left) unit singular vectors v_i, and let $G + \delta G$ have right (or left) unit singular vectors v_i'. Let $\eta = \|\delta G\|_2 / \|G\|_2$. Then

$$\|v_i - v_i'\|_2 \leq \frac{\eta}{\text{absgap}_{\sigma i}} + O(\eta^2).$$

We improve this in the following theorem.

THEOREM 2.16. *Let $G = BD$ be as in Theorem 2.14. Define $G(\epsilon) = (B + \epsilon E)D$ where E is any matrix with unit two-norm. Let $\sigma_i(\epsilon)$ be the ith singular value of $G(\epsilon)$, and assume that $\sigma_i(0)$ is simple so that the corresponding right unit singular vector $v_i(\epsilon)$ and left unit singular vector $u_i(\epsilon)$ are well defined for sufficiently small ϵ. Then*

$$\max(\|v_i(\epsilon) - v_i(0)\|_2, \|u_i(\epsilon) - u_i(0)\|_2) \leq \frac{(n - .5)^{1/2} \kappa(B) \epsilon}{\text{relgap}_{\sigma i}} + O(\epsilon^2).$$

COROLLARY 2.17. *Let $G = BD$ be as in Theorem 2.14. Suppose that $\delta \equiv \|\delta B\|_2 / \sigma_{\min}(B)$ satisfies*

$$\frac{\delta}{1 - \delta} < \text{relgap}_{\sigma i}.$$

Let v_i and u_i be the unit right and left singular vectors of G, respectively, and let v_i' and u_i' be the unit right and left singular vectors of $G' = (B + \delta B)D$, respectively. Then

$$\max(\|v_i - v_i'\|_2, \|u_i - u_i'\|_2) \leq \frac{(n - .5)^{1/2}\delta}{(1 - \delta)((1 - \delta)\text{relgap}_{\sigma i} - \delta)}.$$

There are analogues to Propositions 2.7–2.9 of the last section, obtained by considering $H = G^T G$:

PROPOSITION 2.18. *Let $G = BD$ be as in Theorem 2.14. Let $\sigma_1 \leq \cdots \leq \sigma_n$ be the singular values of G and $d_1 \leq \cdots \leq d_n$ the diagonal entries of D in increasing order. Then*

$$\sigma_{\min}(B) \leq \frac{\sigma_i}{d_i} \leq \sigma_{\max}(B).$$

PROPOSITION 2.19. *Let $G = BD$ be as in Theorem 2.14 with singular values $\sigma_1 \leq \cdots \leq \sigma_n$. Let v_i be the ith right singular vector of G, normalized so that its ith component $v_i(i) = 1$. Then*

$$|v_i(j)| \leq \bar{\bar{v}}_i(j) \equiv (\kappa(B))^3 \cdot \min\left(\frac{\sigma_i}{\sigma_j}, \frac{\sigma_j}{\sigma_i}\right).$$

We also have

$$|v_i(j)| \leq (\kappa(B))^3 \cdot \min\left(\frac{d_i}{d_j}, \frac{d_j}{d_i}\right).$$

PROPOSITION 2.20. *Let $G(\epsilon)$ and $v_i(\epsilon)$ be as in Theorem 2.16, and $\bar{\bar{v}}_i(j)$ be as in Proposition 2.19. Then*

$$|v_i(\epsilon)(j) - v_i(0)(j)| \leq \frac{2(n-1)^{1/2}}{\sigma_{\min}^2(B) \cdot \text{relgap}_{\sigma i}} \cdot \epsilon \cdot \bar{\bar{v}}_i(j) + O(\epsilon^2).$$

There are analogues to all the results in this section for matrices $G = DB$ scaled from the left instead of the right. Thus we can choose to scale either the rows or the columns of G to have unit two-norms, whichever one minimizes the condition number. It is natural to ask if we can do better by considering two-sided diagonal scaling $D_1 G D_2$; to date, we have been unable to formulate a reasonable perturbation theory. To see why, note that if G is triangular, it can be made as close to the identity matrix as desired by two-sided scaling, even though its singular values can be quite sensitive.

2.4. Optimality of the bounds for the singular value decomposition. The results in this section are analogous to, but necessarily weaker than, the results of §2.2. In particular, it is no longer the case that the perturbation bounds for the singular values can be attained by small relative perturbations in the matrix entries.

First, consider the restriction $\|\delta B\|_2 < \sigma_{\min}(B)$. Just as in the symmetric positive definite case, this is necessary so that $B + \delta B$ remains nonsingular. When $B + \delta B$ becomes singular, at least one singular value necessarily loses all relative accuracy. The same kind of block diagonal example as in §2.2 also shows that only one singular value may have its sensitivity depend on $\kappa(B)$, and it might be anywhere in the spectrum (except the very largest singular value).

In order to prove an analogue of Proposition 2.10, we must permit perturbations δB of B which are small in norm but may make large relative changes in tiny entries of B (a similar perturbation was needed to prove that the bound in Proposition 2.15 was attainable).

PROPOSITION 2.21. *Let $G = BD$ with D diagonal and the columns of B having unit two-norm. Then there exists a δB with $\|\delta B\|_2 = \eta < \sigma_{\min}(B)$ such that for $G + \delta G = (B + \delta B)D$ we have, for at least one i,*

$$\frac{\sigma_i(G+\delta G)}{\sigma_i(G)} \geq \left(1 + \frac{\eta}{\sigma_{\min}(B)}\right)^{1/n} \approx 1 + \frac{\eta}{n\sigma_{\min}(B)}.$$

If we restrict δB so that $|\delta B_{ij}/B_{ij}| \leq \eta$, then such a perturbation δB may not exist.

Proof. The proof is very similar to that of Proposition 2.10. Let X be a rank-one matrix of minimal two-norm such that $B + X$ is singular, and let $\delta B = -\eta X$. Then, as in Proposition 2.10, we discover that

$$\prod_i \frac{\sigma_i(G+\delta G)}{\sigma_i(G)} = 1 + \frac{\eta}{\sigma_{\min}(B)},$$

and so at least one term $\sigma_i(G+\delta G)/\sigma_i(G)$ exceeds $(1 + \eta/\sigma_{\min}(B))^{1/n}$. To see that small componentwise relative perturbations are not sufficient, consider the matrix

$$G = B = \begin{bmatrix} 1 & 1 \\ -\epsilon & \epsilon \end{bmatrix}$$

with $\epsilon \ll 1$. The condition number of B is approximately $1/\epsilon$, and relative perturbations of size η in its entries cannot change its singular values by more than a factor of about $(1 \pm \eta)^2$. □

As in Proposition 2.11, our lower bound on the attainable perturbations in the singular vectors requires a dense δB and does not grow with $\kappa(B)$.

PROPOSITION 2.22. *Let $G = BD$, σ_i, u_i, v_i; $\delta G = \delta B D$; and η be as in Proposition 2.15. Let u_i' and v_i' be the unit left and right singular vectors of $G + \delta G$, respectively. Then we can choose δB, $\|\delta B\|_2 \equiv \eta \ll \sigma_{\min}(B)$, so that*

$$\max(\|u_i - u_i'\|_2, \|v_i - v_i'\|_2) \geq \frac{\eta}{2^{3/2}\sigma_{\max}(B)\mathrm{relgap}_{\sigma i}} + O(\eta^2).$$

3. Two-sided Jacobi. In this section, we prove that two-sided Jacobi in floating point arithmetic applied to a positive definite symmetric matrix computes the eigenvalues and eigenvectors with the error bounds of §2.

In this introduction, we present the algorithm and our model of floating point arithmetic. In §3.1, we derive error bounds for the computed eigenvalues. In §3.2, we derive error bounds for the computed eigenvectors.

Let $H_0 = D_0 A_0 D_0$ be the initial matrix, and $H_m = D_m A_m D_m$, where H_m is obtained from H_{m-1} by applying a single Jacobi rotation. Here D_m is diagonal and A_m has unit diagonal as before. All the error bounds in this section contain the factor $\max_m \kappa(A_m)$, whereas the perturbation bounds of §2 are proportional to $\kappa(A_0)$. Therefore, our claim that Jacobi solves the eigenproblem as accurately as predicted in §2 depends on the ratio $\max_m \kappa(A_m)/\kappa(A_0)$ being modest in size. Note that convergence of H_m to diagonal form is equivalent to the convergence of A_m to the identity, or $\kappa(A_m)$ to 1. Thus we expect $\kappa(A_m)$ to be less than $\kappa(A_0)$ eventually.

We have overwhelming numerical evidence that $\max_m \kappa(A_m)/\kappa(A_0)$ is modest in size; in §7, the largest value this ratio attained in random testing was 1.82. Our theoretical understanding of why this ratio is so small is somewhat weaker; we present our theoretical bounds on this ratio in §6.

The essential difference between our algorithm and standard two-sided Jacobi is the stopping criterion. As stated in the introduction (and justified by Theorem 2.3), we set H_{ij} to zero only if $H_{ij}/(H_{ii}H_{jj})^{1/2}$ is small. Otherwise, our algorithm is a simplification of the standard one introduced by Rutishauser [16]. We have chosen a simple version of the algorithm, omitting enhancements such as delayed updates of the diagonals and fast rotations, to make the error analysis clearer (an error analysis of these enhancements is future work).

ALGORITHM 3.1 (Two-sided Jacobi for the symmetric positive definite eigenproblem). tol is a user-defined stopping criterion. The matrix V whose columns are the computed eigenvectors initially contains the identity.

repeat
  for all pairs $i < j$
    /* compute the Jacobi rotation which diagonalizes
    $$\begin{bmatrix} H_{ii} & H_{ij} \\ H_{ji} & H_{jj} \end{bmatrix} \equiv \begin{bmatrix} a & c \\ c & b \end{bmatrix}$$ */
    $\zeta = (b-a)/(2c)$; $t = \text{sign}(\zeta)/(|\zeta| + \sqrt{1+\zeta^2})$
    $cs = 1/\sqrt{1+t^2}$; $sn = cs * t$
    /* update the 2×2 submatrix */
    $H_{ii} = a - c * t$
    $H_{jj} = b + c * t$
    $H_{ij} = H_{ji} = 0$
    /* update the rest of rows and columns i and j */
    for $k = 1$ to n except i and j
      $tmp = H_{ik}$
      $H_{ik} = cs * tmp - sn * H_{jk}$
      $H_{jk} = sn * tmp + cs * H_{jk}$
      $H_{ki} = H_{ik}$; $H_{kj} = H_{jk}$
    endfor
    /* update the eigenvector matrix V */
    for $k = 1$ to n
      $tmp = V_{ki}$
      $V_{ki} = cs * tmp - sn * V_{kj}$
      $V_{kj} = sn * tmp + cs * V_{kj}$
    endfor
  endfor
until convergence (all $|H_{ij}|/(H_{ii}H_{jj})^{1/2} \leq$ tol)

Our model of arithmetic is a variation on the standard one: the floating point result $fl(\cdot)$ of the operation $(\cdot)$ is given by

$$\begin{aligned} fl(a \pm b) &= a(1+\varepsilon_1) \pm b(1+\varepsilon_2), \\ fl(a \times b) &= (a \times b)(1+\varepsilon_3), \\ fl(a/b) &= (a/b)(1+\varepsilon_4), \\ fl(\sqrt{a}) &= \sqrt{a}(1+\varepsilon_5), \end{aligned} \tag{3.1}$$

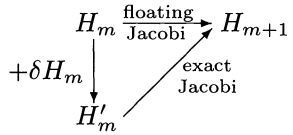
where $|\varepsilon_i| \leq \varepsilon$ and $\varepsilon \ll 1$ is the machine precision. This is somewhat more general

than the usual model, which uses $fl(a \pm b) = (a \pm b)(1 + \varepsilon_1)$, and includes machines like the Cray, which do not have a guard digit. This does not greatly complicate the error analysis, but it is possible that the computed rotation angle may be off by a factor of 2, whereas with a guard digit the rotation angle is always highly accurate. This may adversely affect convergence, but as we see it does not affect the one-step error analysis.

Numerically subscripted ε's denote independent quantities bounded in magnitude by ε. As usual, we make approximations like $(1 + i\varepsilon_1)(1 + j\varepsilon_2) = 1 + (i + j)\varepsilon_3$ and $(1 + i\varepsilon_1)/(1 + j\varepsilon_2) = 1 + (i + j)\varepsilon_3$.

3.1. Error bounds for eigenvalues computed by two-sided Jacobi. The next theorem and its corollary justify our accuracy claims for eigenvalues computed by two-sided Jacobi.

THEOREM 3.1. *Let H_m be the sequence of matrices generated by Algorithm* 3.1 *in finite precision arithmetic with precision ε; that is, H_{m+1} is obtained from H_m by applying a single Jacobi rotation. Then the following diagram:*



commutes in the following sense: The top arrow indicates that H_{m+1} is obtained from H_m by applying one Jacobi rotation in floating point arithmetic. The diagonal arrow indicates that H_{m+1} is obtained from H'_m by applying one Jacobi rotation in exact arithmetic; thus H_{m+1} and H'_m are exactly similar. The vertical arrow indicates that $H'_m = H_m + \delta H_m$. δH_m is bounded as follows. Write $\delta H_m = D_m \delta A_m D_m$. Then

$$\|\delta A_m\|_2 \leq (182(2n-4)^{1/2} + 104)\varepsilon. \tag{3.2}$$

In other words, if $\|\delta A_m\|_2 < \lambda_{\min}(A_m)$, one step of Jacobi satisfies the assumptions needed for the error bounds of §2.

COROLLARY 3.2. *Assume that Algorithm* 3.1 *converges, and that H_M is the final matrix whose diagonal entries we take as the eigenvalues. Write $H_m = D_m A_m D_m$ with D_m diagonal and A_m with ones on the diagonal for $0 \leq m \leq M$. Let λ_j be the jth eigenvalue of H_0 and λ'_j be the jth diagonal entry of H_M. Then to first order in ε, the following error bound holds:*

$$\frac{|\lambda_j - \lambda'_j|}{\lambda_j} \leq (\varepsilon \cdot M \cdot (182(2n-4)^{1/2} + 104) + n \cdot \text{tol}) \cdot \max_{0 \leq m \leq M} \kappa(A_m). \tag{3.3}$$

Remark. In numerical experiments presented in §7, there was no evidence that the actual error bounded in (3.3) grew with increasing n or M.

Proof of Corollary 3.2. Bound (3.3) follows by substituting the bound (3.2) and the stopping criterion into Theorem 2.3. □

Remark. A similar bound can be obtained based on the error bound in Proposition 2.4.

Proof of Theorem 3.1. The proof of the commuting diagram is a tedious computation. Write the 2×2 submatrix of H_{mm} being reduced as

$$\begin{bmatrix} a & c \\ c & b \end{bmatrix} \equiv \begin{bmatrix} d_i^2 & z d_i d_j \\ z d_i d_j & d_j^2 \end{bmatrix},$$

where we assume without loss of generality that $a \geq b$ and $c > 0$. By positive definiteness, $0 < z \leq \bar{z} \equiv (\kappa(A_m) - 1)/(\kappa(A_m) + 1) < 1$. Let a' and b' be the new values of H_{ii} and H_{jj} computed by the algorithm, respectively. Let $x \equiv d_j/d_i \leq 1$. We consider two cases: $x \leq \bar{x} \equiv (\sqrt{5} - 1)/2 \approx .62$, and $x > \bar{x}$.

First consider $x \leq \bar{x}$. Systematic application of formulas (3.1) shows that

$$\begin{aligned}\zeta &= fl((b-a)/(2*c)) \\ &= (1+\varepsilon_4)(((1+\varepsilon_1)b - (1+\varepsilon_2)a)/((1+\varepsilon_3)2c)) \\ &= \frac{(1+\varepsilon_4)(1+\varepsilon_2)}{1+\varepsilon_3}\left(\frac{\tilde{b}-a}{2c}\right),\end{aligned}$$

where $\tilde{b} \equiv (1+\varepsilon_1)b/(1+\varepsilon_2) \equiv (1+\varepsilon_b)b$, $|\varepsilon_b| \leq 2\varepsilon$. Thus $\zeta = (1+\varepsilon_\zeta)(\tilde{b}-a)/(2c)$ where $|\varepsilon_\zeta| \leq 3\varepsilon$.

Let $t(c)$ denote the true value of t (i.e., without rounding error) as a function of a, $\tilde{b}$, and c. Using (3.1) again, we can show $t = (1+\varepsilon_t)t(c)$ where $|\varepsilon_t| \leq 7\varepsilon$.

Next,

$$\begin{aligned}b' &= fl(b+ct) = (1+\varepsilon_5)b + (1+\varepsilon_6)(1+\varepsilon_7)ct \\ &= \frac{(1+\varepsilon_2)(1+\varepsilon_5)}{1+\varepsilon_1}\left(\tilde{b} + \frac{(1+\varepsilon_1)(1+\varepsilon_6)(1+\varepsilon_7)(1+\varepsilon_t)}{(1+\varepsilon_2)(1+\varepsilon_5)}ct(c)\right) \\ &\equiv (1+\varepsilon_{b'})(\tilde{b} + (1+\varepsilon_{ct(c)})ct(c)),\end{aligned} \tag{3.4}$$

where $|\varepsilon_{ct(c)}| \leq 12\varepsilon$ and $|\varepsilon_{b'}| \leq 3\varepsilon$. Since $|t(c)|$ is an increasing function of c, we can write $(1+\varepsilon_{ct(c)})ct(c) = (1+\varepsilon_c)c \cdot t((1+\varepsilon_c)c)$ for some ε_c where $|\varepsilon_c| \leq |\varepsilon_{ct(c)}| \leq 12\varepsilon$.

Now we can define $\tilde{c} \equiv (1+\varepsilon_c)c$, and $\tilde{\zeta}$, $\tilde{t}$, $\tilde{cs}$, and $\tilde{sn}$ as the true values of the untilded quantities computed without rounding error starting from a, $\tilde{b}$, and $\tilde{c}$. $\tilde{cs}$ and $\tilde{sn}$ define the exact Jacobi rotation

$$J_m \equiv \begin{bmatrix} \tilde{cs} & \tilde{sn} \\ -\tilde{sn} & \tilde{cs} \end{bmatrix},$$

which transforms H'_m into H_{m+1} in the commutative diagram in the statement of the theorem: $J_m^T H'_m J_m = H_{m+1}$.

Now we begin constructing δH_m. δH_m is nonzero only in rows and columns i and j. First, we compute its entries outside the 2×2 (i,j) submatrix. Using (3.1) we can show $cs = (1+\varepsilon_{cs})\tilde{cs}$ and $sn = (1+\varepsilon_{sn})\tilde{sn}$, where $|\varepsilon_{cs}| \leq 22\varepsilon$ and $|\varepsilon_{sn}| \leq 30\varepsilon$. Now let H'_{ik} and H'_{jk} denote the updated quantities computed by the algorithm. Then

$$\begin{aligned}H'_{ik} &= fl(cs * H_{ik} - sn * H_{jk}) \\ &= (1+\varepsilon_{10})(1+\varepsilon_8)csH_{ik} - (1+\varepsilon_9)(1+\varepsilon_{11})snH_{jk} \\ &= (1+\varepsilon_{10})(1+\varepsilon_8)(1+\varepsilon_{cs})\tilde{cs}H_{ik} - (1+\varepsilon_9)(1+\varepsilon_{11})(1+\varepsilon_{sn})\tilde{sn}H_{jk} \\ &\equiv \tilde{cs}H_{ik} - \tilde{sn}H_{jk} + \epsilon(H'_{ik}).\end{aligned} \tag{3.5}$$

Similarly,

$$\begin{aligned}H'_{jk} &= fl(sn * H_{ik} + cs * H_{jk}) \\ &= (1+\varepsilon_{14})(1+\varepsilon_{12})(1+\varepsilon_{sn})\tilde{sn}H_{ik} + (1+\varepsilon_{13})(1+\varepsilon_{15})(1+\varepsilon_{cs})\tilde{cs}H_{jk} \\ &\equiv \tilde{sn}H_{ik} + \tilde{cs}H_{jk} + \epsilon(H'_{jk}).\end{aligned} \tag{3.6}$$

Now $x = d_j/d_i$ implies

$$\bar{\zeta} = \frac{b-a}{2\tilde{c}} = \frac{d_j^2 - d_i^2}{2\tilde{z}d_i d_j} = \frac{x^2-1}{2\tilde{z}x},$$

where $\tilde{z} \equiv z(1+\varepsilon_c)$. Then $x \le \bar{x}$ implies

$$|\tilde{t}| = \frac{1}{\frac{1-x^2}{2\tilde{z}x} + \left(1 + \left(\frac{1-x^2}{2\tilde{z}x}\right)^2\right)^{1/2}} \le \frac{\tilde{z}x}{1-\bar{x}^2}.$$

Also, $|\tilde{sn}| \le |\tilde{t}|$, so this last expression is an upper bound on $|\tilde{sn}|$ as well. Substituting this bound on $\tilde{sn}$, $\tilde{cs} \le 1$, $|H_{ik}| \le d_i d_k \bar{z}$, and $|H_{jk}| \le d_j d_k \bar{z}$ into (3.5) and (3.6) yields

$$|\epsilon(H'_{ik})| \le 56\varepsilon d_i d_k \bar{z},$$
$$|\epsilon(H'_{jk})| \le 56\varepsilon d_j d_k \bar{z}/(1-\bar{x}^2).$$

Thus

$$\begin{bmatrix} H'_{ik} \\ H'_{jk} \end{bmatrix} = J_m^T \cdot \begin{bmatrix} H_{ik} \\ H_{jk} \end{bmatrix} + \begin{bmatrix} \epsilon(H_{ik}) \\ \epsilon(H_{jk}) \end{bmatrix}$$
$$= J_m^T \cdot \left(\begin{bmatrix} H_{ik} \\ H_{jk} \end{bmatrix} + J_m \cdot \begin{bmatrix} \epsilon(H_{ik}) \\ \epsilon(H_{jk}) \end{bmatrix} \right) \equiv J_m^T \cdot \left(\begin{bmatrix} H_{ik} \\ H_{jk} \end{bmatrix} + \begin{bmatrix} \delta H_{ik} \\ \delta H_{jk} \end{bmatrix} \right),$$

where $|\delta H_{ik}| \le 112\varepsilon d_i d_k \bar{z}/(1-\bar{x}^2)$ and $|\delta H_{jk}| \le 112\varepsilon d_j d_k \bar{z}/(1-\bar{x}^2)$.

Now we construct the 2×2 submatrix Δ of δH_m at the intersection of rows and columns i and j. We construct it of three components: $\Delta = \Delta_1 + \Delta_2 + \Delta_3$.

Consider the formula $a' = fl(a - c * t)$ for the i,i entry of H_{m+1}. Applying (3.1) systematically, we see that

$$\begin{aligned} a' &= (1+\varepsilon_{18})a - (1+\varepsilon_{17})(1+\varepsilon_{16})ct \\ &= (1+\varepsilon_{18})a - (1+\varepsilon_{17})(1+\varepsilon_{16})(1+\varepsilon_t)ct(c) \\ &= (1+\varepsilon_{18})a - \frac{(1+\varepsilon_{17})(1+\varepsilon_{16})(1+\varepsilon_t)\tilde{c}t(\tilde{c})}{1+\varepsilon_{ct(c)}} \\ &\equiv (1+\varepsilon_{18})a - (1+\varepsilon'_{ct(c)})\tilde{c}t(\tilde{c}), \end{aligned}$$

where $|\varepsilon'_{ct(c)}| \le 21\varepsilon$. Since $a > 0$ and $\tilde{c}t(\tilde{c}) < 0$, we get

$$a' = \left(1 + \frac{\varepsilon_{18}a - \varepsilon'_{ct(c)}\tilde{c}t(\tilde{c})}{a - \tilde{c}t(\tilde{c})}\right)(a - \tilde{c}t(\tilde{c})) \equiv (1+\varepsilon_{a'})(a - \tilde{c}t(\tilde{c})),$$

where $|\varepsilon_{a'}| \le 21\varepsilon$.

Now let

$$\Delta_1 = \begin{bmatrix} 0 & \varepsilon_c c \\ \varepsilon_c c & \varepsilon_b b \end{bmatrix} = \begin{bmatrix} 0 & \tilde{c} - c \\ \tilde{c} - c & \tilde{b} - b \end{bmatrix}.$$

From earlier discussion we see that

$$J_m^T \left(\begin{bmatrix} a & c \\ c & b \end{bmatrix} + \Delta_1 \right) J_m = \begin{bmatrix} a - \tilde{c}t(\tilde{c}) & 0 \\ 0 & \tilde{b} + \tilde{c}t(\tilde{c}) \end{bmatrix}.$$

Next let

$$\Delta_2 = \varepsilon_{a'}\left(\begin{bmatrix} a & c \\ c & b \end{bmatrix} + \Delta_1\right).$$

Thus

$$J_m^T\left(\begin{bmatrix} a & c \\ c & b \end{bmatrix} + \Delta_1 + \Delta_2\right) J_m = (1+\varepsilon_{a'})\begin{bmatrix} a - \tilde{c}t(\tilde{c}) & 0 \\ 0 & \tilde{b} + \tilde{c}t(\tilde{c}) \end{bmatrix}$$
$$= \begin{bmatrix} a' & 0 \\ 0 & b'((1+\varepsilon_{a'})/(1+\varepsilon_b)) \end{bmatrix}.$$

Finally, let

$$\Delta_3 = J_m \begin{bmatrix} 0 & 0 \\ 0 & b'(1-((1+\varepsilon_{a'})/(1+\varepsilon_b))) \end{bmatrix} J_m^T \equiv \begin{bmatrix} \tilde{s}\tilde{n}^2\varepsilon_{b''}b & \tilde{c}\tilde{s}\tilde{n}\varepsilon_{b''}b \\ \tilde{c}\tilde{s}\tilde{n}\varepsilon_{b''}b & \tilde{c}\tilde{s}^2\varepsilon_{b''}b \end{bmatrix},$$

where $|\varepsilon_{b''}| \le |\varepsilon_{a'}| + |\varepsilon_b| \le 24\varepsilon$. Then

$$J_m^T\left(\begin{bmatrix} a & c \\ c & b \end{bmatrix} + \Delta_1 + \Delta_2 + \Delta_3\right) J_m = \begin{bmatrix} a' & 0 \\ 0 & b' \end{bmatrix},$$

as desired. This completes the construction of δH_m. We may bound

$$\|\delta A_m\|_2 \le \left(\frac{112(2n-4)^{1/2}\bar{z}}{1-\bar{x}^2} + 104\right)\varepsilon. \tag{3.7}$$

Now we consider the second case, when $x > \bar{x}$. The only thing that changes in the previous analysis is our analysis of δH_{ik} and δH_{jk}, since $\tilde{s}\tilde{n}$ is no longer small. Instead we substitute the bounds $|\tilde{s}\tilde{n}| \le 1$, $|\tilde{c}\tilde{s}| \le 1$, $|H_{ik}| \le d_i d_k \bar{z} \le d_j d_k \bar{z}/\bar{x}$, and $|H_{jk}| \le d_j d_k \bar{z}$ into (3.5) and (3.6) to get

$$|\epsilon(H'_{ik})| \le 56\varepsilon d_i d_k \bar{z} \quad \text{and} \quad |\epsilon(H'_{jk})| \le 56\varepsilon d_i d_k \bar{z},$$

whence

$$|\delta H_{ik}| \le 112\varepsilon d_i d_k \bar{z} \quad \text{and} \quad |\delta H_{jk}| \le 112\varepsilon d_j d_k \bar{z}/\bar{x},$$

and so

$$\|\delta A_m\|_2 \le \left(\frac{112(2n-4)^{1/2}\bar{z}}{\bar{x}} + 104\right)\varepsilon. \tag{3.8}$$

Finally, we note that our choice of $\bar{x}$ makes the upper bounds in (3.7) and (3.8) both equal, with $1/(1-\bar{x}^2) = 1/\bar{x} < 1.62$, proving the theorem. □

Remark. The quantity $182(2n-4)^{1/2}$ in the theorem may be multiplied by $\max_{m,i\neq j}|A_{m,ij}|$, which is less than one. Thus if the A_m are strongly diagonally dominant, the part of the error term that depends on n is suppressed.

Commutative diagrams like the one in the theorem, where performing one step of the algorithm in floating point arithmetic is equivalent to making small relative errors in the matrix and then performing the algorithm exactly, occur elsewhere in numerical analysis. For example, such a diagram describes an entire sweep of the zero-shift bidiagonal QR algorithm [8], and is the key to the high accuracy achieved by that algorithm.

3.2. Error bounds for eigenvectors computed by two-sided Jacobi. The next two theorems justify our accuracy claims for eigenvectors computed by two-sided Jacobi.

THEOREM 3.3. *Let $V = [v_1, \cdots, v_n]$ be the matrix of unit eigenvectors computed by Algorithm 3.1 in finite precision arithmetic with precision ε. Let $U = [u_1, \cdots, u_n]$ be the true eigenvector matrix. Let $\bar{\kappa} \equiv \max_m \kappa(A_m)$ be the largest $\kappa(A_m)$ of any iterate. Then the error in the computed eigenvectors is bounded in norm by*

$$\|v_i - u_i\|_2 \leq \frac{(n-1)^{1/2}(n \cdot \mathrm{tol} + M \cdot (182(2n-4)^{1/2} + 104)\varepsilon)\bar{\kappa}}{\mathrm{relgap}_{\lambda i}} + 46M\varepsilon. \tag{3.9}$$

Proof. Let $H_0, \cdots, H_M$ be the sequence of matrices generated by the Jacobi algorithm, where H_M satisfies the stopping criterion. Let J_m be the exact Jacobi rotation which transforms H'_m to H_{m+1} in the commuting diagram of Theorem 3.1: $J_m^T H'_m J_m = H_{m+1}$.

We use the approximation that $\mathrm{relgap}_{\lambda i}$ is the same for all H_m, even though it changes slightly. This contributes an $O(\varepsilon^2)$ term to the overall bound (which we ignore), but could be accounted for by using the bounds of Theorem 3.1.

Initially, we compute error bounds for the columns of $J_0 \cdots J_{M-1}$, ignoring any rounding errors occurring in computing their product. Then we incorporate these rounding errors.

We prove by induction that the ith column v_{mi} of $V_m \equiv J_m \cdots J_{M-1}$ is a good approximation to the true ith eigenvector u_{mi} of H_m. In particular, we show that to first order in ε,

$$\|u_i - v_{0i}\|_2 \leq \frac{(n-1)^{1/2}(n \cdot \mathrm{tol} + M \cdot (182(2n-4)^{1/2} + 104)\varepsilon)\bar{\kappa}}{\mathrm{relgap}_{\lambda i}}.$$

The basis of the induction is as follows. $V_M = I$ is the eigenvector matrix for H_M, which is considered diagonal since it satisfies the stopping criterion. Thus the norm error in v_{Mi} follows from plugging the stopping criterion into Theorem 2.5:

$$\|u_{Mi} - v_{Mi}\|_2 \leq \frac{(n-1)^{1/2} \cdot n \cdot \mathrm{tol} \cdot \bar{\kappa}}{\mathrm{relgap}_{\lambda i}}.$$

For the induction step we assume that

$$\|u_{m+1,i} - v_{m+1,i}\|_2 \leq \frac{(n-1)^{1/2}(n \cdot \mathrm{tol} + (M-m-1) \cdot (182(2n-4)^{1/2} + 104)\varepsilon)\bar{\kappa}}{\mathrm{relgap}_{\lambda i}},$$

and try to extend it to m. Consider the commuting diagram of Theorem 3.1. Accordingly, the errors in $V_m = J_m V_{m+1}$ considered as eigenvectors of H'_m are the errors in V_{m+1} premultiplied by J_m. This does not increase them in the two-norm, since J_m is orthogonal. Now we change H'_m to H_m. This increases the norm error in v_{mi} by an amount bounded by plugging the bound for $\|\delta A_m\|_2$ into Theorem 2.5: $(n-1)^{1/2}\bar{\kappa}(182(2n-4)^{1/2} + 104)\varepsilon/\mathrm{relgap}_{\lambda i}$. This proves the induction step.

Finally, consider the errors from accumulating the product of slightly wrong values of J_m in floating point arithmetic. From the proof of Theorem 3.1, we see that the relative errors in the entries of J_m are at most 30ε, and from the usual error analysis of a product of 2×2 rotations, we get $32\sqrt{2}M\varepsilon < 46M\varepsilon$ for the norm error in the product of M rotations. This completes the proof of bound (3.9). □

Now we consider the errors in the individual components of the computed eigenvectors $|u_i(j) - v_i(j)|$. From Proposition 2.9, we see that we can hope to bound this quantity by $O(\varepsilon)\bar{\kappa}\bar{v}_i(j)/\min(\text{relgap}_{\lambda i}, 2^{-1/2})$, where

$$\bar{v}_i(j) \equiv \bar{\kappa}^{3/2} \min\left(\left(\frac{\lambda_i}{\lambda_j}\right)^{1/2} \left(\frac{\lambda_j}{\lambda_i}\right)^{1/2} \right) \tag{3.10}$$

is a modified upper bound for the eigenvector component $v_i(j)$ as in Proposition 2.8. In other words, we may have high relative accuracy even in the tiny components of the computed eigenvectors; this is the case in the example in the introduction and at the end of §2.1.

We use $\bar{v}_i(j)$ as defined in (3.10) for each H_m, even though the values of λ_i and λ_j vary slightly from step to step. This error contributes an $O(\varepsilon^2)$ term to the overall bound (we are ignoring such terms), but could be incorporated using the bounds of Corollary 3.2.

THEOREM 3.4. *Let V, U, and $\bar{\kappa}$ be as in Theorem* 3.3, *and $\bar{v}_i(j)$ be as in* (3.10). *Then we can bound the error in the individual eigencomponents by*

$$|u_i(j) - v_i(j)| \le p(M,n) \cdot \frac{(\text{tol} + \varepsilon) \cdot \bar{\kappa} \cdot \bar{v}_i(j)}{\min(\text{relgap}_{\lambda i}, 2^{-1/2})}. \tag{3.11}$$

Here $p(M,n)$ is a "pivot growth" factor, which is given in (3.21).

Proof. The proof is similar to that of Theorem 3.3. One difference is that we use Proposition 2.9 instead of Theorem 2.5 to bound the errors in the eigenvectors. Another difference, which introduces the growth factor $p(M,n)$, is that we need to use the scaling of the entries of J_m to see how small eigenvector components have small errors; not being able to use the orthogonality of J_m introduces $p(M,n)$. We can only prove an exponentially growing bound for $p(M,n)$, although we believe it to be much smaller.

As in the proof of Theorem 3.3, let $V_m = J_m \cdots J_{M-1}$, where $J_m^T H'_m J_m = H_{m+1}$. Set $V_M = I$. The proof has three parts. In the first part, we show that the ith column of V_0 is a good approximation to the eigenvectors of H_0 in the sense of the theorem. In the second part, we show that the (i,j) entry of $J_0 \cdots J_m$ is bounded by a modest multiple of $\bar{v}_i(j)$. In the third part, we show that the rounding errors committed in computing $J_0 \cdots J_m$ in floating point are small compared to $\bar{v}_i(j)$.

For the first part of the proof we use induction to prove that the ith column v_{mi} of V_m is a good approximation to the true ith eigenvector u_{mi} of H_m. This shows that

$$|u_i(j) - v_{0i}(j)| \le \rho_0 \frac{(\text{tol} + \varepsilon) \cdot \bar{\kappa} \cdot \bar{v}_i(j)}{\min(\text{relgap}_{\lambda i}, 2^{-1/2})}, \tag{3.12}$$

where ρ_0 is a constant (part of the "pivot growth" factor) we need to estimate. The base of the induction follows from plugging the stopping criterion into the bound of Proposition 2.9, yielding

$$|u_{Mi}(j) - v_{Mi}(j)| \le \frac{(2n-2)^{1/2} \cdot n \cdot \text{tol} \cdot \bar{\kappa} \cdot \bar{v}_i(j)}{\min(\text{relgap}_{\lambda i}, 2^{-1/2})} \le \rho_M \frac{(\text{tol} + \varepsilon) \cdot \bar{\kappa} \cdot \bar{v}_i(j)}{\min(\text{relgap}_{\lambda i}, 2^{-1/2})},$$

where $\rho_M \equiv n(2n-2)^{1/2}$. The induction step assumes that

$$|u_{m+1,i}(j) - v_{m+1,i}(j)| \le \rho_{m+1} \frac{(\text{tol} + \varepsilon) \cdot \bar{\kappa} \cdot \bar{v}_i(j)}{\min(\text{relgap}_{\lambda i}, 2^{-1/2})},$$

which we try to extend to m. Consider the commuting diagram of Theorem 3.1. Accordingly, the errors in the columns of $V_m = J_m V_{m+1}$ considered as eigenvectors of H'_m are just the errors in V_{m+1} premultiplied by J_m; let e_{mi} denote this error for the ith column of V_m. Suppose J_m rotates in rows and columns k and l; then e_{mi} is identical to $u_{m+1,i} - v_{m+1,i}$ except for $e_{mi}(k)$ and $e_{mi}(l)$. We may assume without loss of generality that $k < l$ and $d_k \geq d_l$ (d_k^2 and d_l^2 are the diagonal entries of H_m). As in the proof of Theorem 3.1, there are two cases: $x \equiv d_l/d_k \leq \bar{x} \equiv (\sqrt{5}-1)/2$, and $x > \bar{x}$.

In the first case, $x \leq \bar{x}$, we know (as in the proof of Theorem 3.1) that $\tilde{sn}$, the sine in the rotation J_m, is bounded in magnitude by $x/(1-\bar{x}^2)$. Write $|\tilde{sn}| \leq c_m(\lambda_l/\lambda_k)^{1/2}$ instead, where c_m is a modest constant. We can do this because $d_r \approx \lambda_r^{1/2}$ from Proposition 2.7. This lets us bound

$$
\begin{aligned}
\begin{bmatrix} |e_{mi}(k)| \\ |e_{mi}(l)| \end{bmatrix} &= \left| J_m \begin{bmatrix} u_{m+1,i}(k) - v_{m+1,i}(k) \\ u_{m+1,i}(l) - v_{m+1,i}(l) \end{bmatrix} \right| \\
&\leq \begin{bmatrix} |u_{m+1,i}(k) - v_{m+1,i}(k)| + |\tilde{sn}(u_{m+1,i}(l) - v_{m+1,i}(l))| \\ |\tilde{sn}(u_{m+1,i}(k) - v_{m+1,i}(k))| + |u_{m+1,i}(l) - v_{m+1,i}(l)| \end{bmatrix} \\
&\leq \frac{\rho_{m+1}(\mathrm{tol}+\varepsilon)\bar{\kappa}^{5/2}}{\min(\mathrm{relgap}_{\lambda i}, 2^{-1/2})} \\
&\times \begin{bmatrix} \min\left(\left(\frac{\lambda_i}{\lambda_k}\right)^{1/2}, \left(\frac{\lambda_k}{\lambda_i}\right)^{1/2}\right) + c_m\left(\frac{\lambda_l}{\lambda_k}\right)^{1/2} \min\left(\left(\frac{\lambda_i}{\lambda_l}\right)^{1/2}, \left(\frac{\lambda_l}{\lambda_i}\right)^{1/2}\right) \\ c_m\left(\frac{\lambda_l}{\lambda_k}\right)^{1/2} \min\left(\left(\frac{\lambda_i}{\lambda_k}\right)^{1/2}, \left(\frac{\lambda_k}{\lambda_i}\right)^{1/2}\right) + \min\left(\left(\frac{\lambda_i}{\lambda_l}\right)^{1/2}, \left(\frac{\lambda_l}{\lambda_i}\right)^{1/2}\right) \end{bmatrix} \\
&\leq \frac{\rho_{m+1}(\mathrm{tol}+\varepsilon)\bar{\kappa}^{5/2}}{\min(\mathrm{relgap}_{\lambda i}, 2^{-1/2})} \cdot \begin{bmatrix} (1+c_m)\min\left(\left(\frac{\lambda_i}{\lambda_k}\right)^{1/2}, \left(\frac{\lambda_k}{\lambda_i}\right)^{1/2}\right) \\ (1+c_m)\min\left(\left(\frac{\lambda_i}{\lambda_l}\right)^{1/2}, \left(\frac{\lambda_l}{\lambda_i}\right)^{1/2}\right) \end{bmatrix} \\
&= (1+c_m)\frac{\rho_{m+1}(\mathrm{tol}+\varepsilon)\bar{\kappa}}{\min(\mathrm{relgap}_{\lambda i}, 2^{-1/2})} \cdot \begin{bmatrix} \bar{v}_i(k) \\ \bar{v}_i(l) \end{bmatrix}. \qquad (3.13)
\end{aligned}
$$

Now consider case 2, $x > \bar{x}$. Now λ_k and λ_l are reasonably close together. Thus we may bound $|\tilde{sn}|$ simply by 1 in the derivation (3.13). This leads to the same bound with a possibly different c_m; we take the final c_m as the maximum of these two values. This bounds the error in the columns of V_m considered as eigenvectors of H'_m.

Now we change H'_m to H_m. This increases the bound for $|u_{mi}(j) - v_{mi}(j)|$ by an amount bounded by plugging the bound for $\|\delta A_m\|_2$ from Theorem 3.1 into Proposition 2.9: $(2n-2)^{1/2}(182(2n-4)^{1/2}+104)\cdot\varepsilon\cdot\bar{\kappa}\cdot\bar{v}_i(j)/\min(\mathrm{relgap}_{\lambda i}, 2^{-1/2})$. This completes the induction with

(3.14)

$$
\begin{aligned}
&|u_{mi}(j) - v_{mi}(j)| \\
&\quad \leq ((1+c_m)\rho_{m+1} + (2n-2)^{1/2}(182(2n-4)^{1/2}+104)) \cdot \frac{(\mathrm{tol}+\varepsilon)\cdot\bar{\kappa}\cdot\bar{v}_i(j)}{\min(\mathrm{relgap}_{\lambda i}, 2^{-1/2})} \\
&\quad \equiv \rho_m \cdot \frac{(\mathrm{tol}+\varepsilon)\cdot\bar{\kappa}\cdot\bar{v}_i(j)}{\min(\mathrm{relgap}_{\lambda i}, 2^{-1/2})}.
\end{aligned}
$$

Here

(3.15)
$$\rho_m = (1+c_m)\rho_{m+1} + (2n-2)^{1/2}(182(2n-4)^{1/2}+104) \ , \quad \rho_M = n(2n-2)^{1/2}.$$

ρ_m satisfies an exponential error bound, but it is clear from the derivation that linear growth is far more likely than exponential growth. This completes the first part of the proof.

In the second part of the proof we show that the (i,j) entry of $\tilde{V}_m \equiv J_0 \cdots J_m$ is bounded by a modest multiple of $\bar{v}_i(j)$. To do this we prove by induction that

(3.16)
$$|\tilde{v}_{mi}(j)| \le \tau_m \bar{v}_i(j),$$

where $\tilde{V}_m = [\tilde{v}_{m1}, \cdots, \tilde{v}_{mn}]$ and τ_m is a constant (part of the "pivot growth" factor) we need to estimate. The base of the induction is for $m=-1$, i.e., the null product, which we set equal to the identity matrix. This clearly satisfies (3.16) with $\tau_{-1}=1$. Now we assume that (3.16) is true for $m-1$ and try to extend it to m. Suppose J_m rotates in rows and columns k and l. Postmultiplying $\tilde{V}_{m-1}$ by J_m only changes it in columns k and l. Assume as before that $k<l$ and $x = d_l/d_k \le 1$. There are two cases, as before: $x \le \bar{x}$ and $x > \bar{x}$.

First, consider the case $x \le \bar{x}$. We may bound the (j,k) and (j,l) entries of $\tilde{V}_m$ as follows:

(3.17)
$$\begin{aligned}
|[\tilde{v}_{m,j}(k), \quad \tilde{v}_{m,j}(l)]| &= \left| [\tilde{v}_{m-1,j}(k), \tilde{v}_{m-1,j}(l)] \cdot \begin{bmatrix} \tilde{cs} & \tilde{sn} \\ -\tilde{sn} & \tilde{cs} \end{bmatrix} \right| \\
&\le \tau_{m-1}\bar{\kappa}^{3/2} \left[\min\left(\left(\frac{\lambda_j}{\lambda_k}\right)^{1/2}, \left(\frac{\lambda_k}{\lambda_j}\right)^{1/2} \right), \min\left(\left(\frac{\lambda_j}{\lambda_l}\right)^{1/2}, \left(\frac{\lambda_l}{\lambda_j}\right)^{1/2} \right) \right] \\
&\quad \times \begin{bmatrix} 1 & c_m\left(\frac{\lambda_l}{\lambda_k}\right)^{1/2} \\ c_m\left(\frac{\lambda_l}{\lambda_k}\right)^{1/2} & 1 \end{bmatrix} \\
&\le \tau_{m-1}\bar{\kappa}^{3/2}(1+c_m) \\
&\quad \times \left[\min\left(\left(\frac{\lambda_j}{\lambda_k}\right)^{1/2}, \left(\frac{\lambda_k}{\lambda_j}\right)^{1/2} \right), \min\left(\left(\frac{\lambda_j}{\lambda_l}\right)^{1/2}, \left(\frac{\lambda_l}{\lambda_j}\right)^{1/2} \right) \right] \\
&= \tau_{m-1}(1+c_m)[\bar{v}_k(j), \bar{v}_l(j)] \\
&\equiv \tau_m [\bar{v}_k(j), \bar{v}_l(j)].
\end{aligned}$$

In the second case, $x > \bar{x}$, we get a similar bound. Here $\lambda_k \approx \lambda_l$, and we can simply bound $|\tilde{sn}| \le 1$. This yields a slightly different c_m; for the final c_m, we again take the maximum of the two. This ends the second part of the proof with

(3.18)
$$\tau_m = (1+c_m)\tau_{m-1}, \qquad \tau_{-1} = 1.$$

Even though this only yields an exponential upper bound for τ_M, it is clear from the derivation that linear growth is far more likely than exponential growth.

In the third and final part of the proof, we show that the rounding errors in the (i,j) entry of the computed approximation to $\tilde{V}_{m-1}$ is bounded by $O(\varepsilon)\bar{v}_i(j)$. Let $\tilde{J}_m$ be the actual rotation which only approximates J_m. From the proof of Theorem 3.1, we have that $cs = \tilde{cs}(1+\varepsilon_{cs})$ with $|\varepsilon_{cs}| \le 22\varepsilon$, and $sn = \tilde{sn}(1+\varepsilon_{sn})$ with $|\varepsilon_{sn}| \le 30\varepsilon$.

Let $\tilde{\tilde{V}}_m = fl(\tilde{\tilde{V}}_{m-1} * \tilde{J}_m)$ be the actually computed eigenvector matrix after the mth Jacobi rotation. The final computed eigenvector matrix is $V = \tilde{\tilde{V}}_{M-1}$. We use induction to prove that

$$|\tilde{\tilde{v}}_{m,i}(j) - \tilde{v}_{m,i}(j)| \le \chi_m \varepsilon \bar{v}_i(j), \tag{3.19}$$

where $\tilde{\tilde{V}}_m = [\tilde{\tilde{v}}_{m1}, \cdots, \tilde{\tilde{v}}_{mn}]$ and χ_m is a constant (part of the "pivot growth" factor) we need to estimate. The basis is again for $m = -1$ when $\tilde{\tilde{V}}_{-1} = \tilde{V}_{-1} = I$ and $\chi_{-1} = 0$. Now we assume that (3.19) is true for $m-1$ and try to extend it to m. As before, we assume that J_m rotates in rows and columns k and l with $k < l$ and $x = d_k/d_l \le 1$. Write $\tilde{e}_{mi} \equiv \tilde{\tilde{v}}_{mi} - \tilde{v}_{mi}$. The (j,k) and (j,l) entries of $\tilde{\tilde{V}}_m$ are

$$\begin{aligned}
[\tilde{\tilde{v}}_{m,j}(k), \tilde{\tilde{v}}_{m,j}(l)] = [&\tilde{\tilde{v}}_{m-1,j}(k)\tilde{cs}(1+\varepsilon_{cs})(1+\varepsilon_1)(1+\varepsilon_2) \\
&- \tilde{\tilde{v}}_{m-1,j}(l)\tilde{sn}(1+\varepsilon_{sn})(1+\varepsilon_3)(1+\varepsilon_4), \\
&\tilde{\tilde{v}}_{m-1,j}(k)\tilde{sn}(1+\varepsilon_{sn})(1+\varepsilon_5)(1+\varepsilon_6) \\
&+ \tilde{\tilde{v}}_{m-1,j}(l)\tilde{cs}(1+\varepsilon_{cs})(1+\varepsilon_7)(1+\varepsilon_8)] \\
= [&\tilde{v}_{m-1,j}(k)\tilde{cs} - \tilde{v}_{m-1,j}(l)\tilde{sn}, \tilde{v}_{m-1,j}(k)\tilde{sn} + \tilde{v}_{m-1,j}(l)\tilde{cs}] \\
&+ [24\varepsilon_9 \tilde{cs}\tilde{v}_{m-1,j}(k) + 32\varepsilon_{10}\tilde{sn}\tilde{v}_{m-1,j}(l), \\
&\quad 32\varepsilon_{11}\tilde{sn}\tilde{v}_{m-1,j}(k) + 24\varepsilon_{12}\tilde{cs}\tilde{v}_{m-1,j}(l)] \\
&+ [(1+24\varepsilon_9)\tilde{cs}\tilde{e}_{m-1,j}(k) + (1+32\varepsilon_{10})\tilde{sn}\tilde{e}_{m-1,j}(l), \\
&\quad (1+32\varepsilon_{11})\tilde{sn}\tilde{e}_{m-1,j}(k) + (1+24\varepsilon_{12})\tilde{cs}\tilde{e}_{m-1,j}(l)] \\
= [&\tilde{v}_{m,j}(k), \tilde{v}_{m,j}(l)] + I_1 + I_2,
\end{aligned}$$

so that $[\tilde{e}_{m,j}(k), \tilde{e}_{m,j}(l)] = I_1 + I_2$.

As before, there are two cases: $x \le \bar{x}$ and $x > \bar{x}$. Consider the case $x \le \bar{x}$. Using $|\tilde{sn}| \le c_m(\lambda_l/\lambda_k)^{1/2}$, $|\tilde{cs}| \le 1$, and $|\tilde{v}_{m-1,i}(j)| \le \tau_{m-1}\bar{v}_i(j)$, we get

$$\begin{aligned}
|I_1| &\le \varepsilon\tau_{m-1}\bar{\kappa}^{3/2}(24+32c_m) \\
&\quad \times \left[\min\left(\left(\frac{\lambda_j}{\lambda_k}\right)^{1/2}, \left(\frac{\lambda_k}{\lambda_j}\right)^{1/2}\right), \min\left(\left(\frac{\lambda_j}{\lambda_l}\right)^{1/2}, \left(\frac{\lambda_l}{\lambda_j}\right)^{1/2}\right)\right] \\
&= \varepsilon\tau_{m-1}(24+32c_m)[\bar{v}_k(j), \bar{v}_k(l)],
\end{aligned}$$

and

$$|I_2| \le \chi_{m-1}(1+c_m)\varepsilon[\bar{v}_k(j), \bar{v}_k(l)].$$

Taken together, we get

$$\chi_m = (1+c_m)\chi_{m-1} + \varepsilon\tau_{m-1}(24+32c_m), \qquad \chi_{-1} = 0. \tag{3.20}$$

In the second case, $x > \bar{x}$, we get a similar bound with a possibly different c_m. Again, we take the maximum of the two. This completes the third part of the proof.

Finally, combining (3.19) and (3.12) we get

$$|v_i(j) - u_i(j)| \le (\rho_0 + \chi_{M-1})\frac{(\mathrm{tol}+\varepsilon)\cdot\bar{\kappa}\cdot\bar{v}_i(j)}{\min(\mathrm{relgap}_{\lambda_i}, 2^{-1/2})}, \tag{3.21}$$

proving the theorem with $p(M,n) = \rho_0 + \chi_{M-1}$. □

4. One-sided Jacobi. In this section we prove that one-sided Jacobi in floating point arithmetic applied on the right of a general matrix computes the singular values and singular vectors with the error bounds of §2. Here we present our algorithm; the model of arithmetic was presented in §3. In §4.1 we derive error bounds for the computed singular values. In §4.2 we derive error bounds for the computed singular vectors. In §4.3, we present two algorithms for the symmetric positive definite eigenproblem H, which do either left-handed or right-handed Jacobi on the Cholesky factor L of H. The second of these algorithms cannot compute eigenvectors quite as accurately as the first, but it may be much faster than either the first algorithm or two-sided Jacobi.

Let $G_0 = B_0 D_0$ be the initial matrix, and $G_m = B_m D_m$, where G_m is obtained from G_{m-1} by applying a single Jacobi rotation. Here D_m is diagonal and B_m has columns of unit two-norm. All the error bounds in this section contain the factor $\max_m \kappa(B_m)$, whereas the perturbation bounds in §2 are proportional to $\kappa(B_0)$. Therefore, as in §3, our claim that Jacobi computes the singular value decomposition (SVD) as accurately as predicted in §2 depends on the ratio $\max_m \kappa(B_m)/\kappa(B_0)$ being modest. In exact arithmetic, right-handed Jacobi on $G = BD$ is identical to two-sided Jacobi on $H = G^T G = DB^T BD = DAD$, so the question of the growth of $\kappa(B_m) = \kappa(A_m)^{1/2}$ is essentially identical to the question of the growth of $\kappa(A_m)$ in the case of two-sided Jacobi.

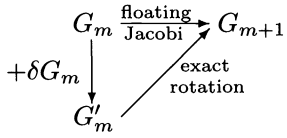
ALGORITHM 4.1 (Right-handed Jacobi for the singular value problem). tol is a user-defined stopping criterion. The matrix V whose columns are the computed right singular vectors initially contains the identity.

```
repeat
    for all pairs i < j
        /* compute [ a  c ; c  b ] ≡ the (i,j) submatrix of G^T G */
        a = Σ_{k=1}^{n} G_ki^2
        b = Σ_{k=1}^{n} G_kj^2
        c = Σ_{k=1}^{n} G_ki * G_kj
        /* compute the Jacobi rotation which diagonalizes [ a  c ; c  b ] */
        ζ = (b − a)/(2c); t = sign(ζ)/(|ζ| + sqrt(1 + ζ^2))
        cs = 1/sqrt(1 + t^2); sn = cs * t
        /* update columns i and j of G */
        for k = 1 to n
            tmp = G_ki
            G_ki = cs * tmp − sn * G_kj
            G_kj = sn * tmp + cs * G_kj
        endfor
        /* update the matrix V of right singular vectors */
        for k = 1 to n
            tmp = V_ki
            V_ki = cs * tmp − sn * V_kj
            V_kj = sn * tmp + cs * V_kj
        endfor
    endfor
until convergence (all |c|/sqrt(ab) ≤ tol)
/* the computed singular values are the norms of the columns of the final G */
```

/* the computed left singular vectors are the normalized columns of the final G */

4.1. Error bounds for singular values computed by right-handed Jacobi. The next theorem and its corollary justify our accuracy claims for singular values computed by right-handed Jacobi. The proofs are analogous to those in §3; details may be found in [10].

THEOREM 4.1. *Let G_m be the sequence of matrices generated by the right-handed Jacobi algorithm in finite precision arithmetic with precision ε; that is G_{m+1} is obtained from G_m by applying a single Jacobi rotation. Then the following diagram:*



commutes in the same way as in Theorem 3.1. *The diagonal arrow indicates that G_{m+1} is obtained from G'_m by applying one Givens (not necessarily Jacobi) rotation in exact arithmetic. δG_m is bounded as follows. Write $\delta G_m = \delta B_m D_m$, where D_m is diagonal such that B_m in $G_m = B_m D_m$ has unit columns. Then*

$$\|\delta B_m\|_2 \leq 72\varepsilon. \tag{4.1}$$

In other words, one step of Jacobi satisfies the assumptions needed for the error bounds of §2.

COROLLARY 4.2. *Assume that Algorithm* 4.1 *converges, and that G_M is the final matrix which satisfies the stopping criterion. For $0 \leq m \leq M$, write $G_m = B_m D_m$ with D_m diagonal and B_m with unit columns. Let σ_j be the jth singular value of G_0 and σ'_j the jth computed singular value. Then to first order in ε the following error bound holds:*

$$\frac{|\sigma_j - \sigma'_j|}{\sigma_j} \leq (72\varepsilon \cdot M + n^2\varepsilon + n \cdot \mathrm{tol}) \cdot \max_{0 \leq k \leq M} \kappa(B_k) + n\varepsilon. \tag{4.2}$$

Remark. A similar bound can be obtained based on the error bound in Proposition 2.15.

4.2. Error bounds for singular vectors computed by right-handed Jacobi. The next two theorems justify our accuracy claims for singular vectors computed by right-handed Jacobi.

THEOREM 4.3. *Let $V = [v_1, \cdots, v_n]$ be the matrix of unit right singular vectors and $U = [u_1, \cdots, u_n]$ be the matrix of unit left singular vectors computed by Algorithm* 4.1 *in finite precision arithmetic with precision ε. Let $V_T = [v_{T1}, \cdots, v_{Tn}]$ and $U_T = [u_{T1}, \cdots, u_{Tn}]$ be the matrices of true unit right and left singular vectors, respectively. Let $\bar{\kappa} \equiv \max_m \kappa(B_m)$ be the largest $\kappa(B_m)$ of any iterate. Then the error in the computed singular vectors is bounded in norm by*

$$\begin{aligned} &\max(\|u_{Ti} - u_i\|_2, \|v_{Ti} - v_i\|_2) \\ &\qquad \leq \frac{(n - .5)^{1/2} \cdot \bar{\kappa} \cdot (72M \cdot \varepsilon + n \cdot \mathrm{tol} + n^2 \cdot \varepsilon)}{\mathrm{relgap}_{\sigma i}} + (9M + n + 1)\varepsilon. \end{aligned} \tag{4.3}$$

Then consider the errors in the individual components of the computed right singular vectors $|v_{Ti}(j) - v_i(j)|$. From Proposition 2.20, we see that we can hope to bound this quantity by $O(\varepsilon)\bar{\kappa}^2\bar{\bar{v}}_i(j)/\text{relgap}_{\sigma i}$, where

$$\bar{\bar{v}}_i(j) \equiv \bar{\kappa}^3 \min\left(\frac{\sigma_i}{\sigma_j}, \frac{\sigma_j}{\sigma_i}\right). \tag{4.4}$$

We use $\bar{\bar{v}}_i(j)$ as defined in (4.4) for each G_m, even though the values of σ_i and σ_j vary slightly from step to step. This error contributes an $O(\varepsilon^2)$ term to the overall bound (which we ignore) but could be incorporated using the bounds of Corollary 4.2.

THEOREM 4.4. *Let V, V_T, and $\bar{\kappa}$ be as in Theorem* 4.3, *and $\bar{\bar{v}}_i(j)$ be as in* (4.4). *Then we can bound the error in the individual components of v_i by*

$$|v_{Ti}(j) - v_i(j)| \leq q(M,n) \cdot \frac{(\text{tol} + \varepsilon) \cdot \bar{\kappa}^2 \cdot \bar{\bar{v}}_i(j)}{\text{relgap}_{\sigma i}}, \tag{4.5}$$

where $q(M,n)$ has a bound similar to that of $p(M,n)$ in Theorem 3.4.

4.3. Using Cholesky followed by one-sided Jacobi for the symmetric positive definite eigenproblem. In this subsection we consider two algorithms for the symmetric positive definite eigenproblem H, both based on performing Cholesky on H, and using one-sided Jacobi to compute the SVD of the Cholesky factor L. The first algorithm (Algorithm 4.2) does left-handed Jacobi on L, returning its left singular vectors as the eigenvectors of H and the squares of its singular values as the eigenvalues of H. The second algorithm (Algorithm 4.4), originally proposed in [22], does Cholesky with complete pivoting (which is equivalent to diagonal pivoting) and then right-handed Jacobi on L, again returning its left singular vectors and squares of its singular values.

The first algorithm, left-handed Jacobi, is about as accurate as two-sided Jacobi, but permits purely column oriented access to the data following the initial Cholesky decomposition; this can have speed advantages on machines with memory hierarchies. The second algorithm, right-handed Jacobi with pivoting, is less accurate than the first because it will not always compute tiny eigenvector components with the accuracy of Theorem 3.4, although it does compute the eigenvalues as accurately, and the eigenvectors with the same norm error bound. However, it can be several times faster than either the first algorithm or two-sided Jacobi.

ALGORITHM 4.2 (Left-handed Jacobi on L without pivoting for the symmetric positive definite eigenproblem H).

1. Form the Cholesky factor L of H: $H = LL^T$.
2. Compute the singular values σ_i and left singular vectors v_i of L using left-handed Jacobi.
3. The eigenvalues λ_i of H are $\lambda_i = \sigma_s^2$. The eigenvectors of H are v_i.

We show that this method is as accurate as using two-sided Jacobi directly on H. The proof involves a new error analysis of Cholesky decomposition, so we begin by restating Cholesky's algorithm in order to establish notation for our error analysis.

ALGORITHM 4.3 (Cholesky decomposition $H = LL^T$ for an $n \times n$ symmetric positive definite matrix H).

for $i = 1$ to n

$L_{ii} = (H_{ii} - \sum_{k=1}^{i-1} L_{ik}^2)^{1/2}$
for $j = i+1$ to n
$L_{ji} = (H_{ji} - \sum_{k=1}^{i-1} L_{jk}L_{ik})/L_{ii}$
endfor
endfor

LEMMA 4.5 (see [10]). *Let L be the Cholesky factor of H computed using Algorithm 4.3 in finite precision arithmetic with precision ε. Then $LL^T = H + E$ where $|E_{ij}| \le (n+5)\varepsilon(H_{ii}H_{jj})^{1/2}$.*

THEOREM 4.6. *Let L be the Cholesky factor of $H = DAD$ computed in floating point arithmetic using Algorithm 4.3. Let σ_i and v_{Li} be the exact singular values and right singular vectors of L^T, and λ_i and v_{Hi} be the eigenvalues and eigenvectors of H. Let $\bar{v}_i(j)$ be as in Proposition 2.8. Then*

$$\frac{|\lambda_i - \sigma_i^2|}{\lambda_i} \le (n^2+5n)\cdot\varepsilon\cdot\kappa(A),$$

$$\|v_{Li} - v_{Hi}\|_2 \le \frac{(n^2+5n)(n-1)^{1/2}\cdot\varepsilon\cdot\kappa(A)}{\text{relgap}_{\lambda i}} + O(\varepsilon^2),$$

$$|v_{Li}(j) - v_{Hi}(j)| \le \frac{(n^2+5n)(2n-2)^{1/2}\cdot\varepsilon\cdot\kappa(A)\cdot\bar{v}_i(j)}{\min(\text{relgap}_{\lambda i}, 2^{-1/2})} + O(\varepsilon^2).$$

Proof. Plug the bound of Lemma 4.5 into Theorem 2.3, Theorem 2.5, and Proposition 2.9. □

Theorem 4.6 implies that the errors introduced by Cholesky are as small as those introduced by two-sided Jacobi. Write $H = DAD$ and $L_A = D^{-1}L$. Since $\|A - L_A L_A^T\|_2 \le (n^2+5n)\varepsilon$, $\kappa(A) \approx (\kappa(L_A))^2$ (unless both are very large). Since the columns of L_A^T have nearly unit two-norm, the accuracy of left-handed Jacobi applied to L is governed by $\kappa(L_A)$. Thus Cholesky followed by left-handed Jacobi on L results in a problem whose condition number $\kappa(L_A)$ is approximately the square root of the condition number of the original problem $\kappa(A)$. Corollary 4.2 and Theorems 4.3 and 4.4 guarantee that the computed eigenvalues and eigenvectors are accurate. In exact arithmetic, left-handed Jacobi on L is the same as two-sided Jacobi on $DAD = H = LL^T = D(L_A L_A^T)D$, so the question of how much $\kappa(L_A)$ can grow during subsequent Jacobi rotations is essentially identical to the question of the growth of $\kappa(A_m)$ during two-sided Jacobi. The second algorithm follows.

ALGORITHM 4.4 (Right-handed Jacobi on L with pivoting for the symmetric positive definite eigenproblem H).

1. Form the Cholesky factor L of H using complete pivoting. Then there is a permutation matrix P such that $P^THP = LL^T$.
2. Compute the singular values σ_i and left singular vectors v_i of L using right-handed Jacobi.
3. The eigenvalues λ_i of H are $\lambda_i = \sigma_s^2$. The eigenvectors of H are Pv_i.

Even if we did not do complete pivoting, Theorem 4.6 would guarantee that the squares of the true singular values of L would be accurate eigenvalues of H, and that the true left singular vectors of L would be accurate eigenvectors of P^THP. Since we are computing left singular vectors of L, Theorem 4.4 does not apply, but from Corollary 4.2, we know that the computed eigenvalues are accurate, and from Theorem 4.3 we know that the computed eigenvectors are accurate in a norm sense.

Numerical experiments in §7 bear out the fact that tiny eigenvector components may not always be computed as accurately by Algorithm 4.4 as by Algorithm 4.2.

Note that Algorithm 4.4 is mathematically equivalent to doing two-sided Jacobi on L^TL, so we in effect take a single step of the symmetric LR algorithm [22] before beginning Jacobi, thus giving Jacobi a "head start." (An analogous head start is attained by preceding left-handed Jacobi for the SVD with a QR decomposition with column pivoting [12].) Writing $L^TL = H_1 = D_1A_1D_1$, we see it is $\kappa(A_1)$, which governs the accuracy of step 2 of the algorithm, as well as its speed, since it is $\kappa(A_1)$ that must be driven to 1. We discuss this in more detail in §6, where we show that $\kappa(A_1)$ can be much smaller than $\kappa(A)$, where $H = DAD$ is the original problem.

There are two other algorithmic variations one might consider: Cholesky with pivoting followed by left-handed Jacobi on L, and Cholesky without pivoting followed by right-handed Jacobi on L. We can measure the quality of both algorithms as we did in the last paragraph: We find symmetric positive definite matrices H_2 and H_3 such that the algorithms are mathematically equivalent to doing two-sided Jacobi on H_2 and H_3, respectively. Then their accuracies and running times depend on $\kappa(A_2)$ and $\kappa(A_3)$ where $H_i = D_iA_iD_i$. One may show that $\kappa(A_2) = \kappa(A)$, so there is no advantage to pivoting and left-handed Jacobi, and also that $\kappa(A_3)$ can greatly exceed $\kappa(A)$, so that right-handed Jacobi without pivoting can be harmful. We do not consider these algorithmic variations further.

5. Bisection and inverse iteration. Here we show that bisection and inverse iteration applied to the symmetric positive definite matrix $H = DAD$ can compute the eigenvalues and eigenvectors within the accuracy bounds section of 2. Let inertia(H) denote the triple (n, z, p) of the number n of negative eigenvalues of H, the number z of zero eigenvalues of H, and the number p of positive eigenvalues of H. These results are simple extensions of Algorithms 3 and 5 in [2], and detailed proofs may be found there and in [10].

ALGORITHM 5.1 (Stably computing the inertia of $H - xI = DAD - xI$).

1. Permute the rows and columns of $A - xD^{-2}$ (which has the same inertia as $H - xI$) and partition it as

$$\begin{bmatrix} A_{11} - xD_1^{-2} & A_{12} \\ A_{21} & A_{22} - xD_2^{-2} \end{bmatrix}$$

so that if $1 - xd^{-2}$ is a diagonal entry of $A_{11} - xD_1^{-2}$, then $xd^{-2} \geq 2n + 1$, where n is the dimension of H.

2. Compute $X = A_{22} - xD_2^{-2} - A_{21}(A_{11} - xD_1^{-2})^{-1}A_{12}$, using Cholesky to compute $(A_{11} - xD_1^{-2})^{-1}A_{12}$.

3. Compute inertia(X) = (neg, zero, pos) using a stable pivoting scheme such as in [4].

4. The inertia of $H - xI$ is $(\text{neg} + \dim(A_{11}), \text{zero}, \text{pos})$.

We need to partition $A - xD^{-2}$ as above in order to make the proof convenient, but it may not be necessary algorithmically [10].

THEOREM 5.1. *Let ε be the machine precision in which Algorithm* 5.1 *is carried out, where we assume that neither overflow nor underflow occur. Then Algorithm* 5.1 *computes the exact inertia of $D(A + \delta A)D - xI$, where $\|\delta A\|_2 = O(\varepsilon)$. Thus Algorithm* 5.1 *can be used in a bisection algorithm to find all the eigenvalues of H to the accuracy of Theorem* 2.3 *or Proposition* 2.4.

ALGORITHM 5.2 (Inverse iteration for computing the eigenvector x of a symmetric positive definite matrix $H = DAD$ corresponding to eigenvalue z). tol is a user-specified stopping criterion.

1. We assume that eigenvalue z has been computed accurately, for example, using Algorithm 5.1.
2. Choose a starting vector y_0; set $i = 0$.
3. Compute the symmetric indefinite factorization LDL^T of $P(A - zD^{-2})P^T$ [4], where P is the same permutation as in Algorithm 5.1, step 1.
4. Repeat
   $i = i + 1$
   Solve $(A - zD^{-2})\tilde{y}_i = y_{i-1}$ for $\tilde{y}_i$ using the LDL^T factorization of step 3.
   $r = 1/\|\tilde{y}_i\|_2$
   $y_i = r \cdot \tilde{y}_i$
   until $(r \leq \text{tol})$
5. $x = D^{-1}y_i$

THEOREM 5.2. *Suppose that Algorithm* 5.2 *terminates with* x *as the computed eigenvector of* $H = DAD$. *Then there is a diagonal matrix* $\hat{D}$ *with* $\hat{D}_{ii} = 1 + O(\text{tol})$ *and a matrix* δA *with* $\|\delta A\|_2 = O(\text{tol})$, *such that* $\hat{D}x$ *is the exact eigenvector of* $D(A + \delta A)D$. *Thus the error in* x *is bounded by Theorem* 2.5, *Corollary* 2.6, *and Proposition* 2.9.

6. Upper bounds for $\max_m \kappa(A_m)/\kappa(A_0)$. As stated in §§3 and 4, our claims about the accuracy to which Jacobi can solve the eigenproblem depend on the ratio $\max_m \kappa(A_m)/\kappa(A_0)$ being modest. Here $H_0 = D_0A_0D_0$ is the initial matrix, and $H_m = D_mA_mD_m$ is the sequence produced by Jacobi (H_{m+1} is obtained from H_m by applying a single Jacobi rotation, D_m is diagonal, and A_m has ones on the diagonal). The reason is that the error bounds for Jacobi are proportional to $\max_m \kappa(A_m)$, and the error bounds of §2 are proportional to $\kappa(A_0)$.

In this section, we present several results explaining why $\max_m \kappa(A_m)/\kappa(A_0)$ should not be expected to grow very much. Recall that convergence of H_m to diagonal form is equivalent to the convergence of A_m to the identity matrix, or of $\kappa(A_m)$ to 1. Thus we expect $\kappa(A_m) < \kappa(A_0)$ eventually. The best situation would be monotonic convergence, but this is, unfortunately, not always the case.

We have not been able to completely explain the extremely good numerical results of §7, that $\max_m \kappa(A_m)/\kappa(A_0)$ never exceeded 1.82, and averaged 1.20 in random experiments. (Wang [23] has found a sequence H_n of matrices of dimension n where this ratio grows slowly with n, reaching 8 for $n = 50$. Changing the sweep strategy eliminated this growth.) A complete theoretical explanation of this remains an open question.

We only speak in terms of two-sided Jacobi in this section. This is no loss of generality because, in exact arithmetic, right-handed Jacobi on G is equivalent to two-sided Jacobi on G^TG.

Our first result shows that $\kappa(A_m)/\kappa(A_0)$ cannot be too large if A_m is obtained from A_0 by a sequence of Jacobi rotations in pairwise disjoint rows and columns. The second result gives a cheaply computable guaranteed upper bound on $\max_m \kappa(A_m)/\kappa(A_0)$ in terms of the Hadamard measure of A_0. This bound is generally quite pessimistic unless the dimension of A is modest and $\kappa(A_0)$ is small—at most a few hundred. The third and fourth results will be for right-handed Jacobi with pivoting (Algorithm 4.4). The third result shows that the wider the range of numbers

on the diagonal of H, the smaller $\kappa(A_1)$ is for that algorithm. This in turn makes it converge faster. The fourth, rather surprising, result is that $\kappa(A_1)$ is bounded by a constant, depending *only* on the dimension n, not on A_0. These last two results lead us to recommend right-handed Jacobi with pivoting Jacobi as the algorithm of choice (unless it is important to get small eigenvector components to high accuracy; see the discussion in §4.3).

PROPOSITION 6.1. *Let H_0 be $n \times n$. Let H_m be obtained from H_0 by applying m Jacobi rotations in pairwise nonoverlapping rows and columns (this means $m \leq n/2$). Write $H_m = D_m A_m D_m$ as before. Then*

$$\frac{\kappa(A_m)}{\kappa(A_0)} \leq \frac{1 + \max_{1\leq k\leq m} |A_{0,2k-1,2k}|}{1 - \max_{1\leq k\leq m} |A_{0,2k-1,2k}|} \leq \min(\kappa(A_0), 2n). \tag{6.1}$$

Also,

$$\frac{\kappa(A_{i+1})}{\kappa(A_i)} \leq \min(\kappa(A_0), 8). \tag{6.2}$$

Furthermore, the spectrum of A_m is independent of D_0, even though the entries of A_m depend on D_0. More precisely, the spectrum of A_m coincides with the spectrum of the pencil $A_0 - \lambda A_0'$, where A_0' coincides with A_0 on every rotated element and is the identity otherwise.

Proof. We begin by deriving a matrix pencil depending only on A_0 whose eigenvalues are the same as A_m. This proves that the eigenvalues of A_m depend only on A_0. We assume without loss of generality that the m Jacobi rotations are in rows and columns (1,2), (3,4), $\cdots$, $(2m-1, 2m)$. This lets us write $J^T H_0 J = H_m$, where J is block diagonal with the 2×2 Jacobi rotations (and possibly ones) on its diagonal. Rewrite this as

$$A_m = (D_m^{-1} J^T D_0) A_0 (D_0 J D_m^{-1}) \equiv Z^T A_0 Z,$$

where Z has the same block diagonal structure as J. Let A_0' be a block diagonal matrix with the same block structure as Z and J, where A_0' is identical to A_0 within its 2×2 blocks, and has ones on its diagonal when J does. Since $H_{m,12} = H_{m,34} = \cdots = 0$, also $A_{m,12} = A_{m,23} = \cdots = 0$. Thus A_m has 2×2 identity matrices on its diagonal matching the block structure of Z, J, and A_0'. Thus $A_m = Z^T A_0 Z$ implies $Z^{-T} Z^{-1} = A_0'$. Therefore, the eigenvalues of $A_m = Z^T A_0 Z$ are identical to those of the pencil $A_0 - \lambda Z^{-T} Z^{-1} = A_0 - \lambda A_0'$.

Now we apply the minimax theorem to bound $\lambda_{\min}(A_m)$ below by

$$\lambda_{\min}(A_m) = \min_{x\neq 0} \frac{x^T A_0 x}{x^T A_0' x} \geq \frac{\min_{\|x\|=1} x^T A_0 x}{\max_{\|x\|=1} x^T A_0' x} = \frac{\lambda_{\min}(A_0)}{1 + \max_{1\leq k\leq m} |A_{0,2k-1,2k}|}. \tag{6.3}$$

We may bound $1+\max_{1\leq k\leq m} |A_{0,2k-1,2k}|$ from above by both $\lambda_{\max}(A_0)$ and 2, yielding

$$\lambda_{\min}(A_m) \geq \frac{\lambda_{\min}(A_0)}{\min(2, \lambda_{\max}(A_0))}. \tag{6.4}$$

Now we bound $\lambda_{\max}(A_m)$ from above. First, by the minimax theorem we may write

$$\lambda_{\max}(A_m) = \max_{x\neq 0} \frac{x^T A_0 x}{x^T A_0' x} \leq \frac{\max_{\|x\|=1} x^T A_0 x}{\min_{\|x\|=1} x^T A_0' x} \leq \frac{\lambda_{\max}(A_0)}{1 - \max_{1\leq k\leq m} |A_{0,2k-1,2k}|},$$

which, when combined with (6.4), yields

$$\kappa(A_m) \leq (\kappa(A_0))^2,$$

proving half of (6.1). For the other half, note that $1 \leq \lambda_{\max}(A_i) \leq n$ for all i, so that $\lambda_{\max}(A_m)/\lambda_{\max}(A_0) \leq n$. Now combine this with (6.4).

Now we show $\lambda_{\max}(A_{i+1}) \leq 4\lambda_{\max}(A_i)$, which, when combined with (6.4), yields (6.2). It suffices to show $\lambda_{\max}(A_1) \leq 4\lambda_{\max}(A_0)$. Write

$$A_0 = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix},$$

where A_{11} is 2×2. Then by the minimax theorem, there exists a conformally partitioned unit vector $x^T = [x_1^T, x_2^T]$ where

$$\lambda_{\max}(A_1) = \frac{x_1^T A_{11} x_1 + 2x_1^T A_{12} x_2 + x_2^T A_{22} x_2}{x_1^T A_{11} x_1 + x_2^T x_2}.$$

Write $x_1^T x_1 = \zeta$ $(0 \leq \zeta \leq 1)$, $x_2^T x_2 = 1 - \zeta$, $x_1^T A_{11} x_1 = \tau_1 \zeta$, and $x_2^T A_{22} x_2 = \tau_2(1-\zeta)$, so that

$$\lambda_{\max}(A_1) = \frac{\tau_1\zeta + 2x_1^T A_{12} x_2 + \tau_2(1-\zeta)}{\tau_1\zeta + 1 - \zeta} \leq 2\frac{\tau_1\zeta + \tau_2(1-\zeta)}{\tau_1\zeta + 1 - \zeta}.$$

The maximum of this last expression over all $0 \leq \zeta \leq 1$ is

$$2 + 2\tau_2 \leq 2 + 2\lambda_{\max}(A_{22}) \leq 4\lambda_{\max}(A_0). \qquad \square$$

Our second bound is based on the Hadamard measure of a symmetric positive definite matrix H:

$$\mathcal{H}(H) \equiv \frac{\det(H)}{\prod_i H_{ii}}.$$

PROPOSITION 6.2. *The Hadamard measure $\mathcal{H}(H)$ has the following properties:*

1. *$\mathcal{H}(H) \leq 1$ and $\mathcal{H}(H) = 1$ if and only if H is diagonal.*
2. *$\mathcal{H}(H) = \mathcal{H}(\tilde{D}H\tilde{D})$ for any nonsingular diagonal $\tilde{D}$.*
3. *Let $H = DAD$ with D diagonal and A with unit diagonal. Then*

$$\lambda_{\min}(A) \geq \frac{\mathcal{H}(H)}{e} = \frac{\det(A)}{e},$$

where $e = \exp(1)$.

4. *Let H' be obtained from H by applying a Jacobi rotation (in exact arithmetic) in rows and columns i and j. Then*

$$\mathcal{H}(H') = \frac{\mathcal{H}(H)}{1 - A_{ij}^2} \geq \mathcal{H}(H).$$

5. *Let $H_0, \cdots, H_m, \cdots$ be a sequence of symmetric positive definite matrices obtained from Jacobi's method in exact arithmetic. Let $H_m = D_m A_m D_m$ with D_m diagonal and A_m with unit diagonal. Then*

$$\max_m \kappa(A_m) \leq \frac{n \cdot e}{\det(A_0)} = \frac{n \cdot e}{\mathcal{H}(H_0)}.$$

Proof. 1. Write the Cholesky decomposition $H = LL^T$. Then

$$H_{11} \cdots H_{nn} = \prod_{i=1}^{n} \left(\sum_{k=1}^{i} L_{ik}^2 \right) \geq \prod_{i=1}^{n} L_{ii}^2 = \det(H).$$

2. $\det(\tilde{D}^2)$ factors out of the numerator and denominator of $\mathcal{H}(\tilde{D}H\tilde{D})$.

3. From part 2 above $\mathcal{H}(H) = \mathcal{H}(A) = \det(A)$, so it suffices to show $\lambda_{\min}(A) \geq \det(A)/e$. Let $0 < \lambda_1 \leq \cdots \leq \lambda_n$ be the eigenvalues of A. Since $\lambda_1 = \det(A)/\prod_{i=2}^{n} \lambda_i$, we need to show $\prod_{i=2}^{n} \lambda_i \leq e$. Now $\sum_{i=2}^{n} \lambda_i \leq \operatorname{tr}(A) = n$. Since $ab \geq (a+x)(b-x)$ for all $a \geq b \geq x \geq 0$, we see that $\prod_{i=2}^{n} \lambda_i$ is greatest when all $\lambda_i = n/(n-1)$, in which case $\prod_{i=2}^{n} \lambda_i = ((n-1)/n)^{n-1} \leq e$.

4. From Proposition 6.1 we have

$$\mathcal{H}(H') = \det(AA'^{-1}) = \det(A)/(1 - A_{ij}^2) = \mathcal{H}(H)/(1 - A_{ij}^2),$$

where $A' = I$ except for $A'_{ij} = A'_{ji} = A_{ij}$.

5. This is directly implied by parts 3 and 4. □

Thus part 5 of this proposition gives us a guaranteed upper bound on $\max_m \kappa(A_m)$ at a cost of about $n^3/6$ flops, compared to $2n^3$ flops per Jacobi sweep ($4n^3$ if accumulating eigenvectors). If we use the algorithm in §4.3, where we must do Cholesky anyway, this upper bound comes nearly for free.

Basically, this upper bound is only useful as long as $\kappa(A_0)$ is quite small and A_0 has low dimension; otherwise, it is much too large to be useful.

Our third and fourth bounds are for right-handed Jacobi with pivoting (Algorithm 4.4). Recall that this algorithm begins by doing Cholesky with complete pivoting on H_0 to get $PH_0P^T = LL^T$, where P is a permutation matrix. Then it does right-handed Jacobi on L, which is equivalent (in exact arithmetic) to two-sided Jacobi on L^TL. Therefore, Algorithm 4.4 essentially starts with $L^TL = H_1 = D_1A_1D_1$.

Our third result, which we state rather informally, is that the larger the range of numbers on the diagonal D^2 of H, the smaller is $\kappa(A_1)$ (this effect was also observed in [22]). We argue as follows. Let $L = DL_A$ be the factor obtained from complete pivoting. Here, L_A has rows of unit two-norm. Since Algorithm 4.4 does right-handed Jacobi on L, its performance depends on the condition number of DL_AD', where D' is chosen diagonal to make the columns of DL_AD' unit vectors. From van der Sluis's theorem [21], we know the condition number of DL_AD' can be at most n times DL_AD^{-1}, so it suffices to examine $\kappa(DL_AD^{-1})$. The effect of complete pivoting is essentially to reorder D so that $D_{ii} \geq D_{i+1,i+1}$, and to keep $L_{A,ii}$ as large as possible. Now $(DL_AD^{-1})_{ii} = L_{A,ii}$ is unchanged, and the subdiagonal entry $(DL_AD^{-1})_{ij} = L_{A,ij}D_{ii}D_{jj}^{-1}$ is multiplied by the factor $D_{ii}D_{jj}^{-1}$, which is between 0 and 1. The more D_{jj} exceeds D_{ii}, the smaller this factor, and the more nearly diagonal DL_AD^{-1} becomes. Since complete pivoting tries to keep the diagonal of L_A large, this improves the condition number.

Our fourth result shows that, surprisingly, $\max_{m \geq 1} \kappa(A_m)$ is bounded independent of H_0.

PROPOSITION 6.3. *Let $PH_0P^T = LL^T$ be the Cholesky decomposition of the $n \times n$ matrix H_0 obtained with complete pivoting. Let $H_1 = L^TL = D_1A_1D_1$. Let $H_m = D_mA_mD_m$, $m > 1$, be obtained from two-sided Jacobi applied to H_1. Then*

1. $\mathcal{H}(H_1) \geq \mathcal{H}(H_0)$.
2. $\mathcal{H}(H_1) \geq 1/n!$. *This bound is attainable.*
3. $\max_{m \geq 1} \kappa(A_m) \leq e \cdot n \,/\mathcal{H}(H_1) \leq e \cdot n \cdot n!$.

Proof. 1. Since $\det(H_1) = \det(H_0)$, it suffices to show that $\prod_i H_{0,ii} \geq \prod_i H_{1,ii}$. Assume without loss of generality that $P = I$. Then $H_{0,ii} = \sum_{k=1}^{i} L_{ik}^2$ and $H_{1,ii} = \sum_{k=i}^{n} L_{ki}^2$. Complete pivoting is equivalent to the fact that $L_{ii}^2 \geq \sum_{k=i}^{j} L_{jk}^2$ for all $j > i$. We wish to prove $\prod_{i=1}^{n} \sum_{k=1}^{i} L_{ik}^2 \geq \prod_{i=1}^{n} \sum_{k=i}^{n} L_{ki}^2$. We systematically use the fact that $ab \geq (a+x)(b-x)$ for $a \geq b \geq x \geq 0$. We illustrate the general procedure in the case of $n = 3$:

$$\begin{aligned}(L_{11}^2)(L_{21}^2 + L_{22}^2)(L_{31}^2 + L_{32}^2 + L_{33}^2) &\geq (L_{11}^2 + L_{21}^2)(L_{22}^2)(L_{31}^2 + L_{32}^2 + L_{33}^2) \\ &\geq (L_{11}^2 + L_{21}^2 + L_{31}^2)(L_{22}^2)(L_{32}^2 + L_{33}^2) \\ &\geq (L_{11}^2 + L_{21}^2 + L_{31}^2)(L_{22}^2 + L_{32}^2)(L_{33}^2).\end{aligned}$$

2. We have

$$\mathcal{H}(H_1) = \frac{\det(L)^2}{\prod_{i=1}^{n}(L^T L)_{ii}} = \frac{\prod_{i=1}^{n} L_{ii}^2}{\prod_{i=1}^{n}(\sum_{k=i}^{n} L_{ki}^2)} = \prod_{i=1}^{n} \frac{L_{ii}^2}{\sum_{k=i}^{n} L_{ki}^2} \geq \prod_{i=1}^{n} \frac{1}{i} = \frac{1}{n!}.$$

To see that this bound is attainable, let $H = LL^T$ where $L_{ii} = \mu^{(i-1)/2}$ and $L_{ij} = (1-\mu)^{1/2}\mu^{(i-1)/2}$. Now let $\mu > 0$ become small.

3. The result follows from part 2 and Proposition 6.2, part 5. □

The example in part 2 of the proposition for which the Hadamard bound is attainable unfortunately has the property that the resulting upper bound in part 3 is a gross overestimate. While the upper bound grows as $e \cdot n \cdot n!$, $\kappa(A_1)$ only grows like $n^{3/2}$. However, $\kappa(A_0)$ grows like $\mu^{-n/2}$, which can be arbitrarily larger than the bound in part 3. The choice $\mu = 0.5$ provides an example in which the upper bound in part 3 can arbitrarily exceed both $\kappa(A_0)$ and $\max_{m \geq 1} \kappa(A_m)$ for large n.

Nonetheless, in numerical experiments the upper bound $e \cdot n/\mathcal{H}(H_1)$ on $\max_{m \geq 1} \kappa(A_m)$ never exceeded 40. We also always observed that $\kappa(A_1) \leq \kappa(A_0)$ in all cases, although this is not true in general [23].

Recently, Slapničar [17] has improved the $e \cdot n \cdot n!$ bound to $O(4^n)$ and has shown that this improved bound is attainable; see also related results in [13].

7. Numerical experiments. In this section we present the results of numerical experiments. Briefly, we tested every error bound of every algorithm presented in this paper, and verified that they held in all examples. In fact, the performance is better than we were able to explain theoretically, both because we could observe little or no growth in actual errors for increasing dimension, and because of the surprisingly small values attained by $\max_m \kappa(A_m)/\kappa(A_0)$ (see §6).

These tests were performed using FORTRAN on a SUN 4/260. The arithmetic was IEEE standard double precision [1], with a machine precision of $\varepsilon = 2^{-53} \approx 10^{-16}$ and over/underflow threshold $10^{\pm 308}$.

There were essentially four algorithms tested: two-sided Jacobi (Algorithm 3.1), one-sided Jacobi (Algorithms 4.1 and 4.2), right-handed Jacobi with pivoting (Algorithm 4.4), and bisection/inverse iteration (Algorithms 5.1 and 5.2). All were used with the stopping criterion $\text{tol} = 10^{-14}$.

Since we claim that these algorithms are more accurate than any other, we tested their accuracy as follows. We considered only symmetric positive definite eigenproblems, and solved every one using every algorithm. The different answers were compared to see if they agreed to the predicted accuracy (which they did). They were also compared to the EISPACK routines tred2/tql2 [18], which implement tridiagonalization followed by QR iteration. Small eigenvalues computed by EISPACK were often negative, indicating total loss of relative accuracy.

For example, the matrix

$$H = \begin{bmatrix} 10^{40} & 10^{19} & 10^{19} \\ 10^{19} & 10^{20} & 10^{9} \\ 10^{19} & 10^{9} & 1 \end{bmatrix}$$

has all its eigenvalues computed to high relative accuracy by Jacobi, whereas QR computes at least one negative or zero eigenvalue, no matter how the rows and columns are ordered.[1] This shows that QR cannot be made to deliver high relative accuracy on appropriately graded matrices, as suggested in [18].

The remainder of this section is organized as follows: Section 7.1 discusses test matrix generation. Section 7.2 discusses the accuracy of the computed eigenvalues. Section 7.3 discusses the accuracy of the computed eigenvectors. Section 7.4 discusses the the growth of $\max_m \kappa(A_m)/\kappa(A_0)$. Section 7.5 discusses convergence rates; here the speed advantage of right-handed Jacobi with pivoting is apparent.

7.1. Test matrix generation. We generated several categories of random test matrices according to three parameters: the dimension n, κ_A, and κ_D. First, we describe the algorithm used to generate a random matrix from these parameters and then the sets of parameters used.

We tested matrices of dimensions $n = 4$, 8, 16, and 50. Since testing involved solving an $n \times n$ eigenproblem after each Jacobi rotation (to evaluate $\kappa(A_m)$) and there are $O(n^2)$ Jacobi rotations required for convergence, testing costs $O(n^5)$ operations per matrix.

Given κ_A, we generated a random symmetric positive definite matrix with unit diagonal and approximate condition number κ_A as follows. We began by generating a diagonal matrix T with diagonal entries in a geometric series from 1 down to $1/\kappa_A$. Then we generated an orthogonal matrix U uniformly distributed with respect to Haar measure [19], and formed UTU^T. Finally, we computed another diagonal matrix K so that $A_0 = KUTU^TK$ had unit diagonal. This last transformation can decrease the condition number of UTU^T, but usually not by much. For 4×4 matrices, it decreased it by as much as a factor of 500, for 8×8 matrices by a factor of 20, for 16×16 matrices by a factor of 5, and for 50×50 matrices by a factor of 1.5. (This decreasing variability is at least partly due to the fact that we ran fewer tests on the larger matrices.) For a more complete discussion of the test matrix generation software, see [9].

Given κ_D, we generated a random diagonal matrix D_0 with diagonal entries whose logarithms were uniformly distributed between 0 and $\log \kappa_D$. This means the diagonal entries themselves were distributed from 1 to κ_D. The uniform distribution of the logarithm essentially means that every decade is equally likely, and so matrices D_0 are generated with entries of widely varying magnitudes.

The resulting random matrix was then $H_0 = D_0 A_0 D_0$.

We generated random matrices with five possible different values of κ_A: 10, 10^2, 10^4, 10^8, and 10^{12}; six possible different values of κ_D: 10^5, 10^{10}, 10^{20}, 10^{30}, 10^{50}, and 10^{100}; and four different dimensions $n = 4$, 8, 16, and 50. This makes a total of $5 \times 6 \times 4 = 120$ different classes of matrices. In each class of dimension $n = 4$ matrices, we generated 100 random matrices, in each class of $n = 8$, we generated 50 random matrices, in each class of $n = 16$, we generated 10 random matrices, and in each class

[1] This was using version 3.5h of Matlab on a SUN 4/260. Later versions of Matlab may get different results. For more analysis, see [7] and [15].

of $n = 50$, we generated one random matrix. This makes a total of 4830 different test matrices.

The matrices had, in some cases, eigenvalues ranging over 200 orders of magnitude (when $\kappa_D = 10^{100}$). The relative gaps relgap_λ ranged from .028 to $2 \cdot 10^{42}$.

7.2. Accuracy of the computed eigenvalues. There are two accuracy bounds for eigenvalues from §2 which we tested. The first one is based on Theorem 2.3 (or Theorem 2.14 together with Theorem 4.6), which says that if λ_i' and λ_i'' are approximations of λ_i computed by two of our algorithms, then

$$Q_1 \equiv \frac{|\lambda_i' - \lambda_i''|}{\kappa(A_0)\lambda_i'}$$

should be $O(\text{tol})$, where $\text{tol} = 10^{-14}$ is our stopping criterion. For two-sided Jacobi and one-sided Jacobi, Q_1 never exceeded $2 \cdot 10^{-15}$. For two-sided Jacobi and right-handed Jacobi with pivoting, Q_1 also never exceeded $2 \cdot 10^{-15}$. Every matrix had an eigenvalue for which Q_1 exceeded $4 \cdot 10^{-18}$, showing that the bound of Theorem 2.3 is attainable, as predicted by Proposition 2.10.

In the case of bisection, we did not run a bisection algorithm to convergence for each eigenvalue, but rather took the eigenvalues λ_i' computed by two-sided Jacobi, made intervals $[(1 - \text{tol} \cdot \kappa(A_0))\lambda_i', (1 + \text{tol} \cdot \kappa(A_0))\lambda_i']$ from each one, and used bisection to verify that each interval contained one eigenvalue (overlapping intervals were merged and the counting modified in the obvious way). All intervals successfully passed this test.

The second accuracy bound is from Proposition 2.4 (or Proposition 2.15 together with Theorem 4.6), which predicts that

$$Q_2 \equiv \frac{|\lambda_i' - \lambda_i''|}{\|D_0 v_i\|_2^2}$$

should be $O(\text{tol})$. Here v_i is the unit eigenvector computed by two-sided Jacobi. For two-sided Jacobi and one-sided Jacobi, Q_2 never exceeded $2 \cdot 10^{-14}$. For two-sided Jacobi and right-handed Jacobi with pivoting, Q_2 never exceeded $9 \cdot 10^{-15}$. Every matrix had an eigenvalue for which Q_2 exceeded $5 \cdot 10^{-16}$, showing that the bound of Proposition 2.4 is attainable, as it predicts.

In the case of bisection, we again made intervals $[\lambda_i' - \text{tol} \cdot \|D_0 v_i\|_2^2, \lambda_i' + \text{tol} \cdot \|D_0 v_i\|_2^2]$ from each eigenvalue λ_i' and verified that each interval contained the proper number of eigenvalues.

Finally, we verified a slightly weakened version of Proposition 2.7, that

$$\lambda_{\min}(A_0) - \text{tol} \le \frac{\lambda_i'}{h_i} \le \lambda_{\max}(A_0) + \text{tol}$$

for the eigenvalues λ_i' computed by two-sided Jacobi. Here h_i is the ith smallest diagonal entry of H_0. Adding and subtracting tol to the upper and lower bounds takes into account the errors in computing λ_i'.

7.3. Accuracy of the computed eigenvectors. There is one bound on the magnitude of the components of the eigenvectors, and two accuracy bounds, one for the norm error and one for the componentwise error.

We begin with a few details about our implementation of inverse iteration. We used the eigenvalues computed by two-sided Jacobi, and the vector of all ones as a starting vector. Convergence always occurred after just one iteration.

The componentwise bound on the magnitude of the eigenvectors is based on Proposition 2.8, which says that the components of the normalized eigenvector v_i should be bounded by

$$|v_i(j)| \le \bar{v}_i(j) \equiv (\kappa(A_0))^{3/2} \cdot \min\left(\left(\frac{\lambda_i}{\lambda_j}\right)^{1/2}, \left(\frac{\lambda_j}{\lambda_i}\right)^{1/2} \right).$$

This was verified for the eigenvectors computed by all four algorithms. We note that since this bound is proportional to $\kappa(A_0)^{3/2}$, it becomes weaker as $\kappa(A_0)$ becomes larger, and indeed becomes vacuous for matrices with $\kappa(A_0)$ large and eigenvalues in a narrow range.

The norm error bounds are based on Theorem 2.5 (or Theorem 2.16 together with Theorem 4.6), which predicts that if v_i' and v_i'' are approximations of the unit eigenvector v_i computed by two of our algorithms, then

$$Q_3 \equiv \frac{\|v_i' - v_i''\|_2}{(\kappa(A_0)/\mathrm{relgap}_{\lambda_i}) + 1}$$

should be $O(\mathrm{tol})$. (We add the 1 in the denominator because a single roundoff error in the largest entry can cause a norm error of ε; see Theorem 3.3 or Theorem 4.3.)

For two-sided Jacobi and one-sided Jacobi, Q_3 never exceeded $3 \cdot 10^{-16}$. For two-sided Jacobi and right-handed Jacobi with pivoting, Q_3 also never exceeded $2 \cdot 10^{-14}$. For two-sided Jacobi and inverse iteration, Q_3 never exceeded $8 \cdot 10^{-14}$. Every matrix had an eigenvector for which Q_3 exceeded 10^{-18} for every pair of algorithms compared, showing that the bound of Theorem 2.5 is nearly attainable, as predicted by Proposition 2.11.

The second accuracy bound is based on Proposition 2.9 (or Proposition 2.20 and Theorem 4.6), which predicts that

$$Q_4 \equiv \frac{|v_i'(j) - v_i''(j)| \min(\mathrm{relgap}_{\lambda_i}, 2^{-1/2})}{\kappa(A_0) \cdot \bar{v}_i(j)}$$

should be $O(\mathrm{tol})$. For two-sided Jacobi and one-sided Jacobi, Q_4 never exceeded $3 \cdot 10^{-17}$. For two-sided Jacobi and inverse iteration, Q_4 never exceeded $3 \cdot 10^{-15}$. For two-sided Jacobi and right-handed Jacobi with pivoting, Q_4 was as large as .02, which is consistent with the fact that right-handed Jacobi with pivoting computes the eigenvectors as left singular vectors of L, for which we only have a normwise error bound (Theorem 4.3). For the other algorithm, Q_4 was only 10^{-30} for matrices with $\kappa(A_0) = 10^{12}$; this reflects the factor $\kappa(A_0)^{5/2}$ in the denominator of Q_4, a weakness of Proposition 2.8. In other words, the componentwise error bounds are generally only interesting for small to medium $\kappa(A_0)$.

7.4. Growth of $\max_m \kappa(A_m)/\kappa(A_0)$. In computing

$$Q_5 \equiv \max_m \kappa(A_m)/\kappa(A_0),$$

we note that a single computation requiring M Jacobi rotations supplied us not just with one value of Q_5, but rather $M - 1$: Since every A_i can be thought of as starting a new eigenvalue computation, we may also measure $\max_{m \ge i} \kappa(A_m)/\kappa(A_i)$ for all $i < M$. Thus, all told, our 4830 different matrices represent over 900,000 data points of Q_5.

TABLE 1
Hadamard upper bound Q_6 on $\max_m \kappa(A_m)/\kappa(A_0)$.

n	κ_A				
	10	10^2	10^4	10^8	10^{12}
4	5.8	13	590	$6.3 \cdot 10^6$	$6.1 \cdot 10^{10}$
8	21	410	$1.1 \cdot 10^7$	$9.1 \cdot 10^{17}$	∞
16	200	$2.7 \cdot 10^5$	$1.8 \cdot 10^{15}$	∞	∞
50	$6.4 \cdot 10^5$	$8.0 \cdot 10^{16}$	∞	∞	∞

The largest value of Q_5 encountered was 1.82. This was for an 8×8 matrix with $\kappa(A_0) = 1.4 \cdot 10^{12}$, and eigenvalues ranging over 133 orders of magnitude. 141 Jacobi rotations (a little over 5 sweeps) were required for convergence, plus 28 more steps (one more sweep) where no work is done to recognize convergence. In Fig. 1, a plot is shown of $\kappa(A_i) - 1$ versus i. We plot $\kappa(A_i) - 1$ instead of $\kappa(A_i)$ in order to see the quadratic convergence of $\kappa(A_i)$ to 1. The graph appears nearly monotonic, except for a slight rise near $i = 20$. This is seen more clearly in Fig. 2, which plots $\max_{m \geq i} \kappa(A_m)/\kappa(A_i)$ versus i. Here the maximal nonmonotonicity of the curve near $i = 20$ is apparent.

Recently, Wang [23] found a family of examples where Q_5 was as large as 8 for matrices up to dimension 50. These matrices have 1 on the diagonal and $1 - \epsilon$ on the offdiagonal, where ϵ is small. However, by using a different pivoting strategy than cyclic-by-rows, namely, the parallel pivoting discussed in Proposition 6.1, this growth could be eliminated.

Now we consider the Hadamard-based upper bound on Q_5 from Proposition 6.2:

$$Q_5 \leq Q_6 \equiv \frac{e \cdot n}{\mathcal{H}(H_0) \cdot \kappa(A_0)}.$$

Table 1 gives the maximum values of this upper bound for different values of dimension n and $\kappa_A \approx \kappa(A_0)$. Recall that the true value of Q_5 never exceeds 1.82. As Proposition 6.2 suggests, this upper bound should not depend on D_0 and indeed the values observed depended very little on D_0.

As can be seen, the Hadamard-based bound is of little use except for very small matrices of modest $\kappa(A_0)$. ∞ means the value overflowed.

Now we consider right-handed Jacobi with pivoting. Let us recall the notation of §6: Let $PH_0P^T = LL^T$ be Cholesky with complete pivoting, and let $L^TL = H_1 = D_1A_1D_1$. As suggested in that section, we expect both $\kappa(A_1)$ to be smaller than $\kappa(A_0)$, and the Hadamard-based upper bound

$$Q_5 \leq Q_7 \equiv \max\left(1, \frac{e \cdot n}{\mathcal{H}(H_1) \cdot \kappa(A_0)}\right)$$

on Q_5 to be much smaller than the one for two-sided Jacobi.

First, $\kappa(A_1)/\kappa(A_0)$ never exceeded $\frac{6}{10}$. In fact, $\kappa(A_1)$ *never exceeded* 40 *for any matrix.* This is quite remarkable. This means that all essential rounding errors occurred during the initial Cholesky decomposition. Finally, the Hadamard-based upper bound Q_7 on Q_5 never exceeded 29. (Recently, Wang [23] found an example where $\kappa(A_1)/\kappa(A_0)$ slightly exceeded 1; in his example, $\kappa(A_0)$ was close to 1.)

7.5. Convergence rates. We begin with a few details on how we counted the number of Jacobi rotations required for convergence. In all algorithms (two-sided Jacobi, one-sided Jacobi, and right-handed Jacobi with pivoting), we stopped when the last $n(n-1)/2$ stopping tests $|H_{ij}| \cdot (H_{ii}H_{jj})^{-1/2} \leq \text{tol}$ succeeded; this means every off-diagonal entry of H satisfies the stopping criterion. In the case of two-sided

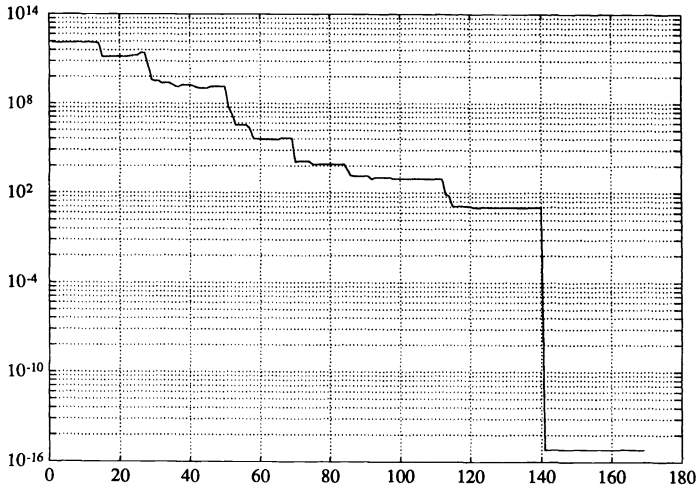


FIG. 1. $\kappa(A_i) - 1$ *versus* i.

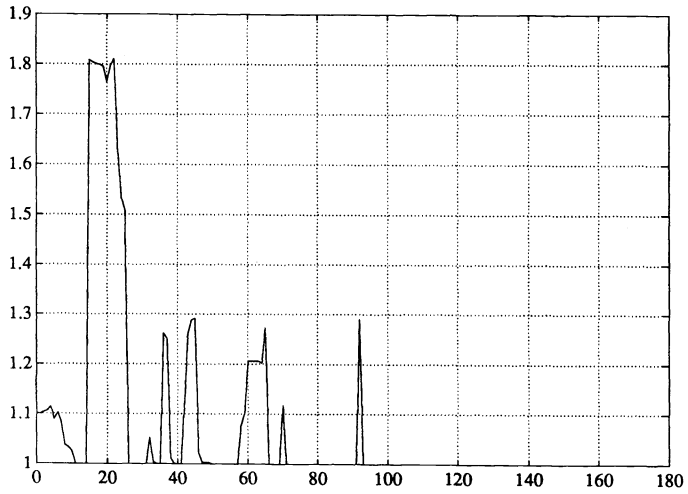


FIG. 2. $\max_{m \geq i} \kappa(A_m)/\kappa(A_i)$ *versus* i.

Jacobi, this means the last $n(n-1)/2$ Jacobi rotations involved almost no work. For the two one-sided Jacobis, however, evaluating the stopping criterion costs three inner products, so the last $n(n-1)/2$ rotations involve a significant amount of work, even if no rotations are performed. This must be kept in mind when comparing the number of rotations for two-sided and one-sided Jacobi.

We used the same standard cyclic pivot sequence for all the algorithms: (1,2), (1,3), $\cdots$, (1,n), (2,3), $\cdots$, (2,n), (3,4), $\cdots$, $(n-1, n)$.

We begin by comparing two-sided Jacobi and one-sided Jacobi. In exact arithmetic, these two algorithms are identical. In practice, they usually took the same number of steps, although one-sided Jacobi did vary from 20 percent faster to 50 per-

TABLE 2
Average number of sweeps for two-sided Jacobi (TsJ) *and right-handed Jacobi with pivoting* (RhJwP).

κ_A	κ_D	Dimension n							
		4		8		16		50	
		TsJ	RhJwP	TsJ	RhJwP	TsJ	RhJwP	TsJ	RhJwP
10	10^5	3.7	3.0	4.9	3.7	5.7	4.4	6.4	5.0
	10^{10}	3.5	2.5	4.6	3.3	5.6	4.1	6.4	5.0
	10^{20}	3.1	2.2	4.5	2.8	5.5	3.6	6.0	4.0
	10^{30}	3.0	2.1	4.6	2.5	5.5	3.4	6.3	4.0
	10^{50}	2.8	1.9	4.4	2.3	5.5	3.1	5.8	4.0
	10^{100}	2.7	1.7	4.5	2.0	5.6	2.6	5.8	3.0
10^2	10^5	3.8	3.0	5.2	3.8	6.4	4.5	7.5	6.0
	10^{10}	3.5	2.5	5.1	3.3	6.2	4.1	7.4	5.0
	10^{20}	3.2	2.2	4.9	2.9	6.2	3.9	7.1	4.0
	10^{30}	3.0	2.0	4.8	2.6	5.8	3.3	6.8	4.1
	10^{50}	2.9	1.9	4.8	2.2	6.1	3.0	6.5	4.0
	10^{100}	2.8	1.6	4.7	2.0	6.0	2.7	6.8	3.4
10^4	10^5	4.0	2.9	5.8	3.6	7.5	4.5	9.2	6.0
	10^{10}	3.7	2.5	5.6	3.3	7.2	4.1	9.3	5.0
	10^{20}	3.2	2.2	5.3	2.9	7.2	3.7	8.5	4.9
	10^{30}	3.1	2.1	5.2	2.6	6.8	3.1	8.2	4.0
	10^{50}	2.9	1.9	5.2	2.4	6.6	3.0	8.5	4.6
	10^{100}	2.7	1.7	4.9	2.2	6.9	2.4	8.0	3.9
10^8	10^5	3.9	2.7	6.4	3.5	9.7	4.1	13.5	6.0
	10^{10}	3.6	2.3	6.3	3.2	9.4	3.8	12.4	5.0
	10^{20}	3.3	2.1	5.7	2.8	8.9	3.5	11.7	4.7
	10^{30}	3.1	2.1	5.5	2.6	8.6	3.4	12.0	4.0
	10^{50}	2.9	1.9	5.3	2.3	8.5	3.1	11.6	4.0
	10^{100}	2.9	1.7	5.1	2.0	8.7	2.6	11.6	4.0
10^{12}	10^5	3.8	2.5	6.8	3.1	10.6	4.0	16.5	6.0
	10^{10}	3.6	2.2	6.4	3.0	10.3	3.9	15.6	5.0
	10^{20}	3.4	2.1	6.0	2.7	9.8	3.5	15.3	5.0
	10^{30}	3.1	2.0	5.8	2.5	10.2	3.3	15.2	4.0
	10^{50}	2.9	1.9	5.6	2.3	9.3	3.2	13.7	3.9
	10^{100}	2.8	1.6	5.2	2.0	8.7	2.7	15.2	3.0

cent slower than two-sided Jacobi on some examples. Hereafter, we will only compare two-sided Jacobi to right-handed Jacobi with pivoting.

The most interesting phenomenon was the speedup experienced by right-handed Jacobi with pivoting with respect to two-sided Jacobi. In Table 2 we present the raw data on the number of sweeps required for convergence.

There are a number of interesting trends exhibited in this table. First, RhJwP (right-handed Jacobi with pivoting) never takes more than six sweeps to converge for any matrix, whereas TsJ (two-sided Jacobi) takes up to 16.5. In fact, RhJwP is almost always faster than TsJ (in one example it took 5 percent longer), and can be up to five times faster (3.0 sweeps versus 15.2 sweeps for $\kappa_A = 10^{12}$, $\kappa_D = 10^{100}$, and $n = 50$). Second, the number of sweeps increases with increasing κ_A for TsJ, but not for RhJwP. Third, the number of sweeps increases with increasing dimension for both TsJ and RhJwP, but much more modestly for RhJwP (from two to three up to six) than for TsJ (from three to four up to fifteen). Thus the running time for RhJwP is much less dependent on the problem size or sensitivity (as measured by κ_A) than TsJ. Fourth, the number of sweeps decreases as κ_D increases, both for TsJ and RhJwP,

but much more markedly for RhJwP (up to a factor of 2) than for TsJ (usually just one sweep).

8. Conclusions. In this paper we have developed new perturbation theory for the eigenvalues and eigenvectors of symmetric positive definite matrices, as well as for eigenvalues of symmetric positive definite pencils. This theory assumes that the perturbations are scaled analogously to the way the matrix is scaled, letting us derive much tighter bounds than in the classical theory. In particular, we get relative error bounds for the eigenvalues and individual components of the eigenvectors, which are (nearly) attainable. The bound for symmetric positive definite pencils may be applied to matrices arising in finite element modeling.

Second, we have shown both through formal error analysis and numerical experiment that Jacobi's method (with a proper stopping criterion) computes the eigenvalues and eigenvectors with these error bounds. We also show that bisection and inverse iteration (applied to the original matrix) attain these bounds. In contrast, methods based on tridiagonalization (such as QR, divide and conquer, traditional bisection, etc.) fail to attain these bounds. In particular, QR can fail to attain these bounds whether or not preceded by tridiagonalization.

We have similar perturbation theorems for the singular value decomposition of a general matrix and the generalized singular values of a pair of matrices, and similar error analyses and numerical experiments for one-sided Jacobi applied to this problem. We may also use one-sided Jacobi to solve the symmetric positive definite eigenproblem.

We have discussed an accelerated version of Jacobi for the symmetric positive definite eigenproblem, which has the property that the more its accuracy exceeds that of QR (or other conventional algorithms), the faster it converges. However, it cannot compute tiny components of eigenvectors as accurately as the other versions of Jacobi, although it computes the eigenvectors with the same norm error bounds. Unless getting the tiny eigenvector components is important, we recommend this accelerated version of Jacobi for the symmetric positive definite eigenproblem.

The quantity $\max_m \kappa(A_m)/\kappa(A_0)$ was seen to be central in the analysis of Jacobi's accuracy. Numerical experiments show it to be much smaller in practice than we can explain. For the accelerated version of Jacobi we provide an inexpensive estimator of $\max_m \kappa(A_m)/\kappa(A_0)$, which works very well in practice. Explaining the excellent behavior of $\max_m \kappa(A_m)/\kappa(A_0)$ is an important open problem.

The error analyses of Jacobi dealt only with the simplest implementations. It would be worthwhile to extend these analyses to cover various enhancements introduced by Veselić, Hari, Rutishauser, and others. These include delayed updates of the diagonal entries and an alternate formula for updating the off-diagonal entries [16], [22], as well as block Jacobi methods.

In future work, we plan to extend these results to the symmetric positive definite generalized eigenproblem, as well as indefinite matrices. Any extension requires an appropriate perturbation theory; therefore, we do not expect to be able to extend the result to all indefinite matrices, since there is no guaranteed way to compute the zero eigenvalues of a singular matrix to "high relative accuracy" without computing them exactly, a feat requiring high precision arithmetic. A class of indefinite matrices for which a suitable perturbation theory exists are the scaled diagonally dominant matrices [2]. The perturbation theory also already exists (at least for eigenvalues) for the symmetric positive definite generalized eigenproblem.

REFERENCES

[1] ANSI/IEEE, IEEE *Standard for Binary Floating Point Arithmetic*, IEEE Press, New York, Std 754-1985 ed., 1985.

[2] J. BARLOW AND J. DEMMEL, *Computing accurate eigensystems of scaled diagonally dominant matrices*, SIAM J. Numer. Anal., 27 (1990), pp. 762–791.

[3] M. BERRY AND A. SAMEH, *Parallel algorithms for the singular value and dense symmetric eigenvalues problems*, J. Comput. Appl. Math., 27 (1989), pp. 191–213.

[4] J. BUNCH AND L. KAUFMAN, *Some stable methods for calculating inertia and solving symmetric linear systems*, Math. Comp., 31 (1977), pp. 163–179.

[5] P. P. M. DE RIJK, *A one-sided Jacobi algorithm for computing the singular value decomposition on a vector computer*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 359–371.

[6] J. DEMMEL, *The condition number of equivalence transformations that block diagonalize matrix pencils*, SIAM J. Numer. Anal., 20 (1983), pp. 599–610.

[7] ———, *On the inherent inaccuracy of implicit tridiagonal* QR, IMA Preprint 963, University of Minnesota, Minneapolis, MN, April 1992.

[8] J. DEMMEL AND W. KAHAN, *Accurate singular values of bidiagonal matrices*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 873–912.

[9] J. DEMMEL AND A. MCKENNEY, *A test matrix generation suite*, Computer Science Department Tech. Report, Courant Institute, New York, July 1989 (LAPACK Working Note #9).

[10] J. DEMMEL AND K. VESELIĆ, *Jacobi's method is more accurate than* QR, Computer Science Department Tech. Report 468, Courant Institute, New York, October 1989 (LAPACK Working Note #15).

[11] G. GOLUB AND C. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, MD, 1989.

[12] V. HARI AND K. VESELIĆ, *On Jacobi methods for singular value decompositions*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 741–754.

[13] N. J. HIGHAM, *Analysis of the Cholesky decomposition of a semi-definite matrix*, in Reliable Numerical Computation, M. G. Cox and S. Hammarling, eds., Clarendon Press, Oxford, 1990, Chap. 9, pp. 161–186.

[14] T. KATO, *Perturbation Theory for Linear Operators*, Second Edition, Springer-Verlag, Berlin, 1980.

[15] J. LE AND B. PARLETT, *On the sensitivity of the tridiagonal* QR *transformation*, SIAM J. Matrix Anal. Appl., 14 (1993), to appear.

[16] B. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice–Hall, Englewood Cliffs, NJ, 1980.

[17] I. SLAPNIČAR, *Upper bound for the condition of the scaled matrix of the symmetric eigenvalue problem*, Lehrgebiet Mathematische Physik, Fern-Universität Hagen, Hagen, Germany, November 1990.

[18] B. T. SMITH, J. M. BOYLE, J. J. DONGARRA, B. S. GARBOW, Y. IKEBE, V. C. KLEMA, AND C. B. MOLER, *Matrix Eigensystem Routines*—EISPACK *Guide*, Lecture Notes in Computer Science 6, Springer-Verlag, Berlin, 1976.

[19] G. W. STEWART, *On efficient generation of random matrices with an application to condition estimation*, SIAM J. Numer. Anal., 17 (1980), pp. 403–409.

[20] ———, *A method for computing the generalized singular value decomposition*, in Matrix Pencils, B. Kågström and A. Ruhe, eds., Springer-Verlag, Berlin, 1983, pp. 207–220. Lecture Notes in Mathematics 973, Proceedings, Pite Havsbad, 1982.

[21] A. VAN DER SLUIS, *Condition numbers and equilibration of matrices*, Numer. Math., 14 (1969), pp. 14–23.

[22] K. VESELIĆ AND V. HARI, *A note on a one-sided Jacobi algorithm*, Numer. Math., 56 (1990), pp. 627–633.

[23] X. WANG, private communication, 1990.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1246–1254, October 1992

016

A TRICYCLIC TRIDIAGONAL EQUATION SOLVER*

DAVID S. DODSON† AND STEWART A. LEVIN‡

Abstract. An improved method for solving tridiagonal equations on CONVEX computers is exhibited. The method employs cyclic reduction by powers of three and has several advantages over conventional power-of-two reduction. The number of divisions per element is cut in half, while the number of multiplications and additions remains almost exactly the same. Memory bank conflicts are minimized because all vector strides are powers of three, i.e., odd. The number of memory accesses is reduced by a quarter. This is especially important for the CONVEX where cyclic reduction is memory limited. Last, the algorithm supports modification in a recently published manner that maximizes the use of full vector register segments throughout the reduction [R. Reuter, *Parallel Comput.*, 8 (1988), pp. 371–376].

Key words. tridiagonal linear systems, cyclic reduction, vector computers

AMS(MOS) subject classifications. 65F05, 65W05

1. Introduction.

Cyclic reduction in various forms is used routinely to improve the speed of solution of tridiagonal systems on vector and parallel computers. Variables are grouped into two disjoint index sets and simultaneously eliminated from the corresponding tridiagonal equations to yield a descending sequence of progressively smaller tridiagonal systems to solve. Because this reduction process and corresponding back substitution are vectorizable, this algorithm is beneficial despite the doubled arithmetic load it incurs when compared to the conventional serial algorithm, e.g., SGTSL from LINPACK [3].

In the past, the index sets have been the even and odd variables, resulting in a halving of the system size at each reduction step. Problems that researchers have wrestled with are memory bank conflicts due to the even strides between vector elements and inefficient use of vector registers due to end effects and short vectors.

In this report we explore a different choice of index sets, specifically every third variable in one set and the rest in the other, and show that it leads to a faster tridiagonal solution algorithm on a CONVEX computer.

2. Tricyclic reduction.

2.1. Algorithm.

We describe the in-place version of power-of-three cyclic reduction. Denote the subdiagonal elements a_j, $j = 1, \ldots, n-1$, the diagonal elements b_j, $j = 0, \ldots, n-1$, the superdiagonal elements c_j, $j = 0, \ldots, n-2$, and the input right-hand side d_j, $j = 0, \ldots, n-1$. Pictorially,

$$\begin{array}{ccccccccc}
a_{3j-2} & b_{3j-2} & c_{3j-2} & & & & & & d_{3j-2} \\
 & a_{3j-1} & b_{3j-1} & c_{3j-1} & & & & & d_{3j-1} \\
 & & a_{3j} & b_{3j} & c_{3j} & & & & d_{3j} \\
 & & & a_{3j+1} & b_{3j+1} & c_{3j+1} & & & d_{3j+1} \\
 & & & & a_{3j+2} & b_{3j+2} & c_{3j+2} & & d_{3j+2}
\end{array}$$

are the matrix entries participating in the reduction steps.

* Received by the editors September 4, 1990; accepted for publication (in revised form) March 4, 1991.

† Mathematical Software Group, CONVEX Computer Corporation, P.O. Box 833851, Dallas, Texas 75083-3851 (dodson@convex.com).

‡ Mobil Research and Development Corporation, P.O. Box 819047, Dallas, Texas 75381-9047 (stew@hanauma.stanford.edu).

The first step is to invert the submatrices

$$\begin{pmatrix} b_{3j+1} & c_{3j+1} \\ a_{3j+2} & b_{3j+2} \end{pmatrix},$$

which we do directly using Cramer's Rule to produce

$$\begin{pmatrix} b'_{3j+2} & -c'_{3j+1} \\ -a'_{3j+2} & b'_{3j+1} \end{pmatrix} = \frac{1}{b_{3j+1}b_{3j+2} - c_{3j+1}a_{3j+2}} \begin{pmatrix} b_{3j+2} & -c_{3j+1} \\ -a_{3j+2} & b_{3j+1} \end{pmatrix}.$$

Here the primed entries may be stored in place of the original unprimed matrix entries. This matrix inverse is then multiplied into the corresponding rows of the original system to bring it into the form

$$\begin{array}{rrrrrrrr} a'_{3j-2} & 1 & 0 & -c''_{3j-2} & & & & d'_{3j-2} \\ -a''_{3j-1} & 0 & 1 & c'_{3j-1} & & & & d'_{3j-1} \\ & & a_{3j} & b_{3j} & c_{3j} & & & d_{3j} \\ & & & a'_{3j+1} & 1 & 0 & -c''_{3j+1} & d'_{3j+1} \\ & & & -a''_{3j+2} & 0 & 1 & c'_{3j+2} & d'_{3j+2} \end{array}$$

where

$$\begin{aligned} d'_{3j+1} &= b'_{3j+2}d_{3j+1} - c'_{3j+1}d_{3j+2}, \\ d'_{3j+2} &= b'_{3j+1}d_{3j+2} - a'_{3j+2}d_{3j+1}, \\ a''_{3j+2} &= a'_{3j+2}a_{3j+1}, \\ a'_{3j+1} &= a_{3j+1}b'_{3j+2}, \\ c''_{3j+1} &= c'_{3j+1}c_{3j+2}, \\ c'_{3j+2} &= c_{3j+2}b'_{3j+1} \end{aligned}$$

replace the previous values of the indicated variables.

To complete the reduction step we eliminate a_{3j} and c_{3j} to produce

$$\begin{array}{rrrrrrrr} a'_{3j-2} & 1 & 0 & -c''_{3j-2} & & & & d'_{3j-2} \\ -a''_{3j-1} & 0 & 1 & c'_{3j-1} & & & & d'_{3j-1} \\ a'_{3j} & 0 & 0 & b'_{3j} & 0 & 0 & c'_{3j} & d'_{3j} \\ & & & a'_{3j+1} & 1 & 0 & -c''_{3j+1} & d'_{3j+1} \\ & & & -a''_{3j+2} & 0 & 1 & c'_{3j+2} & d'_{3j+2} \end{array}$$

with

$$\begin{aligned} b'_{3j} &= b_{3j} - c_{3j}a'_{3j+1} - a_{3j}c'_{3j-1}, \\ d'_{3j} &= d_{3j} - c_{3j}d'_{3j+1} - a_{3j}d'_{3j-1}, \\ a'_{3j} &= a_{3j}a''_{3j-1}, \\ c'_{3j} &= c_{3j}c''_{3j+1}, \end{aligned}$$

again replacing the corresponding unprimed quantities.

We recognize now that the equations in a'_{3j}, b'_{3j}, c'_{3j}, and d'_{3j} now form a tridiagonal system one-third the size of the original system. Thus we reapply the tricyclic reduction to this reduced system.

Having solved the reduced system for the components d''_{3j} of the solution vector, we back-substitute according to the formulas

$$d''_{3j+1} = d'_{3j+1} - a'_{3j+1}d''_{3j} + c''_{3j+1}d''_{3j+3},$$
$$d''_{3j+2} = d'_{3j+2} + a''_{3j+2}d''_{3j} - c'_{3j+2}d''_{3j+3},$$

to obtain the remaining components of the solution.

Straightforward modifications are required at the ends of the tridiagonal systems to avoid computations with undefined indices. The timing results below include these special cases as well as switching to a conventional, scalar, tridiagonal solver for system sizes smaller than 25 equations.

2.2. Operation counts. To analyze this algorithm, we separately tally the operations *load, store, add, multiply, divide,* and *shift.* Here *shift* is the operation of moving the components of a vector one element left or right. On the CONVEX, this may be accomplished by (assembly language) register-to-register vector compression or by storing to memory and then reloading with an offset. Forming the submatrix inverse requires four *load*'s, six *mult*'s, one *add*, and one *div.* Applying the submatrix inverse takes four *load*'s, eight *mult*'s, two *add*'s, and six *store*'s. Finishing the reduction uses four *load*'s, three *shift*'s, six *mult*'s, four *add*'s, and four *store*'s. Finally, the back substitution uses seven *load*'s, one *shift*, four *mult*'s, four *add*'s, and two *store*'s.

The number of reductions and back substitutions is approximately

$$n\sum_{j=1}^{\infty}\frac{1}{3^j} = \frac{n}{2},$$

leading to an average operation count per equation of

$$\tfrac{1}{2}(12\ load + 3\ shift + 20\ mult + 7\ add + 1\ div + 10\ store)$$

for reduction, and

$$\tfrac{1}{2}(7\ load + 1\ shift + 4\ mult + 4\ add + 0\ div + 2\ store)$$

for back substitution.

The comparable operation counts for power-of-two cyclic reduction are

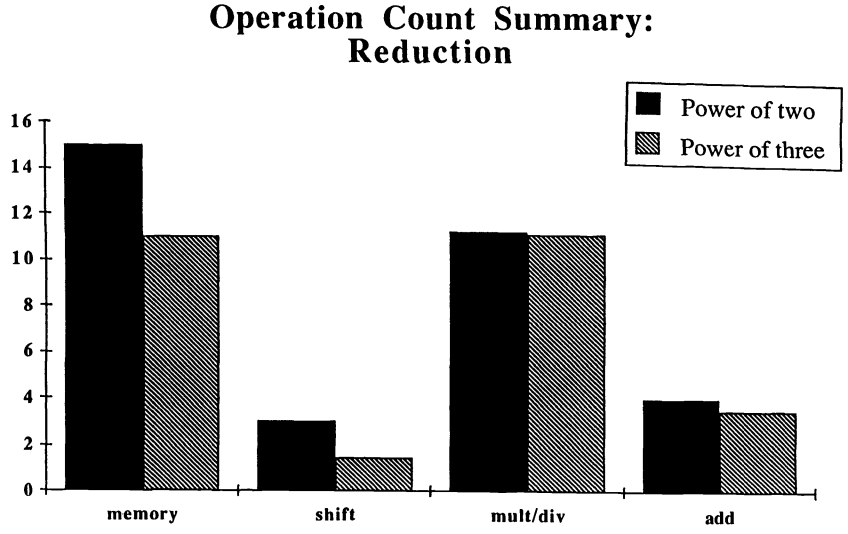
$$8\ load + 3\ shift + 9\ mult + 4\ add + 1\ div + 7\ store$$

for reduction, and

$$4\ load + 1\ shift + 2\ mult + 2\ add + 0\ div + 1\ store$$

for back substitution.

Figure 1 summarizes these operation counts graphically for the CONVEX. Note that we have conservatively equated each division to 2.25 multiplications [1]. It is apparent that on a CONVEX the power-of-three algorithm is never inferior to the power-of-two method in this measure of performance.



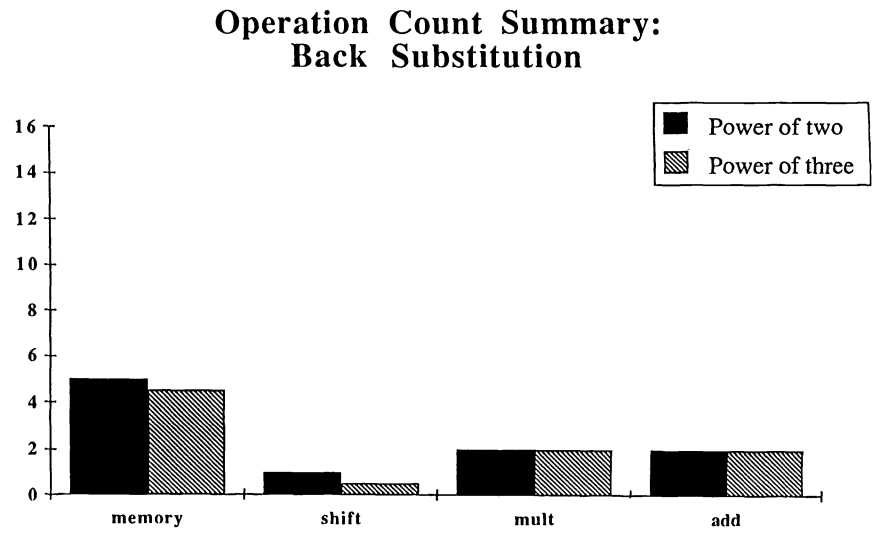


FIG. 1. *Average per-element operation counts for power-of-two versus power-of-three tridiagonal cyclic reduction algorithms on the CONVEX. Tricyclic reduction is always as good or better than the power-of-two method.*

2.3. Analysis. As noted in [9], "cyclic reduction is just Gaussian elimination applied to a permuted system.... As a result, the cyclic reduction algorithm is numerically stable for matrices for which Gaussian elimination is stable without pivoting, such as symmetric positive-definite or diagonally dominant matrices." This applies to power-of-three cyclic reduction as well, when the two-by-two principal subminors are solved by Gaussian elimination without pivoting and even more so when, as above, the subminors are inverted directly. To better appreciate other relative merits of power-of-three versus power-of-two cyclic reduction, we start by looking at some limiting cases.

2.3.1. Fast memory. In this setting, we neglect memory accesses and find that the power-of-three method uses $\frac{1}{2}(24\ mult + 11\ add + 1\ div)$ operations per element and the power-of-two method uses $11\ mult + 6\ add + 1\ div$ operations per element. If, as typical on vector computers, a division is 2 to 3 times slower than a multiplication, then the power-of-three has slightly less arithmetic than power-of-two. An analogous finding for fast Fourier transforms (FFTs) was made in [2], where it was observed that, arithmetically, the power-of-three FFT is about 6 percent more efficient than the power-of-two version.

2.3.2. Fast arithmetic. When arithmetic is much faster than memory access times, we neglect computation time and find that power-of-three reduction uses $\frac{1}{2}(19\ load + 4\ shift + 12\ store)$ operations and power-of-two uses $12\ load + 4\ shift + 8\ store$ operations per equation. In this setting, the power-of-three approach saves 25 percent over power-of-two cyclic reduction, regardless of whether the shift is implemented by memory operations or by register-to-register vector compression. In our fastest FORTRAN versions of these cyclic reduction algorithms, all but one of the *shift* s translate into two memory operations.

For vector computers there is an additional factor that affects memory access speeds. Invariably, the memory is divided into interleaved banks in order to permit faster reference to contiguous vectors. Accessing vectors with an element-to-element stride differing from unity can be significantly slower if the spacing causes particular memory banks to be addressed too quickly in succession. In practice, this usually penalizes access to vectors with strides divisible by a power of two, exactly the case in power-of-two cyclic reduction. To minimize this additional cost, many power-of-two cyclic reduction codes are implemented using additional storage to compact intermediate results into contiguous vectors, thereby avoiding strides greater than two.

For parallel, distributed-memory computers, the memory accesses involve processor-to-processor communication. This is generally much slower than arithmetic. In practice, though, communication is usually fastest for elements spaced a power-of-two apart and then the scales tip towards power-of-two cyclic reduction. An example of this is the hypercube multiprocessor architecture wherein elements spaced by a power-of-two are no more than two hops apart in the physical processor grid. On the other hand, with a processor ring, the amount of communication per step in the reduction is approximately constant. Therefore, the power-of-three is better by the ratio $\log 3 / \log 2$ because there are fewer reduction steps.

2.3.3. Early termination. One of the more useful, though infrequently mentioned, advantages of cyclic reduction is the rapid decrease in the off-diagonal terms of the reduced systems when the original system is diagonally dominant. By predicting or monitoring this decrease, one can terminate the reduction stage early when the

off-diagonal components become negligible.

To attain the same off-diagonal reduction, the power-of-three method will use two-thirds of the number of reduction steps needed by the power-of-two version. This slightly increases the ratio of the number of elements processed by the two methods from its asymptotic value of 1:2 improving the competitiveness of power-of-two reduction a little. More importantly, the power-of-two algorithm has more places it can stop, and thus might shave one or two iterations from the asymptotic 3/2's of the iteration count of power-of-three cyclic reduction. This is potentially a significant savings for large tridiagonal systems. In some cases, this savings may be partially offset by changing the last reduction by three into a reduction by two.

2.3.4. Cache. The first-generation CONVEX C1 required that vector operations on noncontiguous elements first be staged into cache memory. (The C200 Series bypasses cache for all vector operations.) Since cyclic reduction, whether by powers of two or three, involves noncontiguous vector access, cache usage becomes a concern. Our aim is to minimize the number of passes over the data so that cache is "missed" as few times as possible. Because it uses fewer passes over the data, tricyclic reduction has an advantage. This advantage is nullified, however, if the partial reduction in [6] is employed. In this variation, a partial reduction plus a shuffle is used to reduce the tridiagonal system by one (or two) vector register lengths. The result is that the data is traversed only twice: once forward during reduction and once backward during resubstitution. Cache misses are minimized. Additionally, vector register usage is maximized. We are not aware of any comparative round-off study for this variation.

2.3.5. Parallelism. Cyclic reduction can be parallelized to a good degree on the CONVEX C200 Series by the conventional method of distributing the iterations of the stripmine sections of the relevant DO-loops among the available processors. The *shift* operations cause some difficulty for the calculations at each end of a vector segment in a reduction step because some of the required data is assigned to a cooperating processor. This can be handled by synchronized memory references or by duplicating the end calculations in both relevant processors. The latter is advantageous when the *shift* operation is handled by vector-to-vector compression.

Memory bank conflicts are a significant concern for parallel operations. The more simultaneous power-of-two memory accesses occurring in parallel, the slower each becomes. For this reason, tricyclic reduction is to be preferred because memory bank conflicts are resolved by a systematic increase in startup time for memory operations rather than a systematic increase in per-element memory transfer time.

2.4. CONVEX timings. The analysis above has indicated that tricyclic reduction is a choice method for solving tridiagonal equations on CONVEX computers. We have programmed the algorithm in both FORTRAN (compiler version 6.0) and CONVEX assembly language for the C200 Series. Both the FORTRAN and the assembler versions vectorize and parallelize. We have also compiled the FORTRAN version with parallelization disabled. Lastly, for comparative timings, we have programmed power-of-two cyclic reduction in FORTRAN, using auxiliary space to compress intermediate results. This we label *sericr* following standard terminology [4]. Figure 2 displays CPU times for these four algorithms. As predicted, FORTRAN savings are nearly 25 percent for the power-of-three algorithm over the power-of-two version. Assembly programming buys an additional 10 percent efficiency over FORTRAN. The small gain achieved with assembler language demonstrates that the algorithm can be expressed in FORTRAN without hiding the available parallelism, that the method

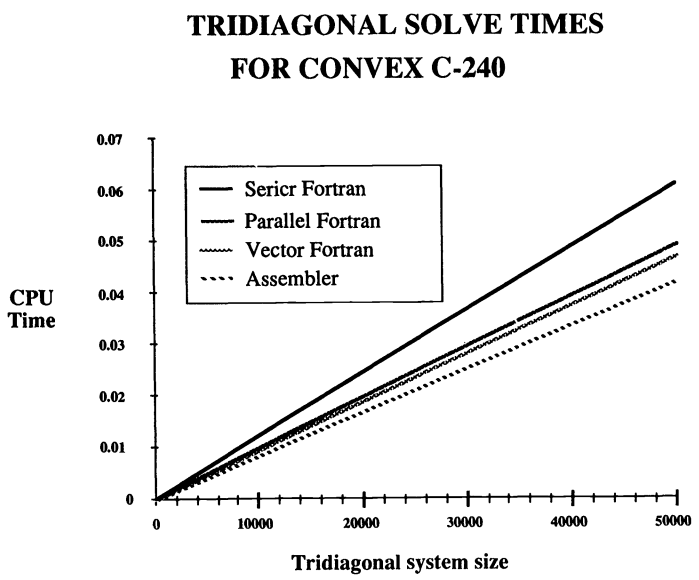


FIG. 2. CPU *times for solving tridiagonal equations using power-of-three and power-of-two cyclic reductions. Because it uses fewer memory operations, tricyclic reduction is around* 25 *percent faster than power-of-two cyclic reduction (sericr) on the CONVEX.*

is matched well to the CONVEX architecture, and that the CONVEX FORTRAN compiler's vectorization, parallelization, and instruction scheduling are quite mature. Figure 3 displays parallel performance of the tricyclic algorithm. For large systems, parallel speedup, i.e., the ratio of CPU time to wall clock time, levels off at around 3.5 out of a possible 4. In comparison, the power-of-two speedup peaks at a lower value of about 3.1. This reflects the fact that the power-of-two algorithm uses more DO-loops with less work per loop and so synchronization occurs more frequently with concomittant loss of parallelism.

3. History and related work. Sweet [7] published a power-of-α block cyclic reduction for the case of symmetric, constant-coefficient block tridiagonal systems in 1974. In that setting, he found that power-of-two cyclic reduction was best, using 40 percent less arithmetic than power-of-three reduction. To our knowledge, vectorized tricyclic reduction for general tridiagonal systems first appeared in 1980, when it was employed by Levin to speed up an implicit finite-difference program running on an IBM 3838 array processor. It was later resurrected for a Cray-1 assembly language exercise in 1986. About the same time, Phuong Vu of Cray Research independently devised the algorithm for use on the Cray-2 [8].

Tricyclic reduction is closely related to the hybrid algorithm GECR [5], [10], a method that applies Gaussian elimination to uniform-sized blocks of consecutive equations to produce one smaller tridiagonal system that is solved by cyclic reduction. Indeed, when GECR is specialized to partitions of size three, a single reduction step in both procedures will produce results identical up to round-off variations. There are two differences, however. First, Cramer's Rule is used in tricyclic reduction as compared to Gaussian elimination in GECR. This cuts the number of divide oper-

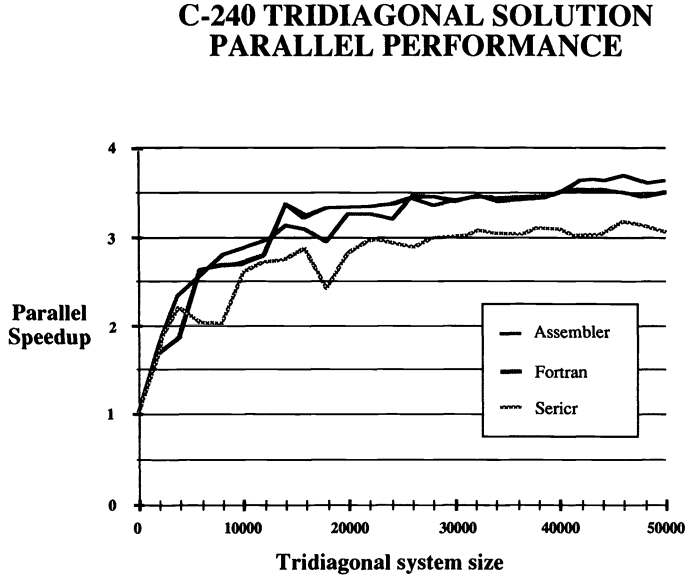


FIG. 3. *Parallel speedups of wall clock times over* CPU *times for power-of-three and power-of-two cyclic reduction on a four-processor CONVEX* C240. *Tricyclic reduction is better because it has fewer loops with more work per loop than power-of-two reduction.*

ations in half compared to GECR. Second, GECR does not recursively reapply the reduction to the smaller resulting system, as it is assumed that a proper choice of partition size renders further parallel reduction unnecessary. For the CONVEX, GECR would subdivide the equations so that the reduced size would be as close as possible to the vector register length of 128. At that point, another algorithm, cyclic reduction, would be applied to this reduced system. We do not yet know whether, in practice, GECR is faster or slower than tricyclic reduction for the CONVEX.

4. Conclusions. Cyclic reduction by powers of three is significantly faster than cyclic reduction by powers of two for solving tridiagonal equations on the CONVEX. The algorithm achieves its better performance primarily by reducing the number of memory accesses needed to solve the system. Additionally, it uses memory more efficiently because vector strides are always odd. Last, its use of half the division operations in fewer loops with more arithmetic per loop produces a somewhat better arithmetic balance and compiler utilization. These performance gains have also been proven on the Cray-2 by independent invention of the algorithm.

REFERENCES

[1] CONVEX *Architecture Reference* (C100 *Series,* C200 *Series*), Fourth Edition, 1989, CONVEX Computer Corp., Document 081-000050-202, Richardson, TX.

[2] J. W. COOLEY AND J. W. TUKEY, *An algorithm for the machine calculation of complex Fourier series*, Math. Comp., 19 (1965), pp. 297–301. (Reprinted in *Digital Signal Processing*, L. R. Rabiner and C. M. Rader, eds., IEEE Press, New York, 1972.)

[3] J. J. DONGARRA, C. B. MOLER, J. R. BUNCH, AND G. W. STEWART, LINPACK: *Users' Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.

[4] R. W. HOCKNEY AND C. R. JESSHOPE, *Parallel Computers*, Adam Hilger, Bristol, 1981.

[5] S. L. JOHNSSON, *Solving tridiagonal systems on ensemble architectures*, SIAM J. Sci. Statist. Comput., 8 (1987), pp. 354–392.
[6] R. REUTER, *Solving tridiagonal systems of linear equations on the* IBM 3090 VF, Parallel Comput., 8 (1988), pp. 371–376.
[7] R. A. SWEET, *A generalized cyclic reduction algorithm*, SIAM J. Numer. Anal., 11 (1974), pp. 506–520.
[8] P. VU, personal communication, 1990.
[9] P. VU AND C. YANG, *Comparing tridiagonal solvers on the* CRAY X-MP/416 *system*, Cray Channels, 10-4 (1988), pp. 22–25.
[10] H. H. WANG, *A parallel method for tridiagonal equations*, ACM Trans. Math. Software, 7 (1981), pp. 170–183.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1255–1263, October 1992

017

A PROJECTIVE METHOD FOR RESCALING A DIAGONALLY STABLE MATRIX TO BE POSITIVE DEFINITE*

H. HU†

Abstract. For a diagonally stable matrix M, a projective method is presented that can find a positive diagonal matrix D such that $DM + M^TD$ is positive definite in finitely many iterations, and that provides an upper bound on the total number of iterations needed for finding such a D. The upper bound obtained is determined by the rescaling measure of positive definiteness of M. The method is extended to solve the problem of finding a positive diagonal matrix D such that $DM + M^TD$ is positive semidefinite if such a D exists.

Key words. positive definite, can be rescaled positive definite, rescaling measure of positive definiteness, orthogonal projection

AMS(MOS) subject classifications. 15A18, 15A24

1. Introduction.

A real square matrix M is said to be *diagonally stable* if there exists a positive diagonal matrix D such that $DM+M^TD$ is positive definite [4]. Such matrices are called "Lyapunov diagonally stable" in [7], "Volterra–Lyapunov stable" in [6], and "can be rescaled positive definite" in [8]. It is well known that diagonally stable matrices play an important role in economics and dynamic systems [1], [3], [4], [5], [7], [11]. Let x be a vector and $D(x)$ be the diagonal matrix with diagonal elements $x_1, \ldots, x_n$. A positive vector x is called a rescaling vector of M if $D(x)M + M^TD(x)$ is positive definite. So far, there are two general methods for finding a rescaling vector of a diagonally stable matrix M. The first method was given by Khalil [11]. He defines a function $g(x)$ to be the smallest eigenvalue of $D(x)M + M^TD(x)$. His method finds a positive x such that $g(x) > 0$ as follows: At iteration k, if $g(x^k) > 0$, then stop, because x^k is a rescaling vector of M; otherwise, find a direction d^k by solving a linear program and then construct x^{k+1} as a convex combination of x^k and d^k such that x^{k+1} is in the interior of the unit box of R^n. If M is diagonally stable, then his method will find a rescaling vector in finitely many iterations. If his method does not stop finitely, then M is not diagonally stable. The second method [8] approaches the problem in the following way: First transform the problem into an infinite system of linear inequalities and then solve the infinite system of linear inequalities by generating and solving a sequence of standard linear programs. The common drawbacks of these methods are that at the kth iteration a linear program with at least k columns needs to be solved and hence, in practice, the number of iterations that can be performed is limited, while no upper bound on the total number of iterations (when M is diagonally stable) has been found for these methods.

Herein we present a projective method for finding a rescaling vector of a diagonally stable matrix M. This method "solves" the same infinite system of linear inequalities as in [8], but each iteration of this method is very simple: The new iterate x^{k+1} is the orthogonal projection of the current iterate x^k on a plane constructed from a unit eigenvector corresponding to the smallest eigenvalue of $D(x^k)M + M^TD(x^k)$. Therefore, the amount of work at iteration k does not increase as k increases. Moreover,

*Received by the editors August 21, 1989; accepted for publication (in revised form) March 7, 1991.

†Department of Mathematical Sciences, Northern Illinois University, DeKalb, Illinois 60115-2888 (hu@math.niu.edu).

we are able to provide an upper bound on the total number of iterations needed for finding a rescaling vector of a diagonally stable matrix. The upper bound we obtained is determined by the rescaling measure of positive definiteness of M. This method is inspired by the orthogonal projection method for solving systems of linear inequalities [2], [9].

The remainder of this paper is organized as follows. In §2, we explain notation and definitions. In §3, we specify the method and prove the convergence. In §4, we provide an upper bound on the total number of iterations needed for finding a rescaling vector of a diagonally stable matrix. In §5, we present computational results. In §6, we extend the method to solve the problem of finding a positive diagonal matrix D such that $DM + M^TD$ is positive semidefinite if such a D exists.

2. Notation and definitions. First, let us extend the definition of a positive definite matrix. We define an $n \times n$ real matrix M, not necessarily symmetric, to be *positive definite* if $x^TMx > 0$ for all $0 \neq x \in R^n$, and to be *positive semidefinite* if $x^TMx \geq 0$ for all $x \in R^n$. By this definition, a matrix M is diagonally stable if and only if there exists a positive vector x such that $D(x)M$ is positive definite. As $D(x)M$ is the rescaling of the rows of M, we say that M *can be rescaled positive definite* if M is diagonally stable.

Let $S^{n-1} = \{x \in R^n : x^Tx = 1\}$ denote the unit sphere in R^n and $S_+^{n-1} = \{x \in S^{n-1} : x \geq 0\}$ denote the set of nonnegative vectors in S^{n-1}.

Let $\|x\|$ be the Euclidean norm of $x \in R^n$ and $\|M\| = (\sum_{i,j=1}^n m_{ij}^2)^{1/2}$ be the Frobenius norm of $M \in R^{n\times n}$. In particular, $\|D(x)\| = \|x\|$.

Given a real square matrix M, we define its *measure of positive definiteness* to be $\lambda[M] = \min\{u^TMu : u \in S^{n-1}\}$. For a real symmetric matrix M, $\lambda[M]$ equals the smallest eigenvalue of M and $\lambda[M] = V[M]^TMV[M]$ where $V[M]$ denotes a unit eigenvector corresponding to $\lambda[M]$. For a nonsymmetric matrix M, $\lambda[M] = \lambda[M + M^T]/2$.

Given a real square matrix M, we define its *rescaling measure of positive definiteness* to be

$$\beta[M] = \max_{x\in S_+^{n-1}} \min_{u\in S^{n-1}} u^TD(x)Mu = \max_{x\in S_+^{n-1}} \lambda[D(x)M].$$

It is known that M can be rescaled positive definite if and only if its rescaling measure of positive definiteness is positive [8].

Given a point $\bar{x} \in R^n$ and a hyperplane $a^Tx = b$, where a, $x \in R^n$ and $b \in R^1$, we say that $\bar{x}$ is on the right side of $a^Tx = b$ if $a^T\bar{x} \geq b$ and $\bar{x}$ is on the wrong side if $a^T\bar{x} < b$.

3. The method. If M can be rescaled positive definite, then $m_{ii} > 0$ for all $i = 1, 2, \ldots, n$. Hence, without loss of generality we may assume that all diagonal elements of M are positive in §§3–5. The following lemma transforms the problem into an infinite system of linear inequalities. The lemma becomes trivial if one knows the identity $(D(u)Mu)^Tx = u^TD(x)Mu$.

LEMMA 1. *M can be rescaled positive definite if and only if the infinite system of linear inequalities*

$$(D(u)Mu)^Tx \geq 1 \quad \textit{for all } u \in S^{n-1} \tag{I}$$

has a solution. Moreover, if $\bar{x}$ is any solution of (I), *then $\bar{x}_i \geq m_{ii}^{-1} > 0$ for all $i = 1, \ldots, n$ and $D(\bar{x})M$ is positive definite* [8].

Lemma 1 tells us that in order to find a rescaling vector, we only need to solve the infinite system of linear inequalities (I). The method that we present "solves" (I) by generating a sequence $\{x^k \in R^n : k = 0, 1, \dots\}$. If $D(x^k)M$ is positive definite, then the method stops; otherwise, the new iterate x^{k+1} is the orthogonal projection of x^k on $(D(u^k)Mu^k)^T x = 1$, where u^k is an approximation of a unit eigenvector corresponding to the smallest eigenvalue of the symmetric matrix $D(x^k)M + M^T D(x^k)$.

METHOD 1.

Step 0.

Let x^0 be a point in R^n such that $x_i^0 > 0$ for all $i = 1, \dots, n$;
let $0 \leq \epsilon < 1$ be a fixed small number;
let $k := 0$.

Step 1.

Find a λ^k such that $|\lambda^k - \lambda[D(x^k)M + M^T D(x^k)]| \leq \epsilon/2$;
if $\lambda^k > \epsilon/2$, then $D(x^k)M$ is positive definite, stop.
Otherwise, find a $u^k \in S^{n-1}$ such that
$\|u^k - V[D(x^k)M + M^T D(x^k)]\| \leq \epsilon/(4\|x^k\| \cdot \|M\|)$.

Step 2.

If $D(u^k)Mu^k \leq 0$, then M cannot be rescaled positive definite, stop.
Otherwise, let $x^{k+1} = x^k - \{(D(u^k)Mu^k)^T x^k - 1\}D(u^k)Mu^k/\|D(u^k)Mu^k\|^2$
be the orthogonal projection of x^k on $(D(u^k)Mu^k)^T x = 1$;
$k := k + 1$;
go to Step 1.

Comments on Method 1. (a) If $\lambda^k > \epsilon/2$, then $\lambda[D(x^k)M + M^T D(x^k)] > 0$, which implies that $D(x^k)M + M^T D(x^k)$ is positive definite and therefore $D(x^k)M$ is positive definite. Because $m_{ii} > 0$ for all i, we know that x^k must be a positive vector.

(b) As x^{k+1} is the projection of x^k on $(D(u^k)Mu^k)^T x = 1$, $(D(u^k)Mu^k)^T x^{k+1} = 1$ and thus $x^{k+1} \neq 0$ for all $k = 0, 1, \dots$.

(c) Since any solution of (I) is positive, if $D(u^k)Mu^k \leq 0$, then (I) has no solution and hence M cannot be rescaled positive definite.

THEOREM 1. *If M can be rescaled positive definite, then Method* 1 *can find a rescaling vector of M in a finite number of iterations.*

Proof. If the method does not stop at Step 1, then

$$\lambda[D(x^k)M + M^T D(x^k)] \leq \lambda^k + \epsilon/2 \leq \epsilon,$$

and

$$\begin{aligned}
&|\, {u^k}^T D(x^k)Mu^k - V[D(x^k)M + M^T D(x^k)]^T D(x^k)MV[D(x^k)M + M^T D(x^k)]\,| \\
&\quad \leq |\, {u^k}^T D(x^k)M(u^k - V[D(x^k)M + M^T D(x^k)])\,| \\
&\qquad + |\,(u^k - V[D(x^k)M + M^T D(x^k)])^T D(x^k)MV[D(x^k)M + M^T D(x^k)]\,| \\
&\quad \leq 2\|x^k\| \cdot \|M\| \cdot \|u^k - V[D(x^k)M + M^T D(x^k)]\| \\
&\quad \leq \epsilon/2.
\end{aligned}$$

Hence

$$\begin{aligned}
(D(u^k)Mu^k)^T x^k &= {u^k}^T D(x^k)Mu^k \\
&\leq V[D(x^k)M + M^T D(x^k)]^T D(x^k)MV[D(x^k)M + M^T D(x^k)] + \epsilon/2 \\
&= \lambda[D(x^k)M + M^T D(x^k)]/2 + \epsilon/2 \\
&\leq \epsilon < 1.
\end{aligned}$$

That is, if Method 1 does not stop at iteration k, then x^k is on the wrong side of $(D(u^k)Mu^k)^T x = 1$. It follows that

$$\begin{aligned} \|M\| \cdot \|x^{k+1} - x^k\| &\geq \|D(u^k)Mu^k\| \cdot \|x^{k+1} - x^k\| \\ &\geq (D(u^k)Mu^k)^T (x^{k+1} - x^k) \\ &\geq 1 - \epsilon. \end{aligned} \tag{1}$$

Let S be the solution set of (I) and $d(x) = \min\{\|x - y\| : y \in S\}$ be the Euclidean distance from x to S. It follows from the assumption of Theorem 1 and Lemma 1 that S is nonempty. Now suppose that the method generates an infinite sequence $\{x^k : k = 0, 1, \dots\}$. For any $y \in S$ we claim that for all $k = 0, 1, \dots,$

$$\|x^{k+1} - y\|^2 \leq \|x^k - y\|^2 - \|x^{k+1} - x^k\|^2. \tag{2}$$

Indeed, let α_k be the angle between $x^k - x^{k+1}$ and $y - x^{k+1}$. Because x^k is on the wrong side of $(D(u^k)Mu^k)^T x = 1$, y is on the right side, and x^{k+1} is the orthogonal projection of x^k on $(D(u^k)Mu^k)^T x = 1$, we know that $\alpha_k \geq \pi/2$. By the cosine law,

$$\begin{aligned} \|x^{k+1} - y\|^2 &= \|x^k - y\|^2 - \|x^{k+1} - x^k\|^2 + 2\|x^{k+1} - y\| \cdot \|x^{k+1} - x^k\| \cos\alpha_k \\ &\leq \|x^k - y\|^2 - \|x^{k+1} - x^k\|^2. \end{aligned}$$

For all $k = 0, 1, \dots,$ let y^k be the point in S nearest to x^k. Using (2) repeatedly, we have that for any fixed $k > 0$,

$$\|x^{n+k} - y^k\|^2 \leq \|x^k - y^k\|^2 = d(x^k)^2 \quad \text{for all } n = 0, 1, \dots . \tag{3}$$

That is, $\{x^{n+k} : n = 0, 1, \dots\}$ is contained in the closed ball $B(y^k, d(x^k))$ with center y^k and radius $d(x^k)$. Therefore, the sequence $\{x^k : k = 0, 1, \dots\}$ is bounded. From (2) and the fact that y^{k+1} is the point in S nearest to x^{k+1},

$$d(x^{k+1}) = \|x^{k+1} - y^{k+1}\| \leq \|x^{k+1} - y^k\| < \|x^k - y^k\| = d(x^k). \tag{4}$$

That is, x^k approaches S steadily. Moreover,

$$\begin{aligned} \|x^{k+1} - y^{k+1}\|^2 &\leq \|x^{k+1} - y^k\|^2 \\ &\leq \|x^k - y^k\|^2 - \|x^{k+1} - x^k\|^2 \\ &= \|x^k - y^k\|^2 \cdot \big(1 - \|x^{k+1} - x^k\|^2 / \|x^k - y^k\|^2\big). \end{aligned}$$

Now let $\tau_k \equiv \big(1 - \|x^{k+1} - x^k\|^2 / \|x^k - y^k\|^2\big)^{1/2}$ for all $k = 0, 1, \dots$; then $0 < \tau_k < 1$ and $\|x^{k+1} - y^{k+1}\| \leq \tau_k \|x^k - y^k\|$. It follows that

$$d(x^{k+1}) \leq \big(\Pi_{i=0}^k \tau_i\big) d(x^0). \tag{5}$$

As $\{\Pi_{i=0}^k \tau_i \geq 0 : k = 1, 2, \dots\}$ is a decreasing sequence, we discuss two cases.

Case 1. $\Pi_{i=0}^\infty \tau_i = 0$. By (5), $d(x^k) \to 0$, which implies that $\{x^k : k = 1, 2, \dots\}$ is a convergent sequence.

Case 2. $\Pi_{i=0}^\infty \tau_i > 0$. This implies that $\tau_i \to 1$. By the definition of τ_k and (4),

$$\|x^{k+1} - x^k\|^2 = (1 - \tau_k^2) d(x^k)^2 \leq (1 - \tau_k^2) d(x^0)^2,$$

and hence $\|x^{k+1} - x^k\| \to 0$.

As both cases contradict (1), we conclude that the method must stop after a finite number of iterations. □

4. Estimation of upper bound. Theorem 1 shows that if M can be rescaled positive definite, then Method 1 will find a positive x^k such that $D(x^k)M$ is positive definite after a finite number of iterations. In this section, we give an upper bound on the total number of iterations needed for finding such an x^k. We assume that $\epsilon = 0$; that is, we calculate the exact smallest eigenvalue and a corresponding unit eigenvector in Step 1. Let us call it Method 1′.

Let $H(x) = \max\{(1 - (D(u)Mu)^T x)^+ : u \in S^{n-1}\}$ be the biggest violation by x of (I). In particular, if $D(x)M$ is not positive definite, then

$$\begin{aligned} H(x) &= \max\{(1 - (D(u)Mu)^T x)^+ : u \in S^{n-1}\} \\ &= \max\{(1 - u^T D(x)Mu)^+ : u \in S^{n-1}\} \\ &= 1 - \lambda[D(x)M] \\ &= 1 - \lambda[D(x)M + M^T D(x)]/2. \end{aligned}$$

Let S and $d(x)$ be defined as before. Let $\{x^k : k = 0, 1, \dots\}$ be generated by Method 1′ and y^k be the point in S nearest to x^k.

LEMMA 2. *If there exists a unit vector $\hat{x}$ such that $(D(u)Mu)^T \hat{x} \geq \gamma > 0$ for all $u \in S^{n-1}$, then $d(x) \leq \gamma^{-1} H(x)$ for all $x \in R^n$* [10].

THEOREM 2. *If M can be rescaled positive definite and $\|M\| = 1$, then an upper bound on the number of iterations of Method 1′ is of order $\beta[M]^{-2}\ln(\beta[M]^{-1})$, where $0 < \beta[M] \leq 1$ is the rescaling measure of positive definiteness of M.*

Proof. Since $\lambda[D(x)M]$ is a continuous function of x and S_+^{n-1} is compact, we may assume that $\beta[M]$ is attained at $x^* \in S_+^{n-1}$. It follows that

$$1 \geq (D(u)Mu)^T x^* = u^T D(x^*)Mu \geq \beta[M] > 0 \quad \text{for all } u \in S^{n-1}.$$

By Lemma 2,

$$\|x^k - y^k\| = d(x^k) \leq \beta[M]^{-1} H(x^k) \quad \text{for all } k = 0, 1, \dots . \tag{6}$$

If Method 1′ does not stop at iteration k, then

$$\begin{aligned} \|x^{k+1} - x^k\| &= \frac{1 - (D(u^k)Mu^k)^T x^k}{\|D(u)Mu\|} \\ &\geq 1 - {u^k}^T D(x^k)Mu^k \\ &= 1 - \lambda[D(x^k)M + M^T D(x^k)]/2 \\ &= H(x^k) \\ &\geq \|x^k - y^k\|\beta[M]. \end{aligned} \tag{7}$$

As in the proof of Theorem 1, we still have

$$\|x^{k+1} - y^{k+1}\|^2 \leq \|x^k - y^k\|^2 - \|x^{k+1} - x^k\|^2. \tag{8}$$

Equations (6), (7), and (8) imply that

$$\|x^{k+1} - y^{k+1}\|^2 \leq \|x^k - y^k\|^2(1 - \beta[M]^2). \tag{9}$$

Using (9) repeatedly, we have $\|x^k - y^k\| \leq (1-\beta[M]^2)^{k/2}\|x^0 - y^0\|$. As x^{k+1} and x^k are contained in the closed ball with center y^k and radius $\|x^k - y^k\|$,

$$\begin{aligned} \|x^{k+1} - x^k\| &\leq 2(1-\beta[M]^2)^{k/2}\|x^0 - y^0\| \\ &\leq 2(1-\beta[M]^2)^{k/2}\beta[M]^{-1}H(x^0) \\ &= (1-\beta[M]^2)^{k/2}\beta[M]^{-1}(2-\lambda^0). \end{aligned} \tag{10}$$

Let $K = 2\ln(\beta[M]/(2-\lambda^0))/\ln(1-\beta[M]^2)$. It is easy to see that if Method 1′ does not stop at iteration k and $k > K$, then $\|x^{k+1} - x^k\| < 1$. On the other hand, if Method 1′ does not stop at iteration k, then $(D(u^k)Mu^k)^T(x^{k+1} - x^k) = 1 - {u^k}^T D(x^k)Mu^k \geq 1$ and hence $\|x^{k+1} - x^k\| \geq 1$. Therefore, the total number of iterations cannot exceed K. The theorem then follows from the fact that $\lim_{\beta[M]\to 0} K/(\beta[M]^{-2}\ln(\beta[M]^{-1})) = \frac{1}{2}$. □

Remarks. (a) Theorem 2 indicates that the amount of work needed for finding a rescaling vector is related to $\beta[M]$. For example, if $\beta[M] = 0.1$, then $\beta[M]^{-2}\ln(\beta[M]^{-1}) \doteq 230$. In practice, if the method does not stop after a large number of iterations, then we know that M cannot be rescaled positive definite or $\beta[M]$ is very small.

(b) There are additional ways to obtain an upper bound on the number of iterations. (1) If we have found a unit vector $\hat{x}$ such that $D(\hat{x})M$ is positive definite, then letting $\lambda = \lambda[D(\hat{x})M + M^T D(\hat{x})]/2$, we know that $\beta[M] \geq \lambda > 0$ and an upper bound on the number of iterations is of order $\lambda^{-2}\ln(\lambda^{-1})$. (2) If we know that there exists a solution $\bar{x}$ of (I) with $\|\bar{x}\| \leq \alpha$, then an upper bound on the number of iterations of Method 1′ is of order $\alpha^2 \ln\alpha$.

5. Computational results. We have coded the method in FORTRAN. We use EISPACK subroutine RS to calculate eigenvalues and eigenvectors, and we assume that RS can return "exact" eigenvalues and eigenvectors [12]. The input data were randomly generated and the program was executed on SUN 3/80.

First, we provide a step-by-step example of using the method to find a rescaling vector.

EXAMPLE.

$$M = \begin{pmatrix} 0.5020 & 0.4730 & 0.6128 \\ -0.1828 & 0.3038 & -0.0828 \\ -0.0804 & -0.0580 & 0.0797 \end{pmatrix}.$$

Iteration 0.

$x^0 = (1.0000, 1.0000, 1.0000)$,

$\lambda^0 = -0.1819$,

$u^0 = (-0.4518, 0.3149, 0.8347)$.

Iteration 1.

$x^1 = (-3.7970, 1.8417, 2.7288)$,

$\lambda^1 = -5.6399$,

$u^1 = (-0.8760, -0.2940, -0.3823)$.

Iteration 2.

$x^2 = (1.5508, 1.6155, 2.5651)$,

$\lambda^2 = -0.1405$,

$u^2 = (-0.4591, 0.3797, 0.8072)$.

Iteration 3.

$x^3 = (-3.0156, 2.7400, 3.9989)$,

$\lambda^3 = -4.6355$,

$u^3 = (-0.8742, -0.2955, -0.3854)$.

Iteration 4.

$x^4 = (1.6299, 2.5435, 3.8564)$,

$\lambda^4 = 0.0793$.

Additional information.

$\lambda^{-2}\ln(\lambda^{-1}) = 73558$, and $\alpha^2\ln(\alpha) = 15743$.

We have solved a number of problems in different dimensions (see Table 1). Based on our computational experience the method seems efficient. The theoretical (order of) upper bounds $\lambda^{-2}\ln(\lambda^{-1})$ and $\alpha^2\ln(\alpha)$ mentioned in the remarks of the last section are very loose and are omitted.

In addition, we have tried a heuristic approach to accelerating the convergence. Instead of projecting x^k on $(D(u^k)Mu^k)^T x = 1$ as in Method 1, we now project x^k on $(D(u^k)Mu^k)^T x = \lambda[D(x^k)M + M^T D(x^k)]$. We have tested this idea using the same data as the problems in Table 1, and it seems that this heuristic approach is more efficient than Method 1 (see Table 2).

Finally, we have implemented the method proposed in [8]. The method is also coded in FORTRAN and executed on SUN 3/80 using the same data as the problems in Table 1 (see Table 3). Based on our computational experience, the heuristic version of the projective method seems most efficient.

TABLE 1

Problem dimension	5×5	6×6	7×7	8×8	9×9
No. of iterations	27	15	32	50	30
CPU time (sec.)	5.28	4.92	14.16	30.40	25.48

TABLE 2

Problem dimension	5×5	6×6	7×7	8×8	9×9
No. of iterations	14	3	11	12	14
CPU time (sec.)	2.80	1.32	5.20	7.64	12.06

TABLE 3

Problem dimension	5×5	6×6	7×7	8×8	9×9
No. of iterations	3	3	7	6	11
CPU time (sec.)	17.48	19.64	45.38	42.54	89.34

6. Rescaling a matrix positive semidefinite. In this section, we discuss how to rescale a matrix positive semidefinite, i.e., how to construct a positive diagonal matrix D such that DM is positive semidefinite if it exists. Such matrices are called "Lyapunov diagonally semistable" in [7]. Without loss of generality, we assume that the diagonal elements of M are nonnegative. It is not difficult to see that M can be

rescaled positive semidefinite if and only if the infinite system of linear inequalities

$$\text{(II)} \qquad \begin{aligned} (D(u)Mu)^T x &\geq 0 \quad \text{for all } u \in S^{n-1}, \text{ and} \\ {e^i}^T x &\geq 1 \quad \text{for all } i = 1, \ldots, n \end{aligned}$$

has a solution, where e^i is the ith unit vector in R^n, and if $\hat{x}$ is a solution of (II), then $D(\hat{x})M$ is positive semidefinite.

METHOD 2.

Step 0.

Let x^0 be a point in R^n such that $x_i^0 \geq 1$ for all $i = 1, \ldots, n$;
let $k := 0$.

Step 1.

Find $\lambda^k = \lambda[D(x^k)M + M^T D(x^k)]$;
if $\lambda^k \geq 0$ and $x_i^k > 0$ for all $i = 1, \ldots, n$,
then $D(x^k)M$ is positive semidefinite and x^k is positive, stop.

Step 2.

Find $u^k = V[D(x^k)M + M^T D(x^k)]$;
find ${e^j}^T x^k - 1 = \min\{{e^i}^T x^k - 1 : i = 1, \ldots, n\}$;
if $(D(u^k)Mu^k)^T x^k \leq {e^j}^T x^k - 1$,
then let $x^{k+1} = x^k - \{(D(u^k)Mu^k)^T x^k\} D(u^k)Mu^k / \|D(u^k)Mu^k\|^2$
be the orthogonal projection of x^k on $(D(u^k)Mu^k)^T x = 0$;
otherwise, let $x^{k+1} = x^k - \{{e^j}^T x^k - 1\} e^j$
be the orthogonal projection of x^k on ${e^j}^T x = 1$;
$k := k + 1$;
go to Step 1.

THEOREM 3. *If M can be rescaled positive semidefinite and Method 2 does not stop finitely, then the sequence $\{x^k : k = 0, 1, \ldots\}$ generated by Method 2 converges to a solution of* (II).

The proof of Theorem 3 is similar to that of Theorem 1 and is omitted. Finally, we point out that if (I) has a solution, then the interior of the solution set S is nonempty. This property enables us to calculate inexact eigenvalues and eigenvectors in Step 1 of Method 1, to prove the finiteness of Method 1, and to obtain an upper bound on the number of iterations of Method 1′ in the case where M can be rescaled positive definite. Because (II) does not have this property, we can only prove the convergence of Method 2 under the assumption that exact eigenvalues and eigenvectors are calculated in Steps 1 and 2.

Acknowledgment. I wish to thank Professor G. B. Dantzig for his suggestions, which helped to improve the bound in Theorem 2.

REFERENCES

[1] K. J. ARROW AND M. MCMANUS, *A note on dynamic stability*, Econometrica, 26 (1958), pp. 448–454.

[2] S. AGMON, *The relaxation method for linear inequalities*, Canad. J. Math., 6 (1954), pp. 382–392.

[3] M. ARAKI, *Application of M-matrices to stability problems of composite dynamical systems*, J. Math. Anal. Appl., 52 (1975), pp. 309–321.

[4] G. P. BARKER, A. BERMAN, AND R. J. PLEMMONS, *Positive diagonal solutions to the Lyapunov equations*, Linear and Multilinear Algebra, 5 (1978), pp. 249–256.

[5] A. BERMAN AND D. HERSHKOWITZ, *Characterization of acyclic D-stable matrices*, Linear Algebra Appl., 58 (1984), pp. 17–32.
[6] G. W. CROSS, *Three types of matrix stability*, Linear Algebra Appl., 20 (1978), pp. 253–263.
[7] D. HERSHKOWITZ AND H. SCHNEIDER, *Scalings of vector spaces and the uniqueness of Lyapunov scaling factors*, Linear and Multilinear Algebra, 17 (1985), pp. 203–226.
[8] H. HU, *An algorithm for rescaling a matrix positive definite*, Linear Algebra Appl., 96 (1987), pp. 131–147.
[9] ———, *Projection methods for solving infinite systems of linear and convex inequalities*, preprint.
[10] H. HU AND Q. WANG, *On approximate solutions of infinite systems of linear inequalities*, Linear Algebra Appl., 114/115 (1989), pp. 429–438.
[11] H. K. KHALIL, *On the existence of positive diagonal P such that* $PA + A^T P < 0$, IEEE Trans. Automat. Control, 27 (1982), pp. 181–184.
[12] B. T. SMITH, J. M. BOYLE, Y. IKEBE, V. C. KLEMA, AND C. B. MOLER, *Matrix Eigensystem Routines Guide*, Springer-Verlag, Berlin, 1970.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1264–1278, October 1992

018

ADAPTIVE CONDITION ESTIMATION FOR RANK-ONE UPDATES OF QR FACTORIZATIONS*

GAUTAM M. SHROFF† AND CHRISTIAN H. BISCHOF‡

Abstract. Many applications involve repeatedly solving linear systems of equations while the coefficient matrix is modified by a rank-one matrix at each iteration. The QR factorization is often used in such situations, and algorithms that update the QR factorizations in $O(n^2)$ time are well known. To avoid excessive round-off error in solving equation systems, it is useful to monitor the condition number of the matrix as the iterations progress. In this paper, *g*eneral (i.e., nonsymmetric) matrices undergoing *r*ank-one changes are considered and an *a*daptive *c*ondition *e*stimation algorithm, "GRACE," is developed to monitor the condition number of the matrices during the update process. The algorithm requires only $O(n)$ overhead beyond the cost of updating the QR factorization. Potential numerical difficulties in the algorithm are analyzed and modifications to overcome these are introduced. These modifications are also applicable to the ACE algorithm of Pierce and Plemmons that handles symmetric updates. Finally, experimental results are presented that demonstrate that GRACE works well in practice.

Key words. adaptive condition estimation, updating, orthogonal factorization

AMS(MOS) subject classifications. 15A12, 15A23, 65F25, 65F35

1. Introduction.

In certain applications we are interested in repeatedly solving systems of linear equations

$$A_k x_k = b_k, \tag{1}$$

where the matrix is modified by low rank updates, typically rank-one updates, at each iteration, i.e.,

$$A_{k+1} = A_k + w_k v_k^T \quad \text{where } A \in \mathcal{R}^{n\times n} \quad \text{and} \quad w_k, v_k \in \mathcal{R}^{n\times 1}. \tag{2}$$

We deal with the problem of adaptively estimating the condition numbers of the matrices A_k. By "adaptive" we mean that we use the estimate at iteration k to compute the estimate at step $k+1$.

When the A_ks are symmetric and positive definite, such as in recursive least squares applications, the linear systems are usually solved via the Cholesky factorization $A_k = R^T R$. Efficient $O(n^2)$ algorithms that update the Cholesky factorization in such situations are well known. Adaptive condition estimation in this setting has been the subject of recent work by Pierce and Plemmons [8] as well as by Ferng, Golub, and Plemmons [3].

If, on the other hand, the A_ks are symmetric indefinite or in general nonsymmetric, such as in quasi-Newton methods for optimization (e.g., Broyden's method), other techniques are usually used for solving the systems involved. One approach would be to note that a rank-one update to A_k can be written as two symmetric rank-one updates to $A_k^T A_k$. If we are willing to solve the linear systems (1) using the Cholesky factorization of $A_k^T A_k$, i.e., the "normal equations," we can update the solutions using

* Received by the editors August 6, 1990; accepted for publication (in revised form) December 10, 1990. This work was supported by the Applied Mathematical Sciences subprogram of the Office of Energy Research, U. S. Department of Energy, under contract W-31-109-Eng-38.

† Applied Mathematics 217-50, California Institute of Technology, Pasadena, California 91125 (gms@ama-1.caltech.edu).

‡ Mathematics and Computer Science Division, Argonne National Laboratory, 9700 S. Cass Ave., Argonne, Illinois 60439 (bischof@mcs.anl.gov).

the same algorithms as for recursive least squares. For condition estimation in this case we could simply treat the problem as two symmetric rank-one updates to $A_k^T A_k$ and use Pierce and Plemmons's adaptive condition estimation (ACE) algorithm twice.

However, it is well known that solving linear systems by the normal equations is relatively less stable since the error is governed by the *square* of the condition number, $(\kappa(A_k))^2$, rather than $\kappa(A_k)$. This is why QR factorizations of A_k are often used instead. Algorithms for updating the QR factorization of general matrices under rank-one updates are also well known [6], and one such algorithm is described in §2. Our goal is to adaptively estimate the condition number of the A_k's in such situations.

The QR updating algorithm requires $O(n^2)$ operations and so we would like our condition estimation algorithm to require only an insignificant amount of overhead, i.e., $O(n)$ at most. (This is similar in spirit to the ACE algorithm of Pierce and Plemmons for the symmetric positive definite case, since their algorithm requires only $O(n)$ extra work as well. The algorithm of Ferng, Golub, and Plemmons, on the other hand, requires $O(n^2)$ extra work.) Our algorithm will fulfill this requirement by utilizing information obtained as a by-product of the QR updating process.

In §2 we briefly review the algorithm for updating the QR factorization. In §3 we derive our *g*eneral *r*ank-one *a*daptive *c*ondition *e*stimation algorithm, which we call GRACE. We discuss the numerical difficulties involved and suggest ways to overcome them. It turns out that some of the modifications introduced for numerical stability will also prevent the breakdown of the ACE algorithm [7], [8]. In §4 we describe the results of numerical experiments using the proposed algorithm. Finally, we present our conclusions in §5.

2. Modifying the QR factorization. An algorithm for updating the QR factorization of a matrix modified by a rank-one matrix is described in Golub and Van Loan [6] and is based on the work of Gill, Golub, Murray, and Saunders [5]. We briefly review this algorithm below.

Given $B = QR$, the QR factorization of $B \in \mathcal{R}^{n\times n}$, we wish to compute the QR factorization of $B + wv^T$. But $B + wv^T = Q(R + uv^T)$ where $u = Q^T w$, thereby reducing the problem to determining the factorization of $R + uv^T$.

We will use $J(i, j, \theta)$ to denote a Givens rotation in the (i, j) plane, i.e., the orthogonal matrix which differs from the identity only in the (i, i), (i, j), (j, i), and (j, j) positions, where it is:

$$\begin{pmatrix} J_{ii} & J_{ij} \\ J_{ji} & J_{jj} \end{pmatrix} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}.$$

The first step is to compute $n-1$ Givens rotations $G = J(1, 2, \nu_1)\cdots J(n-1, n, \nu_{n-1})$ such that Gu is a multiple of e_1, the first column of an identity matrix. We have

$$G \begin{pmatrix} x & x & x & x & & x \\ & x & x & x & & x \\ & & x & x & + & x \\ & & & x & & x \end{pmatrix} \begin{matrix} \\ \\ \\ \\ x & x & x & x \end{matrix} = \begin{pmatrix} x & x & x & x \\ x & x & x & x \\ & x & x & x \\ & & x & x \end{pmatrix} \tag{3}$$

$$R + uv^T \qquad\qquad H + \eta e_1 v^T = \tilde{H}$$

where H and $\tilde{H}$ are upper Hessenberg matrices. Next a second set of $n-1$ Givens rotations $U = J(n-1, n, \mu_1)\cdots J(1, 2, \mu_{n-1})$ is computed to restore $\tilde{H}$ to upper

triangular form:

$$U \begin{pmatrix} x & x & x & x \\ x & x & x & x \\ & x & x & x \\ & & x & x \end{pmatrix} = \begin{pmatrix} x & x & x & x \\ & x & x & x \\ & & x & x \\ & & & x \end{pmatrix}. \tag{4}$$

$$\tilde{H} \qquad\qquad R_{\text{new}}$$

The updated QR factorization is $B + wv^T = QG^TU^TR_{\text{new}} = Q_{\text{new}}R_{\text{new}}$, and $13n^2$ flops[1] are required to compute Q_{new} and R_{new}.

3. General rank-one adaptive condition estimation. Recall that we are considering situations in which a succession of rank-one updates are being performed, and at each step the QR factorization of the current iterate is being updated using the algorithm of the previous section.

Our purpose is to inexpensively monitor the extreme singular values, and consequently the condition number, of the matrix at each step of this process while performing only $O(n)$ extra work in addition to the $O(n^2)$ work required in the QR update itself.

Condition estimation algorithms usually exploit the relationship

$$x^TR = d^T \Longrightarrow \sigma_{\min}(R) \leq \frac{\|d\|_2}{\|x\|_2} \leq \sigma_{\max}(R). \tag{5}$$

In particular, if x is a left singular vector of R corresponding to the smallest singular value of R (in the future we will call such a vector the "smallest left singular vector" of R), then the lower bound in (5) is achieved. Similarly the upper bound is achieved if x is the "largest left singular vector" of R.

We will maintain vectors $x_{\min}$ and $x_{\max}$, each of unit norm, which will be approximations to the smallest and largest left singular vectors of R, the triangular factor of the current iterate B. We also maintain the numbers $\|R^Tx_{\min}\|_2$ and $\|R^Tx_{\max}\|_2$ as approximations to the singular values of R (and also of B, since $\sigma(B) = \sigma(R)$). We derive below the algorithm that will give us new approximate singular vectors and singular value estimates for R_{new}, the triangular factor of $B + wv^T$.

3.1. Derivation. For the moment we restrict the discussion to updating only the smallest singular vector. In the ACE algorithm of Pierce and Plemmons [7], [8], which considers updating the QR factorization when a row is appended to the matrix, the relationship between the new approximate singular vector and the old approximation is of the form

$$\begin{pmatrix} x_{\text{new}} \\ \rho \end{pmatrix} = V \begin{pmatrix} \alpha x_{\text{old}} \\ \beta \end{pmatrix}, \tag{6}$$

where V is the orthogonal matrix used in the update. Their algorithm determines the parameters α, β, and ρ to minimize $\|R_{\text{new}}^Tx_{\text{new}}\|_2$ while constraining x_{new} to be of unit norm.

In our algorithm we are considering general rank-one updates and consequently a simple relationship like (6) does not suffice; a more complicated relation between x_{new} and x_{old} is required.

[1] One *flop* consists of one floating point multiplication and one floating point addition.

Let us assume we are given x_{old}, an approximate left smallest singular vector for R, the triangular factor of B, as well as $d_{\text{old}} = R^T x_{\text{old}}$. We first outline the basic steps of GRACE before giving the details.

Step 1. Let R be partitioned as

$$R = \begin{pmatrix} R_1 & r \\ 0 & \zeta \end{pmatrix} \quad \text{where } R_1 \in \mathcal{R}^{n-1 \times n-1}. \tag{7}$$

Then form x, an approximate singular vector for R_1, from the first $n-1$ components of x_{old} by scaling; also compute $d = R_1^T x$:

$$x = \frac{x_{\text{old}(1:n-1)}}{\|x_{\text{old}(1:n-1)}\|_2} \quad \text{and} \quad d = \frac{d_{\text{old}(1:n-1)}}{\|x_{\text{old}(1:n-1)}\|_2}. \tag{8}$$

Step 2. Recall the first stage of the updating algorithm of §2. Rewrite the first $n-1$ columns of (3) as

$$G \begin{pmatrix} R_1 \\ 0^T \end{pmatrix} = \begin{pmatrix} y^T \\ \tilde{R} \end{pmatrix}, \quad \text{and define } \tilde{x} \text{ by } \begin{pmatrix} \theta \\ \tilde{x} \end{pmatrix} = G \begin{pmatrix} \alpha x \\ \beta \end{pmatrix}, \tag{9}$$

where α, β, and θ are parameters to be determined. Note that $\tilde{R}$ is just the lower left piece of the Hessenberg matrix H in (3) and is therefore upper triangular. Also, y^T consists of the leading $n-1$ components of the first row of H.

From (9) it follows that

$$\tilde{R}^T \tilde{x} = \alpha R_1^T x - \theta y. \tag{10}$$

The motivation for this step is that we can obtain an approximate singular vector $\tilde{x}$ for $\tilde{R}$ by minimizing $\|\tilde{R}^T \tilde{x}\|_2$ while constraining $\|\tilde{x}\|_2 = 1$. This can be viewed as using the ACE technique of Pierce and Plemmons "backwards," since we are removing the row y^T rather than appending it.

Step 3. We now consider the second stage of the QR updating algorithm. Let $\tilde{y} = y + \eta v_{(1:n-1)}$, i.e., y^T consists of the leading $n-1$ components of the first row of $\tilde{H}$ in (3).

We rewrite the first $n-1$ columns of (4) as

$$U \begin{pmatrix} \tilde{y}^T \\ \tilde{R}^T \end{pmatrix} = \begin{pmatrix} \hat{R}^T \\ 0^T \end{pmatrix}, \quad \text{and define } \tilde{x} \text{ by } \begin{pmatrix} \hat{x} \\ \phi \end{pmatrix} = U \begin{pmatrix} \gamma \\ \delta \tilde{x} \end{pmatrix}, \tag{11}$$

with undetermined parameters γ, δ, and ϕ. Note that $\hat{R}$ is just the $n-1 \times n-1$ leading principal submatrix of R_{new} in (4) and is therefore upper triangular.

The following relationship now follows:

$$\hat{R}^T \hat{x} = \delta R_1^T x + \gamma \tilde{y}. \tag{12}$$

So we can obtain an approximate singular vector $\hat{x}$ for $\hat{R}$ by performing another constrained minimization. (This is similar to an ACE step where a row $\tilde{y}^T$ is appended to the matrix $\tilde{R}$.)

Step 4. We have obtained an approximate singular vector for $\hat{R}$, but we need to obtain one for

$$R_{\text{new}} = \begin{pmatrix} \hat{R} & \hat{r} \\ 0^T & \hat{\zeta} \end{pmatrix}, \quad \text{where } \begin{pmatrix} \hat{r} \\ \hat{\zeta} \end{pmatrix} = UG \begin{pmatrix} r \\ \zeta \end{pmatrix} \tag{13}$$

is the last column of the updated triangular factor, R_{new}.

To this end we apply the technique of "incremental condition estimation" or ICE, due to Bischof [1]. Using ICE we form the estimated singular vector for R_{new} as

$$x_{\text{new}} = \begin{pmatrix} \hat{x} \sin\psi \\ \cos\psi \end{pmatrix}, \tag{14}$$

determining the angle ψ to minimize $\|R_{\text{new}}^T x_{\text{new}}\|_2$ while restricting $\|x_{\text{new}}\|_2 = 1$.

To summarize, then, we first obtain an approximate singular vector for R_1 by truncation, for $\tilde{R}$ by a "backward" ACE step, for $\hat{R}$ by an ordinary ACE step, and finally for R_{new} using an ICE step. We now turn to the implementation details, including numerical difficulties and modifications of the basic algorithm that overcome them.

3.2. Implementation. The GRACE algorithm may be implemented as described above, but we can compress the two ACE-like steps (i.e., Steps 2 and 3) into one, giving rise to one constrained minimization problem with three unknowns instead of two problems with two unknowns each. We describe this formulation first and then briefly mention the alternative formulas as well.

Steps 2 and 3 ($x \longrightarrow \hat{x}$). We are given x and $d = R_1^T x$, and we wish to obtain $\hat{x}$, by arriving at values for the unknown parameters α, β, δ, and γ. Let (q_1^T, q_2) be the first row of G, (u_1, u_2^T) the last row of U, and (w_1^T, w_2) the last row of $W = UG$. So

(15)

$$G = \begin{pmatrix} q_1^T & q_2 \\ \square & \square \end{pmatrix}, \quad U = \begin{pmatrix} \square & \square \\ u_1 & u_2^T \end{pmatrix},$$

$$W = UG = \begin{pmatrix} \square & \square \\ w_1^T & w_2 \end{pmatrix}.$$

We note that since G and U are each products of $n-1$ Givens rotations, all these rows can be accumulated in $O(n)$ time.

The following relations involving the unknown parameters α, β, θ, γ, δ, ϕ, and the intermediate vector $\tilde{x}$ follow from (9) and (11):

$$\theta = \alpha q_1^T x + \beta q_2, \tag{16}$$

$$\phi = \gamma u_1 + \delta u_2^T \tilde{x}, \tag{17}$$

and

$$\alpha w_1^T x + \beta w_2 = \theta u_1 + u_2^T \tilde{x}. \tag{18}$$

Now we proceed to set up the minimization problem. Using (10), (12), and (16) we get

$$\hat{R}^T \hat{x} = \gamma \tilde{y} + \delta\alpha(R_1^T x - y q_1^T x) - \delta\beta q_2 y. \tag{19}$$

We already know $d = R_1^T x$. Define $b = d - yq_1^T x$, and introduce the new variables $\hat{\alpha} = \delta\alpha$ and $\hat{\beta} = \delta\beta$. Rewriting (19) we have

$$\hat{R}^T \hat{x} = \gamma\tilde{y} + \hat{\alpha} b - \hat{\beta} q_2 y. \tag{20}$$

Therefore

$$\|\hat{R}^T\hat{x}\|_2^2 = \Psi^T \underbrace{\begin{pmatrix} \tilde{y}^T\tilde{y} & \tilde{y}^T b & -q_2\tilde{y}^T y \\ \tilde{y}^T b & b^T b & -q_2 b^T y \\ -q_2\tilde{y}^T y & -q_2 b^T y & q_2^2 y^T y \end{pmatrix}}_{M} \Psi, \quad \text{where } \Psi = \begin{pmatrix} \gamma \\ \hat{\alpha} \\ \hat{\beta} \end{pmatrix}. \tag{21}$$

We need to minimize the above quadratic form $\Psi^T M \Psi$ subject to the constraint that $\|\hat{x}\| = 1$. From (11) we have

$$\|\hat{x}\|_2^2 = \gamma^2 + \delta^2\|\tilde{x}\|_2^2 - \phi^2. \tag{22}$$

Since $\|x\|_2 = 1$, (9) gives

$$\|\tilde{x}\|_2^2 = \alpha^2 + \beta^2 - \theta^2.$$

Using this in (22) and after some algebra (which we omit) involving the liberal use of equations (16)–(18), we can express $\|\hat{x}\|_2^2$ also as a quadratic form in Ψ,

$$\|\hat{x}\|_2^2 = \Psi^T \underbrace{\begin{pmatrix} 1 - u_1^2 & -u_1 z_1 & -u_1 z_2 \\ -u_1 z_1 & 1 - (q_1^T x)^2 - z_1^2 & -z_2 z_1 - q_2 q_1^T x \\ -u_1 z_1 & -z_2 z_1 - q_2 q_1^T x & 1 - q_2^2 - z_2^2 \end{pmatrix}}_{N} \Psi, \tag{23}$$

where z_1 and z_2 are defined by

$$z_1 = w_1^T x - u_1 q_1^T x \quad \text{and} \quad z_2 = w_2 - u_1 q_2. \tag{24}$$

Minimizing $\Psi^T M \Psi$ subject to $\Psi^T N \Psi = 1$ is equivalent to solving the generalized eigenvalue problem (see Golub and Van Loan [6])

$$M\Gamma = \lambda N \Gamma. \tag{25}$$

The matrices M and N above are symmetric and *semidefinite*. For the moment we will assume that N is positive definite, postponing to a later section discussion of the "singular case" when N is singular or ill conditioned. If N is nonsingular, the pencil (25) is *symmetric definite* and we can always determine three generalized eigenvalues λ and three corresponding eigenvectors Γ. Choosing Γ to be the generalized eigenvector corresponding to the minimum generalized eigenvalue, the vector

$$\Psi = \frac{\Gamma}{\sqrt{\Gamma^T N \Gamma}} \tag{26}$$

minimizes $\Psi^T M \Psi$ with $\Psi^T N \Psi = 1$.

Having computed Ψ, we have γ, $\hat{\alpha}$, and $\hat{\beta}$. The parameter δ is determined by normalizing the intermediate vector $\tilde{x}$,

$$\|\tilde{x}\|_2 = \alpha^2 + \beta^2 - \theta^2 = 1, \tag{27}$$

so that

$$\delta^2 = \hat{\alpha}^2 + \hat{\beta}^2 - \delta^2\theta^2. \tag{28}$$

Writing θ in terms of α and β from (16) we get

$$\delta^2 = (\hat{\alpha} \quad \hat{\beta}) \begin{pmatrix} 1-(q_1^T x)^2 & -q_2 q_1^T x \\ -q_2 q_1^T x & 1-q_2^2 \end{pmatrix} \begin{pmatrix} \hat{\alpha} \\ \hat{\beta} \end{pmatrix}. \tag{29}$$

Now we can compute $\tilde{x}$ from (9) and the desired vector $\hat{x}$ from (11). Once again, since G and U are products of Givens rotations, this requires only $O(n)$ work. The vector $\hat{d} = \hat{R}^T\hat{x}$ is obtained from (20), again with only $O(n)$ work.

Steps 2 and 3 (alternative formulation). Before we proceed to the next step of the algorithm, we mention that Steps 2 and 3 could have been performed independently instead of together as we have above. The following two minimization problems would be involved in such an alternative formulation; we state the formulas below since they will be useful later when discussing the "singular case."

In Step 2, the generalized eigenvalue problem

$$\underbrace{\begin{pmatrix} b^T b & -q_2 b^T y \\ -q_2 b^T y & q_2^2 y^T y \end{pmatrix}}_{M_1} \Gamma_1 = \lambda_1 \underbrace{\begin{pmatrix} 1-(q_1^T x)^2 & -q_2 q_1^T x \\ -q_2 q_1^T x & 1-q_2^2 \end{pmatrix}}_{N_1} \Gamma_1 \tag{30}$$

is solved and we obtain α and β from the smallest eigenvector as

$$\begin{pmatrix} \alpha \\ \beta \end{pmatrix} = \frac{\Gamma_1}{\sqrt{\Gamma_1^T N_1 \Gamma_1}}.$$

In Step 3, the problem is

$$\underbrace{\begin{pmatrix} \tilde{y}^T\tilde{y} & \tilde{y}^T\tilde{d} \\ \tilde{y}^T\tilde{d} & \tilde{d}^T\tilde{d} \end{pmatrix}}_{M_2} \Gamma_2 = \lambda_2 \underbrace{\begin{pmatrix} 1-u_1^2 & -u_1 u_2^T\tilde{x} \\ -u_1 u_2^T\tilde{x} & 1-(u_1^T\tilde{x})^2 \end{pmatrix}}_{N_2} \Gamma_2, \tag{31}$$

where $\tilde{d} = \tilde{R}^T\tilde{x} = \alpha b - \beta q_2 y$ and γ and δ are given by

$$\begin{pmatrix} \gamma \\ \delta \end{pmatrix} = \frac{\Gamma_2}{\sqrt{\Gamma_2^T N_2 \Gamma_2}}.$$

Step 4 ($\hat{x} \longrightarrow x_{\text{new}}$)**.** This step uses the ICE technique of Bischof [1]. From (13) and (14) we have

$$R_{\text{new}}^T x_{\text{new}} = \begin{pmatrix} \hat{R}^T\hat{x}\sin\psi \\ \hat{r}^T\hat{x}\sin\psi + \hat{\zeta}\cos\psi \end{pmatrix}. \tag{32}$$

So we can write

$$\|R_{\text{new}}^T\|_2^2 = (\sin\psi \quad \cos\psi) \underbrace{\begin{pmatrix} \hat{d}^T\hat{d} + (\hat{r}^T\hat{x}) & \hat{\zeta}(\hat{r}^T\hat{x})^2 \\ \hat{\zeta}(\hat{r}^T\hat{x}) & \hat{\zeta}^2 \end{pmatrix}}_{S} \begin{pmatrix} \sin\psi \\ \cos\psi \end{pmatrix}. \tag{33}$$

This quadratic form is minimized by choosing $(\sin\psi \quad \cos\psi)^T$ to be the eigenvector corresponding to the minimum eigenvalue of the *symmetric semidefinite* matrix S. Once we have ψ, x_{new} is given by (14) and $d_{\text{new}} = R_{\text{new}}^T x_{\text{new}}$ is obtained from (32).

This completes the description of the algorithm for updating the approximate smallest singular vector. The algorithm for updating the largest singular vector is almost identical, except that all *minimizations* are replaced by *maximizations*. The extra work involved in the condition estimation can now be calculated; of course, we only consider the $O(n)$ terms. Also, we consider updating both approximate singular vectors, $x_{\max}$ and $x_{\min}$, since some of the work can be shared (i.e., computing q_1, u_2, and w_1).

Step 1:	two norms, four scalings,	$6n$
Step 2:	q_1, u_2, and w_1,	$12n$
	b, and all inner products,	$13n$
	$\tilde{x}$, $\hat{x}$, and $\hat{d}$,	$18n$
Step 3:	$\hat{r}^T\hat{x}$, x_{new} and d_{new},	$6n$
		$55n$

So, the total cost of updating the QR factorization and maintaining the adaptive condition estimates is $13n^2+55n+O(1)$. We now turn to a discussion of the numerical issues involved.

3.3. Numerical issues. In the above description of the algorithm, we have avoided discussion of potential numerical difficulties. We will now identify the potential problem cases and suggest ways to overcome them.

3.3.1. Step 1 can fail. Clearly, if $x_{\text{old}} \approx e_n$, i.e., if the first $n-1$ components of x_{old} are very small or zero, then $\|x_{\text{old}(1:n-1)}\|_2 \approx 0$ and Step 1 breaks down. In this case our algorithm will *fail* and we cannot proceed further. However, consider when such a situation may be expected to arise. If e_n is an approximate *smallest* singular vector, then $R_{(n,n)}$ is in fact a very good estimate for $\sigma_{\min}(R)$. It is likely then that $R_{(n,n)}$ is much smaller than the rest of R and that R is almost singular. (Of course, examples can be constructed when this is not true, for example, the identity matrix. However, it may be expected to hold in the generic case.) Our condition estimation is not really supposed to be working with singular matrices, but is supposed to identify cases when the matrix *becomes* almost singular after an update. We note that if R is well conditioned before the update and becomes singular after the update, the algorithm will work, producing $x_{\text{new}} = e_n$, but it will not work for future updates. In practice, we did not encounter the above problem in any random situation and do not consider it "likely" to occur except in the manner indicated. In the code we check for this situation and insert a call to some other condition estimator such as the one suggested by Bischof [1] or the one due to Cline, Conn, and Van Loan [2], even though these are both $O(n^2)$ estimators.

3.3.2. Singular case, $\det(N) \approx 0$. If the matrix N in the generalized eigenvalue problem (25) is singular or ill conditioned, some of the generalized eigenvalues may be infinite or undefined (i.e., the eigenvalue problem may be ill posed). In such a situation there are two separate numerical problems. The first is to accurately compute the generalized eigenvectors. We do not discuss this problem in detail here since it is covered elsewhere, for example, in the text by Golub and Van Loan [6] and in the paper by Fix and Heiberger [4]. The second problem arises because the eigenvectors Γ corresponding to these "bad" eigenvalues are such that $\Gamma^T N \Gamma$ is zero or very small, rendering the computation of Ψ in (26) unstable.

To investigate when N can be singular, it is instructive to look at the alternative formulation in terms of two 2×2 eigenvalue problems involving the pencils (M_1, N_1) and (M_2, N_2), i.e., (30) and (31). This is justified because if N_1 is singular, a solution $\tilde{x} = 0$ exists and therefore $\gamma = 0$ yields a solution $\hat{x} = 0$, so N must be singular. Alternatively, if N_2 is singular but not N_1, again we have a solution $\hat{x} = 0$ and N is singular. Conversely, if N is singular, either $\tilde{x} = 0$ or not, and one of the above situations must be true.

Consider the matrix N_1,

$$N_1 = \begin{pmatrix} 1 - (q_1^T x)^2 & -q_2 q_1^T x \\ -q_2 q_1^T x & 1 - q_2^2 \end{pmatrix}. \tag{34}$$

The matrix N_2 is similar with q_1 replaced by u_2, q_2 by u_1, and x by $\tilde{x}$. Such a matrix is singular if and only if either

$$\text{(A)} \quad q_1 = 0 \quad \text{and} \quad \begin{pmatrix} 0 \\ 1 \end{pmatrix} \quad \text{is the null vector,} \tag{35}$$

or

$$\text{(B)} \quad q_1 = x\sqrt{q_1^T q_1} \quad \text{and} \quad \begin{pmatrix} \sqrt{q_1^T q_1} \\ q_2 \end{pmatrix} \quad \text{is the null vector.} \tag{36}$$

In Step 2 (9), cases (A) or (B) can arise only if the vector u in the rank-one update uv^T is

$$u = ce_n \quad \text{or} \quad u = \begin{pmatrix} x\sqrt{q_1^T q_1} \\ q_2 \end{pmatrix}. \tag{37}$$

In Step 3 (11), we have similar conditions (A) and (B) but with u_2 instead of q_1, etc. The case (A) can arise only if the vector $\tilde{y}$ in (9) is zero. Case (B) can arise if

$$u_1 \tilde{y}^T + \sqrt{u_2^T u_2}\tilde{x}^T \tilde{R} = 0.$$

This situation is likely to occur if $\tilde{R}$ is singular, in which case we can have $\|u_2\|_2 = 1$ and $u_1 = 0$, i.e., u_2 is the null vector for $\tilde{R}$.

Of course in practice in all the above cases "singular" should become "almost singular" and all equality signs are "almost equal" signs. But we must conclude from this analysis that N may in fact approach singularity in precisely the situations the condition estimation algorithm is supposed to detect: sharp increases in condition number.

As an aside, notice that Step 3 is actually an application of the ACE algorithm of Pierce and Plemmons [7], and such a breakdown is also likely to occur in their recursive least-squares setting for precisely the same reasons.

3.3.3. Modification 1. Our first modification is very simple. First, the matrices M and N are both *symmetric* and *nonnegative definite.* Therefore, the pencil (M, N) is diagonalizable, i.e., there exists a nonsingular matrix X of generalized eigenvectors such that $X^T M X$ and $X^T N X$ are each diagonal (this can be shown using the generalized singular value decomposition; see Golub and Van Loan [6]). Further note that although N can be singular, $\operatorname{rank}(N) > 0$ (i.e., if $N \neq 0$) since $q_1^T q_1 + q_2^2 = 1$ (or $u_2^T u_2 + u_1^2 = 1$). Therefore, at least one of the eigenvectors is such that $\Gamma^T N \Gamma$ is positive.

Now we argue that since we are starting with an approximate smallest/largest singular vector x_{old}, it is often the case that *all* the eigenvalues of the pencil (M, N) are reasonably small/large. So, the modification is to discard the eigenvectors for which $\Gamma^T N \Gamma$ is smaller than some threshold τ, and choose the smallest/largest eigenvector of those that remain. By the preceding argument, there will be at least one such eigenvector.

Some of the experiments presented in the next section are specially constructed so that a singular N often occurs; in other situations it may occur because of ill-conditioning, and this simple strategy appears to work well. Again we mention as an aside that we expect that this idea could also be used effectively in the ACE algorithm for the recursive least-squares setting.

3.3.4. Modification 2. Our experiments indicated that very occasionally our algorithm would fail to track sharp increases in condition numbers. Such situations were not like the singular cases above where N was ill conditioned; however, the smallest singular value estimates for $\hat{R}$ were much larger than the actual value (by about a factor of 100). The diagonal elements of $\hat{R}$, on the other hand, which we know are upper bounds for its smallest singular value, were in fact of the same order as $\sigma_{\min}(\hat{R})$. Thus, though we do not have an explanation for these rare situations when the estimates are not accurate, we can easily detect their occurrence using the diagonal elements of $\hat{R}$, and we propose a second heuristic that appears to correct such problems. At the end of Step 3, we have an approximate smallest singular value for $\hat{R}$ given by $\|\hat{d}\|_2$. If our estimate is significantly larger than the smallest diagonal element, we perform the following correction step using the ICE methodology.

Truncate $\hat{x}$ and obtain a singular vector for its $n-2 \times n-2$ leading submatrix, much as was done in Step 1 for R. Now use one step of ICE to obtain a new singular vector for $\hat{R}$ starting from this truncated singular vector. Note that this costs only $O(n)$ operations.

We may think that such a fix should be done whenever our estimate exceeds the minimum diagonal element of $\hat{R}$, since the latter is obviously a better estimate. However, recall that we are really interested in the estimate for R_{new}, not $\hat{R}$. If we perform the fix too often, it leads to worse estimates for R_{new}, not better. In practice we perform this step only if

$$\text{Estimated } \sigma_{\min}(\hat{R}) > 10 \min_i |\hat{R}_{ii}|.$$

In the next section we present and analyze some numerical experiments, including specially constructed situations where the above modifications become active.

4. Experimental results. In the first set of experiments we start with a random matrix of condition number κ, having singular values that decrease exponentially from 1 down to $\frac{1}{\kappa}$. We then perform a succession of rank-one updates uv^T using random vectors u and v that have been scaled so that $\|u\|_2 = \|v\|_2 = 1$.

Figures 1 and 2 show the results of such experiments with 40×40 matrices for different values of κ. Figure 1 shows the estimated and actual condition numbers in one such experiment for each value $\kappa = 1, 10, 10^6, 10^{11}$. Figure 2 shows the ratio between the actual and estimated condition numbers in three such experiments for each value of κ. The ratios for the experiments shown in Fig. 1 are displayed as solid lines in the corresponding plots of Fig. 2.

First we notice that when we start out with a well-conditioned matrix as in Fig. 1(a), GRACE successfully tracks the gradual increase in condition number as

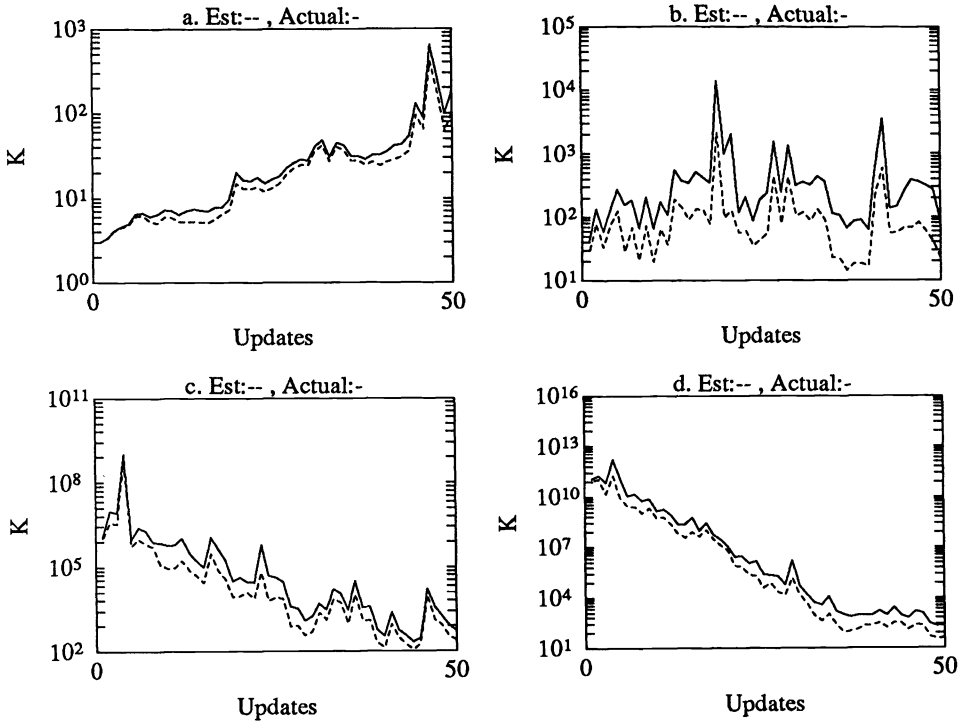


FIG. 1. *Random updates* ($\kappa = 1, 10, 10^6, 10^{11}$, *clockwise*).

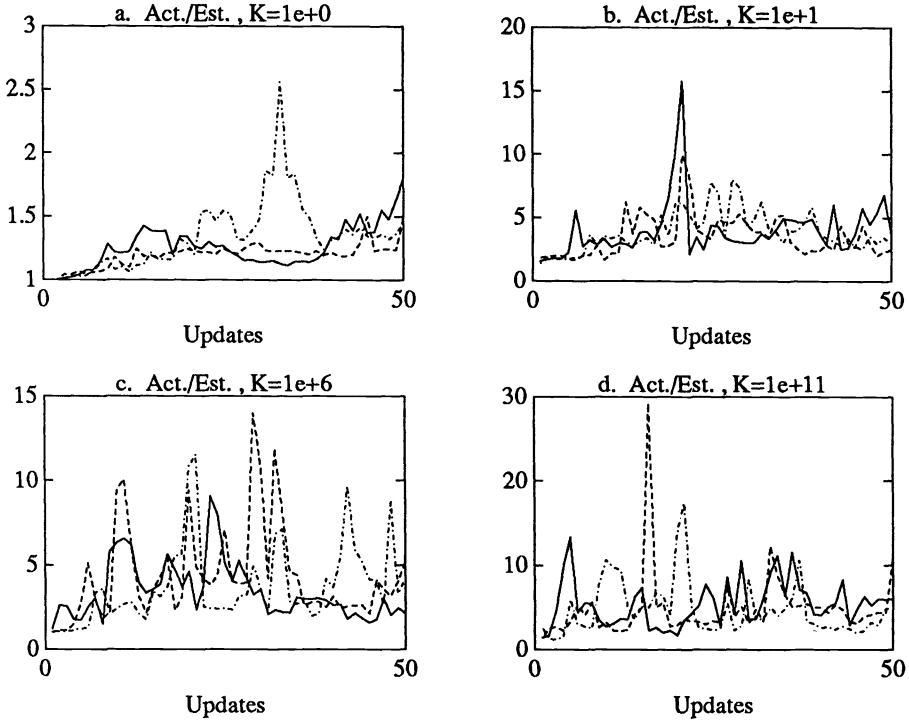


FIG. 2. *Random updates—ratios* ($\kappa = 1, 10, 10^6, 10^{11}$, *clockwise*).

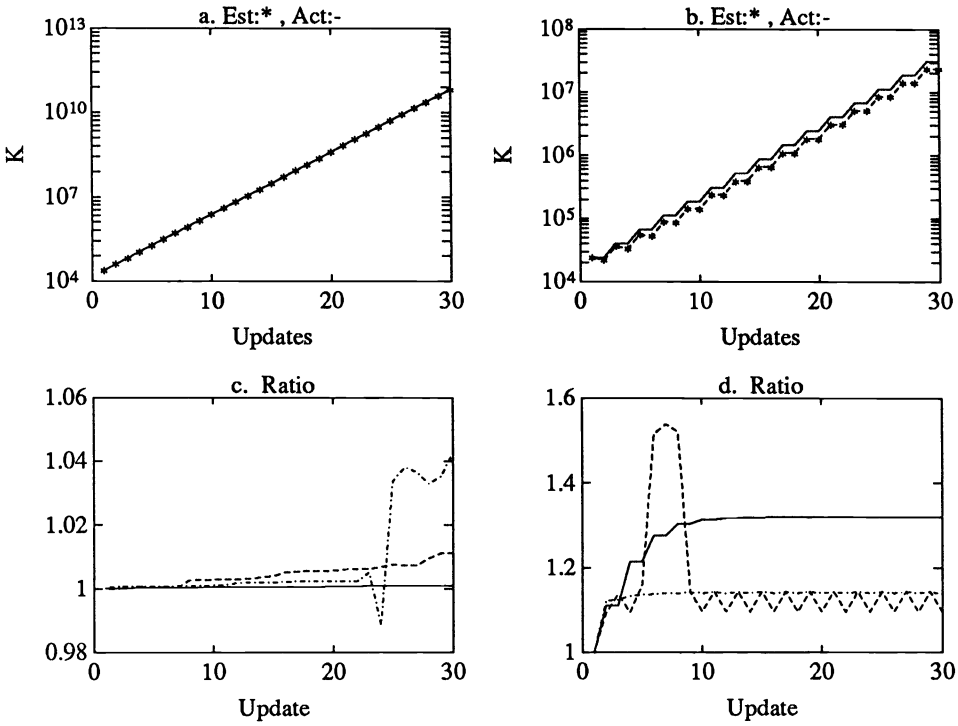


FIG. 3. *Special updates.*

updates progress. From Fig. 2(a) we see that the estimated condition number is usually within a factor of 2 of the actual. This situation is the most probable in practice, i.e., a well-conditioned matrix degrading with time, and GRACE works well in this case.

In Fig. 1(b) the matrix starts out mildly ill conditioned. The updates result in a rapid oscillation of the condition number. Our estimator is able to track these variations, though it is now often off by a larger factor (see Fig. 2(b)). The point to note is that even such rapid oscillations are tracked, and from Fig. 1 we see that the "shape" of the oscillations is reproduced in the estimate.

In Figs. 1(c)–1(d), we start with even more ill-conditioned matrices. The condition number now decreases with time, and the estimator is able to monitor this decrease. We note that even in Fig. 2(d), with very ill conditioned matrices, the estimate is usually off by a factor of less than ten.

Finally, we mention that in all these experiments neither of the modifications discussed in the previous section was required by the algorithm (i.e., a singular case did not occur, neither was the estimated $\hat{\sigma}_{\min}$ much larger than $\min(|\hat{R}_{ii}|)$).

The second set of experiments involves some "special" updates designed to create "singular cases" as discussed in the latter part of the previous section. In each update uv^T, the vectors u and v are chosen to be multiples of the left and right smallest singular vectors of R. This produces a singular case according to (37). More precisely, we start with a random matrix; if at each update step,

$$R = \sum_{i=1}^{n} \sigma_i u_i v_i^T,$$

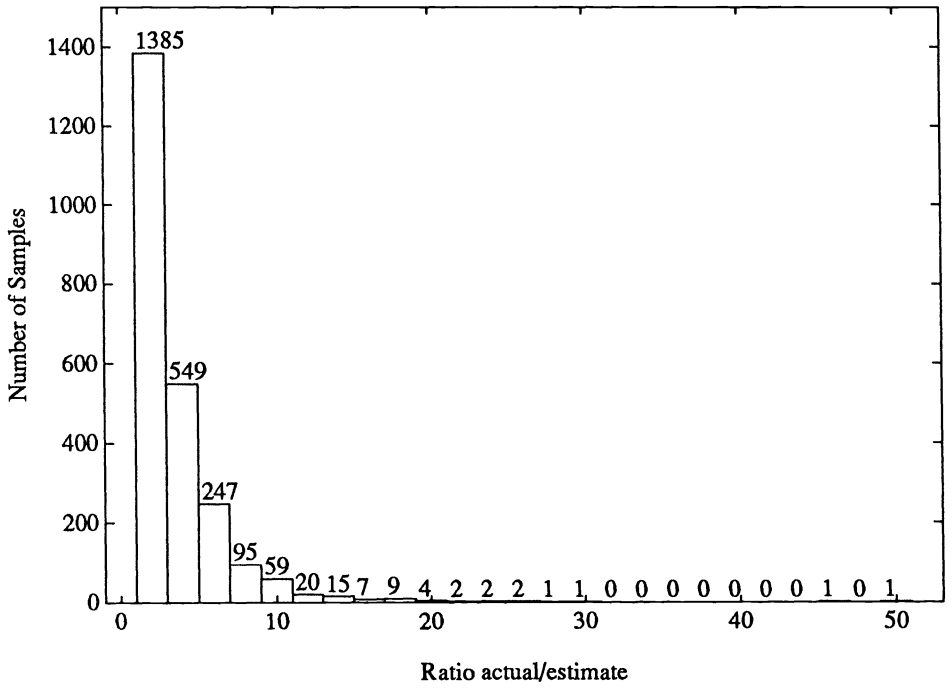


FIG. 4. *Performance for* 2400 *test cases.*

we construct the rank-one update such that $uv^T = \frac{1}{2}\sigma_n u_n v_n^T$. Thus the condition number doubles with every update. Figures 3(a) and 3(c) show the results of three such test cases, with Fig. 3(a) showing the actual and estimated condition numbers and Fig. 3(c) showing the ratio of the two (the erratic behavior of the ratio is largely due to the compressed scale of the figure, but it may be seen that the ratio is always extremely small). In Figs. 3(b) and 3(d) such updates are interleaved with updates in which u and v are multiples of the singular vectors corresponding to σ_{n-1} instead of σ_n. Again the estimate is very close to the actual condition number.

In each of these updates, modification 1, as discussed in the previous section, was activated. The results demonstrate that modification 1 actually works quite well—in its absence the algorithm would break down, and its use results in even better performance for random matrices. We note that we repeated the experiment with updates that were "close" to the special updates by adding small perturbations to the u and v vectors used above, and the behavior remained similar to the above. We conclude that this modification works well and handles the "singular case" in a numerically stable fashion.

We also performed a number of experiments starting with random matrices of different sizes ranging from $n = 20$ to $n = 80$, and condition numbers ranging from $\kappa = 10$ to 10^{11}. On each such matrix fifty random updates were performed. The ratios of actual and estimated condition numbers in all these experiments (a total of 2400 update steps in all) is displayed as a histogram in Fig. 4. For example, in 1385 cases the ratio is less than 3, and, in most cases, the ratio is less than 10.

Finally we mention that the second modification suggested in the previous section is harder to generate, but very occasionally it is in fact required. For example, of the

2400 experiments used for Fig. 4 it was activated in only *one* case, with a 40×40 matrix and condition number around 10^6. Without the modification, the estimate was off by a factor of over 100, whereas with the modification it came within a factor of 3 of the actual value. In general, this modification is very rarely activated, but seems very effective when it is.

We conclude that GRACE is numerically robust and tracks the condition number well in practice.

5. Conclusions. In this paper we developed an algorithm (GRACE) to adaptively monitor the condition number of general (nonsymmetric) matrices as they are modified by a sequence of rank-one updates. The algorithm costs only $O(n)$ extra flops beyond the $O(n^2)$ cost of updating the QR factorization of the matrix with each update. We analyzed potential numerical difficulties in the algorithm and suggested modifications to overcome them.

A number of numerical experiments were presented for random matrices of various sizes, both well conditioned and ill conditioned. The updates included situations in which the condition number increased as well as those in which it decreased, and also cases when the variation in condition number was quite rapid. The results demonstrate that GRACE works well in practice producing an estimate for the condition number that is usually within a small factor of the actual condition number.

Experiments were also carried out that tested the modifications introduced to handle numerical difficulties, and the modifications were seen to work very well in practice. Further, since some of the steps in GRACE use techniques similar to the ACE algorithm developed by Pierce and Plemmons [8] for symmetric updates, these modifications can be used to overcome the numerical difficulties and improve the ACE algorithm as well. (Subsequent to the research in this paper, Pierce and Plemmons [7] have considered these modifications in the context of the ACE algorithm; [7] also contains a more careful investigation of the "singular case" considered in §3.3.2.)

Finally (as was pointed out by one of the referees), the algorithm presented here can be used with minor modifications for rank-one updates of QR factorizations of $m \times n$ matrices (with $m > n$), which could arise in more general least-squares applications.

With the development of this algorithm, $O(n)$ adaptive condition estimation algorithms for the three basic modified QR factorizations are now available: appending/deleting rows or symmetric rank-one updates (ACE), appending columns (ICE), and general rank-one updates (GRACE).

Acknowledgments. We would like to thank the referees for their many helpful comments and suggestions.

REFERENCES

[1] C. H. Bischof, *Incremental condition estimation*, SIAM J. Matrix Anal. Appl., 11 (1989), pp. 312–322.

[2] A. K. Cline, A. R. Conn, and C. F. Van Loan, *Generalizing the* LINPACK *Condition Estimator*, Lecture Notes in Mathematics 909, Springer-Verlag, Berlin, New York, 1982, pp. 73–83.

[3] W. Ferng, G. H. Golub, and R. J. Plemmons, *Adaptive Lanczos methods for recursive condition estimation*, in Proc. SPIE Symposium on Real Time Signal Processing, San Diego, CA, Society of Photo-Optical Instrumentation Engineers, July 1990.

[4] G. Fix and R. Heiberger, *An algorithm for the ill-conditioned generalized eigenvalue problem*, SIAM J. Numer. Anal., 9 (1972), pp. 78–88.

[5] P. E. GILL, G. H GOLUB, W. MURRAY, AND M. A. SAUNDERS, *Methods for modifying matrix factorizations*, Math. Comp., 28 (1974), pp. 505–535.

[6] G. GOLUB AND C. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, Baltimore, 1989.

[7] D. J. PIERCE AND R. J. PLEMMONS, *Fast adaptive condition estimation*, Tech. Report ECA-TR-146, Boeing Computer Services, Seattle, WA, October 1990.

[8] ———, *Tracking the condition number for* RLS *in signal processing*, Math. Control, Sig. Syst., to appear.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1279–1297, October 1992

019

RANK DETECTION METHODS FOR SPARSE MATRICES*

JESSE L. BARLOW† AND UDAYA B. VEMULAPATI‡

Abstract. A method is proposed for estimating the numerical rank of a sparse matrix. The method uses orthogonal factorization along with a one-norm incremental condition estimator that is an adaptation of the LINPACK estimator. This approach allows the use of static storage allocation as is used in SPARSPAK-B, whereas there is no known way to implement column pivoting without dynamic storage allocation. It is shown here that this approach is probably more accurate than the method presently used by SPARSPAK-B. The method is implemented with an overhead of $O(n_U \log n)$ operations, where n_U is the number of nonzeros in the upper triangular factor of the matrix. In theory, it can be implemented in $O(\max\{n_U, n \log n\})$ operations, but this requires the use of a complicated data structure.

It is shown how a variant of this strategy may be implemented on a message-passing architecture. A prototype implementation is done and tests show that the method is accurate and efficient.

Ways in which the condition estimator and the rank detection method can be used are also discussed, along with the rank-revealing orthogonal factorizations of Foster [*Linear Algebra Appl.*, 74 (1986), pp. 47–72] and Chan [*Linear Algebra Appl.*, 88/89 (1987), pp. 67–82].

Key words. sparse matrices, orthogonal factorization, condition estimation, numerical rank

AMS(MOS) subject classifications. 65F50, 65F35, 65F20

1. Introduction.

The problem of rank detection, that is, choosing a set of linearly independent columns from a given matrix within a tolerance of machine precision, is a common subproblem in matrix computations.

We consider here the case of an $s \times n$ matrix C where n and s are large and C is sparse. Traditional methods of rank detection include the singular value decomposition [25, pp. 571–576]; orthogonal factorization with standard column pivoting [3], [12]; rank-revealing orthogonal factorizations (RRQR) [9], [13], [14], [19]; and the threshold strategy in SPARSPAK-B [20], [23], [27]. The singular value decomposition is clearly impractical for large sparse C, so we consider the other three strategies, all of which are based upon orthogonal factorization.

That is, we factor C according to

$$C = Q \begin{pmatrix} U \\ 0 \end{pmatrix} P^T = Q \begin{pmatrix} U_1 & U_2 \\ 0 & 0 \end{pmatrix} P^T, \tag{1}$$

where $U = (U_1\ U_2)$ is a $q \times n$ upper trapezoidal matrix, U_1 is a $q \times q$ nonsingular upper triangular matrix, U_2 is a $q \times (n-q)$ matrix, Q is an $s \times s$ orthogonal matrix, $l = \operatorname{rank}(C)$, and P must be chosen during the course of the factorization.

Ideally, we would have preferred to construct a factorization of the form (1) where U_1 is nonsingular and $\| U_1^{-1} \| < \epsilon^{-1}$ for some prescribed tolerance ϵ and norm $\| \cdot \|$.

* Received by the editors August 23, 1990; accepted for publication (in revised form) August 16, 1991.

† Department of Computer Science, The Pennsylvania State University, University Park, Pennsylvania, 16802-6196 (barlow@cs.psu.edu). The research of this author was supported by Air Force Office of Scientific Research grant AFOSR-88-0161 and was done in part while the author was visiting Oak Ridge National Laboratory under the support of the Applied Mathematical Sciences Research Program of the Office of Energy Research, U.S. Department of Energy, under contract DE-AC05-840R21400 with Martin Marietta Energy Systems, Inc.

‡ Department of Computer Science, University of Central Florida, Orlando, Florida 32816-0362 (vemula@cs.ucf.edu). This research was supported by Office of Naval Research grant N00024-85-C-6041 while the author was a Ph.D. student at the Pennsylvania State University, University Park, Pennsylvania.

It is possible to design such an algorithm, but we know of no algorithm that can do it with an overhead of $O(n_U \log n)$ operations or any comparable complexity. Here we define *overhead* to mean the operations necessary to choose the column ordering.

The RRQR factorizations due to Foster [19] and Chan [13] can actually obtain a U_1 matrix satisfying $\| U_1^{-1} \| \leq \epsilon^{-1}$. However, their overhead is $O(d \cdot n_U)$ where $(n-d)$ is the rank. If $d = O(n)$ (which is possible) or $O(f(n))$ where $f(n) \geq \Omega(\log n)$, then this procedure could be quite expensive.

Foster [19, p. 57] points out that his method can be implemented with a static data structure. We found this to be true, but, as discussed by Björck [11] and §4 of this paper, some of the factor U would have to be stored implicitly. The same could be done with the RRQR technique of Chan [13].

To discuss how P is chosen, we now give one possible context for the use of the factorization (1). Consider the equality-constrained least-squares problem of finding an n-vector y such that

$$\| b - Ay \|_2 = \min_{x \in S_C} \| b - Ax \|_2, \tag{2}$$

where S_C is the set of minimizers of

$$\min_{x \in \Re^n} \| g - Cx \|_2 . \tag{3}$$

Here A is an $m \times n$ matrix, b is an m-vector, g is an s-vector, and $n \leq s + m$. The standard procedures for solving (2) and (3) assume that we can adequately detect the rank of C using a factorization of the form (1) (cf. [4], [5], [3], [37], [23]). For the strategy in [1] and [37], it is necessary to compute the orthogonal factorization of $B = \binom{\tau C}{A}$, where τ is some large weight. Thus the sparsity pattern of B is very important.

Our strategy for choosing P should have two desirable features:

- It should be possible to implement it using a static data structure;
- It should be easy to implement on a message-passing architecture.

To have a static data structure, we choose an initial column ordering P_1 according to the George–Heath strategy from the nonzero structure of $B^T B$. To preserve as much of the sparsity structure as possible, we base the rank detection ordering P upon the sparsity ordering P_1. We also make certain that the factored matrix U fits into a static data structure.

We could discover no way to implement column pivoting without some dynamic storage allocation. It is also difficult to implement on a message-passing architecture because its computations proceed in a lockstep fashion. There is no effective method for exploiting pipelining. We cannot overlap the application and formation of orthogonal factorization, since all column norms must be updated in order to decide which column is the next pivot. Recently, Bischof and Hansen [8] reported some significant progress in this direction using an idea due to Bischof [7] on controlled local pivoting.

For that reason, we design a condition estimation strategy based upon a modification of the LINPACK one-norm estimator of Cline, Stewart, Moler, and Wilkinson [15], [18]. We show that this rank detection method is probably more accurate than the method presently used in SPARSPAK-B [27], [23], a property that column pivoting does not share.

On a sequential computer, our strategy can be implemented with an overhead of $O(\max\{n_U, n \log n\})$ where n_U is the number of nonzeros in U. Since n_U is usually at least $O(n \log n)$, that overhead can be considered to be $O(n_U)$. However,

to obtain that efficiency, we must use Fibonacci heaps [21], which are complicated data structures. Our implementation requires at most $O(n_U \log n)$ operations. The strategy in SPARSPAK-B requires $O(n_U)$ operations. The value n_U is about the same for the two procedures. As is shown in §3, it is easy to exploit pipelining in the implementation of this procedure.

We outline our procedure and discuss its analysis in §2. Computational examples are given in §3. Section 4 is a brief discussion concerning how this method can be used as a preprocessor to an RRQR routine. Section 5 is the conclusion.

2. Rank detection for a sparse underdetermined system. We now give a strategy for finding the column ordering P in (1). It is assumed that we have already obtained the column ordering P_1 through the use of the George–Heath strategy [22], which is given below.

1. Determine the symbolic structure of B^TB.
2. Find a permutation matrix P_1 such that $P_1^TB^TBP_1$ has a sparse Cholesky factor R.
3. Perform the symbolic Cholesky factorization of $P_1^TB^TBP_1$ to generate a storage structure for R.

It should be noted that P_1 is determined using no numerical information. Unfortunately, the rank of a matrix is determined using numerical information and must be done by altering the column ordering during the factorization. Thus U could have fill elements whose positions cannot be predicted by any strategy before the factorization (1). That makes it more difficult to maintain a static data structure.

Björck [11] proved the following result, which we use to show that the algorithms in this section can be implemented using static data structures.

THEOREM 2.1. *Let $B = (b_1, b_2, \ldots, b_n)$ be a column partition of B and let*

$$\hat{B} = (b_{j_1}, b_{j_2}, \ldots, b_{j_k}), \qquad 1 \le j_1 < j_2 < \cdots < j_k \le n$$

be a submatrix of B. Denote the Cholesky factors of B^TB and $\hat{B}^T\hat{B}$ by R and $\hat{R}$, respectively. Then the nonzero structure of $\hat{R}$ is included in the nonzero structure of R.

The reason that column pivoting cannot make use of the above result is that it reorders the columns in a random fashion from the original ordering. Column pivoting skips a column whose norm is less than the maximum and then eliminates it later. The extra fill occurs in the column that is skipped, and it is not possible to predict that fill before factorization.

The first strategy, Algorithm 2.1, just skips columns whose uneliminated parts are negligible, thus no extra storage is necessary. It is a columnwise version of the method used in SPARSPAK-B [23]. The method is explained as a rowwise algorithm by Heath [27]. We call it threshold pivoting.

The column ordering P is denoted by $p_1, p_2, \ldots, p_n$ and $P = (e_{p_1}, \ldots, e_{p_n})$. Without loss of generality, we can assume that the ordering P_1 from SPARSPAK-B is $1, 2, \ldots, n$.

ALGORITHM 2.1 (THRESHOLD PIVOTING STRATEGY).

Assume that we are given a drop tolerance ϵ. Set $p_i \leftarrow i$, $i = 1, 2, \ldots, n$; done $\leftarrow$ false; $k \leftarrow 1$; $q \leftarrow 0$;

while not done **do**

$\gamma \leftarrow \| (c_{q+1,k}, \ldots, c_{s,k})^T \|_2$

if $\gamma \ge \epsilon$ **then**

$q \leftarrow q+1$;
Construct an orthogonal transformation $H_q = \mathrm{diag}(I_{q-1}, \bar{H}_q)$ such that $\bar{H}_q(c_{q,k}, \ldots, c_{s,k})^T = \pm\gamma e_1^{(s-q)}$
Compute $C \leftarrow H_q C$
Set $p_q \leftarrow k$; $k \leftarrow k+1$;
else
temp $\leftarrow k$; $p_q \leftarrow p_{q+1}$; $\ldots$; $p_n \leftarrow$ temp; $k \leftarrow k+1$;
endif
done $\leftarrow q \geq s$ **or** $k > n$
endwhile

Algorithm 2.1 creates no nonzeros in U that cannot be predicted by the George–Heath algorithm [27]. Empirical tests by Heath [27] show that it rarely gives a different value for the rank from that given by column pivoting. We obtain a matrix of the form

$$U = \begin{pmatrix} D_1 & V_1 & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & D_2 & V_2 & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & D_{\alpha-1} & V_{\alpha-1} & \cdot & \cdot \\ 0 & 0 & \cdot & \cdot & \cdot & 0 & 0 & D_\alpha & V_\alpha \end{pmatrix}, \tag{4}$$

where $D_1, D_2, \ldots, D_\alpha$ are upper triangular. More dramatic failures in rank detection are possible for Algorithm 2.1 than for column pivoting. A simple example is the bidiagonal matrix

$$C = \begin{pmatrix} 1 & a & 0 & \cdot & \cdot \\ 0 & 1 & a & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & \cdot & \cdot & 1 & a \\ \cdot & \cdot & \cdot & \cdot & 1 \end{pmatrix}. \tag{5}$$

Clearly, C is considered full rank to machine precision for any value of a, even though it could be arbitrarily badly conditioned. Recently, Foster [20] showed that the random matrix generation scheme given by Stewart [36] yielded badly conditioned upper triangular factors with large diagonal elements with substantial positive probability. This leads us to believe that the example (5) is not altogether rare.

Our improvement to that strategy is designed to make each (D_i, V_i) block in (4) have full row rank to machine precision. First, we discuss the role of condition estimation in column selection. We now give an incremental condition estimator similar in spirit to the one discussed by Bischof [6].

The differences between our estimator and that of Bischof [6] are that our estimator estimates the one-norm instead of the two-norm and our estimator employs "lookahead." Bischof's two-norm estimate requires $O(q^2)$ operations where $q = \mathrm{rank}(C)$. Bischof, Pierce, and Lewis [9] have recently implemented an improvement of that strategy for sparse matrices, but that procedure forces us to use the elimination tree. Our method is a simple modification of the sparse matrix ordering output from, say, SPARSPAK-B.

Also, an estimate without lookahead can fail on 3×3 matrices. For instance, let

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ \tau & -\tau\omega & 2 \end{pmatrix}.$$

For the value $\omega = 1 - \sqrt{2}$, the Bischof algorithm's estimate of the smallest singular value is 1, independent of τ, while the actual two-norm of L^{-1} is about τ^2. In fact, for any 3×3 matrix

$$L^{(3)} = \begin{pmatrix} L^{(2)} & 0 \\ \tau v_3^T & l_{33} \end{pmatrix},$$

if we choose v_3 to be orthogonal to the smallest singular vector of $L^{(2)}$, Bischof's estimate is the same for all values of τ. This type of failure is discussed by Bischof [6] and is possible for any estimator that does not employ lookahead. However Bischof's estimator works well in practice for heuristic reasons given in [6].

Define the sequence of upper triangular matrices $W^{(k)}$, $k = 1, 2, \ldots, l$, by

$$W^{(1)} = (w_{11}) \quad \text{where } w_{11} = \| c_{p_1} \|_2, \tag{6}$$

$$W^{(k+1)} = \begin{pmatrix} W^{(k)} & v_{k+1} \\ 0 & \gamma_{k+1} \end{pmatrix}, \tag{7}$$

where

$$\begin{aligned} v_{k+1} &= (c_{1,p_{k+1}}^{(k)}, \ldots, c_{k,p_{k+1}}^{(k)})^T, \\ \gamma_{k+1} &\equiv \| (c_{k+1,p_{k+1}}^{(k)}, \ldots, c_{s,p_{k+1}}^{(k)})^T \|_2 \\ &= \sqrt{\| c_{p_{k+1}}^{(k)} \|_2^2 - \| v_{k+1} \|_2^2}. \end{aligned}$$

Here

$$c_j^{(k)} = \left(c_{1,j}^{(k)}, \ldots, c_{s,j}^{(k)}\right)^T$$

is column j of C after $H_1, \ldots, H_k$ have been applied.

An estimate $\hat{\sigma}_k$ of $\| [W^{(k)}]^{-1} \|_1$ is formulated recursively as follows. Let

$$L^{(k)} = [W^{(k)}]^T$$

and

$$\begin{aligned} a^{(1)} &= (1), \\ \hat{\sigma}_1 &= 1/w_{11} = 1/\| c_{p_1} \|_2 = 1/\gamma_1, \\ x^{(1)} &= a^{(1)}/w_{11}. \end{aligned}$$

For $k \geq 1$ we now define $\hat{\sigma_{k+1}}$ recursively. Assume that $x^{(k)}$ has been selected at step k to satisfy

$$L^{(k)} x^{(k)} = a^{(k)},$$

where $a^{(k)}$ is a vector of ± 1. Then compute

$$x^{(k+1)} = (x^{(k)}, \xi_{k+1})^T,$$

where

$$\xi_{k+1} = \gamma_{k+1}^{-1}(-\operatorname{sgn}(v_{k+1}^T x^{(k)}) - v_{k+1}^T x^{(k)}).$$

Thus

$$\hat{\sigma}_{k+1} = \max\{\hat{\sigma}_k, |\xi_{k+1}|\} = \| x^{(k+1)} \|_\infty .$$

This procedure is precisely the LINPACK estimator without the lookahead property.

To incorporate lookahead, we consider the partial sums

$$\rho_j^+ = \begin{pmatrix} v_j \\ c_{k+1,j}^{(k)} \end{pmatrix}^T \begin{pmatrix} x^{(k)} \\ \xi_{k+1}^+ \end{pmatrix} \quad \text{and} \quad \rho_j^- = \begin{pmatrix} v_j \\ c_{k+1,j}^{(k)} \end{pmatrix}^T \begin{pmatrix} x^{(k)} \\ \xi_{k+1}^- \end{pmatrix},$$
$$j = k+1, \ldots, n,$$

where

$$\xi_{k+1}^+ = \gamma_{k+1}^{-1}(1 - v_{k+1}^T x^{(k)}), \qquad \xi_{k+1}^- = \gamma_{k+1}^{-1}(-1 - v_{k+1}^T x^{(k)}),$$
$$v_j = (c_{1j}^{(k)}, \ldots, c_{k,j}^{(k)})^T, \qquad j = k+1, \ldots, n.$$

Note that the entry $c_{k+1,j}^{(k)}$ is not known until after we use column p_k to form H_k. The ρ_j can be accumulated throughout the computation. For weights $t_1, t_2, \ldots, t_n > 0$, we then examine

$$\zeta^+ = |\xi_{k+1}^+| + \sum_{j \in Nonz(c^{[k+1]})} t_j \rho_j^+, \qquad \zeta^- = |\xi_{k+1}^-| + \sum_{j \in Nonz(c^{[k+1]})} t_j \rho_j^-, \tag{8}$$

where

$$Nonz(c^{[k+1]}) = \{j : j > k+1, c_{k+1,j}^{(k)} \neq 0\}. \tag{9}$$

Then choose

$$x^{(k+1)} = \begin{cases} (x^{(k)}, \xi_{k+1}^+)^T & \text{if } \zeta^+ \geq \zeta^- , \\ (x^{(k)}, \xi_{k+1}^-)^T & \text{if } \zeta^- > \zeta^+. \end{cases}$$

The choice of weights t_j is heuristic. LINPACK chooses $t_j = u_{p_j,j}^{-1}$. However, we have not computed $u_{p_j,j}^{-1}$ at this point, so we choose $t_j = \gamma_{p_j}^{-1}$, where γ_j is defined in (7). These are lower bounds on the LINPACK weights, that is, $t_j \leq |u_{p_j,j}|^{-1}$. For the set of weights, $t_j = 1$, $j = 1, 2, \ldots, n$, counterexamples have been found where there is dramatic failure [16], [28]. To our knowledge, for either $t_j = |u_{p_j,j}^{-1}|$ or $t_j = \gamma_{p_j,j}^{-1}$, no such counterexamples have been found.

Our algorithm performs the condition estimator upon the most recently formed diagonal block D_i until either

1. it finds a diagonal entry below a tolerance ϵ_1, or
2. the estimate $\| D_i^{-1} \|_1$ exceeds ϵ_2^{-1}.

Here $\epsilon \approx \mu$. In either case, we restart the condition estimator with all $\rho_i = 0$ (implicitly) and then begin block D_{i+1}. Under case (2), we must find a dependent column in D_i. Fortunately, this is easily done with this condition estimator, and we can still use a static data structure.

Note also that under case (2), the condition estimate concludes that

$$\| D_i^{-T} a^{(k)} \|_\infty = \| x^{(k)} \|_\infty = |e_k^T x^{(k)}| = |e_k^T D_i^{-T} a^{(k)}| \geq \epsilon^{-1},$$

since if $|e_k^T x^{(k)}| \neq \| x^{(k)} \|_\infty$, then case (2) occurs for $x^{(k-1)}$. This duality is used by Hager [26], [29] to develop his condition estimator. Thus

$$\epsilon^{-1} \leq |[a^{(k)}]^T D_i^{-1} e_k| \leq \| D_i^{-1} e_k \|_1,$$

since $a^{(k)}$ is a vector of ± 1. We now can locate the dependent column using ideas similar to that originally given by Golub, Klema, and Stewart [24]. That procedure has been generalized to blocks by Hong and Pan [30], but their algorithm is not practical. Solve $D_i h = e_k$. Let ν be an index such that

$$|h_\nu| = \max_{1 \leq j \leq k} |h_j|.$$

Reorder D_i into $\tilde{D}_i$ such that

$$\tilde{D}_i = \left(d_1^{(i)}, \ldots, d_{\nu-1}^{(i)}, d_{\nu+1}^{(i)}, \ldots, d_k^{(i)}, d_\nu^{(i)} \right),$$

and compute

$$\hat{D}_i = \hat{Q}_i \tilde{D}_i = \left(\hat{d}_1^{(i)}, \ldots, \hat{d}_{\nu-1}^{(i)}, \hat{d}_{\nu+1}^{(i)}, \ldots, \hat{d}_k^{(i)}, \hat{d}_\nu^{(i)} \right),$$

where $\hat{Q}_i$ is orthogonal and $\hat{D}_i$ is upper trapezoidal. We can conclude that $\hat{D}_i$ is of rank $k-1$.

That conclusion is based upon the following heuristic. Columns $d_1^{(i)}, \ldots, d_{k-1}^{(i)}$ are considered linearly independent by the condition estimator. The addition of column $d_k^{(i)}$ does not increase the rank and cannot decrease it. Thus D_i must have $k-1$ linearly independent columns. The reordering of the columns is to ensure that the kth component of $d_\nu^{(i)}$ should be of magnitude $|h_\nu|^{-1}$, which is less than $k\epsilon$ and thus negligible. This is a simple column deletion update [32, Chap. 24]. We then let

$$D_i \leftarrow \bar{D}_i = (\hat{d}_1, \ldots, \hat{d}_{\nu-1}^{(i)}, \hat{d}_{\nu+1}^{(i)}, \ldots, \hat{d}_k^{(i)}).$$

The nonzero structure of $\bar{D}_i$ is included in the nonzero structure of the original D_i. We can store $d_\nu^{(i)}$ in a special full vector. Because of structure (4), we need only one full vector of length s for the entire factorization, and that is the only extra storage beyond that allocated by the George–Heath algorithm. That vector would consist of $(d_{\nu_1}^{(1)}, d_{\nu_2}^{(2)}, \ldots, d_{\nu_\alpha}^{(\alpha)})^T$.

We summarize our procedure as Algorithm 2.2.

ALGORITHM 2.2 (CONDITION ESTIMATION RANK DETECTION PROCEDURE).

Assume that we are given the tolerance ϵ that is near machine precision. Let $c^{[q]}$ denote the qth row of C at step q, and let $Nonz(\cdot)$ be defined by (9).

Initializations.

Set $q \leftarrow 0$; done $\leftarrow$ false; $k \leftarrow 1$; first$k \leftarrow 1$; first$q \leftarrow 1$; $p_i \leftarrow i$;
$\rho_i \leftarrow 0$, $\gamma_i \leftarrow \| c_i \|_2 \quad i = 1, 2, \ldots, n$; $\sigma \leftarrow 0$.

while not done **do**
  Compute $\xi \leftarrow \gamma_k^{-1}(-sign(\rho_k) - \rho_k)$
  $\hat{\sigma} \leftarrow \max\{\sigma, |\xi|\}$
  if $\hat{\sigma}^{-1} > \epsilon$ **then**
    $q \leftarrow q + 1$

Construct an orthogonal transformation $H_q = \text{diag}(I_{q-1}, \bar{H}_q)$ such that
$\bar{H}_q(c_{qk}, \ldots, c_{sk})^T = \pm\gamma_k e_1^{(s-q)}$
Compute
$C \leftarrow H_q C$; $p_q \leftarrow k$;
for $j \in Nonz(c^{[q]})$
$\gamma_j^2 \leftarrow \gamma_j^2 - c_{qj}^2$
endfor
$\xi^+ \leftarrow \gamma_k^{-1}(1 - \rho_k)$; $\xi^- \leftarrow \gamma_k^{-1}(-1 - \rho_k)$;
$\zeta^+ \leftarrow |\xi^+|$; $\zeta^- \leftarrow |\xi^-|$;
for $j \in Nonz(c^{[q]})$
$\rho_j^+ \leftarrow \rho_j + c_{qj}\xi^+$; $\rho_j^- \leftarrow \rho_j + c_{qj}\xi^-$;
$\zeta^+ \leftarrow \zeta^+ + \gamma_j^{-1}\rho_j^+$; $\zeta^- \leftarrow \zeta^- + \gamma_j^{-1}\rho_j^-$;
endfor
if $\zeta^+ \geq \zeta^-$ **then**
$\sigma \leftarrow \max\{\sigma, |\xi^+|\}$
$\rho_j \leftarrow \rho_j^+, \quad j \in Nonz(c^{[l]})$
else
$\sigma \leftarrow \max\{\sigma, |\xi^-|\}$
$\rho_j \leftarrow \rho_j^-, \quad j \in Nonz(c^{[q]})$
endif
else
if $\gamma_k \leq \epsilon_1$ **then**
first$q \leftarrow q + 1$; first$k \leftarrow k + 1$;
temp $\leftarrow k$; $p_q \leftarrow p_{q+1}$; $\ldots, p_{n-1} \leftarrow p_n$; $p_n \leftarrow k$;
$\rho \leftarrow 0$, $i = 1, 2, \ldots, n$(implicitly)
else
Let D be the submatrix of C from rows first$q$ through q
and columns first$k$ through k.
Solve $Dh = (0, 0, \cdots, 0, 1)^T$
Let ν be an index such that $|h_\nu| = \max_{(i)} |h_i|$
Move column ν so that it is the last column of D
Compute $D \leftarrow \hat{Q}^T D$
where $\hat{Q}^T$ is an orthogonal transformation that puts D into
upper triangular form.
Apply $\hat{Q}$ to the appropriate rows of C.
first$l \leftarrow l$; first$k \leftarrow k + 1$; $\rho_i \leftarrow 0, i = 1, 2, \ldots, n$ (implicitly).
endif
endif
$k \leftarrow k + 1$; done $\leftarrow l \geq s$ *or* $k \geq n$
endwhile
$$\begin{pmatrix} U \\ 0 \end{pmatrix} = C$$

We note that like Algorithm 2.1, Algorithm 2.2 requires only one pass through the columns of C. The backsolves are with separate matrices $D_i, i = 1, 2, \ldots, \alpha$, and thus require no more work than one backsolve with all of U_1 in (1).

Most of the overhead in Algorithm 2.2 is in updating the ρ_i and γ_i, $i = 1, 2, \ldots, n$. Updating the γ_i is just $O(n_U)$ operations where n_U is the number of nonzeros in U. Normally, updating the ρ_i would also be $O(n_U)$ operations except that they must

be set to zero when we start a new block $D_i, i = 1, 2, \ldots, \alpha$. That can be done by storing only the nonzero values of the ρ_i. In order to make access as fast as possible, we stored the ρ_i in a hierarchical data structure based upon the indices. A Fibonacci heap could be used to get access time down to $O(\max\{n_U, n \log n\})$, or we could use a standard heap for which total access time is $O(n_U \log n)$. We chose the latter because of its simplicity, and because it makes only a second-order contribution to the complexity of the problem. The orthogonal updates operate upon each element of U a constant number of times, and thus require at most $O(n_U)$ operations. Thus the total complexity of the overhead computations is $O(n_U \log n)$. This is only a second-order contribution to the complexity of the computation (1).

In the sequential algorithm, during each step, we generate two possible values for ρ, namely, ρ^+ and ρ^-. Depending on whether $\zeta^+ \geq \zeta^-$, we set ρ to either ρ^+ or ρ^-. But the computation of ζ^+ and ζ^- requires that all columns be updated with the factorization. However, on message-passing machines, we want to overlap the computation of the factorization with the computations of the next column by pipelining. One way of doing this is to compute ζ^+ and ζ^- using columns held by the local processor only (i.e., limited look-up), as opposed to using the information from all the columns of the matrix. The other approach is to postpone the decision of setting ρ to either ρ^+ or ρ^- until the next column is computed. Here we compute ρ^+ and ρ^- as usual, but we do not set ρ. In order to proceed to the next column, we need the value of ρ. Since we do not have the value of ρ, we calculate both possibilities for all variables that depend on the previous value of ρ. The idea is that, by the time the computations are completed for the jth step, the information required to set the value of ρ that corresponds to the $(j-1)$th step would have been known to all processors. Computationally, there is no difference between this parallel implementation and the sequential counterpart. We only try to overlap and pipeline some computations in order to avoid blocking the algorithm to wait for computation of ρ at each step. In the following segment of parallel code, we show how that can be done. One point to note here is that there are some redundant computations in this approach.

During the qth step
for $j \in Nonz(c^{[q]})$ on this processor
$\quad \rho_j^{++} \leftarrow \rho_j^+ + c_{qj}\xi^+;$
$\quad \rho_j^{+-} \leftarrow \rho_j^- + c_{qj}\xi^+;$
$\quad \rho_j^{--} \leftarrow \rho_j^- + c_{qj}\xi^-;$
$\quad \rho_j^{-+} \leftarrow \rho_j^+ + c_{qj}\xi^-$
endfor
Wait for information from the other processors about the previous column
that would allow us to compute ζ_p^+ and ζ_p^- for the previous column.
if $\zeta_p^+ \geq \zeta_p^-$ **then**
$\quad \rho_j^+ = \rho_j^{++};$
$\quad \rho_j^- = \rho_j^{-+}$
else
$\quad \rho_j^+ = \rho_j^{+-};$
$\quad \rho_j^- = \rho_j^{--}$
endif

In our implementations, we used the latter strategy. Algorithm 2.2 has one other useful analytic property.

THEOREM 2.2. *Let* $\text{rank}_1(C)$ *be the rank as determined by Algorithm* 2.1 *for tolerance* ϵ *and let* $\text{rank}_2(C)$ *be the rank as determined by Algorithm* 2.2 *with tolerance* ϵ. *Then, excluding the effects of rounding error,* $\text{rank}_2(C) \leq \text{rank}_1(C)$.

Proof. We show that applying Algorithm 2.2 after Algorithm 2.1 obtains the same rank as applying Algorithm 2.2 by itself. Thus Algorithm 2.2 must obtain at least as small a value for the rank as that for Algorithm 2.1.

Assume that the factorization

$$C = Q_1 \begin{pmatrix} U_1 & U_2 \\ 0 & U_3 \end{pmatrix} P^T$$

is output from Algorithm 2.1. Here, U_1 is an $l \times l$ nonsingular upper triangular matrix, $\text{rank}_1(C) = l$, U_2 is an $l \times (n-l)$ matrix, and U_3 is a matrix whose columns satisfy $\| u_i^{(3)} \|_2 \leq \epsilon$ and are thus negligible.

If we then apply Algorithm 2.2 to

$$U = \begin{pmatrix} U_1 & U_2 \\ 0 & U_3 \end{pmatrix},$$

we get the factorization

$$U = Q_2 \begin{pmatrix} \hat{U}_1 & \hat{U}_2 \\ 0 & \hat{U}_3 \end{pmatrix} \hat{P}^T,$$

where $\hat{U}_1$ is an $\hat{l} \times \hat{l}$ nonsingular upper triangular matrix, $\text{rank}_2(U) \leq \hat{l}$, $\hat{U}_2$ is an $\hat{l} \times (n-\hat{l})$ matrix, and $\hat{U}_3$ is a matrix whose columns satisfy $\|\hat{u}_i^{(3)}\|_2 \leq l\epsilon$. That yields the factorization of C given by

$$C = Q_1 Q_2 \begin{pmatrix} \hat{U}_1 & \hat{U}_2 \\ 0 & \hat{U}_3 \end{pmatrix} \hat{P}^T P^T.$$

Clearly, $\hat{l} \leq l$. Now we must argue that $\hat{l} = \text{rank}_2(C)$.

By uniqueness, results on orthogonal factorization [35, p. 214], and a simple induction argument, Algorithm 2.2 directly applied to C must generate the same coefficients γ_i, ρ_i, $i = 1, 2, \ldots, n$. Thus the same columns are selected for $\hat{U}_1$, and we have the factorization

$$C = \bar{Q} \begin{pmatrix} \hat{U}_1 & \hat{U}_2 \\ 0 & \tilde{U}_3 \end{pmatrix} \bar{P}^T.$$

By uniqueness, $\hat{U}_1$ and $\hat{U}_2$ are the same except for the signs of the rows. $\tilde{U}_3$ can be different but must satisfy $\| \tilde{u}_3^{(i)} \|_2 = \| \hat{u}_3^{(i)} \|_2$ for each column of $\tilde{U}_3$ and $\hat{U}_3$. Hence, $\text{rank}_2(C) = \hat{l} \leq l = \text{rank}_1(C)$. □

The following example shows that column pivoting does not share the property given in Theorem 2.2.

Example 2.1. Let

$$C = \text{diag}(1, s, s^2, \ldots, s^{n-1}) \begin{pmatrix} -c & -c & \cdot & \cdot & -c & 1 \\ 1 & -c & \cdot & \cdot & -c & 0 \\ 0 & 1 & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & 0 & 1 & -c & 0 \\ \cdot & \cdot & \cdot & 0 & 1 & 0 \end{pmatrix} \text{diag}(\beta, \beta^2, \ldots, \beta^{n-1}, 1),$$

where $\beta = 0.9999$, $n = 100$, $c = 0.2$, $c^2 + s^2 = 1$. Let $\mu = 10^{-7}$ be the machine unit. Algorithm 2.1 will produce the factorization $C = QU$, where $u_{100,100} \approx 10^{-8}$. Thus it will conclude that $\text{rank}(C) \leq 99$. Column pivoting will permute column 100 into the pivoting position and then conclude that $\text{rank}(C) = 100$. Note that this is just a slight alteration of a famous example due to Kahan [31]. Thus Algorithm 2.1 can sometimes yield a smaller value for the rank than does column pivoting. This example was improved with the help of one of the referees.

However, column pivoting can sometimes do better than either Algorithm 2.1 or 2.2. Consider the following example, suggested by one of the referees.

Example 2.2. Let

$$C = \begin{pmatrix} a & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & a \end{pmatrix}.$$

The singular values of C are approximately $1, a^2$, and 0. Algorithms 2.1 and 2.2 would both conclude that this matrix has rank 2 if $a > \epsilon$. However, if $a^2 < \epsilon$, the numerical rank of C is 1.

Maximal column pivoting obtains the orthogonal factorization

$$C = \begin{pmatrix} c & s & 0 \\ -s & c & 0 \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \sqrt{1+a^2} & a/\sqrt{1+a^2} & 0 \\ 0 & a^2/\sqrt{1+a^2} & 0 \\ 0 & 0 & 0 \end{pmatrix} P^T,$$

where $c = 1/\sqrt{1+a^2}$, $s = ca$, and P^T is the permutation $(3, 1, 2)$. Note that the numerical rank can be read off of the diagonal.

Our argument for the use of Algorithm 2.2 is mainly that it is more time and storage efficient than column pivoting and about as reliable. These two examples show that the only meaningful way to compare Algorithm 2.2 with column pivoting is empirical tests such as those in §3.

3. Parallel implementation and empirical tests. We implemented a modified version of Algorithm 2.2 on the Intel iPSC/2. This system consists of $P = 2^d$ independent processors, each with its own local memory. Here d is the dimension of the cube. The processors are numbered from 0 to $P - 1$ and there is a connection between two of them if and only if their numbering differs by exactly one digit. Thus no two processors are more than $\log_2 P$ connections apart.

In a straightforward manner, the columns of C are mapped onto the processor in a wrap mapping according to the ordering generated by the George–Heath algorithm. For the sake of simplicity, we may assume that the processors form a ring, i.e., that a ring embedding is used, although other connections are sometimes used for "broadcast" purposes. Each processor makes a decision as to whether to include the next column in the factorization, and sends a message to other processors along with the necessary transformations (if the column is included). The updating of the rest of the columns on the same processor is done only after sending the information to other processors. This procedure can be pipelined very efficiently.

There are a couple of obvious bottlenecks for this algorithm when implemented on a parallel machine. The "back-solve" process, when a bad column is found, involves accessing the partially formed upper triangular factor. Li and Coleman [33], [34] discussed good back-solve procedures for hypercubes that are effective on dense matrices. We used similar techniques in our implementation, but the overall result

TABLE 1
Timing results on factorization with the condition estimator.

No. of processors	Time (secs)
2	40.59
4	28.59
8	20.42
16	14.69
32	10.35
64	7.34
128	5.24

is not impressive on sparse matrices because the arithmetic complexity is quite low compared to the communication overhead. This also causes the algorithm to come to a virtual pause, losing some of the advantages of the asynchronous behavior of the algorithm. However, this happens only occasionally, so we can still expect good speed-ups.

The "lookahead" part of the algorithm, where it needs to find out which value of ρ is to be made permanent, is another bottleneck. We used the compromise lookahead strategy as explained in §2.

3.1. Timing results on the hypercube. The above algorithm was implemented on an Intel iPSC/2 hypercube with 128 nodes, each with four megabytes of memory. The static data structure was generated on another machine (as that was not the part of the problem that we were trying to parallelize) and was fed into the hypercube.

Our data structure, described in detail in [38], allowed us to efficiently execute steps of the form

$$\text{for } j \in Nonz(c^{[l]}).$$

Since this type of step arises in an inner loop of the algorithm, efficient implementation is critical.

The constraint matrices generated for the tests were sparse. The sparsity pattern, as well as the entries of the nonzero elements, were randomly set. While generating the sparsity pattern, the number of nonzeros was selected randomly between a lower and an upper limit (the limits themselves were computed as a fixed fraction of the number of columns in the matrix). The matrix A was taken to be a tridiagonal matrix with diagonal elements as unity and off-diagonal elements as 10^{-3}. The test matrices had up to 10^4 columns and approximately 10^5 nonzeros in the final upper triangular factor. Typical results (averaged over several test matrices) are tabulated in Table 1. The results indicate that each time the number of processors is doubled, the speed-up obtained is approximately 1.4.

3.2. Robustness tests. To test the effectiveness of our algorithm, we tested both the condition estimator and the resulting rank detection procedure. As was suggested by Stewart [36], we generated test matrices of dimension 10, 25, and 50 with a known condition number—the values being 10, 10^3, 10^6, and 10^9. For each of these possibilities, we generated two types of matrices—one where there is a sharp break in the singular value distribution, and the other in which the singular values are exponentially distributed between one and the condition number.

The algorithm always estimated the condition correctly (within two decimal digits accuracy), if there was a sharp break in the singular value distribution, and hence the

TABLE 2
Our condition estimation tests.

k_2	$n = 10$	25	50
10	0.36/0.67	0.33/0.53	0.30/0.43
10^3	0.20/0.58	0.20/0.42	0.22/0.37
10^6	0.11/0.48	0.12/0.36	0.10/0.27
10^9	0.12/0.51	0.12/0.33	0.09/0.26

TABLE 3
LINPACK *condition estimation tests.*

k_2	$n = 10$	25	50
10	0.29/0.46	0.24/0.30	0.17/0.23
10^3	0.29/0.56	0.20/0.33	0.19/0.26
10^6	0.46/0.76	0.20/0.46	0.22/0.35
10^9	0.68/0.86	0.24/0.55	0.23/0.40

results in Table 2 illustrate the case in which there is an exponential distribution of singular values. For each dimension n, 500 test matrices were generated. k_2 is the condition number of the matrix in the Euclidean norm. The numbers quoted are the minimum/average value of the ratio of the estimated condition number to its actual value. The results are rounded to two significant digits.

Comparative results are included in Table 3 for the LINPACK estimator. These tables bound the constant β in (9).

Our tests for rank detection in Tables 4 through 9 were quite similar. The test matrices were generated in exactly the same manner, except that there were only two values of the condition of the matrix, 10^6 and 10^9. We compared Algorithm 2.2 with maximum column pivoting and the SPARSPAK-B procedure. In each of the tables, n is the dimension of the test matrix, and k_2 is the condition number (in the Euclidean norm). Each entry is of the form j/s under each of the min/avg/max columns where j is the rank detected by the algorithm and s is the jth singular value of the matrix. In all of the estimates a cut-off of 10^{-5} was used to detect the rank, and hence the singular value gives us an indication of how accurate the estimate is. The following points summarize the test results.

- Our algorithm always gives a more conservative estimate than the other two strategies. More often than not, it gives a slightly lower value for the rank than the singular value decomposition. In general, for solving least-squares problems, this is better than overestimating the rank. The improvement of our strategy over that in SPARSPAK-B is dramatic.
- Column pivoting seems to obtain a stable value for the rank, with the smallest difference between its maximum and minimum estimates.
- All of the algorithms appear to be reasonably accurate.

4. Use with rank-revealing orthogonal factorization.

4.1. Summary of the algorithm and notes on implementation. We show how our procedure can be merged with an RRQR procedure to perform a more efficient RRQR for a sparse matrix. A scheme similar to this is considered by Barlow [2] in an early version of this work. It should be pointed out that all known procedures that make only one forward pass through the columns of the matrix, such as Algorithms 2.1 and 2.2 and column pivoting procedures, should be considered as preprocessing routines for RRQR procedures.

TABLE 4
Rank detection tests of our algorithm ($k_2 = 1.0e6$).

n	min	avg	max
10	7/1.0e-4	8/2.23-5	9/4.6e-6
25	18/5.7e-5	19/3.2e-5	22/5.6e-6
50	38/3.0e-5	40/1.7e-5	44/5.4e-6

TABLE 5
Rank detection tests of column pivoting algorithm ($k_2 = 1.0e6$).

n	min	avg	max
10	8/2.2 e-5	8/2.2e-5	9/4.6e-6
25	20/1.7e-5	21/1.0e-5	22/5.6e-6
50	42/9.5e-6	44/5.4e-6	46/3.1e-6

The resulting algorithm sacrifices three advantages of Algorithms 2.1 and 2.2.

- We know of no way that it can explicitly store the matrix in statically allocated storage. Described below is an implicit storage scheme (which could not be used for the Businger–Golub [12] or Barlow–Handy [3] column pivoting schemes).
- It does not pipeline well because of its dependence upon back substitution using columns that may not be contiguous.
- Its best complexity bound is $O(d \cdot n_U)$, where d could be $O(n)$ (but it is rarely this large).

Let U be the upper trapezoidal matrix output from Algorithm 2.1 or 2.2. Let U_1 be the left-most subset of columns of U that form an upper triangular matrix, and let U_2 be the remaining columns. Define the functions upper.part$(\cdot)$ and nonupper.part$(\cdot)$ by

$$U_1 = \text{upper.part}(U), \qquad U_2 = \text{nonupper.part}(U).$$

To be consistent with Algorithm 2.2, we use a version of the condition estimator in §2 to develop an implementation of the Foster [19] RRQR routine based upon the one-norm. Chan [13] developed a similar RRQR based upon the two-norm which cannot be adapted to our estimator. Both papers provide rigorous bounds on the accuracy of RRQR procedures.

Since the matrix U will always be upper trapezoidal and is assumed to be output from Algorithm 2.2 (say), we change the condition estimator in §2 so that the weights $t_1, t_2, \ldots, t_q$ in (8) are $t_i = |u_{ii}^{(1)}|^{-1}$, where $U_1 = (u_{ij}^{(1)})$, since these values are now known. The condition estimation procedure is called condest$(W, q, k, \text{rankdef}, \epsilon)$ described below in Algorithm 4.1

ALGORITHM 4.1 (RANK-REVEALING ORTHOGONAL FACTORIZATION).

Assume that we are given a drop tolerance ϵ for the estimate of $\| U_1^{-1} \|_1^{-1}$. Let U be the upper trapezoidal matrix. rankdef is a logical variable that is true if U_1 is rank deficient to the tolerance ϵ. k is the index of the first $k \times k$ principal submatrix of U_1 that is rank deficient or $k = q$ if rankdef is false.

$U_1 \leftarrow$ upper.part(U); $U_2 \leftarrow$ nonupper.part(U);
$q =$ row $-$ dimension(U_1);
condest$(U_1, q, k, \text{rankdef}, \epsilon)$;
while rankdef **do**
  Solve $U_1 h = e_k$
  Let $|h_\nu| = \max_{1 \le i \le q} |h_i|$

TABLE 6
Rank detection tests of threshold pivoting algorithm ($k_2 = 1.0e6$).

n	min	avg	max
10	8/2.2 e-5	9/4.6e-6	10/1.0e-6
25	21/1.0e-5	23/5.6e-6	24/1.8e-6
50	44/5.4e-6	46/3.1e-6	48/1.8e-6

TABLE 7
Rank detection tests of our algorithm ($k_2 = 1.0e9$).

n	min	avg	max
10	4/1.0e-3	5/1.0e-4	6/1.0e-5
25	12/7.5e-5	13/3.2e-5	14/1.3e-5
50	25/4.9e-5	27/1.7e-5	29/7.2e-6

Let Π be a permutation matrix that makes column ν
the last column of U_1.
$U_1 \leftarrow U_1\Pi$
Factor
$U_1 = \hat{Q}\hat{U}_1$ orthogonal factorization
$U_1 \leftarrow \hat{U}_1$; $U_2 \leftarrow \hat{Q}^T U_2; u_{qq}^{(1)} \leftarrow 0$;
$U_1 \leftarrow$ upper.part(U); $U_2 \leftarrow$ nonupper.part(U);
$q =$ row $-$ dimension(U_1);
condest$(U_1, q, k,$ rankdef$, \epsilon)$;
endwhile
$q_{rrqr} =$ row $-$ dimension(U_1)
end algorithm

procedure condest$(W, q, k,$ rankdef$, \epsilon)$;
$\rho_i \leftarrow 0$ $i = 1, 2, \ldots, q$;
done$\leftarrow$ false; rankdef $\leftarrow$ false
$\sigma \leftarrow 0$; $k \leftarrow 1$;
while not done **do**
Compute
$\xi \leftarrow w_{kk}^{-1}(-\text{sign}(\rho_k) - \rho_k)$;
$\hat{\sigma} \leftarrow \max\{\sigma, |\xi|\}$;
if $\hat{\sigma}^{-1} > \epsilon$ **then**
$\xi^+ \leftarrow w_{kk}^{-1}(1 - \rho_k)$; $\xi^- \leftarrow w_{kk}^{-1}(-1 - \rho_k)$;
$\zeta^+ \leftarrow |\xi^+|$; $\zeta^- \leftarrow |\xi^-|$;
for $j \in Nonz(w^{[k]})$;
$\rho_j^+ \leftarrow \rho_j + w_{kj}\xi^+$; $\rho_j^- \leftarrow \rho_j + w_{kj}\xi^-$;
$\zeta^+ \leftarrow \zeta^+ + |w_{jj}^{-1}|\rho_j^+$; $\zeta^- \leftarrow \zeta^- + |w_{jj}^{-1}|\rho_j^-$;
endfor
if $\zeta^+ \geq \zeta^-$ **then**
$\sigma \leftarrow \max\{\sigma, |\xi^+|\}$
$\rho_j \leftarrow \rho_j^+$; $j \in Nonz(w^{[k]})$;
else
$\sigma \leftarrow \max\{\sigma, |\xi^-|\}$
$\rho_j \leftarrow \rho_j^-$; $j \in Nonz(w^{[k]})$;
endif

TABLE 8
Rank detection tests of column pivoting algorithm ($k_2 = 1.0e9$).

n	min	avg	max
10	5/1.03-4	6/1.0e-5	6/1.0e-5
25	13/3.2e-5	14/1.3e-5	15/5.6e-6
50	27/1.7e-5	28/1.1e-5	30/7.9e-6

TABLE 9
Rank detection tests of threshold pivoting algorithm ($k_2 = 1.0e9$).

n	min	avg	max
10	5/1.03-4	6/1.0e-5	7/1.0e-6
25	14/1.3e-5	15/5.6e-6	16/1.3e-6
50	28/1.1e-5	30/7.9e-6	31/3.0e-6

$k \leftarrow k + 1$; done $\leftarrow k \leq q$;
else
done $\leftarrow$ true; rankdef $\leftarrow$ true;
endif
endwhile
end condest

We note that for both Examples 2.1 and 2.2, Algorithm 4.1 obtains the correct value for the rank. It obtains the same factorization as Algorithm 2.2 for Example 2.1 and the same factorization as maximal column pivoting for Example 2.2.

condest$(U_1, q, k, \text{rankdef}, \epsilon)$ requires at most $O(n_U)$ operations. Therefore, Algorithm 4.1 requires $O(d \cdot n_U)$, where d is the number of times through the outer loop. If we use Algorithm 2.1 or 2.2 as a preprocessing routine and q and p_q are the final values of those parameters, it is easy to show that

$$d \leq n - p_q + q - q_{rrqr}.$$

By Theorem 2.2, Algorithm 2.2 never obtains smaller values of q or p_q than Algorithm 2.1; thus this bound is more favorable for Algorithm 2.2.

Note that each time Algorithm 4.1 permutes a vector to the end of U_1, it creates a column whose sparsity pattern cannot be predicted before factorization. Thus to store it explicitly would require some dynamic storage allocation or we could store them in a small dense array. If d is small, this would be the wise method to handle these columns.

Björck [11] proposed the following scheme. Essentially the same scheme was independently discovered by Coleman and Hulbert [17]. Assume that we are storing the original matrix C. Let u_ν be a dependent column found by Algorithm 4.1 and let c_ν be the corresponding column of C. Then u_ν satisfies

$$U_1^T u_\nu = C_1^T c_\nu, \tag{10}$$

where C_1 are the columns of C corresponding to U_1. The corrected seminormal equations [10] can be used to solve (10). It should be noted that to obtain u_ν from c_ν requires one back solve, two forward solves, and two matrix-vector multiplications. If d, the number of columns that we are setting aside this way, is large (as it is for the applications of interest in [11], [17]), this is a practical alternative.

TABLE 10
Results of RRQR *tests.*

Preprocessing algorithm	Rank detected min/avg/max	Improvement over preprocessing	Times condest called
Alg. 2.1	12/13/14	2	3.9
Alg. 2.2	12/13/14	0	1.6
Col. Piv.	12/13/14	1	1.7

4.2. Test results on RRQR algorithm. We carried out tests on random matrices using Algorithm 4.1. The upper trapezoidal matrix U in each case was the result of the factorization of a random matrix that was used in §4.2. The dimension of each matrix was 25 with a condition number of 1.0E-9. The sample size was 500 and ϵ was set to 1.0E5. It is noted that the 12th, 13th, and 14th singular values of the test matrices were 7.5E-5, 3.2E-5, and 1.3E-5, respectively. Table 10 shows the rank detected by Algorithm 4.1 after the preprocessing of the matrix by Algorithm 2.1, Algorithm 2.2, and column pivoting, respectively. From these tests, we conclude that Algorithm 2.2 is a very good preprocessor for a rank-revealing orthogonal factorization.

5. Conclusions. We have demonstrated an accurate algorithm for detecting the numerical rank of a sparse matrix. The method is compatible with any sparse matrix ordering and provably more accurate than the strategy in SPARSPAK-B. It uses only static data structures. On sequential architectures it has an overhead of only $O(n_U \log n)$ operations where n_U is the number of nonzeros in the upper triangular factor. In theory, it can be implemented with an overhead of $O(\max\{n_U, n \log n\})$ operations, but this implementation is very complicated. The method can be efficiently implemented on message-passing architectures.

It has also been noted that this technique can be used to reduce the number of operations in a rank-revealing orthogonal factorization.

Acknowledgments. This problem was originally suggested to Barlow by J. Alan George. Our approach to the problem was inspired by Åke Björck's visit to Penn State. Joseph Liu, Esmond Ng, and Michael Saunders provided sound and helpful suggestions on an early version of this paper [2]. Georg Schnitger pointed out reference [21] on Fibonacci heaps. We also thank the Mathematical Sciences Section at Oak Ridge National Laboratory for the use of the Intel iPSC/2. Finally, we thank C. F. Van Loan, Cindy Robinson-Hubbell, and the insightful, thorough referees.

REFERENCES

[1] J. L. BARLOW, *Error analysis and implementation aspects of deferred correction for equality constrained linear least squares problems*, SIAM J. Numer. Anal., 25 (1988), pp. 1340–1358.

[2] ———, *The accurate solution of sparse weighted and equality constrained least squares problems using a static data structure*, Tech. Report No. CS-89-03, Department of Computer Science, The Pennsylvania State University, University Park, PA, January 1989.

[3] J. L. BARLOW AND S. L. HANDY, *The direct solution of weighted and equality constrained least squares problems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 704–716.

[4] J. L. BARLOW, N. K. NICHOLS, AND R. J. PLEMMONS, *Iterative methods for equality constrained least squares problems*, SIAM J. Statist. Comput., 9 (1988), pp. 892–906.

[5] J. L. BARLOW AND U. B. VEMULAPATI, *A note on deferred correction for equality constrained least squares problems*, SIAM J. Numer. Anal., 29 (1992), pp. 249–256.

[6] C. H. BISCHOF, *Incremental condition estimation*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 312–322.

[7] C. H. BISCHOF, *A parallel QR factorization algorithm with controlled local pivoting*, Tech. Report MCS-P21-1088, Mathematics and Computer Sciences Division, Argonne National Laboratory, Argonne, IL, 1988.
[8] C. H. BISCHOF AND P. C. HANSEN, *Structure-preserving and rank-revealing QR factorizations*, Tech. Report MCS-P100-0989, Mathematical and Computer Science Division, Argonne National Laboratory, Argonne, IL, September 1989.
[9] C. BISCHOF, D. PIERCE, AND J. LEWIS, *Incremental condition estimation for sparse matrices*, SIAM J. Matrix Anal. Appl., 11 (1990), pp. 644–659.
[10] Å. BJÖRCK, *Stability analysis of the method of semi-normal equations for linear least squares problems*, Linear Algebra Appl., 88/89 (1987), pp. 31–48.
[11] ———, *A direct method for sparse least squares problems with lower and upper bounds*, Numer. Math., 54 (1988), pp. 19–32.
[12] P. A. BUSINGER AND G. H. GOLUB, *Linear least squares solutions by Householder transformations*, Numer. Math., 7 (1965), pp. 269–278.
[13] T. F. CHAN, *Rank revealing QR factorizations*, Linear Algebra Appl., 88/89 (1987), pp. 67–82.
[14] T. F. CHAN AND P. C. HANSEN, *Computing truncated singular value decomposition least squares solutions by rank revealing QR-factorizations*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 519–530.
[15] A. K. CLINE, C. B. MOLER, G. W. STEWART, AND J. H. WILKINSON, *An estimate for the condition number of a matrix*, SIAM J. Numer. Anal., 16 (1979), pp. 368–375.
[16] A. K. CLINE AND R. K. REW, *A set of counter-examples to three condition number estimators*, SIAM J. Sci. Statist. Comput., 4 (1983), pp. 602–611.
[17] T. F. COLEMAN AND L. A. HULBERT, *A direct active set algorithm for large, sparse quadratic programs with simple bounds*, Math. Programming, 45 (1989), pp. 373–406.
[18] J. J. DONGARRA, J. R. BUNCH, C. B. MOLER, AND G. W. STEWART, LINPACK *User's Guide*, Society for Industrial and Applied Mathematics, Philadelphia, PA, 1979.
[19] L. V. FOSTER, *Rank and null space calculations using matrix decompositions without column pivoting*, Linear Algebra Appl., 74 (1986), pp. 47–72.
[20] ———, *The probability of large diagonal elements in the QR factorization*, SIAM J. Sci. Statist. Comput., 11 (1990), pp. 531–544.
[21] M. L. FREDMAN AND R. E. TARJAN, *Fibonacci Heaps and their uses in improved network optimization*, in Proc. 25th Annual IEEE Symposium on Foundations of Computer Science, Singer Island, FL, 1984, pp. 338–345.
[22] J. A. GEORGE AND M. T. HEATH, *Solution of sparse linear least squares problems using Givens rotations*, Linear Algebra Appl., 34 (1980), pp. 69–83.
[23] J. A. GEORGE AND E. NG, SPARSPAK: *Waterloo sparse matrix package user's guide for* SPARSPAK-B, Res. Report No. CS-84-37, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, November 1984.
[24] G. H. GOLUB, V. KLEMA, AND G. W. STEWART, *Rank degeneracy and least squares problems*, Tech. Report TR-456, Department of Computer Science, University of Maryland, College Park, MD, 1976.
[25] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins Press, Baltimore, MD, 1989.
[26] W. W. HAGER, *Condition estimates*, SIAM J. Sci. Statist. Comput., 5 (1984), pp. 311–316.
[27] M. T. HEATH, *Some extension of an algorithm for sparse linear least squares problems*, SIAM J. Sci. Statist. Comput., 3 (1982), pp. 223–237.
[28] D. E. HELLER, *More counterexamples to the* LINPACK *condition estimator*, Tech. Report No. CS-81-19, Department of Computer Science, The Pennsylvania State University, University Park, PA, July 1981.
[29] N. J. HIGHAM, *Fortran codes for estimating the one-norm of a real or complex matrix, with applications to condition estimation*, ACM Trans. Math. Software, 14 (1988), pp. 381–396.
[30] Y. P. HONG AND C.-T. PAN, *Rank-revealing* QR *factorizations and* SVD, Tech. Report, Department of Mathematical Sciences, Northern Illinois University, Dekalb, IL, 1990.
[31] W. KAHAN, *Numerical linear algebra*, Canad. Math. Bull., 9 (1966), pp. 757–801.
[32] C. L. LAWSON AND R. J. HANSON, *Solving Least Squares Problems*, Prentice–Hall, Englewood Cliffs, NJ, 1974.
[33] G. LI AND T. F. COLEMAN, *A parallel triangular solver for distributed memory multiprocessors*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 485–502.
[34] ———, *A new method for solving triangular systems on distributed memory message-passing multiprocessors*, SIAM J. Sci. Statist. Comput., 10 (1989), pp. 382–396.
[35] G. W. STEWART, *Introduction to Matrix Computations*, Academic Press, New York, 1973.
[36] ———, *The efficient generation of random orthogonal matrices with an application to condition*

estimators, SIAM J. Numer. Anal., 17 (1980), pp. 403–409.

[37] C. F. VAN LOAN, *On the method of weighting for equality-constrained least-squares problems*, SIAM J. Numer. Anal., 22 (1985), pp. 851–864.

[38] U. B. VEMULAPATI, *Solving Least Squares Problems on Distributed Memory Machines*, Ph.D. thesis, Department of Computer Science, The Pennsylvania State University, University Park, PA, December 1990.

SIAM J. MATRIX ANAL. APPL.
Vol. 13, No. 4, pp. 1298–1313, October 1992

020

MODIFYING THE QR-DECOMPOSITION TO CONSTRAINED AND WEIGHTED LINEAR LEAST SQUARES*

MÅRTEN GULLIKSSON† AND PER-ÅKE WEDIN†

Abstract. A new way of looking at a class of methods for the weighted linear least squares problem $\min_x \|M^{-(1/2)}(b - Ax)\|_2$ where $M = \text{diag}(\mu_i)$ is presented by introducing a modified QR-decomposition with Q M-invariant, i.e., $QMQ^T = M$. One of the main advantages with this approach is that linear constraints are easily incorporated by letting the corresponding diagonal elements in M become zero. Householder reflections are generalized to M-invariant reflections, and an algorithm for solving the constrained and weighted linear least squares problem is described. The system equations (or the augmented system equations) are used to derive condition numbers, and the connection between these condition numbers and the rounding error in the solution is investigated.

Key words. least squares, weights, constraints, QR-decomposition, condition numbers, stability

AMS(MOS) subject classifications. 65F25, 65F35, 65G05

1. Introduction.

1.1. Notation and problem formulation. In a pioneering paper by Gene Golub [10] in 1965 it was shown how the linear least squares problem

$$\min_{x \in \mathbf{R}^n} \|b - Ax\| \tag{1}$$

could be solved by using the QR-decomposition of the matrix A (the norm is always the Euclidean norm if nothing else is stated). In this paper we are going to consider algorithms for an overdetermined system of equations

$$Ax \simeq b, \tag{2}$$

where the overdetermined system of equations (2) corresponds to a weighted and constrained linear least squares problem

$$\min_{x \in \mathbf{R}^n} (b_2 - A_2 x)^T M_2^{-1} (b_2 - A_2 x) \quad \text{s.t.} \quad A_1 x = b_1, \tag{3}$$

where

$$A = \begin{bmatrix} A_1 \\ A_2 \end{bmatrix} \begin{matrix} p \\ q \end{matrix} \quad \text{and} \quad b = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \begin{matrix} p \\ q \end{matrix}, \quad q = m - p,$$

and $M_2 = \text{diag}(\mu_i)$, with $\mu_{i+1} \geq \mu_i$, $i = p+1, \ldots, m$ and $\mu_{p+1} > 0$. It is assumed that $m \geq n \geq p$.

An equivalent formulation of (3) is

$$\begin{bmatrix} 0 & 0 & A_1 \\ 0 & M_2 & A_2 \\ A_1^T & A_2^T & 0 \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ x \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ 0 \end{bmatrix}, \tag{4}$$

* Received by the editors January 11, 1991; accepted for publication (in revised form) January 22, 1992.

† University of Umeå, Institute for Information Processing, S–901 87, Umeå, Sweden (marten@cs.umu.se and pwedin@cs.umu.se).

where λ_1 is the Lagrange vector and $M_2\lambda_2$ is the residual. It is easily seen that (3) has a unique solution if and only if $\operatorname{rank}(A_1) = p$ and $\operatorname{rank}(A) = n$, and we assume that these two conditions are fulfilled.

If we take

$$M = \begin{bmatrix} 0 & 0 \\ 0 & M_2 \end{bmatrix}, \quad M_2 \text{ nonsingular}, \tag{5}$$

we can write (4) as

$$\begin{bmatrix} M & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} \lambda \\ x \end{bmatrix} = \begin{bmatrix} b \\ 0 \end{bmatrix}, \tag{6}$$

where $M\lambda$ is the residual. We generally assume that $M = \operatorname{diag}(\mu_i)$, $\mu_{i+1} \geq \mu_i \geq 0$, $i = 1, \ldots, m-1$. This restriction on M is not necessary in many of the results presented in later sections, but is assumed if nothing else is stated.

If there are no constraints, the matrix M is invertible and we formulate our problem as

$$\min_{x \in \mathbf{R}^n} (b - Ax)^T M^{-1} (b - Ax), \tag{7}$$

where $A \in \mathbf{R}^{m \times n}$, $b \in \mathbf{R}^m$, and $m \geq n$.

For the ordinary least squares problem, we get $M = I_m$, and for an unweighted constrained least squares problem we get $M_2 = I_{m-p}$.

We may write the overdetermined system (2) as $Ax \overset{M}{\simeq} b$ where the "inverse weight matrix" M, by the system of equations (6), exactly defines in which sense b approximates Ax. The system of equations (6) also implies that there exists a matrix B such that $x = Bb$. Just as we avoid computing the inverse of a matrix A when solving a system of equations $Ax = b$, we avoid *explicitly* computing the matrix B. Instead we are going to base our algorithms for (3) on a modified QR-decomposition of the matrix A.

1.2. The basic idea of a modified QR-decomposition. If the matrix A equals $\left[R^T, 0\right]^T$ with R a nonsingular $n \times n$-matrix, then the approximation problem $Ax \overset{M}{\simeq} b$ has the solution $x = \left[R^{-1}, 0\right] b$. The following theorem gives a condition on the matrix Q_1 that makes the problem $Ax \overset{M}{\simeq} b$ equivalent to $Q_1 Ax \overset{M}{\simeq} Q_1 b$.

THEOREM 1.1. *Let the nonsingular matrix Q_1 satisfy the condition*

$$Q_1 M Q_1^T = M. \tag{8}$$

Then $Ax \overset{M}{\simeq} b$ if and only if $Q_1 Ax \overset{M}{\simeq} Q_1 b$.

Proof. Since

$$\begin{bmatrix} Q_1 & 0 \\ 0 & I_{m-n} \end{bmatrix} \begin{bmatrix} M & A \\ A^T & 0 \end{bmatrix} \begin{bmatrix} Q_1^T & 0 \\ 0 & I_{m-n} \end{bmatrix} = \begin{bmatrix} M & Q_1 A \\ A^T Q_1^T & 0 \end{bmatrix},$$

$Ax \overset{M}{\simeq} b$ implies that

$$\begin{bmatrix} M & Q_1 A \\ A^T Q_1^T & 0 \end{bmatrix} \begin{bmatrix} Q_1^{-T} r \\ x \end{bmatrix} = \begin{bmatrix} Q_1 b \\ 0 \end{bmatrix},$$

and hence that $Q_1 Ax \overset{M}{\simeq} Q_1 b$. The converse is proved in exactly the same way. □

A nonsingular matrix that satisfies the identity (8) is called M-invariant. We describe a modification of the QR-decomposition that uses M-invariant reflections $Q_1, \ldots, Q_n$ to transform the given problem $Ax \overset{M}{\simeq} b$ into a triangular system.

The following theorem whose simple proof is analogous to that of Theorem 1.1, shows how the modified QR-decomposition is used to solve the approximation problem $Ax \overset{M}{\simeq} b$.

THEOREM 1.2. *Choose an M-invariant matrix Q and an orthogonal matrix Π such that*

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \Pi^T$$

with R upper triangular. If the matrix in (6) *is invertible then the solution of* (3) (*or equivalently,* (6)) *is given by $x = \Pi(R^{-1}, 0)Q^{-1}b$, and if we partition M as*

$$M = \begin{bmatrix} M_n & 0 \\ 0 & M_{m-n} \end{bmatrix} \tag{9}$$

with M_{m-n} an $(m-n) \times (m-n)$-matrix, the vector λ in (6) *is given by*

$$\lambda = Q^{-T} \begin{bmatrix} 0 & 0 \\ 0 & M_{m-n}^{-1} \end{bmatrix} Q^{-1} b.$$

1.3. Summary, intentions, and related works. We are going to use M-invariant matrices Q_i, i.e., matrices that satisfy $Q_i M Q_i^T = M$, to solve the weighted and constrained linear least squares problem. In §3, we give an algorithm for that purpose and outline the properties of a QR-decomposition with the matrix Q M-invariant.

Before doing so, it is suitable to discuss the general properties of M-invariant matrices, which we do in §2. Note that if M is singular, the matrix Q gets a typical block lower triangular form as is seen in Lemma 2.3. Since we are going to use elementary matrices in our algorithm, we give a detailed analysis of M-invariant matrices of that kind. We especially focus on the M-invariant oblique reflections that play the same role as Householder reflections for ordinary least squares. It is, of course, possible to use M-invariant rotations instead of reflections, and these are discussed in [15].

In a way, we develop a framework that allows us to see the ordinary QR-decomposition, the Powell–Reid method [20] for weighted linear least squares, and the Björck–Golub method [7] for constrained least squares as the same method applied to different problems.

Yet, there are two other fundamentally different ways to use the augmented system: the dual approach developed by Paige (see [17], [18], and [14]), and the direct solution of the augmented system described in [6] and [1].

In §4, we introduce the relevant condition number for (3). We illustrate in a diagram how for some test problems the rounding error in the solution becomes roughly proportional to the condition number. A condition number that depends on the residual was first derived in [12]. The system of equations was used extensively for studying the conditioning in [3] and [4]. A perturbation theory that also covered the rank-deficient case was developed in [21]. The perturbation identity from [21] can be

used for an elementwise perturbation analysis, as shown in [8]. Relevant condition numbers for constrained least squares were given in [9] and a generalization covering rank-deficient equality-constrained least squares problems is given in [23]. Here we use the perturbation theory in [22] that covers both the weighted and constrained linear least squares problem, including the rank-deficient case.

In §5, we use the earlier test problem to show the connection between different condition numbers and rounding errors when our algorithm is used with and without column pivoting.

Finally, we discuss the stability of the method. We have not yet proved results as strong as we think necessary. We present the results we have derived and try to identify the main difficulties in the backward error analysis. The standard error analysis for the ordinary QR-decomposition computed with reflections can be found in [24, pp. 440–441]. An error analysis for the special case when there is only one weight (in the limit when the weight tends to infinity, we get the Björck–Golub algorithm) can be found in [2].

2. M-invariant matrices.

2.1. General properties. We begin with the basic definition. Assume that $Q \in \mathbf{R}^{m\times m}$ and $M \in \mathbf{R}^{m\times m}$; then Q is said to be *M-invariant* if it is nonsingular and $QMQ^T = M$.

The identity matrix is, of course, M-invariant, and the following lemma characterizes the inverse and transpose of an M-invariant matrix as well as the product of several M-invariant matrices (the proof is trivial).

LEMMA 2.1. *If Q is M-invariant, then Q^{-1} is also M-invariant and if in addition M is nonsingular, then Q^T is M^{-1}-invariant. If Q_1 and Q_2 are M-invariant, then Q_1Q_2 and Q_2Q_1 are also M-invariant.*

From Lemma 2.1 we now have all the properties of M-invariant matrices to state the following theorem.

THEOREM 2.2. *The M-invariant matrices $Q \in \mathbf{R}^{m\times m}$ form a group with the identity matrix I_m as the identity.*

The last lemma concerns the structure of Q when M is given by (5).

LEMMA 2.3. *Assume that Q is nonsingular and M has the same properties as in* (5). *Then Q is M-invariant if and only if*

$$Q = \begin{bmatrix} Q_{11} & 0 \\ Q_{21} & Q_{22} \end{bmatrix}, \qquad Q_{22}M_2Q_{22}^T = M_2,$$

and Q_{11} is nonsingular.

Proof. Take

$$Q = \begin{bmatrix} Q_{11} & Q_{12} \\ Q_{21} & Q_{22} \end{bmatrix}$$

and use the special form of M in (5) to write $QMQ^T = M$ on the form

$$QMQ^T = \begin{bmatrix} Q_{12}M_2Q_{12}^T & Q_{12}M_2Q_{22}^T \\ Q_{22}M_2Q_{12}^T & Q_{22}M_2Q_{22}^T \end{bmatrix} = \begin{bmatrix} 0 & 0 \\ 0 & M_2 \end{bmatrix}. \tag{10}$$

From $(Q_{12}M_2^{1/2})(Q_{12}M_2^{1/2})^T = 0$ it follows that $Q_{12} = 0$. Obviously, (10) is satisfied if and only if $Q_{12} = 0$ and $Q_{22}M_2Q_{22}^T = M_2$, which proves the lemma. □

2.2. Elementary M-invariant matrices. In this section we derive the explicit form of elementary M-invariant matrices Q, i.e.,

$$Q = I - 2cd^T, \qquad c, d \in \mathbf{R}^m, \tag{11}$$

and then look at how c and d should be chosen so that

$$Qa = -\alpha e_1, \qquad \alpha \neq 0, \tag{12}$$

when $Q^2 = I$, i.e., Q is a reflector. Before dealing with elementary M-invariant matrices we look at the norm of a general elementary matrix. We then need the following lemma, which we state without proof.

LEMMA 2.4. *Assume that $u, v \in \mathbf{R}^n$ with $\|u\| = \|v\| = 1$; then*

$$\max_{\|x\|=1} u^T x \cdot v^T x = \tfrac{1}{2}(1 + u^T v). \tag{13}$$

Using Lemma 2.4 with the fact that $\|Q\|^2 = \max_{\|x\|=1} \|Qx\|^2 = 1+4\max_{\|x\|=1} d^T x \cdot (\|c\|^2 d - c)^T x$, we immediately get that

$$\|Q\| = \eta + \sqrt{\eta^2 - 2\gamma + 1}, \tag{14}$$

where $\eta = \|d\|\,\|c\|$ and $\gamma = c^T d$. If Q is a reflector, we have $\gamma = 1$ and (14) can be simplified to

$$\|Q\| = \eta + \sqrt{\eta^2 - 1}. \tag{15}$$

We start by examining the case when M is nonsingular and then the case when M has the same structure as in (5). M nonsingular gives us the following lemma.

LEMMA 2.5. *Assume that $Q = I - 2cd^T$, $d^T M d \neq 0$, and Q is M-invariant with M nonsingular. Then*

$$Q = I - 2Mdd^T/d^T Md \quad \text{with } Q^2 = I,$$

i.e., Q is a reflector.

Proof. The condition $QMQ^T = M$ gives, with $Q = I - 2cd^T$ by direct identification, that $Q = I - 2Mdd^T/d^T Md$ if $d^T Md \neq 0$ and $(Md)_i \neq 0, i = 1, \ldots m$. The second restriction is eliminated by the nonsingularity of M. The proof of $Q^2 = I$ is trivial.

The uniqueness of Q is given by the fact that an orthogonal reflection is unique and that $\tilde{Q} = M^{-1/2} Q M^{1/2}$ is an orthogonal reflection. □

The norm of Q when M is nonsingular is given by (15) where $\eta = \|d\|\,\|Md\|/d^T Md$. Obviously, in an algorithm where c and d are constructed, it is essential to keep Md as parallel to d as possible if we want Q to be well conditioned.

The form of the elementary M-invariant matrices for the second case with M as in (5) is given by the lemma below.

LEMMA 2.6. *Assume that $Q = I - 2cd^T$ with Q M-invariant and that M has the form as in* (5). *Then either $d^T Md \neq 0$ and*

$$Q = I - 2Mdd^T/d^T Md \quad \text{with } Q^2 = I \tag{16}$$

or

$$Q = I - 2\begin{bmatrix} c_1 \\ c_2 \end{bmatrix}\begin{bmatrix} d_1 \\ 0 \end{bmatrix}^T \quad \text{with } Q^2 = I \quad \text{if and only if } c_1^T d_1 = 1, \tag{17}$$

where in the latter case we have used the partition

$$c = \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} \begin{matrix} p \\ q \end{matrix} \quad \text{and} \quad d = \begin{bmatrix} d_1 \\ d_2 \end{bmatrix} \begin{matrix} p \\ q \end{matrix}.$$

Proof. Using the partition defined in the lemma and $QMQ^T = M$, we get

$$\begin{bmatrix} 0 & 0 \\ 0 & M_2 \end{bmatrix} = \begin{bmatrix} c_1 c_1^T \cdot d_2^T M_2 d_2 & c_1 c_1^T \cdot d_2^T M d_2 + c_1 d_2^T M_2 \\ \text{symmetric} & (I_q + d_2 c_2^T) M_2 (I_q + d_2 c_2^T)^T \end{bmatrix},$$

which implies that either $c_1 = 0$ or $d_2^T M_2 d_2 = 0$. In the former case, we may choose c_2 and d_2 as in Lemma 2.5 and the form and uniqueness of (16) is proved. For the latter case, we get by the nonsingularity of M_2 that $d_2 = 0$. We finally get $c_1^T d_1 = 1$ directly from $Q^2 = I$ and (17) is proved. □

When Q is given by (16), the Euclidean norm given by (15) is mainly determined by

$$\eta = \frac{\|M_2 d_2\| \, \|d\|}{d_2^T M_2 d_2}, \tag{18}$$

which gets large when d_2 becomes almost orthogonal to $M_2 d_2$. If Q has the other form (17) and $Q^2 = I$, then

$$\eta = \|c\| \, \|d_1\| \tag{19}$$

must be small if Q is to be well conditioned.

2.3. M-invariant reflections. If we impose the condition in (12) on Q when M is nonsingular, we get by Lemma 2.5 that

$$Qa = a - 2\frac{d^T a}{d^T M d} M d = -\alpha e_1, \qquad \alpha \neq 0, \tag{20}$$

and thus $Md \in \operatorname{span}\{a, e_1\}$ or

$$Md = a + \alpha e_1. \tag{21}$$

Multiplying α with the sign of the first element in a makes it easier to calculate our reflection with small rounding errors (this is not necessary, however; see Parlett [19]). Another argument for the sign is that the corresponding rotation (see [15]) gets a minimal rotational angle which seems natural.

The form of Md in (20) gives, after some straightforward calculations, that

$$\alpha = \operatorname{sign}(a_1) \sqrt{\mu_1 a^T M^{-1} a}, \tag{22}$$

which determines Q. If we define $N = M^{-1}\mu_1 = \operatorname{diag}(\mu_1/\mu_i)$, we can write Q as

$$Q = I - \frac{(a + \alpha e_1)(Na + \alpha e_1)^T}{\alpha(\alpha + a_1)}, \tag{23}$$

where $\alpha = \operatorname{sign}(a_1)\sqrt{a^T N a}$. For the sake of simplicity, in what follows, we assume that $a_1 \geq 0$.

With this form on Q we have that

$$\eta = \frac{\|a + \alpha e_1\| \, \|Na + \alpha e_1\|}{\alpha(\alpha + a_1)} \geq \frac{\|a\|}{\alpha + a_1} \geq \frac{\|a\|}{2\alpha} \tag{24}$$

with subsequent ill-conditioned Q when $a + \alpha e_1$ is nearly orthogonal to $Na + \alpha e_1$, i.e., when α is much smaller than $\|a\|$.

We now turn to the case when M is given by (5) and Q is given by Lemma 2.6. The form of Q in (16) is not suitable when we want (12) to be satisfied because according to (20) and (21) we then must have $a_i = 0, i = 2, \ldots, p$, which is a severe restriction on a. Now make the partition

$$a = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \begin{matrix} p \\ q \end{matrix}$$

and observe that a_1 is now a vector. We use the other form of Q in (17) where Q is a reflector, i.e., $c_1^T d_1 = 1$ and choose $c = (a + \|a_1\| e_1)/(2\|a_1\|)$ and $d_1 = (a_1 + \|a_1\| e_1)/(\|a_1\| + a_1^{(1)})$ where $a_1^{(1)}$ denotes the first element in a_1. The final Q can then be written as

$$Q = I - \frac{(a + \alpha e_1)\left(\begin{bmatrix} a_1 \\ 0 \end{bmatrix} + \alpha e_1\right)^T}{\|a_1\| \, (\|a_1\| + a_1^{(1)})}. \tag{25}$$

If we compare (23) with (25) and assume that $\lim_{\mu_p \to 0} \mu_i/\mu_p = 1$ for $i < p$ we see that when

$$M \to \begin{bmatrix} 0 & 0 \\ 0 & M_2 \end{bmatrix}$$

or equivalently, $N \to \operatorname{diag}(I_p, 0)$, we have that

$$Q \text{ in (23)} \to Q \text{ in (25)} \tag{26}$$

and $\alpha \to \operatorname{sign}(a_1^{(1)})\|a_1\|$.

Using these limits and (24), we see that Q is ill conditioned when

$$\frac{\|a\|}{\|2a_1\|} \gg 1.$$

Note that (26) suggests that we could define our problem (1) with $M + \epsilon I$ instead of just M, and then let ϵ tend to zero demanding the solution $x(\epsilon)$ to be continuous. The advantage is, of course, that the M-invariant matrix Q is unique and (26) is given directly.

We conclude this section by giving an algorithm for computing the M-invariant reflection Q. We have chosen to use a stylized MATLAB notation similar to that used in [11].

ALGORITHM RPRvec.

Given an n-vector a and an n-vector μ with the inverse of the weights, this function computes vectors u and v such that $Qa = (I - uv^T/u(1))a$ is zero in all but the first component.

function $[u, v]$ =**RPRvec**(a, μ)
  n =**length**(a); ν =**ones**(n); $u = a$; $v = a$
  for $i = 2 : n$
    if $\mu_i \neq 0$
      $\nu_i = \mu_1/\mu_i$
    end
    $v_i = \nu_i a_i$
  end
  $\alpha = \mathbf{sign}(a_1)\sqrt{a^T v}$
  $u_1 = u_1 + \alpha$; $v_1 = u_1$
  $u = u/\alpha$; $v = v/\alpha$
end

3. The use of M-invariant transformations to solve the weighted and constrained linear least squares problem.

3.1. Existence and uniqueness. Let us consider existence and uniqueness problems connected with the modified QR-decomposition

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix} \tag{27}$$

when A is nonsingular. If we start by analyzing the case when M is nonsingular, the modified QR-decomposition exists with Q M-invariant and the upper triangular matrix R unique if the diagonal elements in R are positive. This result follows directly from the properties of the ordinary QR-decomposition where Q is orthogonal. If we assume that A has the modified QR-decomposition in (27) and M is nonsingular we have

$$M^{-1/2}A = (M^{-1/2}QM^{1/2}) \begin{bmatrix} M^{-1/2}R \\ 0 \end{bmatrix}$$

with $M^{-1/2}QM^{1/2}$ orthogonal and $M^{-1/2}R$ upper triangular. If the matrix $M^{-1/2}R$ has positive diagonal elements, it is unique (see Björck [3]) and hence the matrix R is unique too.

We now turn to the case when M is singular and especially has the form as in (5). We then have the following theorem describing the existence and uniqueness of the modified QR-decomposition. For the proofs of the next theorem, see [15].

THEOREM 3.1. *Assume that Q is M-invariant with M as in* (5) *and that A is nonsingular. Partition*

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{matrix} p \\ q \end{matrix}, \qquad \begin{matrix} p & n-p \end{matrix}$$

where p is the number of linear constraints to problem (1). *A decomposition*

$$A = Q \begin{bmatrix} R \\ 0 \end{bmatrix}$$

with R upper triangular and diagonal elements $r_{ii} > 0$, $i = 1, \ldots, n$ exists if and only if A_{11} is nonsingular. The matrix R is unique if and only if M is nonsingular, i.e., there are no constraints, i.e., $p = 0$.

The role of the M-invariant transformations when solving the weighted linear least squares problem is evident from Theorem 1.1.

3.2. An algorithm for the constrained and weighted linear least squares problem. It is now quite obvious how M-invariant transformations can be used to solve (1). It is not as obvious what kind of pivoting strategy we shall use. The strategy is, in step k, to find the smallest weighted norm of the columns in $A_{22}^{(k)}$ such that

$$\gamma_{j_{\max}}^{(k)} = \max_{k \le j \le n} \gamma_j^{(k)}, \qquad \gamma_j^{(k)} = \sum_{i=k}^{m} |a_{ij}^{(k)}|^2 \frac{\mu_k}{\mu_i}. \tag{28}$$

We then exchange column $j_{\max}$ with column k in $A^{(k)}$ if $j_{\max} \neq k$. Observe that this will give exactly the same procedure as in the Powell–Reid algorithm if all the weights are equal to one. There will be an additional work load of approximately $\frac{1}{2}mn^2 - \frac{1}{3}n^3$ flops for our method if we do not use the fact that

$$\gamma_j^{(k+1)} = \frac{\mu_{k+1}}{\mu_k}(\gamma_j^{(k)} - (a_{kj}^{(k)})^2), \qquad j = k+1, \ldots, n. \tag{29}$$

We can expect numerical trouble when the quotient μ_{k+1}/μ_k becomes very large, but this possibility is easy to check and avoid merely by calculating $\gamma_j^{(k+1)}$ using the definition in (28). A short analysis of the numerical behavior of the updating formula (28) including an algorithm determining when to recalculate the weighted norms can be found in [15].

It is also possible to use row pivoting, but it seems that there is no real need for that, as can be seen from the numerical examples in [14]. It is also seen that the example in [20], which gives bad results with the Powell–Reid algorithm if the weight is large and no row pivoting is used, causes no problem here because we have already sorted the weights in decreasing order *and* the problem is well conditioned.

We now give a description of the algorithm which we call the revised Powell–Reid algorithm (RPR).

ALGORITHM RPR.

Given $A \in \mathbf{R}^{m \times n}$ with $m \ge n$ and the inverse of the weights in the vector μ, where $\mu_i \le \mu_{i+1}, i = 1, \ldots, m-1$, this function computes the factorization $A\Pi = QR$, where Π is a permutation matrix permuting the columns according to (28) with $k = 1$. The matrix A is overwritten by R in its upper triangular part, and information about Q is stored in the strict lower part of A. An output vector u is needed for additional information about Q and the element $pcol_r$ contains the position of the column that was permuted with column r in step r.

function $[A, u, pcol, r] =$**RPR**(A, μ)
  Determine γ according to (28); $\tau = \max_{1 \le j \le n} \gamma_j$
  Find smallest p with $1 \le p \le n$ so that $\gamma_p = \tau$; $r = 0$
  while $\tau > 0$
    $r = r + 1$
    $pcol_r = p$; Swap γ_r with γ_p; Swap $a_{:,r}$ with $a_{:,p}$
    $[u_{r:m}, v_{r:m}] =$ **RPRvec**$(a_{r:m,r}, \mu_{r:m})$
    for $j = r : n$
      $t = -v_{r:m}^T a_{r:m,j}/u_r$
      $a_{r:m,j} = a_{r:m,j} + u_{r:m}t$
    end
    $a_{r+1:m,r} = u_{r+1:m}$
    if $r < n$

```
        Update γ according to (29);  τ = max_{r+1≤j≤n} γ_j
        Find smallest p with r + 1 ≤ p ≤ n so that γ_p = τ
      else
        τ = 0
      end
    end
end
```

The work load for this algorithm is the same as for the method of Powell and Reid and test examples in [14] indicate that our method is at least as accurate as the Powell–Reid method.

Using the output from Algorithm RPR, i.e., A, u, and *pcol*, it is an easy task to calculate the solution $x = \Pi(R^{-1}, 0)Q^{-1}b$ (assuming that the full-rank conditions in the introduction are fulfilled). An implementation of Algorithm RPR and a solver in FORTRAN can be found in [14].

4. Condition numbers and rounding errors for the algorithm.

4.1. The perturbation identity. The condition number of a computational problem is a measure of how sensitive the solution of the problem is to perturbations of the input data. In our case the matrix A and vector b are the input data. For a certain problem, we generally assume the inverse weight matrix M to be fixed, but it is perfectly possible to consider the effect on the solution of a perturbation δM of the matrix M, too.

It can be shown (see [22]) that the inverse of the matrix in (4) can be written as

$$\begin{bmatrix} M & A \\ A^T & 0 \end{bmatrix}^{-1} = \begin{bmatrix} H & B^T \\ B & -BMB^T \end{bmatrix}$$

where B is a generalized inverse of A. The solution of problem (7) is given by $x = Bb$ and $\lambda = Hb$. Assume that the perturbed matrix in (4) is nonsingular and take $\tilde{x} = x + \delta x$ and $\tilde{\lambda} = \lambda + \delta\lambda$ as the solution of the perturbed problem. It is then fairly straightforward to get the identity

$$\delta x = -B\delta A\tilde{x} + B\delta b - B\delta M\tilde{\lambda} + BMB^T\delta A^T\tilde{\lambda}. \tag{30}$$

4.2. Normwise perturbation bounds. If we define

$$\kappa = \|A\|\,\|B\|, \qquad \kappa_\lambda = \|A\|\,\|BM^{1/2}\|/\|M^{1/2}\|, \tag{31}$$

we have from (30) that

$$\frac{\|\delta x\|}{\|x\|} \le \kappa\left(\epsilon + \frac{\|\delta b\|}{\|Ax\|} + \frac{\|\delta M\|\,\|\lambda\|}{\|Ax\|}\right) + \kappa_\lambda^2\epsilon\frac{\|M\|\,\|\lambda\|}{\|Ax\|} + \mathbf{O}(\epsilon^2), \tag{32}$$

where $\epsilon = \|\delta A\|/\|A\|$. It is easily seen that $\kappa_\lambda \le \kappa$ and that κ_λ is not changed when M is multiplied with a scalar.

To know the condition of our problem (1), it is necessary to estimate $\|B\|$ and $\|BMB^T\|$. From Theorem 1.2, $B = [R^{-1}, 0]\,Q^{-1}$ and we have

$$\|B\| = \|[R^{-1}, 0]Q^{-1}\|, \tag{33}$$

an expression not easily simplified unless $M = I_m$ or $M_2 = I_{m-n}$. Using the relation $M = QMQ^T$ and the explicit form of the solution in Theorem 1.2, we get

$$BMB^T = \Pi R^{-1}[I_n, 0]M \begin{bmatrix} I_n \\ 0 \end{bmatrix} R^{-T}\Pi^T = \Pi R^{-1} M_n R^{-T}\Pi^T, \tag{34}$$

and we easily deduce that $\|BM^{1/2}\| = \|R^{-1}M_n^{1/2}\|$. Hence we have from (31) that

$$\kappa = \|A\| \, \| \left[R^{-1}, 0\right] Q^{-1}\|, \qquad \kappa_\lambda = \|A\| \, \|R^{-1}M_n^{1/2}\|/\|M^{1/2}\|. \tag{35}$$

The estimate (32) looks complicated and, superficially, it would seem suitable to use the *condition number* of the system matrix

$$S = \begin{bmatrix} M & A \\ A^T & 0 \end{bmatrix}$$

that occurs in (6). It is also true that $\|S\| \, \|S^{-1}\| \geq \|A\| \, \|B\|$. However, we cannot allow perturbation in all elements of S and, in many cases, the condition number of S is much larger than $\|A\| \, \|B\|$.

If the vector b is perturbed into $b + \delta b$ and the matrix A is left unchanged, then we get the sharp estimate

$$\frac{\|\delta x\|}{\|x\|} \leq \kappa \frac{\|\delta b\|}{\|x\| \, \|A\|} \leq \kappa \frac{\|\delta b\|}{\|Ax\|},$$

and κ is just as relevant a condition number for a general inverse weight matrix M as for an ordinary system of equations where $B = A^{-1}$. However, with respect to perturbations of A, the weight matrix makes a difference. If an ordinary system of equations $Ax = b$ is perturbed into $(A + \delta A)(x + \delta x) = b$, then

$$\lim_{\epsilon \to 0} \left[\sup_{\|\delta A\|/\|A\| \leq \epsilon} \frac{\|\delta x\|/\|x\|}{\epsilon} \right] = \|A\| \, \|A^{-1}\|.$$

In this case the condition number is independent of the vector b. For the weighted least squares problem, we get the following fairly sharp estimate with respect to perturbations in A directly from (32):

$$\lim_{\epsilon \to 0} \left[\sup_{\|\delta A\|/\|A\| \leq \epsilon} \frac{\|\delta x\|/\|x\|}{\epsilon} \right] \leq \kappa + \kappa_\lambda^2 \frac{\|M\| \, \|\lambda\|}{\|Ax\|}.$$

It is seen that this estimate critically depends on the vector b in its second term.

From the discussion above it is natural to define the true condition number for perturbations only in A as

$$\kappa_A = \kappa + \kappa_\lambda^2 \frac{\|M\| \, \|\lambda\|}{\|Ax\|}. \tag{36}$$

4.3. Componentwise perturbation bounds and row equilibration. A componentwise perturbation estimate can just as easily be derived from (30). Let $|\delta A| \leq \epsilon |A|$, $|\delta b| \leq \epsilon |b|$, and $|\delta M| \leq \epsilon M$; then

$$|\delta x|/\epsilon \leq |B|(|A| \, |x| + |b|) + |BM| \, |\lambda| + |BMB^T| \, |\delta A|^T \, |\lambda| + \mathbf{O}(\epsilon). \tag{37}$$

Row equilibration might be useful if we get much sharper bounds from (37) than from (32) for a certain problem class. The revised Powell–Reid algorithm for (3) is still applicable. If the matrix DA, with $D = \operatorname{diag}(D_1, D_2)$ a diagonal scaling matrix with strictly positive elements, is much more well balanced than A, we can change A to DA, b to Db, and M_2 to $M_2 D_2^{-1}$. Finally, we sort the equations so that we have $\mu_i \leq \mu_{i+1}$. However, when A is well scaled, the componentwise perturbation bound (37) will not lead to sharper bounds than (32).

4.4. An illustrative example. Since $B = A^\dagger AB$ and $A^\dagger = BAA^\dagger$, the condition number $\kappa = \|A\| \, \|B\|$ satisfies the inequality

$$\max\{\|AB\|, \kappa_2(A)\} \leq \kappa \leq \kappa_2(A)\|AB\|, \qquad \kappa_2(A) = \|A\| \, \|A^\dagger\|. \tag{38}$$

The norm of the projection AB is greater than 1 if it is not an orthogonal projection and B equals $A^\dagger$. It is easy to prove (38), but the conclusion that can be drawn from it is far from trivial. Our weighted least squares problem can get a large condition number $\|A\| \, \|B\|$ for exactly two reasons: $\kappa_2(A)$ is large or the norm of the projection AB is large. The condition number $\kappa_2(A)$ does not depend on M and

$$\|AB\| = \left\| Q \begin{bmatrix} I_n & 0 \\ 0 & 0 \end{bmatrix} Q^T \right\|$$

does not depend on the scaling in $\mathbf{R}^n$. We want an example where we can vary $\kappa_2(A)$ and $\|AB\|$ independently. The following example has that property. Set

$$A = \begin{bmatrix} 1 & 1 & 5 & 4 \\ 1 & 2 & 4 & 2 \\ 1 & 3 & 3 & \epsilon_1 \\ 1 & 0 & 6 & 1 \\ 1 & 6 & \epsilon_2 & 2 \end{bmatrix}$$

and $b = M\lambda + Ax$, where $\lambda = d_1 + (5/\epsilon_1 - 1)d_2$, $d_1 = [-1, -1, 1, 1, 0]^T$, $d_2 = [1, -2, 1, 0, 0]^T$. Then we get the solution $x = [-12, 1, 3, 3,]^T$ for any inverse weight matrix M. The parameters ϵ_1 and ϵ_2 determine the conditioning of the problem. For $\epsilon_1 = 1$ and $\epsilon_2 = 10$, we get a well-conditioned problem. Using MATLAB notation, we have $\lim_{\epsilon_1 \to 0} \sigma_3(A_{1:3,:}) = 0$ and $\lim_{\epsilon_2 \to 0} \sigma_4(A) = 0$, i.e., for decreasing ϵ_1 the submatrix $A_{1:3,:}$ converges to a rank-deficient matrix and for decreasing ϵ_2 the whole matrix A approaches a rank-deficient matrix.

Let us first choose M as the unit matrix to get an ordinary linear least squares problem. In this case, $\kappa = \kappa_\lambda = \|A\| \, \|A^\dagger\|$ and the ordinary QR-decomposition without column pivoting is a stable method. Assuming that the computed vector x is the solution of a problem with A slightly perturbed, we have that $\log(\|\delta x\|/\|x\|) \simeq \log(\kappa) + \log(\|\delta A\|/\|A\|)$ so that when varying ϵ_2 from 10 to 10^{-6} (and thus creating a more and more ill conditioned problem) the logarithm of the relative error will vary linearly with the logarithm of ϵ_2. This can easily be illustrated in a figure similar to Fig. 1.

To get the norm of the projection $\|AB\|$ large we take $\epsilon_2 = 10$ and choose $\mu_1 \leq \mu_2 \leq \mu_3$ very small but fixed and vary ϵ_1 from 1 to 10^{-6}. The norm of the projection $\|AB\|$ then increases while $\kappa_2(A) = \|A\|\|A^\dagger\|$ stays moderately great. From (32) we have $\log(\|\delta x\|/\|x\|) \simeq \log(\kappa_A) + \log(\|\delta A\|/\|A\|)$ and, again, the logarithm of the rounding errors grows linearly with the logarithm of the true condition number κ_A.

5. The connection between condition numbers and rounding errors. When we use the Powell–Reid method to solve the weighted least squares problem (7), we first transform the given problem (3) to

$$\min_x \|\tilde{b} - \tilde{A}x\| \tag{39}$$

with $\tilde{b} = M^{-1/2}b$ and $\tilde{A} = M^{-1/2}A$ before solving problem (39) with the QR-decomposition. Even if the rounding errors introduced by multiplying A with $M^{-1/2}$ and b with $M^{-1/2}$ can be ignored, we do something potentially dangerous. The condition number of $\tilde{A}$ is $\tilde{\kappa} = \|M^{-1/2}A\| \, \|(M^{-1/2}A)^{\dagger}\|$ and it goes to infinity when μ_1/μ_m goes to zero. The condition number for problem (39) that corresponds to the condition number κ_A defined in (36) is here called the "false" condition number and is defined as

$$\tilde{\kappa}_A = \tilde{\kappa} + \tilde{\kappa}^2 \frac{\|\tilde{b} - \tilde{A}x\|}{\|\tilde{A}x\|}. \tag{40}$$

It can be shown that $\tilde{\kappa}_A$ is almost proportional to $\tilde{\kappa}$ when $\tilde{\kappa}_A$ is much larger than κ_A. Hence, in this case, it is not the residual term that gives us any trouble.

Take $\epsilon_1 = 1$, $\epsilon_2 = 10$, and $\mu = [\eta, \eta, \eta, 1, 1]$ in the example of the previous section. In Fig. 1 the true condition number and the relative error of the solution with and without column pivoting are plotted as a function of the "false" condition number when the parameter η goes to zero.

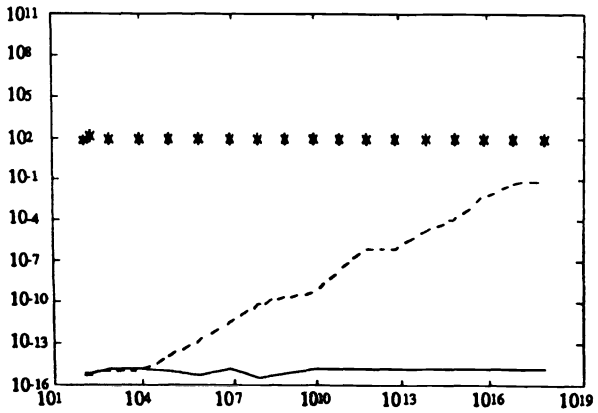


FIG. 1. *True condition number* (∗) *and relative error for our method with* (—) *and without* (- -) *column pivoting as a function of the false condition number.*

In Fig. 1 it is seen that the true condition number does not change with the "false." Neither do the rounding errors when we use column pivoting. The effect is very different when column pivoting is not used in that the rounding errors become almost proportional to the "false" condition number. For a harmless problem with κ_A small and $\tilde{\kappa}_A$ large we have almost no precision left in the result. The reason is, of course, that the submatrix $A_{11} = A_{1:3,1:3}$ in our example is singular, and when η goes to zero the first three equations of $Ax \overset{M}{\simeq} b$ become satisfied with equality.

6. On the error analysis of the modified QR-decomposition with column pivoting. It is evident from the discussion above that when solving the weighted linear least squares problem with the modified QR-decomposition, we get transformation matrices that may have very large elements (and consequently very large norm). The question then arises as to whether the algorithm is numerically stable or not. We only discuss the results we have achieved so far. For a more detailed discussion on error analysis for linear least squares, see, e.g., [5] or [16].

It is quite natural that most error analyses have been done on algorithms using Gauss transformations or orthogonal transformations. Obvious reasons are that Gauss transformations have very nice structural properties as well as being relatively cheap to stabilize, and the orthogonal transformations preserve the Euclidean and the Frobenius norm of the transformed vector or matrix and thus do not enlarge errors.

On the other hand, it is not evident that unstable transformations, i.e., transformations not necessarily having relatively small elements, give a backward unstable algorithm; the LU-decomposition of a matrix A with Gauss transformations using *column* pivoting is backward stable despite the fact that the transformations can be very ill conditioned.

Our results of error analysis concern the important special case where the system matrix looks like

$$S = \begin{bmatrix} 0 & 0 & A_1 \\ 0 & \zeta I_q & A_2 \\ A_1^T & A_2^T & 0 \end{bmatrix}, \qquad 0 < \zeta < \infty,$$

i.e., we want to solve (3) with $M_2 = \zeta I_q$, which is the linear least squares problem

$$\min_{x \in \mathbf{R}^n} \|b_2 - A_2 x\| \quad \text{s.t.} \quad A_1 x = b_1. \tag{41}$$

We assume, without any loss of generality, that $\zeta = 1$. For the proof of Theorem 6.1 we refer to [13]. This result is not entirely new and similar results can be found in [2], but the approach here is different.

THEOREM 6.1. *If $\hat{x}$ is the computed solution we get by solving problem* (41) *with the modified* QR*-decomposition using column pivoting, then*

$$(A + F)\hat{x} \overset{M}{\simeq} b + f. \tag{42}$$

We have the following partitioned bound for the error matrix F:

$$\begin{bmatrix} \|F_1\|_F \\ \|F_2\|_F \end{bmatrix} \leq \begin{bmatrix} c_1 p^2 \|A_1\|_F \mathbf{u} \\ c_2(q(n-p) + \sqrt{n}\, p(n+p^2)\tau^p)\, \|A_2\|_F \mathbf{u} \end{bmatrix} + O(\mathbf{u}^2), \tag{43}$$

where $1 \leq \tau \leq 1+\sqrt{2}$ is a growth factor and $\mathbf{u}$ is the roundoff unit. For the right-hand side we have

$$\|f_1\| = 0 \tag{44}$$

and

$$\|f_2\| \leq c_f(p + (n-p)(m-n+q))\, \|b_2\| \mathbf{u} + O(\mathbf{u}^2). \tag{45}$$

The three constants c_1, c_2, and c_f are of modest size and are independent of the problem size.

We have, as mentioned in the introduction, not proved a theorem analogous to Theorem 6.1 in the general case. The main reason is that the transformations are no longer divided into Gauss transformations and Householder reflections, but we now have general M-invariant transformations with neither the structure of the Gauss transformations nor the Euclidean or Frobenius norm invariant properties of the Householder reflections.

Acknowledgment. The authors are grateful to the referees for constructive criticism and suggestions for improvements of the manuscript and especially for the comment on row equilibration.

REFERENCES

[1] M. ARIOLI, I. S. DUFF, AND P. DE RIJK, *On the augmented system approach to sparse least-squares problems*, Numer. Math., 55 (1989), pp. 667–684.

[2] J. L. BARLOW AND S. L. HANDY, *The direct solution of weighted and equality constrained least squares problems*, SIAM J. Sci. Statist. Comput., 9 (1988), pp. 704–716.

[3] Å. BJÖRCK, *Iterative refinement of linear least squares solutions* I, BIT, 7 (1967), pp. 257–278.

[4] ———, *Iterative refinement of linear least squares solutions* II, BIT, 8 (1968), pp. 8–30.

[5] ———, *Error analysis of least squares algorithms*, Tech. Report LiTH-MAT-R-1988-52, Department of Mathematics, Linköping University, Linköping, Sweden, 1988.

[6] ———, *Pivoting and stability in the augmented system method*, Tech. Report, Department of Mathematics, Linköping University, Linköping, Sweden, 1991.

[7] Å. BJÖRCK AND G. H. GOLUB, *Iterative refinement of linear least squares solution by Householder transformation*, BIT, 7 (1967), pp. 322–337.

[8] J. W. DEMMEL AND N. J. HIGHAM, *Improved error bounds for undetermined system solvers*, Numer. Anal. Report No. 189, Department of Mathematics, University of Manchester, Manchester, U.K., 1990.

[9] L. ELDÉN, *Perturbation theory for the least squares problem with linear equality constraints*, SIAM J. Numer. Anal., 17 (1980), pp. 338–350.

[10] G. H. GOLUB, *Numerical methods for solving linear least squares problems*, Numer. Math., 7 (1976), pp. 206–216.

[11] G. H. GOLUB AND C. F. VAN LOAN, *Matrix Computations*, Second Edition, The Johns Hopkins University Press, 1989.

[12] G. H. GOLUB AND J. H. WILKINSON, *Note on the iterative refinement of least squares solution*, Numer. Math., 9 (1966), pp. 139–148.

[13] M. E. GULLIKSSON, *Error analysis of an algorithm for constrained linear least squares*, Tech. Report UMINF 91-01, Institute of Information Processing, University of Umeå, Umeå, Sweden, 1990.

[14] ———, *On some algorithms for constrained and weighted linear least squares*, Tech. Report UMINF 90-178, Institute of Information Processing, University of Umeå, Umeå, Sweden, 1990.

[15] M. E. GULLIKSSON AND P.-Å. WEDIN, *Modifying the* QR *decomposition to constrained and weighted linear least squares*, Tech. Report UMINF 91-02, Institute of Information Processing, University of Umeå, Umeå, Sweden, 1991.

[16] N. J. HIGHAM, *Computing error bounds for regression problems*, Tech. Report No. 179, Department of Mathematics, University of Manchester, Manchester, U.K., 1989.

[17] C. C. PAIGE, *Fast numerically stable computations for generalized linear least squares problems*, SIAM J. Numer. Anal., 16 (1979), pp. 165–171.

[18] ———, *Computer solution and perturbation analysis of generalized linear least squares problems*, Math. Comp., 33 (1980), pp. 171–183.

[19] B. N. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice–Hall, Englwood Cliffs, NJ, 1980.

[20] M. J. D. POWELL AND J. K. REID, *On applying Householder transformations to linear least squares problems*, in IFIP Congress, North Holland Publishing Company, Amsterdam, 1969, pp. 122–126.

[21] P.-Å. WEDIN, *Perburbation theory for pseudo-inverses*, BIT, 13 (1973), pp. 217–232.

[22] P.-Å. WEDIN, *Perturbation theory and condition numbers for generalized and constrained linear least squares problems*, Tech. Report UMINF–125.85, Institute of Information Processing, University of Umeå, Umeå, Sweden, 1985.

[23] M. WEI, *Perturbation theory for the rank-deficient equality constrained least squares problem*, SIAM J. Numer. Anal., 29 (1992), to appear.

[24] J. H. WILKINSON, *The Algebraic Eigenvalue Problem*, Clarendon Press, Oxford, England, 1965.