JOURNAL OF COMPUTATIONAL PHYSICS 54, 87–94 (1984)

Computational Methods in Linear Algebra

J. M. VARAH

Department of Computer Science, University of British Columbia, Vancouver, British Columbia, Canada

Received June 15, 1981

This paper is the written version of a talk given at the Workshop on Computational Seismology Salishan Lodge, Oregon, March 24–26, 1981. State of the art and recent developments in computational linear algebra, including linear systems, least squares techniques, the singular value decomposition, and eigenvalue problems are reviewed briefly.

1. INTRODUCTION

This paper attempts to survey recent developments in computational linear algebra; it was presented to an audience of seismologists and geophysicists. The aim was to provide a brief description of the state of the art in the major applications areas (including references to methods and software), and to alert the audience to some interesting recent work with potential application. The survey is divided into three sections: linear systems, least squares methods, and eigenvalue problems. Methods for dense and sparse matrices are considered in each section. Instead of the usual plethora of references, we have chosen to mention a few particularly good references for each of the subjects discussed.

2. LINEAR SYSTEMS

2A. *Dense Stored Sysems* ($Ax = b$)

As is well known, the standard algorithm here is Gaussian elimination, which applied to the $n \times n$ matrix A directly produces the factorization $A = LU$ with L unit lower triangular and U upper triangular, using $O(n^3)$ operations. Pivoting is often required as well, so that the elements of L and U produced are not large; *row pivoting* is normally used which produces the factorization $PA = LU$, with P some permutation matrix. Row and column pivoting can also be used; this produces $P_1 A P_2 = LU$.

The computed solution $\bar{x}$ from Gaussian elimination has the very nice property that it is the exact solution of a perturbed system $(A + \delta A)\bar{x} = b$, with $\|\delta A\| \cong f(n) \cdot \varepsilon$, where $f(n)$ is a slowly growing function of n and ε is the machine

0021-9991/84 $3.00

precision. This means that the residual $\|b - A\bar{x}\|$ is always about the size of the machine precision, and that the error in $\bar{x}$ satisfies

$$\frac{\|x - \bar{x}\|}{\|x\|} \leqslant \kappa(A) \cdot f(n) \cdot \varepsilon,$$

where $\kappa(A) = \|A\| \cdot \|A^{-1}\| =$ condition number of A. Thus it is very important when solving any linear system to get an estimate of $\kappa(A)$; one should at least solve a second perturbed system as well, $A(x + \delta x) = b + \delta b$, for then

$$\frac{\|\delta x\|}{\|x\|} \cong \kappa(A) \frac{\|\delta b\|}{\|b\|},$$

so that the condition number is approximated by the normalized ratios of the solution and data.

A good reference for this material is Forsythe and Moler [6]. Many good codes are now available, with many specializations possible with different structures assumed in A. One reliable set of codes is LINPACK, available from Argonne National Laboratories. This includes an estimation for $\kappa(A)$, and the authors have also produced a "Users Guide" (Dongerra *et al.* [3]).

Although, in general, a dense $n \times n$ system requires $O(n^3)$ operations for the LU factorization, sometimes the special nature of the matrix can be exploited to find different solution algorithms which are faster. One example of this is the well-known Levinson algorithm for Toeplitz matrices. This is $O(n^2)$ and has recently been shown to be as stable as the usual LL^T factorization for A positive definite (see [1]).

2B. *Matrices with Special Structure*

When the matrix A of the linear system has a particular structure (in its nonzero elements), it is usually advantageous to devise some modification of the normal Gaussian elimination procedure. We mention two examples here.

(i) *A banded.* Suppose the nonzero elements of A are confined to a narrow band along the diagonal, say with m nonzero elements on each side of the diagonal. Then clearly the elimination procedure only involves m rows at each stage, and the factorization involves only $O(m^2 n)$ operations. It is particularly simple if pivoting is not required during the factorization, so that $A = LU$, for then L and U have the same banded form as A. In many cases the factorization will be stable without pivoting (e.g., if A is diagonally dominant or symmetric positive definite) and it often pays to *try* a factorization without pivoting, checking the growth of the elements as it proceeds.

This simple factorization can also be used when only *part* of A is banded: for example, consider $Ax = b$,

$$A = \begin{pmatrix} B & c \\ d^T & e \end{pmatrix}, \qquad b = \begin{pmatrix} f \\ g \end{pmatrix}, \qquad x = \begin{pmatrix} y \\ z \end{pmatrix},$$

with B banded. We can use the LU factorization for B to get x as follows: first form $B = LU$; then using this, solve for $p = B^{-1}c$, $q = B^{-1}f$. Now we can get z from $z = (g - d^T q)/(e - d^T p)$ and y from $y = q - zp$. Of course, one could also perform the LU decomposition of A itself with the same saving of time, but the above illustrates how *partitioning* the matrix can lead to improvements.

(ii) *A block-banded.* Sometimes A will have a banded form where each nonzero element is itself a matrix—the simplest case is block-tridiagonal,

$$A = \begin{pmatrix} B_1 & C_1 & 0 & \\ A_2 & B_2 & C_2 & \\ 0 & \ddots & \ddots & \ddots \end{pmatrix}.$$

Working with the block-elements, one can generate a block-LU factorization

$$A = \begin{pmatrix} I & & 0 & \\ L_2 & I & & \\ & L_3 & I & \\ 0 & \ddots & \ddots & \end{pmatrix} \begin{pmatrix} U_1 & C_1 & 0 & \\ & U_2 & C_2 & \\ & & \ddots & \ddots \\ 0 & & & \end{pmatrix}.$$

Of course one must be careful that the factorization is stable just as in the scalar case. Notice that no *fill-in* has occurred; the nonzero elements in the factorized matrices occupy the same locations as in A.

A variant of this type of matrix occurs in systems arising in the solution of boundary value problems in ordinary differential equations by the collocation method. Then A has the form

$$A = \begin{pmatrix} F_0 & & & & & \\ F_1 & G_1 & & & & \\ & F_2 & \ddots & & & \\ & & \ddots & \ddots & & \\ & & & F_{n-1} & G_{n-1} & \\ & & & & G_n & \end{pmatrix},$$

with F_i, G_i $p \times p$ for $i = 1,\ldots, n-1$, and F_0 $q \times p$, G_n $(p-q) \times p$. Again, we can factor A in a stable way without changing the structure by *alternating* the pivoting and elimination procedure between rows and columns. See Varah [18] and Diaz, Fairweather, and Keast [2].

2C. *Sparse Systems*

For large sparse systems of equations, there has been an enormous amount of research done in the last several years, and we can only direct the reader to some

good references. The material naturally divides itself into direct methods, where variants of the LU factorization are used directly, and iterative methods, where successive approximations of the solution are formed.

(i) *Direct Methods.* For general symmetric positive definite systems, where pivoting is not required, a tremendous saving can be made by *ordering* the equations and unknowns differently, so that when the factorization is performed, the amount of work or storage required is lessened. One technique for doing this is the minimal degree algorithm, which at each stage looks for the row with the fewest nonzero elements.

When more structure is imposed on the matrix A, more specialized methods can be used to great advantage. One example of this is the nested dissection algorithm of George [7] which, applied to systems arising from $n \times n$ grid problems (finite element problems, e.g.), reduced the time required from $O(n^4)$ to $O(n^3)$. Another example is the fast Poisson solvers, specialized for solving the five-point discrete approximation to the Poisson equation $\nabla^2 u = f$ on a rectangle. Using either a fast Fourier transform approach or the cyclic reduction approach, the time required is reduced to $O(n^2 \log n)$. Codes for these Poisson solvers are available, for example, from Lawrence Livermore Labs. See Swartztrauber [16].

Good references for direct methods on sparse matrices are George and Liu [8] and Duff [4]. Duff also has papers on iterative methods. In addition, some codes are now available: the Yale Sparse Matrix Package, available from the IMSL library, Houston (see also Eisenstat, *et al.* [5]); SPARSPAK, developed by George and Liu, available from the Waterloo Research Institute, University of Waterloo, Waterloo, Ontario, Canada; and the Harwell Sparse Matrix Codes, available from the Computer Science and Systems Division, UKAEA Harwell, England.

(ii) *Iterative Methods.* Here we attempt to approximate the solution of $Ax = b$ by a sequence of vectors $x^{(k)}$, $k = 0, 1, 2, \ldots$. The standard techniques like SOR and ADI are well known and well described in [19]. More recently, there has been interest in solving symmetric positive definite systems by the conjugate gradient technique. Here $x^{(k)}$ is obtained from $x^{(k-1)}$ and $Ax^{(k-1)}$ in such a way that we minimize the quadratic form $(x - \bar{x})^{\mathrm{T}} A(x - \bar{x})$ for $\bar{x} = x^{(k)}$ over the k-dimensional subspace spanned by $\{x^{(1)}, Ax^{(1)}, \ldots, A^{k-1}x^{(1)}\}$. Also convergence to the solution x occurs in at most n steps, and in k steps if there are only k distinct eigenvalues of A.

Because of this last property, it seems to be a good idea to *precondition* the problem: iterate using $L^{-1}AL^{-\mathrm{T}}$ rather than A, choosing L so that the matrix has eigenvalues clustered near unity. For example, one could use L from the Cholesky factorization of a nearby matrix $A + \delta A$. This is called the incomplete Cholesky factorization, and references are Meijerink and van der Vorst [14] and Manteuffel [13].

For unsymmetric sparse systems, in particular systems arising from elliptic PDEs with large first order terms, some success has been reported by using the splitting

$A = M - N$, with $M = (A + A^T)/2$ (if this is positive definite) and $N = -((A - A^T)/2)$, giving the iteration

$$Mx^{(k+1)} = Nx^{(k)} + b,$$

and accelerating this with the usual Chebyshev acceleration technique (see Manteuffel [12]).

3. Least Squares and the Singular Value Decomposition

The basic linear least squares problem is to solve the overdetermined system $Ax = b$, A $m \times n$ with $m > n$, by choosing x to minimize $\|Ay - b\|_2$ over all possible n-vectors y. There are two basic methods for this, assuming A has full column rank $(=n)$: forming and solving the normal equations $A^TAx = A^Tb$, or performing a QR factorization $A = QR$ (Q orthogonal, R upper triangular) and solving the first n equations of $Rx = Q^Tb$. When the columns of A are nearly linearly dependent (so A^TA is badly conditioned) the QR technique is more accurate, particularly if the overdetermined system is nearly consistent. However, this dependency usually means one is working with too many variables x and the ill-conditioning problem can be removed by decreasing the number of variables.

When A is a large sparse matrix, then A^TA is also probably sparse, whereas the QR factors usually are not, so the normal equations become more attractive. Recently, work has been done on more efficient ways to solve these normal equations in the sparse case; for example, by using a preconditioned conjugate gradient technique (see article by Björck in Duff [4]).

To determine whether the columns of A are linearly dependent, or more generally what the effective rank of A is, the most useful technique is that of the singular value decomposition (SVD): this forms $A = UDV^T$ with U and V orthogonal and D diagonal with diagonal elements $\sigma_i = \textit{singular values}$ of A $(\sigma_1 \geqslant \sigma_2 \cdots \geqslant \sigma_n \geqslant 0)$. If ε is the noise level, a good measure of rank A is the number of singular values larger than ε. The SVD algorithm is due to Golub and his co-workers (see, e.g., Golub and Reinsch [9]) and is available in the IMSL library or the NAG library in Oxford. A good reference for this least squares material is Lawson and Hanson [10].

One interesting application of the SVD is to ill-posed problems (e.g., some inverse problems in geophysics). We are given an $n \times n$ system $Ax = b$ with A ill-conditioned and we want an approximate solution x of reasonable size with $\|Ax - b\|$ small, with x perhaps expressed in terms of only the low-order modes (eigenvectors) of the system. If one computes the SVD, the system becomes $D(V^Tx) = U^Tb$, or $Dy = z$, and a "good" solution is obtained by taking $y_i = z_i/\sigma_i$ for $i = 1,..., k$, and $y_i = 0$ for $i > k$. Then our solution is $x^{(k)} = \sum_1^k (z_i/\sigma_i)\, v^{(i)}$, i.e., in terms of only the first k singular vectors of A. Typically k is chosen so that we include those singular vectors whose corresponding singular values are above the noise level. See Varah [17].

4. Eigenvalue Problems

For dense stored matrices, there are standard QR-type routines available for the $Ax = \lambda x$ problem, for example, in EISPACK (available from Argonne National Labs). This has various routines dependent on the structure and symmetry of A. For A symmetric, the eigenvalues λ_i are well-conditioned; that is, when the elements of A are changed by $O(\varepsilon)$, each $\lambda_i \to \lambda_i + \varepsilon_i$, where $\varepsilon_i = O(\varepsilon)$. Thus if the elements of A are known to 5 digits, the eigenvalues should also be accurate to about 5 digits. This is *not* true of the eigenvector components; they are only well conditioned for eigenvalues which are well separated. For eigenvalues which are close together, the individual eigenvectors are poorly determined by the data; only the invariant subspace spanned by the set of eigenvectors is well determined.

For a nonsymmetric matrix A, the QR method produces an orthogonal similarity transformation to triangular form, which gives the eigenvalues on the diagonal, and from which inverse iteration can be used to produce the eigenvectors. Now the conditioning is more delicate; the eigenvalues may be poorly conditioned, but this can be checked by calculating the *condition numbers* s_i for each λ_i. If $x^{(i)}$ and $y^{(i)}$ are the normalized right and left eigenvectors, then s_i is the *cosine* of the angle between them,

$$s_i = \frac{y^{(i)\mathrm{T}} x^{(i)}}{\| y^{(i)} \|_2 \| x^{(i)} \|_2}.$$

For noise level ε in the elements of A, the eigenvalue λ_i will only be correct to $O(\varepsilon / s_i)$. Notice that for A symmetric, $x^{(i)} = y^{(i)}$ so $s_i = 1$. Most codes do not automatically compute these s_i, but it does seem a good idea. Alternatively, as in solving linear equations, one can instead compute the eigenvalues for A and a perturbation $A + \delta A$ and gauge the accuracy in each λ_i by the changes in the computed eigenvalues.

Codes also exist (in EISPACK, e.g.) for the generalized eigenvalue problem $Ax = \lambda Bx$. Usually A is symmetric and B symmetric and positive definite. If B is positive definite, one method is to use the Cholesky factorization $B = LL^{\mathrm{T}}$ and convert the problem back to the normal (symmetric) problem $L^{-1}AL^{-\mathrm{T}}x = \lambda x$. However, when B is nearly singular (yet still positive definite), this clearly produces large elements and so gives unsatisfactory results, even though *some* of the eigenvalues λ_i may be well determined by the problem. One alternative in this case is the QZ algorithm of Moler and Stewart [11] which simultaneously triangularizes A and B, and which actually works for general complex matrices A and B.

For large sparse symmetric matrices, the Lanczos algorithm has enjoyed renewed popularity lately. This forms a sequence of vectors $\{v^{(i)}\}$ spanning the space $\{v^{(1)}, Av^{(1)}, A^2v^{(1)}, \ldots\}$ by the iteration

$$\beta_{i+1} v^{(i+1)} = Av^{(i)} - \alpha_i v^{(i)} - \beta_i v^{(i-1)},$$

with β_i chosen so that $\|v^{(i)}\|_2 = 1$ and $\alpha_i = v^{(i)T} A v^{(i)}$. Then the tridiagonal matrix

$$T = \begin{pmatrix} & \ddots & & \ddots & \\ \ddots & & \alpha_i & & \beta_i \\ & \beta_i & & \ddots & \ddots \\ \mathrm{O} & & \ddots & & \end{pmatrix}$$

has eigenvalues which approximate those of A. A good reference for this material, and for symmetric matrices in general, is Parlett [15].

Finally, lest we leave the impression that all these problems are solved, consider the generalized eigenvalue problem $Ax = \lambda Bx$ with A and B banded, very large, and unfortunately not symmetric. Such problems arise in the finite element solution of structures problems in civil engineering, and one usually wants the eigenvalues λ of algebraically largest real part. If A and B were symmetric, we could use the Sylvester inertia theorem to estimate the eigenvalues (see Parlett [15]). And, if A and B were not too large, we could resort to the QZ algorithm mentioned earlier. Instead, one is left with very little in the way of proven techniques; about the best we can do is to try a generalized Rayleigh quotient iteration, in which we guess vectors $x^{(0)}$, $y^{(0)}$, and $k = 0, 1, 2, \ldots$, set

$$\lambda_{k+1} = \frac{y^{(k)T} A x^{(k)}}{y^{(k)T} B x^{(k)}}$$

and solve

$$(A - \lambda_{k+1} B)\, x^{(k+1)} = x(k), \qquad y^{(k+1)T}(A - \lambda_{k+1} B) = y^{(k)T}.$$

Hopefully $\lambda_k \to \lambda$ desired if we start fairly close to it, although this is not at all assured.

References

1. G. Cybenko, *SISSC* **1** (1980), 303.
2. Diaz, Fairweather, and Keast, "Fortran Packages for Solvig Almost Block Diagonal Linear Systems by Modified Alternate Row and Column Elimination," Technical Report 148/81, Comp. Sci. Dept., Univ. of Toronto, 1981.
3. Dongarra, Bunch, Moler, and Stewart, "LINPACK Users Guide," Soc. Ind. Appl. Math. Philadelphia, 1979.
4. I. Duff, "Sparse Matrices and their Uses," Prentice–Hall, Englewood Cliffs, N.J., 1980.
5. Eisenstat, Gursky, Schultz, and Sherman, *TOMS*, (1981), in press.
6. G. Forsythe and C. Moler, "Computer Solution of Linear Algebraic Systems," Prentice–Hall, Englewood Cliffs, N.J., 1967.
7. J. A. George, *SINUM* **10** (1973), 345.
8. A. George and J. Liu, "Computer Solution of Large Sparse Positive Definite Systems," Prentice–Hall, Englewood Cliffs, N.J., 1981.

9. GOLUB AND REINSCH, *Numer. Math.* **14** (1970), 403.
10. C. LAWSON AND R. HANSON, "Solving Linear Least Squares Problems," Prentice–Hall, Philadelphia, 1974.
11. MOLER AND STEWART, *SINUM* **10** (1973), 241.
12. T. MANTEUFFEL, *Numer. Math.* **28** (1977), 307.
13. T. MANTEUFFEL, Shifted incomplete Cholesky factorization, Sparse Matrix Proceedings 1978 (Duff and Stewart, Eds.), Soc. Ind. Appl. Math., Philadelphia, 1978.
14. MEIJERINK AND VAN DER VORST, *Math. Comput.* **31**(1977), 148.
15. B. PARLETT, "The Symmetric Eigenvalue Problem," Prentice–Hall, Englewood Cliffs, N.J., 1980.
16. P. SWARTZTRAUBER, *SIAM Rev.* **19** (1977), 490.
17. J. VARAH, *SINUM* **10** (1973), 257.
18. J. VARAH, *SINUM* **13** (1976), 71.
19. L. HAGEMAN AND D. YOUNG, "Iterative Methods for Linear Systems," Academic Press, New York, 1982.