JOURNAL OF COMPUTATIONAL PHYSICS 70, 138–144 (1987)

A New Splitting to Solve a Large Hermitian Eigenproblem

C. M. M. NEX

Cavendish Laboratory, Madingley Road,
Cambridge CB3 OHE, United Kingdom

Received April 30, 1986; revised September 26, 1986

A particular case of Ruhe's convergent splitting is presented, which gives a rapidly converging method of estimating the lowest eigenvalues and corresponding vectors of a large hermitian matrix, when a leading principal minor of that matrix provides a good approximation to the desired eigensolution. This is particularly relevant, for example, to the total energy calculations of solids. © 1987 Academic Press, Inc.

1. INTRODUCTION

In solid state physics matrices arise (e.g., in total energy calculations [1]) which are large and hermitian, and for which the lowest few eigensolutions are required. They are generated in a discretisation process which has the property that successive approximations of increasing accuracy incorporate the earlier matrix as a leading principal minor. Thus lower dimensional matrix problems may be solved to give an initial estimate of the more accurate solution required. This is a common situation in finite difference and expansion methods for solving differential equations, when one may use relatively few points to obtain a crude approximate solution, and then add points or terms to obtain a more accurate one.

The present version of Ruhe's splitting algorithm [2] allows the solution to the smaller problem to be efficiently incorporated in the calculation of the eigensolution of the large one of real interest. In its block form it may be regarded as a form of subspace iteration [3], with shifts of origin (as in inverse iteration) designed to accelerate convergence to the eigenvalues of interest; these are obtained from a Rayleigh–Ritz type of formulation from the iteration vectors. The method may also be viewed as an iterative improvement of the "folding-down" of physical perturbation theory [4] or as repeated application of Brillouin–Wigner perturbation theory, (see, e.g., Ziman [5]).

This algorithm requires only the equivalent of a matrix-vector product at each step, while inverse iteration and recent improvements to the Lanczos algorithm [6, 7] require a matrix factorisation, an order n^3 process. For the large, full matrices arising in total energy calculations this would be prohibitively expensive. While the ordinary Lanczos algorithm [3] is quite effective on these problems, it still takes more steps than does, for instance, inverse iteration. Some work is

0021-9991/87 $3.00

typically involved in providing approximations to eigenvalues at the top of the spectrum, which are not of interest. The present algorithm is an attempt to combine some of the speed of inverse iteration with the efficiency of the Lanczos algorithm for problems with the structure defined above.

The rate of convergence of the algorithm appears to be close to that of inverse iteration, given the same starting approximation, but avoids the time-consuming matrix inversion that method requires.

In the following sections we first present the algorithm, both in a simple single vector form and a block form. Then the relationship to folding-down is indicated, and finally we give some numerical results for some relatively small (of dimension about 200) test matrices provided by R. Needs.

2. The Algorithm

The matrix is partitioned so that the leading principal minor is treated "exactly," while the rest of the matrix is incorporated in the iterative scheme to find the lowest eigenvalues. This corresponds to a "block" Gauss–Seidel splitting in the sense of Ruhe [2]. The $n \times n$ matrix H of the eigenproblem is first partitioned so that A is $m \times m$ and B is $n-m \times n-m$, and the n-vectors correspondingly ($\dagger$ indicates complex conjugate transpose and the superscript refers to the partitioned vectors)

$$H = \begin{pmatrix} A & C \\ C^\dagger & B \end{pmatrix} \quad \text{and} \quad \text{vectors} \quad \mathbf{x} = \begin{pmatrix} \mathbf{x}^1 \\ \mathbf{x}^2 \end{pmatrix}. \tag{2.1}$$

The dimension, m, of A is assumed large enough that its eigensolution provides an adequate initial approximation to that of H, while it is small enough that A can be retained in fast store, and that time spent in its factorisation is not excessive. B is further partitioned into its diagonal elements (D) and its strict upper and lower triangles (U and $\mathrm{U}^\dagger$)

$$B = \mathrm{D} + \mathrm{U} + \mathrm{U}^\dagger. \tag{2.2}$$

With this partitioning scheme we have a matrix which is computationally convenient to invert at each step of the algorithm. The two matrices of the splitting are, in Ruhe's notation at the iteration step s,

$$V_s = \begin{pmatrix} A - (\lambda_s + r_s^{(1)})\, I & C \\ 0 & \mathrm{D} + \mathrm{U} - (\lambda_s + r_s^{(2)})\, I \end{pmatrix} \quad \text{and} \quad H_s = -\begin{pmatrix} r_s^{(1)} I & 0 \\ C^\dagger & \mathrm{U}^\dagger + r_s^{(2)} I \end{pmatrix}. \tag{2.3}$$

The iteration is then $V_s \mathbf{x}_{s+1} = H_s \mathbf{x}_s$.

The two r parameters are small and are chosen at each step to avoid singularity of the matrix to be inverted. Note also that the Gauss–Seidel algorithm proceeds

from the bottom–up, which agrees with the ideas of physical perturbation theory. Explicitly, the sth step of Ruhe's procedure for a single vector becomes

$$(\mathrm{D}-\lambda_s I-r_s^{(2)} I)\,\mathbf{x}_{s+1}^2 = -C^\dagger \mathbf{x}_s^1 - (\mathrm{U}^\dagger + r^{(2)} I)\,\mathbf{x}_s^2 - \mathrm{U}\mathbf{x}_{s+1}^2 \tag{2.4a}$$

$$(A-\lambda_s I-r_s^{(1)} I)\,\mathbf{x}_{s+1}^1 = -C\mathbf{x}_{s+1}^2 - r_s^{(1)}\,\mathbf{x}_s^1. \tag{2.4b}$$

The Rayleigh quotient can be used to estimate the next eigenvalue approximation, $\lambda_{s+1} = \mathbf{x}_{s+1}^\dagger H\,\mathbf{x}_{s+1}$, as all the necessary matrix-vector products have been calculated in forming $\mathbf{x}_{s+1}$,

$$\lambda_{s+1} = \mathrm{Real}\,\frac{\begin{pmatrix}(r_s^{(1)}+\lambda_s)\,\mathbf{x}_{s+1}^{1\dagger}\,\mathbf{x}_{s+1}^1 + \mathbf{x}_{s+1}^{1\dagger}\,C\mathbf{x}_{s+1}^2 + \mathbf{x}_{s+1}^{2\dagger}\,\mathrm{D}\mathbf{x}_{s+1}^2 \\ + 2\mathbf{x}_{s+1}^{2\dagger}\,\mathrm{U}\mathbf{x}_{s+1}^2 - r_s^{(1)}\,\mathbf{x}_{s+1}^{1\dagger}\,\mathbf{x}_s^1\end{pmatrix}}{\mathbf{x}_{s+1}^\dagger\,\mathbf{x}_{s+1}}. \tag{2.4c}$$

As all the matrix vector products in (2.4c) have been calculated in the earlier steps, the total time is dominated by that required for a matrix times a vector. The inversion implicit in (2.4b) may be computed at each step of the iteration (using Gaussian elimination, e.g.), or it may be conveniently effected using a complete eigensolution of A, calculated once at the commencement of the process (using reduction to tridiagonal form and the QR algorithm). In either case A is a relatively small matrix whose size determines which of these approaches is the more appropriate: the time for this computation should be much less than that required in the other parts of the operation—that for a large matrix times a vector multiplication.

In the block version of the algorithm a small two-sided eigenproblem is constructed at each step of the algorithm (R and Λ are the obvious diagonal matrices and X is the matrix whose columns are the vector iterates, with the superscripts referring to the partitioning)

$$\mathrm{D}\,X_{s+1}^2 - X_{s+1}^2\,(\Lambda_s + R_s^2) = -C^\dagger X_s^1 - X_s^2 R_s^2 - \mathrm{U}^\dagger X_s^2 - \mathrm{U}\,X_{s+1}^2 \tag{2.5a}$$

$$A\,X_{s+1}^1 - X_{s+1}^1\,(\Lambda_s + R_s^1) = -C\,X_{s+1}^2 - X_s^1 R_s^1. \tag{2.5b}$$

Instead of λ_{s+1} the matrix $H^* = X_{s+1}^\dagger H\,X_{s+1}$ can be accumulated as

$$\begin{aligned} H^* = {} & X_{s+1}^{1\dagger} X_{s+1}^1 (\Lambda_s + R_s^1) + (X_{s+1}^{1\dagger} C\,X_{s+1}^2)^\dagger + X_{s+1}^{2\dagger}\,\mathrm{D}\,X_{s+1}^2 \\ & + (X_{s+1}^{2\dagger}\,\mathrm{U}\,X_{s+1}^2)^\dagger + X_{s+1}^{2\dagger}\,\mathrm{U}\,X_{s+1}^2 - X_{s+1}^{1\dagger} X_s^1 R_s^1. \end{aligned} \tag{2.5c}$$

The small $(m \times m)$ eigenproblem is solved (with $S_{s+1}^* = X_{s+1}^\dagger X_{s+1}$)

$$H_{s+1}^*\,Y_{s+1} = S_{s+1}^*\,Y_{s+1}\,\Lambda_{s+1} \tag{2.6}$$

to give an orthonormal set of eigenvectors Y_{s+1} (the problem is still hermitian) and a new approximation to the eigenvalues Λ_{s+1}. Then the new iterate X_{s+1}^* is given by $X_{s+1}^* = X_{s+1}\,Y_{s+1}$. The matrix X_{s+1}^* and new approximation to the eigenvalues Λ_{s+1} have the useful property that they satisfy

$$\Lambda_{s+1} = X_{s+1}^{*\dagger} H\,X_{s+1}^*.$$

3. Relationship to Other Methods

The folding-down algorithm [4] relates the eigensolution of a smaller, non-linear problem, to that of the original problem using Brillouin–Wigner perturbation theory [5]

$$[A + C(\lambda I - B)^{-1} C^{\dagger}]\, \mathbf{x} = \lambda \mathbf{x}. \tag{3.1}$$

The inversion implicit in this formula is then approximated in some way, usually by choosing a value of λ close to the eigenvalue(s) of interest and then computing the inverse either of $(\lambda I - B)$ or of the diagonal approximation to it. In the latter case we obtain a simple way of calculating a second approximation, λ_1, to an eigenvalue of H, from the first approximation (λ_0 is an eigenvalue, and $\mathbf{x}_0$ an eigenvector of the sub-matrix A)

$$\lambda_1 = \lambda_0 + \mathbf{x}_0^{\dagger}[C(\lambda_0 I - D)^{-1} C^{\dagger}]\, \mathbf{x}_0. \tag{3.2}$$

This is the same result we would obtain by using only the first two terms in (2.4c) and neglecting U in (2.4a), taking the vectors $\mathbf{x}_s^1 = \mathbf{x}_0$ and $\mathbf{x}_s^2 = \mathbf{0}$ (setting the shifts $r^{(i)}$ to zero). Folding down may thus be viewed as the first step of a simplified splitting procedure. This straightforward approximation provides an effective first improvement to the eigenvalue estimate in any iterative scheme, and we advocate using it to start the current process.

A method proposed by Davidson [8], before Ruhe [2] provided a general theory of such algorithms, proceeds with a Jacobi-type splitting, rather than the Gauss–Seidel one of the current method. As in the case of linear equations, we would expect the latter to be significantly better than the former, particularly with the bottom-up application particularly suitable for matrices arising in the total energy calculations [1]. In addition Ruhe's theory [2] enables shifts (r_s of (2.3)) to be incorporated to avoid problems of near-singularity in the matrix implicitly inverted in each algorithm. Davidson's method also starts with the solution of the eigenproblem for a submatrix, but without the folding-down refinement. The dimension of this matrix is the same as the number of eigenvectors required, but at each step of the algorithm the dimension of the sub-eigenproblem increases by 1. In the new splitting we suggest initially solving a slightly larger sub-problem, but at each iterative step the dimension of the eigenproblem is only the number of eigenvectors required.

4. Numerical Results

Three test matrices of dimension 180, 210, and 211 were kindly provided by Richard Needs, and were used to assess the performance of the algorithm in some practical examples. The new method was compared with inverse iteration from the

TABLE I

Number of Iterations Needed to Achieve an Accuracy of 10^{-6} in Corresponding Eigenvalues of Three Test Matrices, Using Inverse Iteration and the New Splitting.

	180 matrix		210 matrix		211 matrix	
Eigenvalue	inverse iteration	new splitting	inverse iteration	new splitting	inverse iteration	new splitting
1	4	5	3	4	6	6
2	3	4	3	4	5	5
3	4	5	4	5	5	5
4	5	5	4	5	5	5
5	4	5	3	5	7	6
6	3	6	4	5	7	6
7	7	5	4	6	7	6
8	6	10	4	6	5	7

same starting vectors, and with only one matrix factorisation per eigenvalue (with shift determined by folding-down) in the latter algorithm. The diagonal matrices (shifts) R_s^i in (2.5) were chosen automatically so that $A_s^i + R_s^i$ was not in a neighbourhood of zero (the performance was not sensitive to the precise size of the shifts, as long as they were reasonably small). Table I gives the number of iterations per eigenvalue for each of the first 8 eigenvalues of each matrix and Table II shows the approximations to the eigenvalues by the 21×21 leading minor and the result of folding down. From the same starting vectors, both inverse iteration and the new block splitting converged to the same eigensolutions.

A standard Lanczos procedure (EA15AD) from the Harwell subroutine library [9] was also used for comparison. On the test matrices this typically used 65 matrix-vector products, compared with the 45 (see Table I) needed by the new algorithm. The computation time for both algorithms was dominated by these figures. Typical overall timings on an IBM 3081 (finding the lowest 8 eigensolutions of a matrix of order 200) were 8 sec for the new algorithm, while for inverse iteration with only one matrix factorisation per eigenvalue, the figure was 16 sec. The standard NAG [10] reduction to tridiagonal form and subsequent solution took of order 20 sec.

As the results of Table I indicate, the new splitting requires a few more iterations per eigenvalue than inverse iteration, which converges reasonably fast, given the good starting approximations. The factor seems to vary from unity to 1.5. The large saving in time for the present algorithm comes from not needing to perform the initial factorisation of the matrix (taking of order n^3 operations) used in inverse iteration. The time per step is dominated by that required for a matrix-vector multiplication (of order n^2 operations) and as these are reasonably straightforward they can be implemented efficiently in "pipe-lining" and "parallel" computers. The block

TABLE II

Lowest Eigenvalues of Test Matrices and Their Initial Approximations

Matrix size	21×21 sub-matrix	Folding back	Accurate
180×180	−0.0593164	−0.16579	−0.1468799
	0.5256818	0.45172	0.4507970
	0.6421063	0.57933	0.5748814
	0.7121972	0.63058	0.6238321
	0.9912691	0.88472	0.8814751
	1.0476261	0.92751	0.9269680
	1.0890217	0.98800	0.9747979
	1.2466530	1.1104	1.0973136
			1.2052475
			1.2948314
210×210	0.0418995	−0.056156	−0.0451855
	0.1198890	0.054332	0.0594346
	0.1650400	0.099727	0.1049581
	0.1988641	0.13539	0.1416183
	0.2545238	0.20033	0.2043467
	0.2811117	0.24499	0.2415569
	0.3272672	0.29675	0.2921596
	0.4647282	0.36469	0.3502600
			0.3696753
			0.3874926
211×211	0.001148074	−0.37224	−0.6471234
	0.003747227	−0.41601	−0.4977474
	0.003967037	−0.41764	−0.5036803
	0.004772899	−0.41716	−0.5031567
	0.06144023	−0.38896	−0.5922293
	0.06265675	−0.38878	−0.5918449
	0.06303806	−0.38873	−0.5917248
	0.5453997	0.41891	0.3409741
			−0.5183999
			0.3411610
			0.3417584

form of the algorithm enables degenerate and near-degenerate eigenvalues to be computed with no special modifications.

For the type of large matrices occurring in total energy calculations, where a principal minor provides a reasonable approximation to the eigensolution, we conclude that the new splitting can provide advantages in speed over other methods in finding the lowest few eigenvalues and vectors.

ACKNOWLEDGMENTS

I am very grateful to Volker Heine and Richard Needs for posing this problem and to Richard Needs for providing the test matrices used as examples.

REFERENCES

1. J. IHM, A. ZUNGER AND M. L. COHEN, *J. Phys. C* **12**, 4409 (1979).
2. A. RUHE, *J. Comput. Phys.* **19**, 110 (1975).
3. B. N. PARLETT, The Symmetric Eigenvalue Problem (Prentice–Hall, Englewood Cliffs, NJ, 1980).
4. M. L. COHEN AND V. HEINE, "The Fitting of Pseudopotentials to Experimental Data and Their Subsequent Application," *Solid State Physics Vol.* **24**, edited by H. Ehrenreich, F. Seitz, and D. Turnbull (Academic Press, New York, 1970), p. 245.
5. J. M. ZIMAN, *Elements of Advanced Quantum Theory* (CUP, Cambridge, 1969).
6. T. ERICSSON AND A. RUHE, *Math. Comput.* **34**, 1251 (1980).
7. B. NOUR-OMID, B. PARLETT AND R. TAYLOR, *Int. J. Numer. Method Eng.* **19**, 859 (1983).
8. E. R. DAVIDSON, *J. Comput. Phys.* **17**, 87 (1975).
9. M. J. HOPPER, ED., *Harwell Subroutine Library* (UKAEA, England, 1984).
10. *NAG Fortran Library* (Oxford, 1983).