J. Phys. A: Math. Gen. 23 (1990) 5479-5491. Printed in the UK

Symmetric matrix methods for Schrödinger eigenvectors

Brian Grieves and Dennis Dunn

Physics Department, University of Reading, Whiteknights, Reading RG6 2AF, UK

Received 30 March 1990

Abstract. The one-dimensional single-particle Schrödinger eigenvalue equation is represented by a *generalized matrix eigenvalue equation* of the form

$$A\Psi = Ea\Psi$$

where A is a symmetric matrix, a is a symmetric positive-definite matrix and Ψ is a column vector. The local error in making the representation is proportional to δ^4 where δ is the step size. Several methods of solution which make use of the *sparsity* of the matrices are investigated.

1. Introduction

The determination of the eigenvalues and eigenfunctions of the single-particle Schrödinger equation is an important aspect of quantum physics. Such solutions are often the starting point for many-body calculations and there is considerable interest in the computational aspects of the solution [1-4].

In situations of special symmetry the three-dimensional Schrödinger equation can be reduced to a one-dimensional (radial) equation and in this paper we restrict ourselves to such one-dimensional equations. There are, of course, situations in physics where the problem is inherently one-dimensional—polymers for example—and in such situations the solutions of the one-dimensional single-particle Schrödinger equation would be the natural starting point for investigations of electron correlations.

A typical requirement is for the eigenfunctions corresponding to the energy eigenvalues in a certain range. If integrals over eigenfunctions are needed then it is convenient if these eigenfunctions are all evaluated at the same set of coordinate points.

The 'typical' method of solution is a 'shooting method' based on the Numerov-Cooley algorithm or some modification of this [3, 5, 6]. The method is to guess a value for an energy eigenvalue, to integrate the equation outwards from the origin and inwards from 'infinity' and to attempt to match the two solutions at some intermediate point. The degree of mismatch provides some estimate for the next 'guess' and the procedure is iterated until the mismatch is small enough.

Improved versions of this shooting method involve working with a phase function rather than with the wavefunction itself [7-9]. A discussion of some of the numerical problems which can arise in this approach has been given by Pruess [21]. In such schemes one eigenvalue and eigenfunction are evaluated at a time and in many of the schemes the step-size is variable. This means that different eigenfunctions are generally determined at different sets of coordinates and so the evaluation of integrals involving the eigenfunctions is inconvenient.

We present a scheme, based on symmetric matrices, whereby all the eigenvalues in a certain energy range, together with their eigenfunctions, are determined simultaneously, and in which the eigenfunctions are evaluated at a common set of coordinates. The idea for such a scheme was introduced by Cooney *et al* [1] but their method had a much poorer accuracy than the Numerov-Cooley algorithm. Our method is to replace the differential equation by a difference equation, as in the Numerov algorithm, with a fixed step size, and to write the set of difference equations as a single *symmetric generalized matrix eigenvalue equation.* There are many library subroutines for the numerical solution of such matrix eigenvalue equations. A brief introduction to our new method has been presented previously [10].

The differential Schrödinger equation is replaced by a matrix equation

$$A\Psi = Ea\Psi \tag{1.1}$$

where A is a *symmetric matrix,* $\boldsymbol{a}$ is a *symmetric positive-definite matrix* and Ψ is a *column vector.* The local error in representing the differential equation by the difference equation is proportional to δ^4 where δ is the step size.

We investigate *five* schemes for the determination of a selection of the eigenvalues and eigenvectors of (1.1). Four of these schemes make use of the *sparsity* of the matrices; this is important since if high accuracy is required then N, the order of the matrices, may need to be very large and it may not be possible to store the complete matrices. The fifth scheme, which uses EISPACK subroutines [11], does not make use of the sparsity and is included simply to show the limitations of such schemes.

2. The algorithm

In dimensionless units the Schrödinger equation is

$$-\frac{\mathrm{d}^2}{\mathrm{d}x^2}\Psi(x) + V(x)\Psi(x) = E\Psi(x). \tag{2.1}$$

We want to transform this *differential* eigenvalue equation into a *difference* equation. A simple procedure for doing this, first presented by Cooney *et al* [1], is to approximate the second derivative at a point $x = x_n$ by

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\Psi(x_n) = \frac{(\Psi_{n+1} + \Psi_{n-1} - 2\Psi_n)}{\delta^2} \tag{2.2}$$

where $\Psi_n = \Psi(x_n)$ and $x_{n+1} - x_n = \delta$.

The resulting equation can be put into matrix form

$$A\Psi = E\Psi \tag{2.3}$$

where A is an $N \times N$ *symmetric* matrix and Ψ is a column vector. There are many standard library procedures for determining the eigenvalues and eigenvectors of this eigensystem. Unfortunately equation (2.3) is a poor representation of the original differential equation (2.1): the leading term in the local error relative to $E\Psi_n$ is

$$\frac{\delta^2}{12}\Psi_n^{(4)} \tag{2.4}$$

where $\Psi^{(4)}$ denotes the fourth derivative. In order to produce accurate results δ is required to be very small and hence, for a given x-range, the order of the matrix very large: the size of the x-range is $N\delta$. An improvement on this simple algorithm is the Numerov-Cooley algorithm [5, 6]. In this the difference equation representing the Schrödinger equation is

$$-\frac{(\Psi_{n+1}+\Psi_{n-1}-2\Psi_n)}{\delta^2}$$

$$=\tfrac{5}{6}(E-V_n)\Psi_n+\tfrac{1}{12}(E-V_{n+1})\Psi_{n+1}+\tfrac{1}{12}(E-V_{n-1})\Psi_{n-1} \tag{2.5}$$

and the local error at $x=x_n$ is reduced to

$$\frac{\delta^4}{12}\Psi_n^{(6)}. \tag{2.6}$$

The matrix form of this equation is

$$A\Psi = E\boldsymbol{a}\Psi \tag{2.7}$$

where A and $\boldsymbol{a}$ are *tridiagonal matrices*. $\boldsymbol{a}$ is *symmetric* and *positive-definite* but the matrix A is *asymmetric*. Methods exist for the solution of this *asymmetric generalized eigensystem*, but they are not nearly as efficient as the methods available for the symmetric case. The reason is that for the *asymmetric* case there is no guarantee that a complete set of solutions to (2.7) exists whereas when A is *symmetric* and $\boldsymbol{a}$ is *symmetric* and *positive-definite* there is a *complete set* of eigensolutions. For a discussion see Wilkinson [22] or Dennery and Krzywicki [23]. We therefore seek a second-order difference equation representation of the Schrödinger equation with the same order of local error (δ^4) as the Numerov-Cooley algorithm but which yields matrices A and $\boldsymbol{a}$ which are *tridiagonal and symmetric* and $\boldsymbol{a}$ *positive-definite.*

The non-zero elements of the required matrices A and $\boldsymbol{a}$ are denoted by

$$\begin{aligned} A_{nn} &= A_n \\ A_{n\,n+1} &= A_{n+1\,n} = B_n \\ a_{nn} &= a_n \\ a_{n\,n+1} &= a_{n+1\,n} = b_n \end{aligned} \tag{2.8}$$

so that a section of the matrix equation looks like

$$\begin{bmatrix} B_{n-2} & A_{n-1} & B_{n-1} & \dots & \dots \\ \dots & B_{n-1} & A_n & B_n & \dots \\ \dots & \dots & B_n & A_{n+1} & B_{n+1} \end{bmatrix}\begin{bmatrix}\Psi_{n-1}\\ \Psi_n \\ \Psi_{n+1}\end{bmatrix}$$

$$= E\begin{bmatrix} b_{n-2} & a_{n-1} & b_{n-1} & \dots & \dots \\ \dots & b_{n-1} & a_n & b_n & \dots \\ \dots & \dots & b_n & a_{n+1} & b_{n+1} \end{bmatrix}\begin{bmatrix}\Psi_{n-1}\\ \Psi_n \\ \Psi_{n+1}\end{bmatrix}. \tag{2.9}$$

The resulting difference equation is

$$B_{n-1}\Psi_{n-1}+A_n\Psi_n+B_n\Psi_{n+1}=E(b_{n-1}\Psi_{n-1}+a_n\Psi_n+b_n\Psi_{n+1}) \tag{2.10}$$

and we want to determine the coefficients so that the local error in the representation is as small as possible.

The method of selecting these coefficients is as follows: since we do not want the coefficients to depend on the unknown eigenvalue E, we replace the terms involving $E\Psi_n$, and so on, by using the exact Schrödinger equation (2.1). We should point out that this is a device solely for determining the coefficients and is not part of the method of solution. The resulting equation, which now involves second derivatives, is

$$\begin{aligned}&(B_{n-1}-b_{n-1}V_{n-1})\Psi_{n-1}+(A_n-a_nV_n)\Psi_n+(B_n-b_nV_{n+1})\Psi_{n+1}\\&\qquad+(b_{n-1}\Psi^{(2)}_{n-1}+a_n\Psi^{(2)}_n+b_n\Psi^{(2)}_{n+1})=0.\end{aligned}\tag{2.11}$$

We then demand that this equation be, at least to some order in δ, satisfied identically for an arbitrary Ψ. The way that this is achieved is to express $\Psi_{n\mp1}$ and $\Psi^{(2)}_{n\mp1}$ in terms of a Taylor expansion about $x=x_n$ and then to equate to zero the coefficients of as many derivatives as possible. For an alternative approach where the coefficients are chosen to depend on (the unknown) E see Ixaru and Rizea [3]. The coefficients of the first six derivatives of $\Psi(x)$ at $x=x_n$ are:

$$\begin{aligned}&0:\ \{(B_{n-1}-b_{n-1}V_{n-1})+(A_n-a_nV_n)+(B_n-b_nV_{n+1})\}\\&1:\ \delta\{(B_n-b_nV_{n+1})-(B_{n-1}-b_{n-1}V_{n-1})\}\\&2:\ \frac{\delta^2}{2}\left\{(B_n-b_nV_{n+1})+(B_{n-1}-b_{n-1}V_{n-1})+\frac{2}{\delta^2}(b_n+b_{n-1}+a_n)\right\}\\&3:\ \frac{\delta^3}{6}\left\{(B_n-b_nV_{n+1})-(B_{n-1}-b_{n-1}V_{n-1})+\frac{6}{\delta^2}(b_n-b_{n-1})\right\}\\&4:\ \frac{\delta^4}{24}\left\{(B_n-b_nV_{n+1})+(B_{n-1}-b_{n-1}V_{n-1})+\frac{12}{\delta^2}(b_n+b_{n-1})\right\}\\&5:\ \frac{\delta^5}{120}\left\{(B_n-b_nV_{n+1})-(B_{n-1}-b_{n-1}V_{n-1})+\frac{20}{\delta^2}(b_n-b_{n-1})\right\}\\&6:\ \frac{\delta^6}{720}\left\{(B_n-b_nV_{n-1})+(B_{n-1}-b_{n-1}V_{n-1})+\frac{30}{\delta^2}(b_n+b_{n-1})\right\}.\end{aligned}\tag{2.12}$$

All the odd coefficients, not just the ones shown here, can be made zero by the solution

$$\begin{aligned}&b_n=b\\&B_n=B+b(V_{n+1}+V_n)\end{aligned}\tag{2.13}$$

where B and b are constants, yet to be determined. Equating the zero and second-order coefficients to zero yields the relations

$$\begin{aligned}&a_n=-[2b+\delta^2(B+bV_n)]\\&A_n=-[2b+\delta^2(B+bV_n)]V_n-(B+bV_n).\end{aligned}\tag{2.14}$$

Using these results, the coefficient of the fourth derivative is

$$\frac{\delta^4}{12}\left(B+bV_n+\frac{12b}{\delta^2}\right)$$

and this obviously cannot be made zero for all n. The best that we can do is to choose

$$B=-b\left(\bar{V}+\frac{12}{\delta^2}\right)\tag{2.15}$$

where $\bar{V}$ is some 'average' potential. With this choice the fourth-order coefficient is

$$\frac{\delta^4}{12} b(V_n - \bar{V}).$$

The constant b is arbitrary; it simply sets the overall scaling. We choose $b = 1/12$ to give (approximately) the same scaling as in the Numerov-Cooley algorithm ((2.5), (2.7)). With this choice the complete set of coefficients is

$$\begin{aligned}
a_n &= \frac{5}{6} - \frac{\delta^2}{12}(V_n - \bar{V}) \\
b_n &= \frac{1}{12} \\
A_n &= \frac{2}{\delta^2} + \frac{(2\bar{V} + V_n(8 - \delta^2(V_n - \bar{V})))}{12} \\
B_n &= -\frac{1}{\delta^2} + \frac{(V_{n+1} + V_n - \bar{V})}{12}.
\end{aligned} \tag{2.16}$$

The leading term in the local error, relative to $E\Psi_n$ is

$$\frac{\delta^4}{144}(V_n - \bar{V})\Psi_n^{(4)} + \frac{\delta^4}{240}\Psi_n^{(6)} \tag{2.17}$$

which is the same order as in the Numerov-Cooley algorithm. $\bar{V}$ can be used to reduce further the local error.

Equations (2.7), (2.8) and (2.16) almost completely determine the algorithm. The 'almost' is necessary because we have not yet discussed boundary conditions. The difference equations (2.10) and (2.11) do not (necessarily) apply at the boundary values $n = 1$ and $n = N$. Hence the expressions for the coefficients (2.16) may not hold for these values. It turns out, however, that only the diagonal elements need to be modified.

3. Boundary conditions

We denote the left-hand and right-hand boundaries by $x = X_{\mathrm{L}}$ and $x = X_{\mathrm{R}}$. The typical situation we have in mind is where the eigenfunctions $\Psi_\nu(x)$ are required on the positive axis $0 < x < \infty$, the values on the negative axis being determined by symmetry or where, as in the radial equation of the spherically symmetric three-dimensional Schrödinger equation, the negative axis does not exist. The typical boundaries are therefore $X_{\mathrm{L}} = 0$ and X_{R} being some approximation to infinity.

The boundary conditions are assumed to have the form

$$\Psi(x) = 0$$

or $$\Psi^{(1)}(x) = \alpha\Psi(x) \tag{3.1}$$

at $x = X_{\mathrm{L}}$ or X_{R}, where α is a constant.

Consider first the boundary condition $\Psi(X_{\mathrm{L}}) = 0$. In this case the first point x_1 is taken to be one step in from the boundary: $x_1 = X_{\mathrm{L}} + \delta$. The difference equation for $n = 1$ is (2.10) with the terms Ψ_0 omitted. However, since these terms are identically zero we can formally leave them in the equation. The analysis leading to (2.16) can then be applied exactly, so this equation also applies to the boundary values $n=1$. Similarly (2.16) applies to $n=N$ if the right-hand boundary condition is $\Psi(X_R)=0$. The expression (2.17) for the local error also applies at $n=1$ and $n=N$ for these boundary conditions.

If the right-hand boundary condition is $\Psi^{(1)}(X_R)=\alpha\Psi(X_R)$ for some constant α (which may be zero), then x_N is taken to occur at the boundary, $x_N=X_R$. The relevant difference equations are (2.10) and (2.11) but with the terms Ψ_{N+1} omitted. The analysis proceeds as before except that the Taylor expansion of Ψ_{N-1} now incorporates the boundary condition:

$$\Psi_{N-1}=\Psi_N(1-\alpha\delta)+\frac{\delta^2}{2}\Psi_N^{(2)}-\frac{\delta^3}{6}\Psi_N^{(3)}+\dots \tag{3.2}$$

We then equate to zero the coefficients of the various derivatives (except $\Psi_N^{(1)}$) as before. The result is that the off-diagonal terms B_N and b_N are as given by (2.16) but that the diagonal terms are

$$\begin{aligned} a_N&=\frac{5}{12}-\frac{\delta^2}{24}(V_N-\bar{V})\\ A_N&=a_NV_N+(1-\alpha\delta)\left(\frac{1}{\delta^2}-\frac{(V_N-\bar{V})}{12}\right). \end{aligned} \tag{3.3}$$

Similarly if the left-hand boundary condition is $\Psi^{(1)}(X_L)=\beta\Psi(X_L)$, x_1 is taken to be X_L and the expressions for A_1 and a_1 are

$$\begin{aligned} a_1&=\frac{5}{12}-\frac{\delta^2}{24}(V_1-\bar{V})\\ A_1&=a_1V_1+(1+\beta\delta)\left(\frac{1}{\delta^2}-\frac{(V_1-\bar{V})}{12}\right). \end{aligned} \tag{3.4}$$

The values of the off-diagonal terms B_1 and b_1 are correctly predicted by (2.16). However, for these boundary conditions the local errors at the boundary are, in general, much larger. The leading terms in the local errors at $n=1$ and $n=N$ for these boundary conditions are respectively:

$$\begin{aligned} &-\left(\frac{\delta}{12}-\frac{\delta^3}{72}(V_1-\bar{V})\right)\Psi_1^{(3)}+\frac{\delta^4}{288}(V_1-\bar{V})\Psi^{(4)}\\ &\left(\frac{\delta}{12}-\frac{\delta^3}{72}(V_N-\bar{V})\right)\Psi_N^{(3)}+\frac{\delta^4}{288}(V_N-\bar{V})\Psi^{(4)}. \end{aligned} \tag{3.5}$$

In two important cases the situation can be retrieved: in the case $X_L=0$ and when the required eigenfunctions are *even* the third-order derivative $\Psi_1^{(3)}$ is zero and so the local error is proportional to δ^4 as before. If X_R is in the asymptotic region of the eigenfunctions then the purpose of the boundary condition is simply to exclude the exponentially growing term, and the solution is not sensitive to the precise form of the boundary condition. So, although the local error is the large value quoted above, this is not significant to the resulting eigenvalues and eigenfunctions.

In the typical case the left-hand boundary condition will be either $\Psi=0$ for odd eigenfunctions or $\Psi^{(1)}=0$ for even eigenfunctions; the right-hand boundary condition will be $\Psi^{(1)}(X_R)=-[V(X_R)-\bar{E}]^{1/2}\Psi_R(X)$ where $\bar{E}$ is some estimate of the energy eigenvalue. In most cases, the results are not very sensitive to the value of $\bar{E}$ and since our aim is to determine all the eigenvalues and eigenvectors in a certain energy range, a value near the top of the range is a suitable choice for $\bar{E}$.

4. Generalized eigenvalue problem

The determination of the eigenvalues and eigenvectors of the one-dimensional Schrödinger equation has been reduced to the solution of the *symmetric generalized eigenvalue problem*:

$$A\Psi = E\boldsymbol{a}\Psi \tag{4.1}$$

where A and $\boldsymbol{a}$ are tridiagonal symmetric $N \times N$ matrices. If we want very high accuracy or very highly excited eigenvectors (or both) then N may be required to have a large value, say 10 000.

As we have emphasized already, we are guaranteed a complete set of eigensolutions if, in addition, $\boldsymbol{a}$ is *positive-definite* and most library software subroutines for the solution of such eigensystems make this additional assumption. In the present case $\boldsymbol{a}$ does not turn out automatically to be positive-definite but we can *choose* it to be so by restricting the allowed range of the parameter $\bar{V}$. In order to ensure $\boldsymbol{a}$ is positive-definite it is sufficient to require that each diagonal element a_n be greater or equal to 1/6. The corresponding restriction on $\bar{V}$ depends on the particular boundary conditions in effect (see (2.16), (3.3) and (3.4)). For vanishing eigenfunctions at both ends of the range $\bar{V}$ must satisfy

$$\bar{V} > V_{\mathrm{max}} - \frac{8}{\delta^2} \tag{4.2}$$

where V_{max} is the largest value of V_n. This still leaves some freedom in the choice of $\bar{V}$ so as to reduce the local error. From now on we assume that $\boldsymbol{a}$ has been chosen to be positive-definite.

The general strategy for the solution of this generalized eigenvalue problem is as follows.

(1) Determine the Cholesky factors $\boldsymbol{L}$ and $\boldsymbol{L}^{\mathrm{T}}$ of $\boldsymbol{a}$ such that

$$\boldsymbol{a} = \boldsymbol{L}\boldsymbol{L}^{\mathrm{T}} \tag{4.3}$$

where the superscript T indicates the transpose and $\boldsymbol{L}$ is lower triangular. The great advantage of Cholesky factorization over most other factorization schemes is that the Cholesky factors perserve the *sparsity* of the matrix $\boldsymbol{a}$ and in the present case $\boldsymbol{L}$ has only a non-zero diagonal and one non-zero sub-diagonal, the remaining terms being zero.

(2) Transform the equation to a conventional eigenvalue problem

$$\boldsymbol{A}_{\mathrm{C}}\boldsymbol{x} = E\boldsymbol{x} \tag{4.4}$$

where $\boldsymbol{A}_{\mathrm{C}}$, the Cholesky transform of $\boldsymbol{A}$, is

$$\boldsymbol{A}_{\mathrm{C}} = \boldsymbol{L}^{-1}\boldsymbol{A}\boldsymbol{L}^{-\mathrm{T}} \tag{4.5}$$

and the eigenvectors $\boldsymbol{x}$ are related to the original eigenvectors Ψ by

$$\boldsymbol{x} = \boldsymbol{L}^{\mathrm{T}}\Psi. \tag{4.6}$$

(3) Solve the conventional eigenvalue problem (3.4) for the eigenvalues E_ν in the required range and their corresponding eigenvectors $\boldsymbol{x}_\nu$ by some method which exploits the sparsity of $\boldsymbol{A}$ and $\boldsymbol{L}$.

(4) Obtain the eigenvectors Ψ_ν from $\boldsymbol{x}_\nu$ by solving (4.6).

It should be emphasized here that although $\boldsymbol{A}$ and $\boldsymbol{L}$ are sparse matrices A_C is definitely not and it is important that A_C is not used explicitly. We explain below how this can be achieved.

The leading contenders for the solution of large sparse matrix eigenvalue problems are methods based on the Lanczos algorithm and those based on subspace iteration [12].

We have investigated two Lanczos schemes. The first was devised by Reid and Parlett [13] and implemented by Reid as subroutine EA15 in the Harwell Subroutine Library (14). This is based on the simple Lanczos algorithm but has a rather elaborate scheme for determining when the algorithm has converged. The second was devised by Scott and Parlett [15] and implemented by Scott as subroutine SILASO [16]. This is the *state of the art* in Lanczos procedures and is a block (or band) Lanczos scheme with selective orthogonalization.

In EA15 the matrix A_C is required only in a user-written subroutine which takes two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ as inputs and produces as output

$$\boldsymbol{v} = \boldsymbol{v} + A_C\boldsymbol{u}. \tag{4.7}$$

If A_C is used directly then, since it is a dense matrix, the operation (4.7) requires additions and multiplications of order N^2. An alternative scheme to calculate $A_C\boldsymbol{u}$ is:

(i) first, solve $\boldsymbol{u} = \boldsymbol{L}^T\boldsymbol{p}$ for $\boldsymbol{p}$

(ii) second, calculate $\boldsymbol{q} = \boldsymbol{A}\boldsymbol{p}$

(iii) finally, solve $\boldsymbol{q} = \boldsymbol{L}\boldsymbol{r}$ for $\boldsymbol{r}$ which is the required result.

This apparently convoluted scheme in fact requires only of the order of $7N$ additions and multiplications and so takes roughly a fraction $7/N$ of the time of the direct method.

SILASO requires a user-written subroutine which solves the equation $A_C\boldsymbol{u} = \boldsymbol{v}$ for $\boldsymbol{u}$. Again it is advantageous not to use A_C directly. The procedure is:

(i) first, calculate $\boldsymbol{p} = \boldsymbol{L}\boldsymbol{v}$

(ii) second, solve $\boldsymbol{A}\boldsymbol{q} = \boldsymbol{p}$ for $\boldsymbol{q}$

(iii) finally, calculate $\boldsymbol{u} = \boldsymbol{L}^T\boldsymbol{q}$ as the required result.

This is much quicker than solving the equation directly because in (ii) the tridiagonal nature of $\boldsymbol{A}$ can be exploited.

The subspace iteration procedure we have used is the NAG [17] subroutine FO2FJF which is based on the SIMITZ algorithm of Nikolai [18] which, in turn, is based on the Algol procedure RITZIT of Rutishauser [19]. This does not require the explicit factorization of the matrix $\boldsymbol{a}$. Two user-written subroutines are required: one forms the '$\boldsymbol{a}$ inner product' $\boldsymbol{u}^T\boldsymbol{a}\boldsymbol{v}$ of two vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ and the other solves the equation $\boldsymbol{A}\boldsymbol{u} = \boldsymbol{v}$ for $\boldsymbol{u}$. Both of these subroutines can take full advantage of the simple structures of $\boldsymbol{A}$ and $\boldsymbol{a}$.

We have included for comparison an investigation of the recommended EISPACK procedure for the generalized eigenvalue problem. This uses the EISPACK routines (in this order) REDUC, TRED1, BISECT, TINVIT, TRBAK1 and finally REBAK (11). These procedures take no account of sparsity of the matrices $\boldsymbol{A}$ and $\boldsymbol{a}$ and the matrix A_C is explicitly determined (by REDUC) and so we would not expect them to be very efficient for large N.

The final scheme we have tested was suggested by one of the referees of this paper. This involves modifications to two of the EISPACK subroutines, BISECT and TINVIT.

BISECT uses Sturm sequences and bisection applied to (in our notation) $(\boldsymbol{A}-E\boldsymbol{1})$ in order to determine the eigenvalues in a certain interval; TINVIT applies inverse iteration to $(\boldsymbol{A}-E\boldsymbol{1})\psi$ to determine the corresponding eigenvectors. Both of the procedures can, since $\boldsymbol{A}$ and $\boldsymbol{a}$ are symmetric and tridiagonal and $\boldsymbol{a}$ is positive-definite, be directly applied to the generalized eigenvalue system $(\boldsymbol{A}-E\boldsymbol{a})$. The necessary modifications to BISECT and TINVIT are relatively simple and eliminate the need for the subroutines REDUC, TRED1, TRBAK1 and REBAK. The storage requirements are also greatly reduced since the modified procedures only require the storage of two symmetric tridiagonal matrices. In the numerical tests this scheme is designated MOD-EIS (for modified EISPACK).

5. Numerical tests

In order to test these five procedures we wanted examples where the eigenvalues and eigenvectors are known exactly, we chose the radial equation of the hydrogen atom and the one-dimensional harmonic oscillator. In dimensionless form the Schrödinger equation for the one-dimensional harmonic oscillator is

$$-\frac{\mathrm{d}^2}{\mathrm{d}x^2}\Psi + x^2\Psi = E\Psi \tag{5.1}$$

and the exact eigenvalues and (unnormalized) eigenvectors are

$$\begin{aligned} E_\nu &= (2\nu+1) \\ \Psi_\nu(x) &= \exp(-x^2/2)H_\nu(x) \end{aligned} \tag{5.2}$$

where $H_\nu(x)$ is a Hermite polynomial [20]. The radial Schrödinger equation for the hydrogen atom, again in dimensionless units, is

$$-\frac{\mathrm{d}^2}{\mathrm{d}r^2}R(r) + \left(\frac{\ell(\ell+1)}{r^2} - \frac{2}{r}\right)R(r) = ER(r). \tag{5.3}$$

The exact eigenvalues and (unnormalized) eigenvectors are

$$\begin{aligned} E_{\nu\ell} &= -\frac{1}{\nu^2} \\ R_{\nu\ell}(r) &= r^{\ell+1}\exp(-r/\nu)L^{2\ell+1}_{\nu-\ell-1}(2r/\nu) \end{aligned} \tag{5.4}$$

where $L^n_m(x)$ is a generalized Laguerre polynomial [20].

These two cases represent very different problems. The harmonic oscillator is the 'ideal problem'. The required step size δ and the required box size $N\delta$ vary only slowly with ν:

$$\delta \approx \nu^{-1/2}$$
$$N\delta \approx \nu^{1/2}$$

where ν is the quantum number of the largest required eigenvalue.

In contrast, the hydrogen atom is far from ideal, the required step size is approximately independent of ν whereas the required box size is

$$N\delta \approx \nu^2$$

where again ν is the quantum number of the largest required eigenvalue. This means that for a given accuracy the matrix order is $N \approx \nu$ for the harmonic oscillator and is $N \approx \nu^2$ for the hydrogen atom.

Table 1. Harmonic oscillator states $\nu = 1, 3, 5, 7, 9, 11, 13, 15$.

Software routine	Number of points	Spacing δ	RMS eigenfunction error	CPU time (s)
NAG	100	8.0×10^{-2}	4.1×10^{-4}	0.65
SILASO	100	8.0×10^{-2}	4.1×10^{-4}	1.22
HARWELL	100	8.0×10^{-2}	4.1×10^{-4}	0.30
MOD-EIS	100	8.0×10^{-2}	3.7×10^{-4}	0.12
EISPACK	100	8.0×10^{-2}	4.1×10^{-4}	1.27
NAG	400	2.0×10^{-2}	7.7×10^{-6}	2.52
SILASO	400	2.0×10^{-2}	7.9×10^{-6}	3.54
HARWELL	400	2.0×10^{-2}	7.7×10^{-6}	3.44
MOD-EIS	400	2.0×10^{-2}	7.7×10^{-6}	0.40
EISPACK	400	2.0×10^{-2}	7.9×10^{-6}	96.0
NAG	1300	6.9×10^{-3}	2.5×10^{-8}	11.1
SILASO	1300	6.9×10^{-3}	6.1×10^{-8}	13.3
HARWELL	1300	6.9×10^{-3}	2.5×10^{-8}	34.9
MOD-EIS	1300	6.9×10^{-3}	2.3×10^{-8}	1.38
NAG	4000	2.5×10^{-3}	3.8×10^{-10}	35.1
SILASO	4000	2.5×10^{-3}	4.9×10^{-8}	33.2
MOD-EIS	4000	2.5×10^{-3}	4.7×10^{-10}	4.24
NAG	7000	1.4×10^{-3}	4.1×10^{-11}	66.4
MOD-EIS	7000	1.4×10^{-3}	3.6×10^{-10}	7.58
MOD-EIS	11 000	1.3×10^{-3}	1.5×10^{-10}	11.8

For the harmonic oscillator the two tests for which the results are reported here are:

(i) the determination of the odd eigenfunctions in the range 0–32

(ii) the determination of the odd eigenfunctions in the range 32–64.

Each test has been performed for a range of required accuracies in the calculated eigenvectors. For the hydrogen atom we present the results for the $\ell = 1$ eigenfunctions in energy range -0.051 to -0.014 ($\nu = 5$–8).

Tables 1, 2 and 3 show the results obtained from these five routines for a range of array sizes N. The results were obtained using double-precision arithmetic on an Amdahl 5870 using a virtual machine size of 4 Mbytes. All five routines could be used for $N \leqslant 400$ but for $N \geqslant 7000$ only the NAG and MOD-EIS routines were successful. With a larger machine size the NAG routine has been used successfully with $N = 20\,000$. There was some problem with the accuracy in SILASO; in the harmonic oscillator examples with $N = 4000$ the tables show that the NAG and MOD-EIS routines produced results which were two orders of magnitude more accurate.

6. Conclusions

We have demonstrated that symmetric matrix methods can be used with advantage to determine groups of eigenvalues and eigenfunctions of the one-dimensional Schrödinger equation. Of the particular software packages we have tested, the NAG routine FO2FJF presented the least difficulty and was the quickest. However even this could not compete in speed with the modified EISPACK subroutines MOD-EIS. The main

Table 2. Harmonic oscillator states $\nu = 17, 19, 21, 23, 25, 27, 29, 31$.

Software routine	Number of points	Spacing δ	RMS eigenfunction error	CPU time (s)
NAG	100	1.0×10^{-1}	8.4×10^{-3}	0.71
SILASO	100	1.0×10^{-1}	8.4×10^{-3}	2.21
HARWELL	100	1.0×10^{-1}	8.4×10^{-3}	0.31
MOD-EIS	100	1.0×10^{-1}	8.0×10^{-3}	0.11
EISPACK	100	1.0×10^{-1}	8.4×10^{-3}	1.27
NAG	400	2.5×10^{-2}	3.5×10^{-5}	2.66
SILASO	400	2.5×10^{-2}	3.5×10^{-5}	3.96
HARWELL	400	2.5×10^{-2}	3.5×10^{-5}	3.54
MOD-EIS	400	2.5×10^{-2}	3.4×10^{-5}	0.38
EISPACK	400	2.5×10^{-2}	3.6×10^{-5}	99.9
NAG	1300	8.5×10^{-3}	4.2×10^{-7}	10.3
SILASO	1300	8.5×10^{-3}	4.5×10^{-7}	13.6
HARWELL	1300	8.5×10^{-3}	4.2×10^{-7}	35.8
MOD-EIS	1300	8.5×10^{-3}	4.0×10^{-7}	1.29
NAG	4000	2.9×10^{-3}	5.6×10^{-9}	32.1
SILASO	4000	2.9×10^{-3}	1.2×10^{-6}	39.8
MOD-EIS	4000	2.9×10^{-3}	5.5×10^{-9}	4.09
NAG	7000	1.7×10^{-3}	7.1×10^{-10}	65.0
MOD-EIS	7000	1.7×10^{-3}	7.4×10^{-10}	7.26
MOD-EIS	11 000	1.1×10^{-3}	2.9×10^{-10}	11.5

Table 3. Hydrogen radial eigenfunctions $\ell = 1$; $\nu = 5, 6, 7, 8$.

Software routine	Number of points	Spacing δ	RMS eigenfunction error	CPU time (s)
NAG	400	5.0×10^{-1}	7.5×10^{-3}	1.24
SILASO	400	5.0×10^{-1}	7.6×10^{-3}	2.33
HARWELL	400	5.0×10^{-1}	7.6×10^{-3}	3.10
MOD-EIS	400	5.0×10^{-1}	7.6×10^{-3}	0.21
EISPACK	400	5.0×10^{-1}	7.6×10^{-3}	94.7
NAG	1300	1.9×10^{-1}	4.6×10^{-4}	4.23
SILASO	1300	1.9×10^{-1}	4.6×10^{-4}	6.69
HARWELL	1300	1.9×10^{-1}	4.6×10^{-4}	29.1
MOD-EIS	1300	1.9×10^{-1}	4.6×10^{-4}	0.68
NAG	4000	6.9×10^{-2}	2.3×10^{-5}	18.2
SILASO	4000	6.9×10^{-2}	2.2×10^{-5}	21.9
MOD-EIS	4000	6.9×10^{-2}	2.3×10^{-5}	2.18
NAG	7000	4.3×10^{-2}	5.5×10^{-6}	32.2
MOD-EIS	7000	4.3×10^{-2}	5.5×10^{-6}	3.72
MOD-EIS	11 000	2.7×10^{-2}	1.5×10^{-6}	5.95

difficulty with the Harwell routine EA15 is that it requires a huge amount ($\propto N^2$) of secondary (disc or tape) storage. In our system the secondary storage is disc and it was very easy to exceed our disc-space allocation. For example, with $N = 10\,000$ and using 8-byte (double-precision) floating point numbers, EA15 requires 800 Mbytes of disc-space! Both the Harwell subroutine EA15 and Scott's SILASO required more main memory space than the NAG routine. In a particular configuration with 4 Mbytes of main memory, the largest matrix sizes that we could accommodate were:

EA15 . . . $N = 6000(1300)$

SILASO . . . $N = 4000$

NAG . . . $N = 7000$

MOD-EIS . . . $N = 11\,000$

EISPACK . . . $N = 425$.

For EA15, $N = 6000$ is the largest array size for which the program will start in this configuration. However, the program runs out of secondary (disc) storage. With a secondary disc size of 25 Mbytes the largest N was 1300.

The modified EISPACK procedure MOD-EIS is small enough to be used on a PC and is particularly suitable for student projects. The original EISPACK procedure is clearly not suitable for large arrays.

Acknowledgments

We would like to thank our anonymous referee for suggesting what is clearly the best programming scheme for implementing our algorithm. The work of one of us (BG) was supported by the United Kingdom Science and Engineering Research Council.

References

[1] Cooney P J, Kanter E P and Vager Z 1981 *Am. J. Phys.* **49** 76-7
[2] Eckert M 1989 *J. Comput. Phys.* **82** 147-60
[3] Ixaru L Gr and Rizea M 1987 *J. Comput. Phys.* **73** 306-24
[4] Searles D J and von Nagy-Felsobuki E I 1988 *Am. J. Phys.* **56** 444-8
[5] Numerov B 1933 *Publ. Obs. Cent. Astrophys. Russ.* **2** 188
[6] Cooley J W 1961 *Math. Comput.* **1** 363
[7] Bailey P B 1966 *SIAM J. Appl. Math.* **14** 242-9
[8] Pryce J D 1977 *IMA Numer. Anal. Newsletter* **1** (3)
[9] Pryce J D 1981 *University of Bristol, Computer Science Technical Report CS-81-01*
[10] Dunn D and Grieves B 1989 *J. Phys. A: Math. Gen.* **22** L1093-6
[11] Garbow B S, Boyle J M, Dongarra J J and Moler C B 1977 *Matrix Eigensystem Routines—EISPACK Guide Extension* (Berlin: Springer)
[12] Parlett B N 1980 *The Symmetric Eigenvalue Problem* (Englewood Cliffs, NJ: Prentice-Hall)
[13] Parlett B N and Reid J K 1981 *IMA J. Numer. Anal.* **1** 135-55
[14] Harwell Subroutine Library, Computer Science Division, Harwell Laboratory, Oxfordshire OX11 0PA, UK
[15] Parlett B N and Scott D 1979 *Math. Comput.* **33** 217-38
[16] SILASO is available via NETLIB
[17] NAG Ltd, Wilkinson House, Jordon Hill Road, Oxford OX2 8DR, UK; NAG Inc, 1101 31st Street, Suite 100, Downers Grove, IL 60515-1263 USA
[18] Nikolai P J 1979 *ACM Trans. Math. Software* **5** 118-25

[19] Rutishauser H 1969 *Numer. Math.* **13** 4–13
[20] Abramowitz M and Stegun I 1964 *Handbook of Mathematical Functions* (Washington, DC: US Govt Printing Office)
[21] Pruess S 1988 *J. Comput. Phys.* **75** 493–7
[22] Wilkinson J H 1965 *The Algebraic Eigenvalue Problem* (Oxford: Clarendon)
[23] Dennery P and Krzywicki A 1967 *Mathematics for Physicists* (New York: Harper and Row)