Eigenvalues of $Ax = \lambda Bx$ with band symmetric A and B

By G. Peters and J. H. Wilkinson*

Describes an efficient algorithm for the calculation of specified eigenvalues of $Ax = \lambda Bx$ with band symmetric A and B, the latter being positive definite. Each eigenvalue is isolated using the Sturm sequence property of leading principal minors of $A - \lambda B$ and is then computed accurately using a modified version of successive linear interpolation.

(Received November 1968)

1. Introduction

In some engineering problems the eigenvalues of $Ax = \lambda Bx$ are required, where A and B are symmetric and B is positive definite, (Gupta, 1968). When A and B are full matrices the reduction to the standard symmetric eigenproblem is probably the most efficient method. If we write

$$B = LL^T, \tag{1.1}$$

where L is lower-triangular, then $Ax = \lambda Bx$ is equivalent to

$$(L^{-1}AL^{-T})(L^Tx) = \lambda(L^Tx), \tag{1.2}$$

or

$$Py = \lambda y. \tag{1.3}$$

(Here and later L^{-T} has been used in place of the cumbersome $(L^{-1})^T$.) This has been described by Wilkinson (1965) and an ALGOL version has been published by Martin and Wilkinson (1968).

When A and B are band matrices and the band-widths are much smaller than the order of A and B it is important to take advantage of the band form. This can indeed be done in the above algorithm when computing L and P, but although L is a band matrix, L^{-1} is full and hence P itself is a full matrix. Methods in which the band forms of A and B are preserved throughout are of particular interest. A number of such algorithms have been developed at the National Physical Laboratory over the last six years and in this paper we describe one of the more efficient among them.

2. Sturm sequence property

It is well known that if P is real and symmetric the leading principal minors of $P - \lambda I$ form a Sturm sequence. Consequently the number of eigenvalues greater than λ is equal to the number of agreements in sign between consecutive members of the sequence $det\,(P_r - \lambda I)(r = 0, 1, \ldots, n)$, where $det\,(P_0 - \lambda I) = 1$ by definition and P_r is the leading principal sub-matrix of order r of P.

We now show that if L and P are defined by (1.1) and (1.2) then the sign of $det\,(A_r - \lambda B_r)$ is the same as that of $det\,(P_r - \lambda I)$. We write

$$B = LL^T,\ L = \begin{bmatrix} L_r & 0 \\ M_r & N_r \end{bmatrix} \tag{2.1}$$

where L_r is the leading principal sub-matrix of order r; clearly

$$L_rL_r^T = B_r,\ L^{-1} = \begin{bmatrix} L_r^{-1} & 0 \\ X & Y \end{bmatrix},\ P_r = L_r^{-1}A_rL_r^{-T}. \tag{2.2}$$

Hence we have

$$\begin{aligned} det\,(A_r - \lambda B_r) &= det\,(L_r(L_r^{-1}A_rL_r^{-T} - \lambda I)L_r^T) \\ &= det\,(L_r(P_r - \lambda I)L_r^T) \\ &= (det\,L_r)^2\,det\,(P_r - \lambda I), \end{aligned} \tag{2.3}$$

and since $det\,L_r \neq 0$ this means that $det\,(A_r - \lambda B_r)$ has the same sign as $det\,(P_r - \lambda I)$. The number of eigenvalues of $Ax = \lambda Bx$ which are greater than λ is therefore equal to the number of agreements in sign between consecutive members of the sequence $det\,(A_r - \lambda B_r)(r = 0, 1, \ldots, n)$.

An apparent difficulty arises with consecutive zero principal minors. This may be illustrated by considering an extreme form associated with diagonal matrices. Suppose

$$A = \begin{bmatrix} 4 & & \\ & 1 & \\ & & 2 \end{bmatrix} \text{ and } B = \begin{bmatrix} 1 & & \\ & 1 & \\ & & 1 \end{bmatrix} \tag{2.4}$$

then with $\lambda = 4$ all principal minors of $A - \lambda B$ are zero, though it is certainly not true that there are three eigenvalues greater than 4. No problem arises in the computational algorithm since we do not actually determine the principal minors explicitly but only the sign of $det\,(A_{r+1} - \lambda B_{r+1})$ in terms of the sign of $det\,(A_r - \lambda B_r)$. This is done by comparing the signs of the pivotal elements, the products of which give the minors. The effect is the same as would be achieved by replacing a zero pivot by a very small quantity though in fact the substitution is not performed explicitly.

Our problem then is to compute all the leading principal minors of $A - \lambda B$ (i.e. $A_n - \lambda B_n$) in an economical manner. We cannot use the Cholesky decomposition of $A_n - \lambda B_n$ because this matrix is not in general positive definite; the LDL^T decomposition (Martin, Peters and Wilkinson, 1965) will in general be unstable for the same reason. The LU decomposition with interchanges is numerically stable, but unfortunately this gives only the leading principal minors of $A_n - \lambda B_n$ *with its rows permuted.*

There are two triangularisations of $A_n - \lambda B_n$ which give the required minors. One is a modification of Givens' triangularisation, and the other is an elimination type method. The basic feature of both is that the first

* *Division of Numerical and Applied Mathematics, National Physical Laboratory, Teddington, Middlesex*

r rows are reduced as far as possible before introducing row $r + 1$. They have been described by Wilkinson (1965) pp. 236–240, for the case when the matrix to be factorised is full, and an ALGOL version of the elimination method has been given by Martin and Wilkinson (1967) specially designed for the band symmetric case. In the ALGOL procedure the factors L and U were preserved since they were required for the solution of linear equations. Here we are concerned only with determinants (for the calculation of the number of eigenvalues greater than λ, only the signs of the minors are required), and the storage can be reduced. For a band of full width $2m + 1$, only $m + 1$ rows are involved at each stage of the reduction and hence in addition to the $n(m + 1)$ storage locations required for each of A and B (taking advantage of symmetry) only $(m + 1)(2m + 1)$ extra locations are required. If, however, the eigenvectors are required, the L and U must be stored in order to perform inverse iteration.

The Sturm sequence property together with the method of bisection can be used to locate the eigenvalues of $Ax = \lambda Bx$ exactly as described by Givens (1954) for $Ax = \lambda x$ in the case when A is symmetric and tridiagonal. An ALGOL version of this tridiagonal case has been given by Barth, Martin and Wilkinson (1967).

3. Use of linear interpolation

When the bands are of appreciable width it is desirable to restrict the number of evaluations of the Sturm sequence. This can be done by switching to a technique with a higher convergence rate when a suitable approximation to an eigenvalue has been determined. The required eigenvalues are the zeros of $det\ (A_n - \lambda B_n)$ and this function is provided by the factorisation algorithm. Once an eigenvalue has been isolated it is attractive to use some method of successive linear interpolation for the corresponding zero of $f(\lambda) = det\ (A_n - \lambda B_n)$.

Alternatively interpolation can be performed on the function $g(\lambda)$ defined by

$$g(\lambda) = det\ (A_n - \lambda B_n)/det\ (A_{n-1} - \lambda B_{n-1}) \quad (3.1)$$

This function has the same zeros as $f(\lambda)$, and since the zeros of $det\ (A_{n-1} - \lambda B_{n-1})$ separate those of $f(\lambda)$ at least in the weak sense, it is easy to show that $\frac{dg}{d\lambda} \leqslant -1$ for all λ. Further if $f(\lambda)$ has a zero of multiplicity r then $det\ (A_{n-1} - \lambda B_{n-1})$ has this zero with multiplicity $r - 1$ and hence all zeros of $g(\lambda)$ are simple. As one might expect from this $g(\lambda)$ is also more satisfactory when $f(\lambda)$ has pathologically close zeros. Finally the computation of $f(\lambda)$ involves scaling problems which are largely absent in the evaluation of $g(\lambda)$. In fact in the algorithm given by Barth *et al.* (1967) for the tridiagonal problem $A - \lambda I$, it was the function $g(\lambda)$ which was used, rather than $f(\lambda)$, for precisely this reason.

Unfortunately it may well happen that a simple zero of $det\ (A_n - \lambda B_n)$ is also a zero of $det\ (A_{n-1} - \lambda B_{n-1})$ or that $det\ (A_n - \lambda B_n)$ has a zero pathologically close to a zero of $det\ (A_{n-1} - \lambda B_{n-1})$. Such zeros are concealed in the ratio and we would be forced to use the bisection process all the time for their location.

It might be thought that this would be a comparatively rare phenomenon, but experience shows this to be untrue. It has been discussed in a somewhat different context by Wilkinson (1958). A good example is given by the tridiagonal matrices of order 21 defined by

$$a_{ii} = 11 - i,\ a_{i,\,i+1} = a_{i+1,\,i} = 1,\ b_{ii} = 1,$$
$$b_{i,\,i+1} = b_{i+1,\,i} = 0 \quad (3.2)$$

A number of the eigenvalues of $det\ (A_{21} - \lambda B_{21})$ agree with those of $det\ (A_{20} - \lambda B_{20})$ to more than ten decimals and hence are completely concealed in the ratio. Our experience over the last six years would seem to indicate that except when A and B are tridiagonal (in which case the percentage of time involved in scaling problems is higher) the use of $f(\lambda)$ is to be preferred.

Experiments have been made with two linear interpolation procedures. These have been described by Wilkinson (1967) but since they are still not readily accessible in any standard journal, the second of these procedures which has proved the more effective is described in the appendix.

Kahan and Varah (1966) have described experiments made in connexion with the problem $A - \lambda I$ where A is symmetric and tridiagonal. They effectively work with the function $g(\lambda)$ and use the first of the linear interpolation procedures described by Wilkinson (1967). For tridiagonal matrices the function $g(\lambda)$ may be determined from the sequence

$$g_1(\lambda) = a_{11} - \lambda,\ g_i(\lambda) = a_{ii} - \lambda - (a_{i,\,i-1})^2/g_{i-1}(\lambda)$$
$$(i = 2, \ldots, n)\ g(\lambda) = g_n(\lambda) \quad (3.3)$$

and since the $a^2_{i,i-1}$ may be determined once and for all this is particularly attractive. The Sturm sequence count is given by the number of positive $g_i(\lambda)$. This

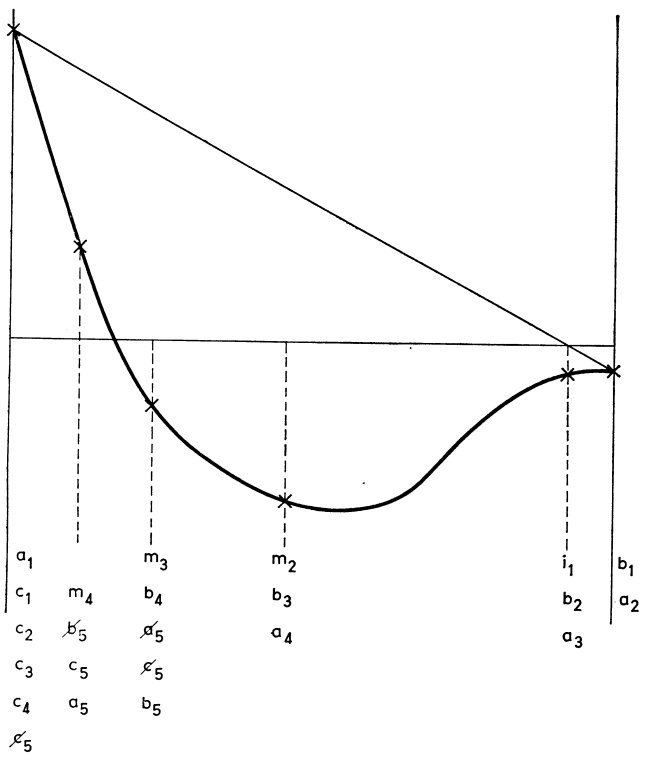

Fig. 1
Discarded provisional points indicated—e_5

method extends immediately to the problem $A - \lambda B$, with A and B tridiagonal, giving

$$g_1(\lambda) = a_{11} - \lambda b_{11},\ g_i(\lambda) = a_{ii} - \lambda b_{ii} - (a_{i,i-1} - \lambda b_{i,i-1})^2/g_{i-1}(\lambda) \quad g(\lambda) = g_n(\lambda) \qquad (3.4)$$

and this has been used at the National Physical Laboratory on ACE and KDF9. The ALGOL procedure given by Kahan and Varah contains a number of attractive scaling and programming features which were not included in our programs.

4. The preferred algorithm

The algorithm may best be described by considering its use for the calculation of all eigenvalues between a and b with $a > b$. The Sturm sequence counts x and y are first determined for a and b. Assuming that the eigenvalues are ordered so that $\lambda_1 \geqslant \lambda_2 \geqslant \ldots \geqslant \lambda_n$, this shows that $\lambda_{x+1}, \lambda_{x+2}, \ldots, \lambda_y$ lie in the given range and a and b can be taken as upper and lower bounds for all of them. To determine each eigenvalue the method of bisection combined with the Sturm sequence count is used until upper and lower bounds p and q are found which contain that eigenvalue *and no others*. Every time a Sturm sequence count is made, the upper and lower bounds for each of the required eigenvalues is updated as described by Givens (1954), and as given in the ALGOL procedure by Barth *et al.* (1967).

When a simple eigenvalue has been isolated in this way, $f(p)$ and $f(q)$ have opposite signs and the method of interpolation can then be used to locate that eigenvalue to the required accuracy.

Notice that this means that multiple eigenvalues are determined entirely by bisection since we never reach an interval containing only one such eigenvalue. However, if it is of multiplicity r, all r eigenvalues are found in the one set of bisections. Similarly if there are close eigenvalues, bisection must be continued further than usual before the first of them is isolated, but in isolating it we obtain comparatively good initial upper and lower bounds for the neighbouring eigenvalues. Simple zeros of $f(\lambda)$ which are also zeros of $det\ (A_{n-1} - \lambda B_{n-1})$ (or close to such zeros) are of no special significance with this algorithm.

Sometimes specific eigenvalues, $\lambda_s \geqslant \lambda_{s+1} \geqslant \ldots \geqslant \lambda_t$ are required rather than those between prescribed values. In this case initial upper and lower bounds are required for these eigenvalues. Such bounds are given by $\pm|\lambda|/\mu$ where $|\lambda|$ is the eigenvalue of A of largest absolute value and μ is the smallest eigenvalue of B (necessarily positive), but this involves substantial computation. The quantity k defined by

$$k = ||A||_\infty / ||B||_\infty$$

is calculated and the Sturm sequence is determined for $\lambda = k, 4k, 4^2k, \ldots$ until an upper bound is obtained for λ_s. Similarly the Sturm sequence is determined for $\lambda = -k, -4k, -4^2k, \ldots$ until a lower bound is obtained for λ_t. The upper and lower bounds for $\lambda_s, \ldots, \lambda_t$ can, of course, be updated during this initial stage.

5. Numerical examples and assessment

As illustration of the use of this algorithm we give a number of numerical examples.

Example 1. A and B are of order 20 and are bands of full width 7.

$$a_{ii} = 51 - i \qquad a_{ij} = 1,\ 0 < |i-j| \leqslant 3$$
$$b_{ii} = 41 - i \qquad b_{ij} = 1,\ 0 < |i-j| \leqslant 3$$

All eigenvalues between ± 10 were required. In **Table 1** we give the computed eigenvalues, and the number of bisection and interpolation steps required for each one. We stress that our interpolation routine is itself a combination of interpolation and bisection, and bisections do occur from time to time in these examples.

Table 1

EIGENVALUES	NUMBER OF BISECTION STEPS	NUMBER OF INTERPOLATION STEPS
+1·2362 2996 622	4	23
+1·2543 8078 474	7	10
+1·2619 2368 457	1	8
+1·2694 3952 847	0	5
+1·2773 9754 724	1	8
+1·2856 3483 441	0	7
+1·2940 9698 102	2	6
+1·3030 1061 009	0	6
+1·3125 0454 161	1	6
+1·3226 0009 164	0	7
+1·3333 9423 801	3	8
+1·3450 0343 860	0	8
+1·3575 7195 730	0	9
+1·3713 1462 185	2	9
+1·3866 8413 225	0	5
+1·4034 7245 976	0	10
+1·4222 3523 837	2	12
+1·4475 1739 434	1	8
+1·4704 2713 163	0	10
+1·4952 1305 093	0	13
	24	178

The total number of steps is 202 an average of 10·1 per eigenvalue. Since the main work involved is in the factorisation, the distinction between the preliminary bisection steps and the interpolation steps is not important.

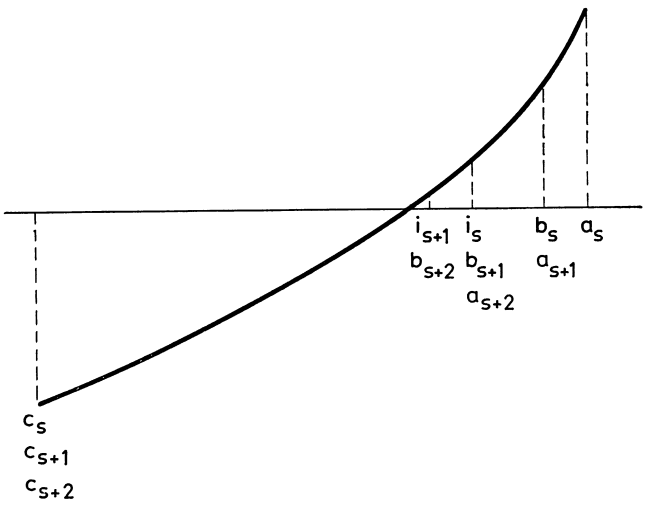

Fig. 2

Example 2. A and B are of order 20, A is tridiagonal and $B = I$. A is defined by

$$a_{2i,2i} = +10^4, \quad a_{2i+1,2i+1} = -10^4,$$
$$a_{i,i+1} = a_{i+1,i} = 1$$

This example was used by Kahan and Varah (1966). The eigenvalues occur in equal and opposite pairs, ten being close to $+10^4$ and the other ten close to -10^4. **Table 2** gives the computed results. It will be seen that most of the work is done in the initial sets of bisections associated with the lowest root and the eleventh root, that is

Table 2

EIGENVALUES $\times 10^{-4}$	NUMBER OF BISECTION STEPS	NUMBER OF INTERPOLATION STEPS
−1·0000 0001 955	31	5
−1·0000 0001 826	0	2
−1·0000 0001 624	0	2
−1·0000 0001 366	3	5
−1·0000 0001 075	0	4
−1·0000 0000 778	1	5
−1·0000 0000 500	1	3
−1·0000 0000 267	0	4
−1·0000 0000 099	4	3
−1·0000 0000 011	0	6
+1·0000 0000 011	30	7
+1·0000 0000 099	0	3
+1·0000 0000 267	3	4
+1·0000 0000 500	0	3
+1·0000 0000 778	0	5
+1·0000 0001 075	1	4
+1·0000 0001 365	0	5
+1·0000 0001 624	4	2
+1·0000 0001 826	1	2
+1·0000 0001 955	0	5
	—	—
	79	79
	—	—

the first of each of the clusters. In all, 79 bisections and 79 interpolations were required for 20 eigenvalues. For matrices of the same type but of order 100 the average was a little lower.

Example 3. A and B are of order 25 and are bands of full width 11. A is the identity matrix and B is the matrix defined by

$$B = \begin{bmatrix} X & -I & & & \\ -I & X & -I & & \\ & -I & X & -I & \\ & & -I & X & -I \\ & & & -I & X \end{bmatrix}$$

$$X = \begin{bmatrix} 4 & -1 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & 0 \\ 0 & -1 & 4 & -1 & 0 \\ 0 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & -1 & 4 \end{bmatrix}$$

This matrix arises in connexion with finite difference approximations to the eigenvalue problem $\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = \lambda u$ for a square region with a six by six mesh. Eigenvalues between 0·19 and 0·35 were requested. There are four eigenvalues each of multiplicity two, and one eigenvalue of multiplicity five, and hence all were located entirely by bisections. **Table 3** gives the computed results. Notice that the higher the multiplicity the more effective is the procedure. The presence of close or multiple eigenvalues does not affect the accuracy. (The problem $Ix = \lambda Bx$ rather than $Bx = \lambda x$ is given since it provides the reader with a more complete test of the procedure.) Matrix problems arising from the same partial differential equation but using a finer mesh have been solved to give a more severe test of the effectiveness of the procedures on zeros of high multiplicity.

Table 3

EIGENVALUES $\times 10$	NUMBER OF BISECTION STEPS	NUMBER OF INTERPOLATION STEPS
+2·0000 0000 000	38	0
+2·0000 0000 000	0	0
+2·1132 4865 405	35	0
+2·1132 4865 405	0	0
+2·5000 0000 000	35	0
+2·5000 0000 000	0	0
+2·5000 0000 000	0	0
+2·5000 0000 000	0	0
+2·5000 0000 000	0	0
+3·0600 2309 436	36	0
+3·0600 2309 436	0	0
+3·3333 3333 333	35	0
+3·3333 3333 333	0	0
	—	–
	179	0
	—	–

Example 4. A and B are of order 21, A is defined by $a_{ii} = 11 - i (i = 1, \ldots, 10)$, $a_{ii} = i - 11 (i = 11, \ldots, 21)$, $a_{i,i+1} = a_{i+1,i} = 1$ and $B = I$. This matrix has no multiple eigenvalues but the largest eigenvalues are equal in pairs to working accuracy and there are several other close pairs. The first of a close pair requires a larger number of bisections and fewer interpolations; the second often requires no bisections. The eigenvalue pairs which are equal to working accuracy are found in one set of bisections, no interpolations being required. **Table 4** gives the computed results.

Our procedure usually gives eigenvalues of almost the optimum accuracy for the precision of the computation. For matrices of modest order the straightforward reduction to the standard eigenvalue problem (Martin and Wilkinson (1968)) will usually be faster particularly if all eigenvalues are required.

However, this reduction has disadvantages when B is ill-conditioned with respect to inversion (see Wilkinson (1965), p. 344) and moreover when A and B are narrow bands of high order, it may well be ruled out because of the storage required by the full matrix P. Even when this is not so our procedure will usually be more effective for such problems when only a few selected eigenvalues are required. The ability to select specific eigenvalues is a great advantage.

When the matrices A and B are derived from differential equations the elements are usually given by quite

simple formulae and in this case there may be no need to store A and B explicitly. In Example 3 it would clearly be quite practical to compute each row of $A - \lambda B$ as required. The storage requirements are then reduced to $(m + 1)(2m + 1)$ locations; notice that this is independent of n.

An alternative method has been developed at NPL which is based on inverse iteration and the use of the generalised Rayleigh Quotient x^TAx/x^TBx. This has the advantage of giving the corresponding eigenvectors. However, it is necessary to store U at least when the triangular factorisation is computed and hence the extreme economy of storage cannot be achieved. It is not easy to ensure convergence to specific vectors and after finding r vectors it is usually necessary to orthogonalise with respect to these in order to ensure that the next computed vector is a 'new' vector. Experiments are continuing with methods which combine inverse iteration with the Sturm sequence count.

For the calculation of a small number of specific eigenvalues of large band matrices the method discussed in this paper is generally the most effective we have used, though obviously there are matrices for which the use of the function $g(\lambda)$ is to be preferred.

Table 4

EIGENVALUES	NUMBER OF BISECTION STEPS	NUMBER OF INTERPOLATION STEPS
$-1{\cdot}1254\ 4152\ 212_{10} + 0$	1	18
$+2{\cdot}5380\ 5817\ 097_{10} - 1$	4	11
$+9{\cdot}4753\ 4367\ 529_{10} - 1$	0	10
$+1{\cdot}7893\ 2135\ 269_{10} + 0$	1	11
$+2{\cdot}1302\ 0921\ 936_{10} + 0$	0	8
$+2{\cdot}9610\ 5888\ 418_{10} + 0$	5	8
$+3{\cdot}0430\ 9929\ 258_{10} + 0$	0	8
$+3{\cdot}9960\ 4820\ 137_{10} + 0$	4	5
$+4{\cdot}0043\ 5402\ 345_{10} + 0$	0	12
$+4{\cdot}9997\ 8247\ 772_{10} + 0$	11	6
$+5{\cdot}0002\ 4442\ 501_{10} + 0$	0	9
$+6{\cdot}0002\ 1752\ 225_{10} + 0$	17	9
$+6{\cdot}0002\ 3403\ 158_{10} + 0$	0	10
$+7{\cdot}0039\ 5179\ 860_{10} + 0$	20	9
$+7{\cdot}0039\ 5220\ 955_{10} + 0$	0	5
$+8{\cdot}0389\ 4111\ 584_{10} + 0$	24	8
$+8{\cdot}0389\ 4112\ 280_{10} + 0$	0	10
$+9{\cdot}2106\ 7864\ 736_{10} + 0$	36	0
$+9{\cdot}2106\ 7864\ 736_{10} + 0$	0	0
$+1{\cdot}0746\ 1941\ 829_{10} + 1$	35	0
$+1{\cdot}0746\ 1941\ 829_{10} + 1$	0	0
	158	157

Appendix

Modified successive linear interpolation

We describe here a method of successive linear interpolation for the calculation of a zero of a real function $f(x)$, given values b and c such that $f(b)f(c) < 0$ and $f(x)$ is continuous in (b, c). This method is due to van Wijngaarden, Zonneveld, Dijkstra and Dekker (1963). We shall not distinguish between interpolation and extrapolation and refer to both as interpolation.

The algorithm is an extension of the process of successive linear interpolation defined by

$$x_1 = b,\ x_2 = c,$$

$$x_{r+1} = (x_r f(x_{r-1}) - x_{r-1} f(x_r))/(f(x_{r-1}) - f(x_r)).$$

When this process converges to a simple zero α of $f(x)$ we have ultimately

$$x_{r-1} - \alpha \sim C(x_r - \alpha)^p, \text{ where } p = \tfrac{1}{2}(5^{1/2} + 1).$$

The asymptotic convergence rate is therefore very satisfactory; if the computation of $f'(x)$ involves as much work as that of $f(x)$ (often it takes much more work) the method is more efficient than that of Newton. However, even when $f(b)$ and $f(c)$ are of opposite signs, an extrapolatory step may well give an iterate outside (b, c) and lead to convergence to a value outside the interval or even to divergence. (Newton's method has a similar weakness.)

In the modified algorithm this difficulty is overcome by combining interpolation with bisection, the latter being used to avoid unacceptable iterates resulting from an interpolation. At the beginning of the rth step three points a_r, b_r, c_r are involved and these are such that

$$f(b_r)f(c_r) < 0 \text{ and } |f(b_r)| \leqslant |f(c_r)|.$$

The initial points a_1, b_1, c_1 are chosen as follows:

If $|f(b)| \leqslant |f(c)|$ then $b_1 = b,\ c_1 = c,\ a_1 = c_1$.

If $|f(b)| > |f(c)|$ then $b_1 = c,\ c_1 = b,\ a_1 = c_1$.

Clearly these satisfy the required conditions. The rth step is then as follows:

(i) Determine a point i_r by interpolation between a_r and b_r.

(ii) Determine m_r the mid-point of b_r and c_r.

(iii) If i_r is between b_r and m_r then it is 'accepted'. Otherwise the interpolated point is rejected and m_r is 'accepted' in its place.

(iv) Take as provisional new values

$a_{r+1} = b_r$, $b_{r+1} = i_r$ or m_r (whichever is 'accepted'), $c_{r+1} = c_r$.

(v) If b_{r+1} and c_{r+1} satisfy the conditions

$$f(b_{r+1})f(c_{r+1}) < 0 \text{ and } |f(b_{r+1})| \leqslant |f(c_{r+1})|$$

we can proceed to the next step, otherwise the provisional values are adjusted as follows.

If $f(c_{r+1})f(b_{r+1}) > 0$ then we take $c_{r+1} = b_r$; this ensures that $f(c_{r+1})f(b_{r+1}) < 0$ because of the conditions

established before the rth step. We now have to make sure that $|f(b_{r+1})| \leqslant |f(c_{r+1})|$ (with current values of b_{r+1} and c_{r+1} of course). If this is not so we can interchange b_{r+1} and c_{r+1} and take a_{r+1} to be the same as the new c_{r+1}. The right conditions now hold for the beginning of step $r + 1$.

Fig. 1 illustrates the importance of the use of bisection steps. The first step gives an acceptable interpolate and $b_2 = i_1$. In the next step interpolation between a_2 and b_2 gives a point outside the initial interval. Hence $b_3 = m_2$. Similarly interpolation between b_3 and a_3 is unacceptable and $b_4 = m_3$. Interpolation between a_4 and b_4 gives a point inside the original interval but not between m_4 and b_4 so that provisionally $b_5 = m_4$ and $a_5 = b_4$. In the first four steps c_1, c_2, c_3, c_4 and c_5 are all at the initial point. However, we now have $f(b_5)f(c_5) > 0$ and hence this c_5 is discarded and is transferred to b_4 (i.e. a_5). Since this leaves $|f(b_5)| > |f(c_5)|$ the roles of b_5 and c_5 must be interchanged. From this point onward no bisection steps are necessary.

The stopping criterion is important in all iterative procedures. It is tempting to use the criterion $|b_r - i_r| < \epsilon$, but this is unreliable. For example in Fig. 1 if $|f(b_1)| \ll |f(c_1)|$ the quantity $|b_1 - i_1|$ will be very small although neither b_1 nor i_1 is near the required zero. In fact it is quite possible for i_1 to be equal to b_1 to working accuracy.

Since the zero is between b_r and c_r at every step the criterion $|b_r - c_r| < \epsilon$ appears to be satisfactory, but unfortunately it may never be satisfied. In **Fig. 2** for example one can see that $c_r = c_s (r \geqslant s)$, while b_r tends to the zero monotonically from above. Hence $|b_r - c_r|$ tends to a finite limit. A simple stratagem overcomes this difficulty and also deals with the problem which arises when $i_r = b_r$ to working accuracy.

Suppose the stopping criterion is $|b_r - c_r| < tol$. Then if $|i_s - b_s| < tol$, the i_s is replaced by $i_s + sign\,(c_s - b_s) \times tol$. The effect of this can be seen in connexion with Fig. 2. It ensures that a b_{r+1} is finally obtained which is beyond the zero. When this happens $f(b_{r+1})f(c_{r+1}) = f(b_{r+1})f(c_s) > 0$ and c_{r+1} is switched in the normal way, immediately giving a b_{r+1} and c_{r+1} straddling the root with $|b_{r+1} - c_{r+1}| < tol$.

The specification of the quantity *tol* presents some problems. It has been our experience that some combination of a relative tolerance and an absolute tolerance is desirable and we have taken

$$tol = 4eps_1|b_r| + eps_2$$

where eps_1 is related to the machine precision and eps_2 is the absolute error which is regarded as permissible in the eigenvalue of smallest modulus.

In practice it is common for computed function values to vary over a range which exceeds that permitted for standard floating-point numbers. We therefore give an ALGOL procedure which works with function values $f(x)$ which are represented by a two word aggregate (p, q) of which p is an integer (positive or negative), a multiple of 4, and q is a standard floating-point number with $\frac{1}{16} \leqslant |q| < 1$ and $f(x) = 2^p q$. ALGOL procedures for computing determinants produced at the National Physical Laboratory have been designed to give results in this form.

```
real procedure fzero (a, b, macheps, atol, function);
value a, b, macheps, atol;
real a, b, macheps, atol;
procedure function;
comment This procedure finds the zero of a real function f(z)
        given points a and b at which the function takes
        opposite signs. It uses a procedure function (z, zf, zx)
        which, given z, forms zf and zx such that
        (1) zx is an integer, a multiple of 4
        (2) zf is a standard floating point number,
                0·0625 ⩽ abs(zf) < 1
        (3) f(z) = zf × 2 ↑ zx
        macheps is the relative machine precision and atol is
        the absolute tolerance;
begin   integer ax, bx, cx;
        real af, bf, c, cf, fb, fa, int, mid, tol, rtol;
        rtol := 4 × macheps;
        function(a, af, ax);
        function(b, bf, bx);
        goto entry;
  cont: if ax > bx then
        begin fa := af;
              fb := bf × 2 ↑ (bx − ax);
        end
        else
        begin fa := af × 2 ↑ (ax − bx);
              fb := bf;
        end;
        int := (if fa ≠ fb then (a × fb − b × fa)/(fb − fa)
                else mid);
        if abs(int − b) < tol then
        int := b + sign(c − b) × tol;
        a := b;
        af := bf;
        ax := bx;
        b := (if sign(int − mid) = sign(b − int) then int
              else mid);
        function(b, bf, bx);
        if bf = 0 then goto root;
        if sign(cf) = sign(bf) then
 entry: begin c := a;
              cf := af;
              cx := ax;
        end;
        if bx > cx ∨ bx = cx ∧ abs(bf) ⩾ abs(cf) then
        begin a := b;
              af := bf;
              ax := bx;
              b := c;
              bf := cf;
              bx := cx;
              c := a;
              cf := af;
              cx := ax;
        end;
        mid := (b + c)/2;
        tol := rtol × abs(b) + atol;
        if abs(mid − b) > tol then goto cont;
  root: fzero := b;
end fzero
```

Acknowledgements

The work described above has been carried out at the National Physical Laboratory. We should like to thank Mr. E. L. Albasiny for reading the manuscript and making many very valuable suggestions.

References

BARTH, W., MARTIN, R. S., and WILKINSON, J. H. (1967). Calculations of the eigenvalues of a symmetric tridiagonal matrix by the method of bisection, *Num. Math.*, **9**, pp. 386–393.

GIVENS, J. W. (1954). Numerical computation of the characteristic values of a real symmetric matrix. Oak Ridge National Laboratory, ORNL–1574.

GUPTA, K. K. (1968). Free vibrations of single branch structural systems, to be published in *Journal Inst. Math. and its Appl.*

KAHAN, W., and VARAH, J. (1966). Two working algorithms for the eigenvalues of a symmetric tridiagonal matrix. Technical Reports No. CS43, Stanford University.

MARTIN, R. S., PETERS, G., and WILKINSON, J. H. (1965). Symmetric decomposition of a positive definite matrix, *Num. Math.*, **7**, pp. 362–383.

MARTIN, R. S., and WILKINSON, J. H. (1967). Solution of symmetric and unsymmetric band equations and the calculation of eigenvectors of band matrices, *Num. Math.*, **9**, pp. 279–301.

MARTIN, R. S., and WILKINSON, J. H. (1968). Reduction of the symmetric eigenproblem $Ax = \lambda Bx$ and related problems to standard form, *Num. Math.*, **11**, pp. 99–110.

WIJNGAARDEN, A., ZONNEVELD, J. A., and DIJKSTRA, E. W. (1963). Programs AP200 and AP230 De serie AP200, edited by T. J. Dekker, The Mathematical Centre, Amsterdam.

WILKINSON, J. H. (1968). The calculation of the eigenvectors of codiagonal matrices, *Comp. J.*, **1**, pp. 148–152.

WILKINSON, J. H. (1965). *The Algebraic Eigenvalue Problem*, Oxford University Press, London.

WILKINSON, J. H. (1967). Two algorithms based on successive linear interpolation, Technical Report No. CS60, Stanford University.

Future papers

The following papers have been accepted for publication but, owing to pressure of space, have had to be held over to the next issue:

L. F. Blake, R. E. Lawson and I. M. Yuille. A ring processing package for use with FORTRAN or a similar high-level language.

This paper describes a software package that enables associative data structures to be represented in a computer store by means of rings of address pointers connecting blocks of data in an orderly manner.

D. Barton, S. R. Bourne and J. P. Fitch. An algebra system.

This paper describes a computing system that enables problems of manipulative algebra involving a number of elementary functions to be simply and efficiently programmed. The system has been designed with particular reference to the problems involved in the explicit calculation of the Riemann tensor and associated quantities.

M. J. M. Bernal and J. R. Whiteman. Numerical treatment of biharmonic boundary value problems with re-entrant boundaries.

Variants of the methods proposed by Motz and Woods for producing accurate approximations to the exact solution of a harmonic boundary value problem for which the region of definition contains a re-entrant corner are used to solve a similar type of biharmonic problem.

M. M. Chawla. Estimation of errors of Gauss–Chebyshev quadratures.

Estimation of errors of Gauss–Chebyshev quadratures in terms of the Chebyshev coefficients of the integrand is discussed.

S. McKee and A. R. Mitchell. Alternating direction methods for parabolic equations in two space dimensions with a mixed derivative.

An alternating direction implicit method, which requires the solution of two tridiagonal sets of equations at each time step, is derived for solving a parabolic equation with variable coefficients in two space dimensions with a mixed derivative.