11.3 Eigenvalues and Eigenvectors of a Tridiagonal Matrix

Evaluation of the Characteristic Polynomial

Once our original, real, symmetric matrix has been reduced to tridiagonal form, one possible way to determine its eigenvalues is to find the roots of the characteristic polynomial Pn (À) directly. The characteristic polynomial of a tridiagonal matrix can be evaluated for any trial value of λ by an efficient recursion relation (see, for example, Acton). The polynomials of lower degree produced during the recurrence form a Sturmian sequence that can be used to localize the eigenvalues to intervals on the real axis. A root-finding method such as bisection or Newton's method can then be employed to refine the intervals. The corresponding eigenvectors can then be found by inverse iteration (see §11.7).

Procedures based on these ideas can be found in the Handbook and in EISPACK. If, however, more than a small fraction of all the eigenvalues and eigenvectors are required, then the factorization method next considered is much more efficient.

The QR and QL Algorithms

The basic idea behind the QR algorithm is that any real matrix can be decomposed in the form

$$A = Q \cdot R, \qquad (11.3.1)$$

where Q is orthogonal and R is upper triangular. For a general matrix, the decomposition is constructed by applying Householder transformations to annihilate successive columns of A below the diagonal.

Now consider the matrix formed by writing the factors in (11.3.1) in the opposite order:

$$A' = R.Q \qquad (11.3.2)$$

Since Q is orthogonal, equation (11.3.1) gives $R = Q^{T} . A$. Thus equation (11.3.2) becomes

$$A' = Q^{T} .A . Q \qquad (11.3.3)$$

We see that A' is an orthogonal transformation of A.

You can verify that a QR transformation preserves the following properties of a matrix: symmetry, tridiagonal form, and Hessenberg form (to be defined in §11.5).

There is nothing special about choosing one of the factors of A to be upper triangular; one could equally well make it lower triangular. This is called the QL algorithm, since

$$A = Q \cdot L \qquad (11.3.4)$$

Where L is lower triangular. (The standard, but confusing, nomenclature R and L stands for whether the right or left of the matrix is nonzero.)

Recall that in the Householder reduction to tridiagonal form in §11.2, we started in the nth (last) column of the original matrix. To minimize round-

off, we then exhorted you to put the biggest elements of the matrix in the lower right-hand corner, if you can. If we now wish to diagonalize the resulting tridiagonal matrix, the QL algorithm will have smaller round-off than the QR algorithm, so we shall use QL henceforth.

The QL algorithm consists of a sequence of orthogonal transformations

$$A_s = Q_s \cdot L_s$$
$$A_{s+1} = L_s \cdot Q_s \quad (= Q_s^T \cdot A_s \cdot Q_s) \qquad (11.3.5)$$

The following (nonobvious!) theorem is the basis of the algorithm for a general matrix A: (i) If A has eigenvalues of different absolute value $|\lambda i|$, then As -> [lower triangular form] as s -> infinity. The eigenvalues appear on the diagonal in increasing order of absolute magnitude. (ii) If A has an eigenvalue $|\lambda i|$ of multiplicity p, As -> [lower triangular form] as s -> infinity, except for a diagonal block matrix of order p, whose eigenvalues -> $|\lambda i|$. The proof of this theorem is fairly lengthy; see, for example, Stoer and Bulirsch.

The workload in the QL algorithm is O(n^3) per iteration for a general matrix, which is prohibitive for a general matrix. However, the workload is only O(n) per iteration for a tridiagonal matrix and O(n^2) for a Hessenberg matrix, which makes it highly efficient on these forms.

In this section we are concerned only with the case where A is a real, symmetric, tridiagonal matrix. All the eigenvalues).i are thus real. According to the theorem, if any $|\lambda i|$ has a multiplicity p, then there must be at least p - 1 zeros on the sub- and superdiagonal. Thus the matrix can be split into submatrices that can be diagonalized separately, and the complication of diagonal blocks that can arise in the general case is irrelevant.

In the proof of the theorem quoted above, one finds that in general a superdiagonal element converges to zero like

$$a_{ij}^{(s)} \sim \left(\frac{\lambda i}{\lambda j}\right)^s \qquad (11.3.6)$$

Although $\lambda i < \lambda j$, convergence cau be slow if λi is close to λj. Convergence can be accelerated by the technique of shifting: If k is any constant. then A - k1 has eigenvalues λi - k. If we decompose

$$A_s - k_s 1 = Q_s \cdot L_s \qquad (11.3.7)$$

so that

$$A_{s+1} = L_s \cdot Q_s + k_s 1$$
$$= Q_s^T \cdot A_s \cdot Q_s \qquad (11.3.8)$$

then the convergence is determined by the ratio

$$\frac{\lambda i - k_s}{\lambda j - k_s} \qquad (11.3.9)$$

The idea is to choose the shift ks at each stage to maximize the rate of convergence. A good choice for the shift initially would be ks close to $\lambda 1$, the smallest eigenvalue. Then the first row of off-diagonal elements would tendd rapidly to zero. However, $\lambda 1$ is not usually known a priori. A very effective strategy in practice (although there is no proof that it is optimal) is to compute the eigenvalues of the leading 2 x 2 diagonal submatrix of A. Then set ks equal to the eigenvalue closer to a11.

More generally, suppose you have already found r - 1 eigenvalues of A. Then you can deflate the matrix by crossing out the first r - 1 rows and and columns leaving

$$\mathbf{A} = \begin{bmatrix} 0 & & \cdots & \cdots & & 0 \\ & \ddots & & & & \\ & 0 & & & & \\ \vdots & & d_r & e_r & & \vdots \\ \vdots & & e_r & d_{r+1} & & \\ & & & \cdots & & 0 \\ & & & & d_{n-1} & e_{n-1} \\ 0 & & \cdots & 0 & e_{n-1} & d_n \end{bmatrix} \qquad (11.3.10)$$

Choose ks equal to the eigenvalue of the leading 2 x 2 submatrix that is closerto dr. One can show that the convergence of the algorithm with this strategy is generally cubic (and at worst quadratic for degenerate eigenvalues). This rapid convergence is what makes the algorithm so attractive.

Note that with shifting, the eigenvalues no longer necessarily appear on the diagonal in order of increasing absolute magnitude. The routine EIGSRT ($11.1) can be used if required.

As we mentioned earlier, the QL decomposition of a general matrix is effected by a sequence of Householder transformations. For a tridiagonal matrix, however, it is more efficient to use plane rotations Ppq. One uses the sequence P12,P23,...,Pn-1,n to annihilate the elements a12,a23,...,an-1,n. By symmetry, the subdiagonal elements a21, a32,...,an,n-1 will be annihilated too. Thus each Qs is a product of plane rotations:

$$Q_s^T = P_1^{(s)} \cdot P_2^{(s)} \cdots P_{n-1}^{(s)} \qquad (11.3.11)$$

where P_i annihilates $a_{i,i+1}$. Note that it is Q^T in equation (11.3.11), not Q, because we defined $L = Q^T \cdot A$.

QL Algorithm with Implicit Shifts

The algorithm as described so far can be very successful. However when the elements of A differ widely in order of magnitude, subtracting a large ks from the diagonal elements can lead to loss of accuracy for the small eigenvalues. This difficulty is avoided by the QL algorithm with implicit shifts. The implicit QL algorithm is mathematically equivalent to the original QL algorithm, but the computation does not require **ks1** to be actually subtracted from **A**.

The algorithm is based on the following lemma: If A is a symmetric nonsingular matrix and $B = Q^T \cdot A \cdot Q$, where Q is orthogonal and B is tridiagonal with positive off-diagonal elements, then Q and B are fully determined when the last row of Q^T is specified. Proof: Let q_i^T denote the ith row vector of the matrix Q^T. Then q_i is the ith column vector of the matrix Q^T. The relation $B \cdot Q^T = Q^T \cdot A$ can be written

$$\begin{bmatrix} \beta_1 & \gamma_1 & & & & \\ \alpha_2 & \beta_2 & \gamma_2 & & & \\ & & \vdots & & & \\ & & & \alpha_{n-1} & \beta_{n-1} & \gamma_{n-1} \\ & & & & \alpha_n & \beta_n \end{bmatrix} \cdot \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_{n-1}^T \\ \mathbf{q}_n^T \end{bmatrix} = \begin{bmatrix} \mathbf{q}_1^T \\ \mathbf{q}_2^T \\ \vdots \\ \mathbf{q}_{n-1}^T \\ \mathbf{q}_n^T \end{bmatrix} \cdot \mathbf{A} \quad (11.3.12)$$

The nth row of this matrix equation is

$$\alpha_n \, q_{n-1}^T + \beta_n \, q_n^T = q_n^T \cdot A \qquad (11.3.13)$$

Since Q is orthogonal,

$$q_n^T \cdot q_m = \delta_{nm} \qquad (11.3.14)$$

Thus if we postmultiply equation (11.3.13) by q_n we find

$$\beta_n = q_n^T \cdot A \cdot q_n \qquad (11.3.15)$$

which is known since q_n is known. Then equation (11.3.13) gives

$$\alpha_n \, q_{n-1}^T = Z_{n-1}^T \qquad (11.3.16)$$

where

$$Z_{n-1}^T \sim q_n^T \cdot A - \beta_n \, q_n^T \qquad (11.3.17)$$

is known. Therefore

$$\alpha_n^2 = z_{n-1}^T z_{n-1} \qquad (11.3.18)$$

or

$$\alpha_n = |z_{n-1}| \qquad (11.3.19)$$

and

$$q_{n-1}^T = z_{n-1}^T / \alpha_n \qquad (11.3.20)$$

(where α_n is nonzero by hypothesis). Similarly, one can show by induction that if we know q_n, q_{n-1}, ... ,q_{n-j} and the α's, β's and γ'S up to level n - i, one can determine the quantities at level n - (j + 1).

To apply the lemma in practice, suppose one can somehow find a tridiagonal matrix $\bar{A}_{s+1}$ such that

$$\bar{A}_{s+1} = \bar{Q}_s^T \cdot \bar{A}_s \cdot \bar{Q}_s \qquad (11.3.21)$$

where $\bar{Q}_s^T$ is orthogonal and has the same last row as Q_s^T in the original QL algorithm, Then

$$\bar{Q}_s = Q_s \text{ and } \bar{A}_{s+1} = A_{s+1} .$$

Now, in the original algorithm, from equation (11.3.11) we see that the last row of Q_s^T is the same as the last row of $P_{n-1}^{(s)}$. But recal! that $P_{n-1}^{(s)}$ is a plane rotation designed to annihilate the (n-1, n) element of $A_s - ks1$.

A simple calculation using the expression (11.1.1) shows that it has parameters

$$c = \frac{dn - ks}{\text{sqrt}(en^2+(dn-ks)^2)} \qquad s = \frac{-en-1}{\text{sqrt}(en^2+(dn-ks)^2)} \qquad (11.3.22)$$

The matrix $P_{n-1}^{(s)} \cdot A_s \cdot P_{n-1}^{(s)T}$ is tridiagonal with 2 extra elements:

$$\begin{array}{|ccccc|} \hline \multicolumn{5}{|l|}{\ldots} \\ & x & x & x & \\ & & x & x & \mathbf{x} \\ & & & x & x \\ & & & \mathbf{x} & x \\ \hline \end{array} \qquad (11.3.23)$$

We must now reduce this to tridiagonal form with an orthogonal matrix whose last row is $[0, 0, \ldots, 0, 1]$ so that the last row of $\bar{Q}_s^T$ will stay equal to $P_{n-1}^{(s)}$. This can be done by a sequence of Householder or Givens transformations.

For the e special form of the matrix (11.3.23), Givens is better. We rotate in the plane (n-2, n-1) to annihilate the (n-2, n) element. [By symmetry, the (n, n-2) element will also be zeroed. This leaves us with tridiagonal form except for extra elements (n-3, n-1) and (n-1, n-3). We annihilate these with a rotation in the (n-3, n-2) plane, and so on. Thus a sequence of n-2 Givens rotations are required. The result is that

$$Q_s^T = \bar{Q}_s^T = \bar{P}_1^{(s)} \cdot \bar{P}_2^{(s)} \cdots \bar{P}_{n-2}^{(s)} \cdot \bar{P}_{n-1}^{(s)} \qquad (11.3.24)$$

where the P's are the Givens rotations and P_{n-1} is the same plane rotation as in the original algorithm. Then equation (11.3.21) gives the next iterate of A. Note that the shift ks enters implicitly through the parameters (11.3.22).

The routine TQLI ("Tridiagonal QL Implicit"}, based algorithmically on the Handbook and EISPACK implementations, works extremely well in practice. The number of iterations for the first few eigenvalues might be 4 or 5 say, but meanwhile the off-diagonal elements in the lower right-hand corner have been reduced too. The later eigenvalues are liberated with very little work. The average number of iterations per eigenvalue is typically 1.3 - 1.6. The operation count per iteration is O(n), with a fairly large effective coefficient, say ~ 20n. The total operation count for the diagonalization is then ~ 20n x (1.3 - 1.6)n ~ $30n^2$. If the eigenvectors are required. the commented statements are included and there is an additional, much larger, workload of about $3n^3$ operations.

[From BIBLI 08].