Chapter 18

Computing Eigenvalues and Eigenvectors

After the problem of solving a linear system, the problem of computing the eigenvalues and the eigenvectors of a real or complex matrix is one of most important problems of numerical linear algebra. Several methods exist, among which we mention Jacobi, Givens–Householder, divide-and-conquer, QR iteration, and Rayleigh–Ritz; see Demmel [48], Trefethen and Bau [176], Meyer [125], Serre [156], Golub and Van Loan [80], and Ciarlet [41]. Typically, better performing methods exist for special kinds of matrices, such as symmetric matrices.

In theory, given an $n \times n$ complex matrix A, if we could compute a Schur form $A = UTU^*$, where T is upper triangular and U is unitary, we would obtain the eigenvalues of A, since they are the diagonal entries in T. However, this would require finding the roots of a polynomial, but methods for doing this are known to be numerically very unstable, so this is not a practical method.

A common paradigm is to construct a sequence (P_k) of matrices such that $A_k = P_k^{-1}AP_k$ converges, in some sense, to a matrix whose eigenvalues are easily determined. For example, $A_k = P_k^{-1}AP_k$ could become upper triangular in the limit. Furthermore, P_k is typically a product of "nice" matrices, for example, orthogonal matrices.

For general matrices, that is, matrices that are not symmetric, the QR iteration algorithm, due to Rutishauser, Francis, and Kublanovskaya in the early 1960s, is one of the most efficient algorithms for computing eigenvalues. A fascinating account of the history of the QR algorithm is given in Golub and Uhlig [79]. The QR algorithm constructs a sequence of matrices (A_k), where A_{k+1} is obtained from A_k by performing a QR-decomposition $A_k = Q_kR_k$ of A_k and then setting $A_{k+1} = R_kQ_k$, the result of swapping Q_k and R_k. It is immediately verified that $A_{k+1} = Q_k^*A_kQ_k$, so A_k and A_{k+1} have *the same eigenvalues*, which are the eigenvalues of A.

The basic version of this algorithm runs into difficulties with matrices that have several eigenvalues with the same modulus (it may loop or not "converge" to an upper triangular matrix). There are ways of dealing with some of these problems, but for ease of exposition, we first present a simplified version of the QR algorithm which we call basic QR algorithm. We prove a convergence theorem for the basic QR algorithm, under the rather restrictive hypothesis that the input matrix A is diagonalizable and that its eigenvalues are nonzero and have distinct moduli. The proof shows that the part of A_k strictly below the diagonal converges to zero and that the diagonal entries of A_k converge to the eigenvalues of A.

Since the convergence of the QR method depends crucially only on the fact that the part of A_k below the diagonal goes to zero, it would be highly desirable if we could replace A by a similar matrix U^*AU easily computable from A and having lots of zero strictly below the diagonal. It turns out that there is a way to construct a matrix $H = U^*AU$ which is almost triangular, except that it may have an extra nonzero diagonal below the main diagonal. Such matrices called, *Hessenberg matrices*, are discussed in Section 18.2. An $n \times n$ diagonalizable Hessenberg matrix H having the property that $h_{i+1i} \neq 0$ for $i = 1, \ldots, n-1$ (such a matrix is called *unreduced*) has the nice property that its eigenvalues are all distinct. Since every Hessenberg matrix is a block diagonal matrix of unreduced Hessenberg blocks, *it suffices to compute the eigenvalues of unreduced Hessenberg matrices.* There is a special case of particular interest: symmetric (or Hermitian) positive definite tridiagonal matrices. Such matrices must have real positive distinct eigenvalues, so the QR algorithm converges to a diagonal matrix.

In Section 18.3, we consider techniques for making the basic QR method practical and more efficient. The first step is to convert the original input matrix A to a similar matrix H in Hessenberg form, and to apply the QR algorithm to H (actually, to the unreduced blocks of H). The second and crucial ingredient to speed up convergence is to add shifts.

A shift is the following step: pick some σ_k, hopefully close to some eigenvalue of A (in general, λ_n), QR-factor $A_k - \sigma_k I$ as

$$A_k - \sigma_k I = Q_k R_k,$$

and then form

$$A_{k+1} = R_k Q_k + \sigma_k I.$$

It is easy to see that we still have $A_{k+1} = Q_k^* A_k Q_k$. A judicious choice of σ_k can speed up convergence considerably. If H is real and has pairs of complex conjugate eigenvalues, we can perform a double shift, and it can be arranged that we work in real arithmetic.

The last step for improving efficiency is to compute $A_{k+1} = Q_k^* A_k Q_k$ without even performing a QR-factorization of $A_k - \sigma_k I$. This can be done when A_k is unreduced Hessenberg. Such a method is called QR iteration with implicit shifts. There is also a version of QR iteration with implicit double shifts.

If the dimension of the matrix A is very large, we can find approximations of some of the eigenvalues of A by using a truncated version of the reduction to Hessenberg form due to Arnoldi in general and to Lanczos in the symmetric (or Hermitian) tridiagonal case. *Arnoldi iteration* is discussed in Section 18.4. If A is an $m \times m$ matrix, for $n \ll m$ (n much smaller than m) the idea is to generate the $n \times n$ Hessenberg submatrix H_n of the full Hessenberg matrix H (such that $A = UHU^*$) consisting of its first n rows and n columns; the matrix U_n consisting of the first n columns of U is also produced. The Rayleigh–Ritz method consists in computing the eigenvalues of H_n using the QR- method with shifts. These eigenvalues, called *Ritz values*, are approximations of the eigenvalues of A. Typically, extreme eigenvalues are found first.

Arnoldi iteration can also be viewed as a way of computing an orthonormal basis of a *Krylov subspace*, namely the subspace $\mathcal{K}_n(A, b)$ spanned by $(b, Ab, \ldots, A^n b)$. We can also use Arnoldi iteration to find an approximate solution of a linear equation $Ax = b$ by minimizing $\|b - Ax_n\|_2$ for all x_n is the Krylov space $\mathcal{K}_n(A, b)$. This method named GMRES is discussed in Section 18.5.

The special case where H is a symmetric (or Hermitian) tridiagonal matrix is discussed in Section 18.6. In this case, Arnoldi's algorithm becomes *Lanczos' algorithm*. It is much more efficient than Arnoldi iteration.

We close this chapter by discussing two classical methods for computing a single eigenvector and a single eigenvalue: power iteration and inverse (power) iteration; see Section 18.7.

18.1 The Basic QR Algorithm

Let A be an $n \times n$ matrix which is assumed to be diagonalizable and invertible. The basic QR algorithm makes use of two very simple steps. Starting with $A_1 = A$, we construct sequences of matrices (A_k), (Q_k) (R_k) and (P_k) as follows:

$$\begin{array}{lr} \text{Factor} & A_1 = Q_1 R_1 \\ \text{Set} & A_2 = R_1 Q_1 \\ \text{Factor} & A_2 = Q_2 R_2 \\ \text{Set} & A_3 = R_2 Q_2 \\ & \vdots \\ \text{Factor} & A_k = Q_k R_k \\ \text{Set} & A_{k+1} = R_k Q_k \\ & \vdots \end{array}$$

Thus, A_{k+1} is obtained from a QR-factorization $A_k = Q_k R_k$ of A_k by swapping Q_k and R_k. Define P_k by

$$P_k = Q_1 Q_2 \cdots Q_k.$$

Since $A_k = Q_k R_k$, we have $R_k = Q_k^* A_k$, and since $A_{k+1} = R_k Q_k$, we obtain

$$A_{k+1} = Q_k^* A_k Q_k. \qquad (*_1)$$

An obvious induction shows that

$$A_{k+1} = Q_k^* \cdots Q_1^* A_1 Q_1 \cdots Q_k = P_k^* A P_k,$$

that is

$$A_{k+1} = P_k^* A P_k. \tag{$*_2$}$$

Therefore, A_{k+1} and A are similar, so they have the same eigenvalues.

The basic QR iteration method consists in computing the sequence of matrices A_k, and in the ideal situation, to expect that A_k "converges" to an upper triangular matrix, more precisely that the part of A_k below the main diagonal goes to zero, and the diagonal entries converge to the eigenvalues of A.

This ideal situation is only achieved in rather special cases. For one thing, if A is unitary (or orthogonal in the real case), since in the QR decomposition we have $R = I$, we get $A_2 = IQ = Q = A_1$, so the method does *not* make any progress. Also, if A is a real matrix, since the A_k are also real, if A has complex eigenvalues, then the part of A_k below the main diagonal can't go to zero. Generally, the method runs into troubles whenever A has distinct eigenvalues with the same modulus.

The convergence of the sequence (A_k) is only known under some fairly restrictive hypotheses. Even under such hypotheses, this is not really genuine convergence. Indeed, it can be shown that the part of A_k below the main diagonal goes to zero, and the diagonal entries converge to the eigenvalues of A, but the part of A_k above the diagonal *may not converge*. However, for the purpose of finding the eigenvalues of A, this does not matter.

The following convergence result is proven in Ciarlet [41] (Chapter 6, Theorem 6.3.10 and Serre [156] (Chapter 13, Theorem 13.2). It is rarely applicable in practice, except for symmetric (or Hermitian) positive definite matrices, as we will see shortly.

Theorem 18.1. *Suppose the (complex) $n \times n$ matrix A is invertible, diagonalizable, and that its eigenvalues $\lambda_1, \ldots, \lambda_n$ have different moduli, so that*

$$|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0.$$

If $A = P\Lambda P^{-1}$, where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$, and if P^{-1} has an LU-factorization, then the strictly lower-triangular part of A_k converges to zero, and the diagonal of A_k converges to Λ.

Proof. We reproduce the proof in Ciarlet [41] (Chapter 6, Theorem 6.3.10). The strategy is to study the asymptotic behavior of the matrices $P_k = Q_1 Q_2 \cdots Q_k$. For this, it turns out that we need to consider the powers A^k.

Step 1. Let $\mathcal{R}_k = R_k \cdots R_2 R_1$. We claim that

$$A^k = (Q_1 Q_2 \cdots Q_k)(R_k \cdots R_2 R_1) = P_k \mathcal{R}_k. \tag{$*_3$}$$

We proceed by induction. The base case $k = 1$ is trivial. For the induction step, from $(*_2)$, we have

$$P_k A_{k+1} = A P_k.$$

Since $A_{k+1} = R_k Q_k = Q_{k+1} R_{k+1}$, we have

$$P_{k+1}\mathcal{R}_{k+1} = P_k Q_{k+1} R_{k+1} \mathcal{R}_k = P_k A_{k+1} \mathcal{R}_k = A P_k \mathcal{R}_k = A A^k = A^{k+1}$$

establishing the induction step.

Step 2. We will express the matrix P_k as $P_k = Q\widetilde{Q}_k D_k$, in terms of a diagonal matrix D_k with unit entries, with Q and $\widetilde{Q}_k$ unitary.

Let $P = QR$, a QR-factorization of P (with R an upper triangular matrix with positive diagonal entries), and $P^{-1} = LU$, an LU-factorization of P^{-1}. Since $A = P\Lambda P^{-1}$, we have

$$A^k = P\Lambda^k P^{-1} = QR\Lambda^k LU = QR(\Lambda^k L \Lambda^{-k})\Lambda^k U. \tag{$*_4$}$$

Here, Λ^{-k} is the diagonal matrix with entries λ_i^{-k}. The reason for introducing the matrix $\Lambda^k L \Lambda^{-k}$ is that its asymptotic behavior is easy to determine. Indeed, we have

$$(\Lambda^k L \Lambda^{-k})_{ij} = \begin{cases} 0 & \text{if } i < j \\ 1 & \text{if } i = j \\ \left(\dfrac{\lambda_i}{\lambda_j}\right)^k L_{ij} & \text{if } i > j. \end{cases}$$

The hypothesis that $|\lambda_1| > |\lambda_2| > \cdots > |\lambda_n| > 0$ implies that

$$\lim_{k \mapsto \infty} \Lambda^k L \Lambda^{-k} = I. \tag{$\dagger$}$$

Note that it is to obtain this limit that we made the hypothesis on the moduli of the eigenvalues. We can write

$$\Lambda^k L \Lambda^{-k} = I + F_k, \quad \text{with} \quad \lim_{k \mapsto \infty} F_k = 0,$$

and consequently, since $R(\Lambda^k L \Lambda^{-k}) = R(I + F_k) = R + RF_kR^{-1}R = (I + RF_kR^{-1})R$, we have

$$R(\Lambda^k L \Lambda^{-k}) = (I + RF_kR^{-1})R. \tag{$*_5$}$$

By Proposition 9.11(1), since $\lim_{k \mapsto \infty} F_k = 0$, and thus $\lim_{k \mapsto \infty} RF_kR^{-1} = 0$, the matrices $I + RF_kR^{-1}$ are invertible for k large enough. Consequently for k large enough, we have a QR-factorization

$$I + RF_kR^{-1} = \widetilde{Q}_k \widetilde{R}_k, \tag{$*_6$}$$

with $(\widetilde{R}_k)_{ii} > 0$ for $i = 1, \ldots, n$. Since the matrices $\widetilde{Q}_k$ are unitary, we have $\left\| \widetilde{Q}_k \right\|_2 = 1$, so the sequence $(\widetilde{Q}_k)$ is bounded. It follows that it has a convergent subsequence $(\widetilde{Q}_\ell)$ that converges to some matrix $\widetilde{Q}$, which is also unitary. Since

$$\widetilde{R}_\ell = (\widetilde{Q}_\ell)^*(I + RF_\ell R^{-1}),$$

we deduce that the subsequence $(\widetilde{R}_\ell)$ also converges to some matrix $\widetilde{R}$, which is also upper triangular with positive diagonal entries. By passing to the limit (using the subsequences), we get $\widetilde{R} = (\widetilde{Q})^*$, that is,

$$I = \widetilde{Q}\widetilde{R}.$$

By the uniqueness of a QR-decomposition (when the diagonal entries of R are positive), we get

$$\widetilde{Q} = \widetilde{R} = I.$$

Since the above reasoning applies to any subsequences of $(\widetilde{Q}_k)$ and $(\widetilde{R}_k)$, by the uniqueness of limits, we conclude that the "full" sequences $(\widetilde{Q}_k)$ and $(\widetilde{R}_k)$ converge:

$$\lim_{k\mapsto\infty} \widetilde{Q}_k = I, \quad \lim_{k\mapsto\infty} \widetilde{R}_k = I.$$

Since by $(*_4)$,

$$A^k = QR(\Lambda^k L \Lambda^{-k})\Lambda^k U,$$

by $(*_5)$,

$$R(\Lambda^k L \Lambda^{-k}) = (I + RF_k R^{-1})R,$$

and by $(*_6)$

$$I + RF_k R^{-1} = \widetilde{Q}_k \widetilde{R}_k,$$

we proved that

$$A^k = (Q\widetilde{Q}_k)(\widetilde{R}_k R \Lambda^k U). \tag{$*_7$}$$

Observe that $Q\widetilde{Q}_k$ is a unitary matrix, and $\widetilde{R}_k R \Lambda^k U$ is an upper triangular matrix, as a product of upper triangular matrices. However, some entries in Λ may be negative, so we can't claim that $\widetilde{R}_k R \Lambda^k U$ has positive diagonal entries. Nevertheless, we have another QR-decomposition of A^k,

$$A^k = (Q\widetilde{Q}_k)(\widetilde{R}_k R \Lambda^k U) = P_k \mathcal{R}_k.$$

It is easy to prove that there is diagonal matrix D_k with $|(D_k)_{ii}| = 1$ for $i = 1, \ldots, n$, such that

$$P_k = Q\widetilde{Q}_k D_k. \tag{$*_8$}$$

The existence of D_k is consequence of the following fact: If an invertible matrix B has two QR factorizations $B = Q_1R_1 = Q_2R_2$, then there is a diagonal matrix D with unit entries such that $Q_2 = DQ_1$.

The expression for P_k in $(*_8)$ is that which we were seeking.

Step 3. Asymptotic behavior of the matrices $A_{k+1} = P_k^* A P_k$.

Since $A = P\Lambda P^{-1} = QR\Lambda R^{-1}Q^{-1}$ and by $(*_8)$, $P_k = Q\widetilde{Q}_k D_k$, we get

$$A_{k+1} = D_k^*(\widetilde{Q}_k)^* Q^* QR\Lambda R^{-1}Q^{-1}Q\widetilde{Q}_k D_k = D_k^*(\widetilde{Q}_k)^* R\Lambda R^{-1}\widetilde{Q}_k D_k. \tag{$*_9$}$$

Since $\lim_{k\mapsto\infty}\widetilde{Q}_k = I$, we deduce that

$$\lim_{k\mapsto\infty}(\widetilde{Q}_k)^* R\Lambda R^{-1}\widetilde{Q}_k = R\Lambda R^{-1} = \begin{pmatrix} \lambda_1 & * & \cdots & * \\ 0 & \lambda_2 & \cdots & * \\ \vdots & & \ddots & \vdots \\ 0 & 0 & \cdots & \lambda_n \end{pmatrix},$$

an upper triangular matrix with the eigenvalues of A on the diagonal. Since R is upper triangular, the order of the eigenvalues is preserved. If we let

$$\mathcal{D}_k = (\widetilde{Q}_k)^* R\Lambda R^{-1}\widetilde{Q}_k, \tag{$*_{10}$}$$

then by $(*_9)$ we have $A_{k+1} = D_k^*\mathcal{D}_k D_k$, and since the matrices D_k are diagonal matrices, we have

$$(A_{k+1})_{jj} = (D_k^*\mathcal{D}_k D_k)_{ij} = \overline{(D_k)_{ii}}(D_k)_{jj}(\mathcal{D}_k)_{ij},$$

which implies that

$$(A_{k+1})_{ii} = (\mathcal{D}_k)_{ii}, \quad i = 1, \ldots, n, \tag{$*_{11}$}$$

since $|(D_k)_{ii}| = 1$ for $i = 1, \ldots, n$. Since $\lim_{k\mapsto\infty}\mathcal{D}_k = R\Lambda R^{-1}$, we conclude that the strictly lower-triangular part of A_{k+1} converges to zero, and the diagonal of A_{k+1} converges to Λ. □

Observe that if the matrix A is real, then the hypothesis that the eigenvalues have distinct moduli implies that the eigenvalues are all real and simple.

The following Matlab program implements the basic QR-method using the function qrv4 from Section 12.8.

```
function T = qreigen(A,m)
T = A;
for k = 1:m
  [Q R] = qrv4(T);
  T = R*Q;
end
end
```

Example 18.1. If we run the function qreigen with 100 iterations on the 8×8 symmetric matrix

$$A = \begin{pmatrix} 4 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 4 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 4 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 4 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 4 \end{pmatrix},$$

we find the matrix

$$T = \begin{pmatrix} 5.8794 & 0.0015 & 0.0000 & -0.0000 & 0.0000 & -0.0000 & 0.0000 & -0.0000 \\ 0.0015 & 5.5321 & 0.0001 & 0.0000 & -0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0 & 0.0001 & 5.0000 & 0.0000 & -0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0 & 0 & 0.0000 & 4.3473 & 0.0000 & 0.0000 & 0.0000 & 0.0000 \\ 0 & 0 & 0 & 0.0000 & 3.6527 & 0.0000 & 0.0000 & -0.0000 \\ 0 & 0 & 0 & 0 & 0.0000 & 3.0000 & 0.0000 & -0.0000 \\ 0 & 0 & 0 & 0 & 0 & 0.0000 & 2.4679 & 0.0000 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0.0000 & 2.1206 \end{pmatrix}.$$

The diagonal entries match the eigenvalues found by running the `Matlab` function `eig(A)`.

If several eigenvalues have the same modulus, then the proof breaks down, we can no longer claim (†), namely that

$$\lim_{k \mapsto \infty} \Lambda^k L \Lambda^{-k} = I.$$

If we assume that P^{-1} has a suitable "block LU-factorization," it can be shown that the matrices A_{k+1} converge to a block upper-triangular matrix, where each block corresponds to eigenvalues having the same modulus. For example, if A is a 9×9 matrix with eigenvalues λ_i such that $|\lambda_1| = |\lambda_2| = |\lambda_3| > |\lambda_4| > |\lambda_5| = |\lambda_6| = |\lambda_7| = |\lambda_8| = |\lambda_9|$, then A_k converges to a block diagonal matrix (with three blocks, a 3×3 block, a 1×1 block, and a 5×5 block) of the form

$$\begin{pmatrix} \star & \star & \star & * & * & * & * & * & * \\ \star & \star & \star & * & * & * & * & * & * \\ \star & \star & \star & * & * & * & * & * & * \\ 0 & 0 & 0 & \star & * & * & * & * & * \\ 0 & 0 & 0 & 0 & \star & \star & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \star & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \star & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \star & \star & \star & \star \\ 0 & 0 & 0 & 0 & \star & \star & \star & \star & \star \end{pmatrix}.$$

See Ciarlet [41] (Chapter 6 Section 6.3) for more details.

Under the conditions of Theorem 18.1, in particular, if A is a symmetric (or Hermitian) positive definite matrix, the eigenvectors of A can be approximated. However, when A is not a symmetric matrix, since the upper triangular part of A_k does not necessarily converge, one has to be cautious that a rigorous justification is lacking.

Suppose we apply the QR algorithm to a matrix A satisfying the hypotheses of Theorem Theorem 18.1. For k large enough, $A_{k+1} = P_k^* A P_k$ is nearly upper triangular and the diagonal entries of A_{k+1} are all distinct, so we can consider that they are the eigenvalues of A_{k+1}, and thus of A. To avoid too many subscripts, write T for the upper triangular matrix obtained by settting the entries of the part of A_{k+1} below the diagonal to 0. Then we can find the corresponding eigenvectors by solving the linear system

$$Tv = t_{ii}v,$$

and since T is upper triangular, this can be done by bottom-up elimination. We leave it as an exercise to show that the following vectors $v^i = (v^i_1, \ldots, v^i_n)$ are eigenvectors:

$$v^1 = e_1,$$

and if $i = 2, \ldots, n$, then

$$v^i_j = \begin{cases} 0 & \text{if } i+1 \leq j \leq n \\ 1 & \text{if } j = i \\ -\dfrac{t_{jj+1}v^i_{j+1} + \cdots + t_{ji}v^i_i}{t_{jj} - t_{ii}} & \text{if } i-1 \geq j \geq 1. \end{cases}$$

Then the vectors $(P_kv^1, \ldots, P_kv^n)$ are a basis of (approximate) eigenvectors for A. In the special case where T is a diagonal matrix, then $v^i = e_i$ for $i = 1, \ldots, n$ and the columns of P_k are an orthonormal basis of (approximate) eigenvectors for A.

If A is a real matrix whose eigenvalues are not all real, then there is some complex pair of eigenvalues $\lambda + i\mu$ (with $\mu \neq 0$), and the QR-algorithm cannot converge to a matrix whose strictly lower-triangular part is zero. There is a way to deal with this situation using upper Hessenberg matrices which will be discussed in the next section.

Since the convergence of the QR method depends crucially only on the fact that the part of A_k below the diagonal goes to zero, it would be highly desirable if we could replace A by a similar matrix U^*AU easily computable from A having lots of zero strictly below the diagonal. We can't expect U^*AU to be a diagonal matrix (since this would mean that A was easily diagonalized), but it turns out that there is a way to construct a matrix $H = U^*AU$ which is almost triangular, except that it may have an extra nonzero diagonal below the main diagonal. Such matrices called Hessenberg matrices are discussed in the next section.

18.2 Hessenberg Matrices

Definition 18.1. An $n \times n$ matrix (real or complex) H is an *(upper) Hessenberg matrix* if it is almost triangular, except that it may have an extra nonzero diagonal below the main diagonal. Technically, $h_{jk} = 0$ for all (j, k) such that $j - k \geq 2$.

The 5×5 matrix below is an example of a Hessenberg matrix.

$$H = \begin{pmatrix} * & * & * & * & * \\ h_{21} & * & * & * & * \\ 0 & h_{32} & * & * & * \\ 0 & 0 & h_{43} & * & * \\ 0 & 0 & 0 & h_{54} & * \end{pmatrix}.$$

The following result can be shown.

Theorem 18.2. *Every $n \times n$ complex or real matrix A is similar to an upper Hessenberg matrix H, that is, $A = UHU^*$ for some unitary matrix U. Furthermore, U can be constructed as a product of Householder matrices (the definition is the same as in Section 13.1, except that W is a complex vector, and that the inner product is the Hermitian inner product on $\mathbb{C}^n$). If A is a real matrix, then U is an orthogonal matrix (and H is a real matrix).*

Theorem 18.2 and algorithms for converting a matrix to Hessenberg form are discussed in Trefethen and Bau [176] (Lecture 26), Demmel [48] (Section 4.4.6, in the real case), Serre [156] (Theorem 13.1), and Meyer [125] (Example 5.7.4, in the real case). The proof of correctness is not difficult and will be the object of a homework problem.

The following functions written in `Matlab` implement a function to compute a Hessenberg form of a matrix.

The function `house` constructs the normalized vector u defining the Householder reflection that zeros all but the first entries in a vector x.

```
function [uu, u] = house(x)
tol = 2*10^(-15);   % tolerance
uu = x;
p = size(x,1);
% computes l^1-norm of x(2:p,1)
n1 = sum(abs(x(2:p,1)));
if n1 <= tol
   u = zeros(p,1);  uu = u;
else
   l = sqrt(x’*x);  % l^2 norm of x
   uu(1) = x(1) + signe(x(1))*l;
   u = uu/sqrt(uu’*uu);
end
end
```

The function `signe(z)` returms -1 if $z < 0$, else $+1$.

The function `buildhouse` builds a Householder reflection from a vector uu.

```
function P = buildhouse(v,i)
% This function builds a Householder reflection
%   [I 0 ]
%   [0 PP]
%   from a Householder reflection
%   PP = I - 2uu*uu’
%   where uu = v(i:n)
%   If uu = 0 then P - I
%

n = size(v,1);
if v(i:n) == zeros(n - i + 1,1)
   P = eye(n);
else
   PP = eye(n - i + 1) - 2*v(i:n)*v(i:n)’;
   P = [eye(i-1) zeros(i-1, n - i + 1); zeros(n - i + 1, i - 1) PP];
end
end
```

The function `Hessenberg1` computes an upper Hessenberg matrix H and an orthogonal matrix Q such that $A = Q^{\top}HQ$.

```
function [H, Q] = Hessenberg1(A)
%
%   This function constructs an upper Hessenberg
%   matrix H and an orthogonal matrix Q such that
%   A = Q’ H Q

n = size(A,1);
H = A;
Q = eye(n);
for i = 1:n-2
  %  H(i+1:n,i)
    [~,u] = house(H(i+1:n,i));
  %  u
    P =  buildhouse(u,1);
    Q(i+1:n,i:n) = P*Q(i+1:n,i:n);
    H(i+1:n,i:n) = H(i+1:n,i:n) - 2*u*(u’)*H(i+1:n,i:n);
    H(1:n,i+1:n) = H(1:n,i+1:n) - 2*H(1:n,i+1:n)*u*(u’);
end
end
```

Example 18.2. If

$$A = \begin{pmatrix} 1 & 2 & 3 & 4 \\ 2 & 3 & 4 & 5 \\ 3 & 4 & 5 & 6 \\ 4 & 5 & 6 & 7 \end{pmatrix},$$

running `Hessenberg1` we find

$$H = \begin{pmatrix} 1.0000 & -5.3852 & 0 & 0 \\ -5.3852 & 15.2069 & -1.6893 & -0.0000 \\ -0.0000 & -1.6893 & -0.2069 & -0.0000 \\ 0 & -0.0000 & 0.0000 & 0.0000 \end{pmatrix}$$

$$Q = \begin{pmatrix} 1.0000 & 0 & 0 & 0 \\ 0 & -0.3714 & -0.5571 & -0.7428 \\ 0 & 0.8339 & 0.1516 & -0.5307 \\ 0 & 0.4082 & -0.8165 & 0.4082 \end{pmatrix}.$$

An important property of (upper) Hessenberg matrices is that if some subdiagonal entry $H_{p+1p} = 0$, then H is of the form

$$H = \begin{pmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{pmatrix},$$

where both H_{11} and H_{22} are upper Hessenberg matrices (with H_{11} a $p \times p$ matrix and H_{22} a $(n-p) \times (n-p)$ matrix), and the eigenvalues of H are the eigenvalues of H_{11} and H_{22}. For example, in the matrix

$$H = \begin{pmatrix} * & * & * & * & * \\ h_{21} & * & * & * & * \\ 0 & h_{32} & * & * & * \\ 0 & 0 & h_{43} & * & * \\ 0 & 0 & 0 & h_{54} & * \end{pmatrix},$$

if $h_{43} = 0$, then we have the block matrix

$$H = \begin{pmatrix} * & * & * & * & * \\ h_{21} & * & * & * & * \\ 0 & h_{32} & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & h_{54} & * \end{pmatrix}.$$

Then the list of eigenvalues of H is the concatenation of the list of eigenvalues of H_{11} and the list of the eigenvalues of H_{22}. This is easily seen by induction on the dimension of the block H_{11}.

More generally, every upper Hessenberg matrix can be written in such a way that it has diagonal blocks that are Hessenberg blocks whose subdiagonal is not zero.

Definition 18.2. An upper Hessenberg $n \times n$ matrix H is *unreduced* if $h_{i+1i} \neq 0$ for $i = 1, \ldots, n-1$. A Hessenberg matrix which is not unreduced is said to be *reduced*.

The following is an example of an 8×8 matrix consisting of three diagonal unreduced Hessenberg blocks:

$$H = \begin{pmatrix} \star & \star & \star & * & * & * & * & * \\ \mathbf{h_{21}} & \star & \star & * & * & * & * & * \\ \mathbf{0} & \mathbf{h_{32}} & \star & * & * & * & * & * \\ 0 & 0 & 0 & \star & \star & \star & * & * \\ 0 & 0 & 0 & \mathbf{h_{54}} & \star & \star & * & * \\ 0 & 0 & 0 & \mathbf{0} & \mathbf{h_{65}} & \star & * & * \\ 0 & 0 & 0 & 0 & 0 & 0 & \star & \star \\ 0 & 0 & 0 & 0 & 0 & 0 & \mathbf{h_{87}} & \star \end{pmatrix}.$$

An interesting and important property of unreduced Hessenberg matrices is the following.

Proposition 18.3. *Let H be an $n \times n$ complex or real unreduced Hessenberg matrix. Then every eigenvalue of H is geometrically simple, that is,* $\dim(E_\lambda) = 1$ *for every eigenvalue λ, where E_λ is the eigenspace associated with λ. Furthermore, if H is diagonalizable, then every eigenvalue is simple, that is, H has n distinct eigenvalues.*

Proof. We follow Serre's proof [156] (Proposition 3.26). Let λ be any eigenvalue of H, let $M = \lambda I_n - H$, and let N be the $(n-1) \times (n-1)$ matrix obtained from M by deleting its first row and its last column. Since H is upper Hessenberg, N is a diagonal matrix with entries $-h_{i+1i} \neq 0$, $i = 1, \ldots, n-1$. Thus N is invertible and has rank $n-1$. But a matrix has rank greater than or equal to the rank of any of its submatrices, so $\mathrm{rank}(M) = n-1$, since M is singular. By the rank-nullity theorem, $\mathrm{rank}(\mathrm{Ker}\, N) = 1$, that is, $\dim(E_\lambda) = 1$, as claimed.

If H is diagonalizable, then the sum of the dimensions of the eigenspaces is equal to n, which implies that the eigenvalues of H are distinct. $\square$

As we said earlier, a case where Theorem 18.1 applies is the case where A is a symmetric (or Hermitian) positive definite matrix. This follows from two facts.

The first fact is that if A is Hermitian (or symmetric in the real case), then it is easy to show that the Hessenberg matrix similar to A is a Hermitian (or symmetric in real case) *tridiagonal matrix*. The conversion method is also more efficient. Here is an example of a symmetric tridiagonal matrix consisting of three unreduced blocks:

$$H = \begin{pmatrix} \alpha_{\mathbf{1}} & \beta_{\mathbf{1}} & \mathbf{0} & 0 & 0 & 0 & 0 & 0 \\ \beta_{\mathbf{1}} & \alpha_{\mathbf{2}} & \beta_{\mathbf{2}} & 0 & 0 & 0 & 0 & 0 \\ \mathbf{0} & \beta_{\mathbf{2}} & \alpha_{\mathbf{3}} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \alpha_{\mathbf{4}} & \beta_{\mathbf{4}} & \mathbf{0} & 0 & 0 \\ 0 & 0 & 0 & \beta_{\mathbf{4}} & \alpha_{\mathbf{5}} & \beta_{\mathbf{5}} & 0 & 0 \\ 0 & 0 & 0 & \mathbf{0} & \beta_{\mathbf{5}} & \alpha_{\mathbf{6}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \alpha_{\mathbf{7}} & \beta_{\mathbf{7}} \\ 0 & 0 & 0 & 0 & 0 & 0 & \beta_{\mathbf{7}} & \alpha_{\mathbf{8}} \end{pmatrix}.$$

Thus the problem of finding the eigenvalues of a symmetric (or Hermitian) matrix reduces to the problem of finding the eigenvalues of a symmetric (resp. Hermitian) tridiagonal matrix, and this can be done much more efficiently.

The second fact is that if H is an upper Hessenberg matrix and if it is diagonalizable, then there is an invertible matrix P such that $H = P\Lambda P^{-1}$ with Λ a diagonal matrix consisting of the eigenvalues of H, such that P^{-1} has an LU-decomposition; see Serre [156] (Theorem 13.3).

As a consequence, since any symmetric (or Hermitian) tridiagonal matrix is a block diagonal matrix of unreduced symmetric (resp. Hermitian) tridiagonal matrices, by Proposition 18.3, we see that the QR algorithm applied to a tridiagonal matrix which is symmetric (or Hermitian) positive definite converges to a diagonal matrix consisting of its eigenvalues. Let us record this important fact.

Theorem 18.4. *Let H be a symmetric (or Hermitian) positive definite tridiagonal matrix. If H is unreduced, then the QR algorithm converges to a diagonal matrix consisting of the eigenvalues of H.*

Since every symmetric (or Hermitian) positive definite matrix is similar to a tridiagonal symmetric (resp. Hermitian) positive definite matrix, we deduce that we have a method for finding the eigenvalues of a symmetric (resp. Hermitian) positive definite matrix (more accurately, to find approximations as good as we want for these eigenvalues).

If A is a symmetric (or Hermitian) matrix, since its eigenvalues are real, for some $\mu > 0$ large enough (pick $\mu > \rho(A)$), $A + \mu I$ is symmetric (resp. Hermitan) positive definite, so we can apply the QR algorithm to an upper Hessenberg matrix similar to $A + \mu I$ to find its eigenvalues, and then the eigenvalues of A are obtained by subtracting μ.

The problem of finding the eigenvalues of a symmetric matrix is discussed extensively in Parlett [135], one of the best references on this topic.

The upper Hessenberg form also yields a way to handle singular matrices. First, checking the proof of Proposition 14.21 that an $n \times n$ complex matrix A (possibly singular) can be factored as $A = QR$ where Q is a unitary matrix which is a product of Householder reflections and R is upper triangular, it is easy to see that if A is upper Hessenberg, then Q is also upper Hessenberg. If H is an unreduced upper Hessenberg matrix, since Q is upper Hessenberg and R is upper triangular, we have $h_{i+1i} = q_{i+1i}r_{ii}$ for $i = 1 \ldots, n-1$, and since H is unreduced, $r_{ii} \neq 0$ for $i = 1, \ldots, n-1$. Consequently H is singular iff $r_{nn} = 0$. Then the matrix RQ is a matrix whose last row consists of zero's thus we can deflate the problem by considering the $(n-1) \times (n-1)$ unreduced Hessenberg matrix obtained by deleting the last row and the last column. After finitely many steps (not larger that the multiplicity of the eigenvalue 0), there remains an invertible unreduced Hessenberg matrix. As an alternative, see Serre [156] (Chapter 13, Section 13.3.2).

As is, the QR algorithm, although very simple, is quite inefficient for several reasons. In the next section, we indicate how to make the method more efficient. This involves a lot of work and we only discuss the main ideas at a high level.

18.3 Making the QR Method More Efficient Using Shifts

To improve efficiency and cope with pairs of complex conjugate eigenvalues in the case of real matrices, the following steps are taken:

(1) Initially reduce the matrix A to upper Hessenberg form, as $A = UHU^*$. Then apply the QR-algorithm to H (actually, to its unreduced Hessenberg blocks). It is easy to see that the matrices H_k produced by the QR algorithm remain upper Hessenberg.

(2) To accelerate convergence, use *shifts*, and to deal with pairs of complex conjugate eigenvalues, use *double shifts*.

(3) Instead of computing a QR-factorization explicitly while doing a shift, perform an *implicit shift* which computes $A_{k+1} = Q_k^* A_k Q_k$ without having to compute a QR-factorization (of $A_k - \sigma_k I$), and similarly in the case of a double shift. This is the most intricate modification of the basic QR algorithm and we will not discuss it here. This method is usually referred as *bulge chasing*. Details about this technique for real matrices can be found in Demmel [48] (Section 4.4.8) and Golub and Van Loan [80] (Section 7.5). Watkins discusses the QR algorithm with shifts as a bulge chasing method in the more general case of complex matrices [187, 188].

Let us repeat an important remark made in the previous section. If we start with a matrix H in upper Hessenberg form, if at any stage of the QR algorithm we find that some subdiagonal entry $(H_k)_{p+1p} = 0$ *or is very small*, then H_k is of the form

$$H_k = \begin{pmatrix} H_{11} & H_{12} \\ 0 & H_{22} \end{pmatrix},$$

where both H_{11} and H_{22} are upper Hessenberg matrices (with H_{11} a $p \times p$ matrix and H_{22} a $(n-p) \times (n-p)$ matrix), and the eigenvalues of H_k are the eigenvalues of H_{11} and H_{22}. For example, in the matrix

$$H = \begin{pmatrix} * & * & * & * & * \\ h_{21} & * & * & * & * \\ 0 & h_{32} & * & * & * \\ 0 & 0 & h_{43} & * & * \\ 0 & 0 & 0 & h_{54} & * \end{pmatrix},$$

if $h_{43} = 0$, then we have the block matrix

$$H = \begin{pmatrix} * & * & * & * & * \\ h_{21} & * & * & * & * \\ 0 & h_{32} & * & * & * \\ 0 & 0 & 0 & * & * \\ 0 & 0 & 0 & h_{54} & * \end{pmatrix}.$$

Then we can recursively apply the QR algorithm to H_{11} and H_{22}.

In particular, if $(H_k)_{nn-1} = 0$ or is very small, then $(H_k)_{nn}$ is a good approximation of an eigenvalue, so we can delete the last row and the last column of H_k and apply the QR algorithm to this submatrix. This process is called *deflation*. If $(H_k)_{n-1n-2} = 0$ or is very small, then the 2×2 "corner block"

$$\begin{pmatrix} (H_k)_{n-1n-1} & (H_k)_{n-1n} \\ (H_k)_{nn-1} & (H_k)_{nn} \end{pmatrix}$$

appears, and its eigenvalues can be computed immediately by solving a quadratic equation. Then we deflate H_k by deleting its last two rows and its last two columns and apply the QR algorithm to this submatrix.

Thus it would seem desirable to modify the basic QR algorithm so that the above situations arises, and this is what shifts are designed for. More precisely, under the hypotheses of Theorem 18.1, it can be shown (see Ciarlet [41], Section 6.3) that the entry $(A_k)_{ij}$ with $i > j$ converges to 0 as $|\lambda_i/\lambda_j|^k$ converges to 0. Also, if we let r_i be defined by

$$r_1 = \left|\frac{\lambda_2}{\lambda_1}\right|, \quad r_i = \max\left\{\left|\frac{\lambda_i}{\lambda_{i-1}}\right|, \left|\frac{\lambda_{i+1}}{\lambda_i}\right|\right\}, \quad 2 \leq i \leq n-1, \quad r_n = \left|\frac{\lambda_n}{\lambda_{n-1}}\right|,$$

then there is a constant C (independent of k) such that

$$|(A_k)_{ii} - \lambda_i| \leq C r_i^k, \quad 1 \leq i \leq n.$$

In particular, if H is upper Hessenberg, then the entry $(H_k)_{i+1i}$ converges to 0 as $|\lambda_{i+1}/\lambda_i|^k$ converges to 0. Thus if we pick σ_k close to λ_i, we expect that $(H_k - \sigma_k I)_{i+1i}$ converges to 0 as $|(\lambda_{i+1} - \sigma_k)/(\lambda_i - \sigma_k)|^k$ converges to 0, and this ratio is much smaller than 1 as σ_k is closer to λ_i. Typically, we apply a shift to accelerate convergence to λ_n (so $i = n-1$). In this case, both $(H_k - \sigma_k I)_{nn-1}$ and $|(H_k - \sigma_k I)_{nn} - \lambda_n|$ converge to 0 as $|(\lambda_n - \sigma_k)/(\lambda_{n-1} - \sigma_k)|^k$ converges to 0.

A *shift* is the following modified QR-steps (switching back to an arbitrary matrix A, since the shift technique applies in general). Pick some σ_k, hopefully close to some eigenvalue of A (in general, λ_n), and QR-factor $A_k - \sigma_k I$ as

$$A_k - \sigma_k I = Q_k R_k,$$

and then form

$$A_{k+1} = R_k Q_k + \sigma_k I.$$

Since

$$\begin{aligned} A_{k+1} &= R_k Q_k + \sigma_k I \\ &= Q_k^* Q_k R_k Q_k + Q_k^* Q_k \sigma_k \\ &= Q_k^* (Q_k R_k + \sigma_k I) Q_k \\ &= Q_k^* A_k Q_k, \end{aligned}$$

A_{k+1} is similar to A_k, as before. If A_k is upper Hessenberg, then it is easy to see that A_{k+1} is also upper Hessenberg.

If A is upper Hessenberg and if σ_i is exactly equal to an eigenvalue, then $A_k - \sigma_k I$ is singular, and forming the QR-factorization will detect that R_k has some diagonal entry equal to 0. Assuming that the QR-algorithm returns $(R_k)_{nn} = 0$ (if not, the argument is easily adapted), then the last row of $R_k Q_k$ is 0, so the last row of $A_{k+1} = R_k Q_k + \sigma_k I$ ends with σ_k (all other entries being zero), so we are in the case where we can deflate A_k (and σ_k is indeed an eigenvalue).

The question remains, what is a good choice for the shift σ_k?

Assuming again that H is in upper Hessenberg form, it turns out that when $(H_k)_{nn-1}$ is small enough, then a good choice for σ_k is $(H_k)_{nn}$. In fact, the rate of convergence is quadratic, which means roughly that the number of correct digits doubles at every iteration. The reason is that shifts are related to another method known as inverse iteration, and such a method converges very fast. For further explanations about this connection, see Demmel [48] (Section 4.4.4) and Trefethen and Bau [176] (Lecture 29).

One should still be cautious that the QR method with shifts does not necessarily converge, and that our convergence proof no longer applies, because instead of having the identity $A^k = P_k \mathcal{R}_k$, we have

$$(A - \sigma_k I) \cdots (A - \sigma_2 I)(A - \sigma_1 I) = P_k \mathcal{R}_k.$$

Of course, the QR algorithm loops immediately when applied to an orthogonal matrix A. This is also the case when A is symmetric but not positive definite. For example, both the QR algorithm and the QR algorithm with shifts loop on the matrix

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}.$$

In the case of symmetric matrices, Wilkinson invented a shift which helps the QR algorithm with shifts to make progress. Again, looking at the lower corner of A_k, say

$$B = \begin{pmatrix} a_{n-1} & b_{n-1} \\ b_{n-1} & a_n \end{pmatrix},$$

the *Wilkinson shift* picks the eigenvalue of B closer to a_n. If we let

$$\delta = \frac{a_{n-1} - a_n}{2},$$

it is easy to see that the eigenvalues of B are given by

$$\lambda = \frac{a_n + a_{n-1}}{2} \pm \sqrt{\delta^2 + b_{n-1}^2}.$$

It follows that

$$\lambda - a_n = \delta \pm \sqrt{\delta^2 + b_{n-1}^2},$$

and from this it is easy to see that the eigenvalue closer to a_n is given by

$$\mu = a_n - \frac{\mathrm{sign}(\delta) b_{n-1}^2}{(|\delta| + \sqrt{\delta^2 + b_{n-1}^2})}.$$

If $\delta = 0$, then we pick arbitrarily one of the two eigenvalues. Observe that the Wilkinson shift applied to the matrix

$$A = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

is either $+1$ or -1, and in one step, deflation occurs and the algorithm terminates successfully.

We now discuss double shifts, which are intended to deal with pairs of complex conjugate eigenvalues.

Let us assume that A is a real matrix. For any complex number σ_k with nonzero imaginary part, a *double shift* consists of the following steps:

$$\begin{aligned} A_k - \sigma_k I &= Q_k R_k \\ A_{k+1} &= R_k Q_k + \sigma_k I \\ A_{k+1} - \overline{\sigma}_k I &= Q_{k+1} R_{k+1} \\ A_{k+2} &= R_{k+1} Q_{k+1} + \overline{\sigma}_k I. \end{aligned}$$

From the computation made for a single shift, we have $A_{k+1} = Q_k^* A_k Q_k$ and $A_{k+2} = Q_{k+1}^* A_{k+1} Q_{k+1}$, so we obtain

$$A_{k+2} = Q_{k+1}^* Q_k^* A_k Q_k Q_{k+1}.$$

The matrices Q_k are complex, so we would expect that the A_k are also complex, but remarkably we can keep the products $Q_k Q_{k+1}$ real, and so the A_k also real. This is highly desirable to avoid complex arithmetic, which is more expensive.

Observe that since

$$Q_{k+1} R_{k+1} = A_{k+1} - \overline{\sigma}_k I = R_k Q_k + (\sigma_k - \overline{\sigma}_k) I,$$

we have

$$\begin{aligned} Q_k Q_{k+1} R_{k+1} R_k &= Q_k (R_k Q_k + (\sigma_k - \overline{\sigma}_k) I) R_k \\ &= Q_k R_k Q_k R_k + (\sigma_k - \overline{\sigma}_k) Q_k R_k \\ &= (A_k - \sigma_k I)^2 + (\sigma_k - \overline{\sigma}_k)(A_k - \sigma_k I) \\ &= A_k^2 - 2(\Re \sigma_k) A_k + |\sigma_k|^2 I. \end{aligned}$$

If we assume by induction that matrix A_k is real (with $k = 2\ell+1, \ell \geq 0$), then the matrix $S = A_k^2 - 2(\Re\sigma_k)A_k + |\sigma_k|^2 I$ is also real, and since $Q_k Q_{k+1}$ is unitary and $R_{k+1}R_k$ is upper triangular, we see that

$$S = Q_k Q_{k+1} R_{k+1} R_k$$

is a QR-factorization of the real matrix S, thus $Q_k Q_{k+1}$ and $R_{k+1}R_k$ can be chosen to be real matrices, in which case $(Q_k Q_{k+1})^*$ is also real, and thus

$$A_{k+2} = Q_{k+1}^* Q_k^* A_k Q_k Q_{k+1} = (Q_k Q_{k+1})^* A_k Q_k Q_{k+1}$$

is real. Consequently, if $A_1 = A$ is real, then $A_{2\ell+1}$ is real for all $\ell \geq 0$.

The strategy that consists in picking σ_k and $\overline{\sigma}_k$ as the complex conjugate eigenvalues of the corner block

$$\begin{pmatrix} (H_k)_{n-1n-1} & (H_k)_{n-1n} \\ (H_k)_{nn-1} & (H_k)_{nn} \end{pmatrix}$$

is called the *Francis shift* (here we are assuming that A has be reduced to upper Hessenberg form).

It should be noted that there are matrices for which neither a shift by $(H_k)_{nn}$ nor the Francis shift works. For instance, the permutation matrix

$$A = \begin{pmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$$

has eigenvalues $e^{i2\pi/3}, e^{i4\pi/3}, +1$, and neither of the above shifts apply to the matrix

$$\begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}.$$

However, a shift by 1 does work. There are other kinds of matrices for which the QR algorithm does not converge. Demmel gives the example of matrices of the form

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & h & 0 \\ 0 & -h & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$

where h is small.

Algorithms implementing the QR algorithm with shifts and double shifts perform "exceptional" shifts every 10 shifts. Despite the fact that the QR algorithm has been perfected since the 1960's, it is still an open problem to find a shift strategy that ensures convergence of all matrices.

Implicit shifting is based on a result known as the *implicit Q theorem*. This theorem says that if A is reduced to upper Hessenberg form as $A = UHU^*$ and if H is unreduced

($h_{i+1i} \neq 0$ for $i = 1, \ldots, n-1$), then the columns of index $2, \ldots, n$ of U are determined by the first column of U up to sign; see Demmel [48] (Theorem 4.9) and Golub and Van Loan [80] (Theorem 7.4.2) for the proof in the case of real matrices. Actually, the proof is not difficult and will be the object of a homework exercise. In the case of a single shift, an implicit shift generates $A_{k+1} = Q_k^* A_k Q_k$ without having to compute a QR-factorization of $A_k - \sigma_k I$. For real matrices, this is done by applying a sequence of Givens rotations which perform a bulge chasing process (a Givens rotation is an orthogonal block diagonal matrix consisting of a single block which is a 2D rotation, the other diagonal entries being equal to 1). Similarly, in the case of a double shift, $A_{k+2} = (Q_k Q_{k+1})^* A_k Q_k Q_{k+1}$ is generated without having to compute the QR-factorizations of $A_k - \sigma_k I$ and $A_{k+1} - \overline{\sigma}_k I$. Again, $(Q_k Q_{k+1})^* A_k Q_k Q_{k+1}$ is generated by applying some simple orthogonal matrices which perform a bulge chasing process. See Demmel [48] (Section 4.4.8) and Golub and Van Loan [80] (Section 7.5) for further explanations regarding implicit shifting involving bulge chasing in the case of real matrices. Watkins [187, 188] discusses bulge chasing in the more general case of complex matrices.

The `Matlab` function for finding the eigenvalues and the eigenvectors of a matrix A is `eig` and is called as `[U, D] = eig(A)`. It is implemented using an optimized version of the QR-algorithm with implicit shifts.

If the dimension of the matrix A is very large, we can find approximations of some of the eigenvalues of A by using a truncated version of the reduction to Hessenberg form due to Arnoldi in general and to Lanczos in the symmetric (or Hermitian) tridiagonal case.

18.4 Krylov Subspaces; Arnoldi Iteration

In this section, we denote the dimension of the square real or complex matrix A by m rather than n, to make it easier for the reader to follow Trefethen and Bau exposition [176], which is particularly lucid.

Suppose that the $m \times m$ matrix A has been reduced to the upper Hessenberg form H, as $A = UHU^*$. For any $n \leq m$ (typically much smaller than m), consider the $(n+1) \times n$ upper left block

$$\widetilde{H}_n = \begin{pmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \cdots & h_{2n} \\ 0 & h_{32} & h_{33} & \cdots & h_{3n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{nn-1} & h_{nn} \\ 0 & \cdots & 0 & 0 & h_{n+1n} \end{pmatrix}$$

of H, and the $n \times n$ upper Hessenberg matrix H_n obtained by deleting the last row of $\widetilde{H}_n$,

$$H_n = \begin{pmatrix} h_{11} & h_{12} & h_{13} & \cdots & h_{1n} \\ h_{21} & h_{22} & h_{23} & \cdots & h_{2n} \\ 0 & h_{32} & h_{33} & \cdots & h_{3n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & h_{nn-1} & h_{nn} \end{pmatrix}.$$

If we denote by U_n the $m \times n$ matrix consisting of the first n columns of U, denoted $u_1, \ldots, u_n$, then the matrix consisting of the first n columns of the matrix $UH = AU$ can be expressed as

$$AU_n = U_{n+1}\widetilde{H}_n. \tag{$*_1$}$$

It follows that the nth column of this matrix can be expressed as

$$Au_n = h_{1n}u_1 + \cdots + h_{nn}u_n + h_{n+1n}u_{n+1}. \tag{$*_2$}$$

Since $(u_1, \ldots, u_n)$ form an orthonormal basis, we deduce from $(*_2)$ that

$$\langle u_j, Au_n \rangle = u_j^* A u_n = h_{jn}, \quad j = 1, \ldots, n. \tag{$*_3$}$$

Equations $(*_2)$ and $(*_3)$ show that U_{n+1} and $\widetilde{H}_n$ can be computed iteratively using the following algorithm due to Arnoldi, known as *Arnoldi iteration*:

```
Given an arbitrary nonzero vector $b \in \mathbb{C}^m$, let $u_1 = b/\|b\|$;
for $n = 1, 2, 3, \ldots$ do
  $z := Au_n$;
  for $j = 1$ to $n$ do
    $h_{jn} := u_j^* z$;
    $z := z - h_{jn}u_j$
  endfor
  $h_{n+1n} := \|z\|$;
  if $h_{n+1n} = 0$ quit
  $u_{n+1} = z/h_{n+1n}$
```

When $h_{n+1n} = 0$, we say that we have a *breakdown* of the Arnoldi iteration.

Arnoldi iteration is an algorithm for producing the $n \times n$ Hessenberg submatrix H_n of the full Hessenberg matrix H consisting of its first n rows and n columns (the first n columns of U are also produced), not using Householder matrices.

As long as $h_{j+1j} \neq 0$ for $j = 1, \ldots, n$, Equation $(*_2)$ shows by an easy induction that u_{n+1} belong to the span of $(b, Ab, \ldots, A^n b)$, and obviously Au_n belongs to the span of $(u_1, \ldots, u_{n+1})$, and thus the following spaces are identical:

$$\text{Span}(b, Ab, \ldots, A^n b) = \text{Span}(u_1, \ldots, u_{n+1}).$$

The space $\mathcal{K}_n(A, b) = \mathrm{Span}(b, Ab, \ldots, A^{n-1}b)$ is called a *Krylov subspace*. We can view Arnoldi's algorithm as the construction of an orthonormal basis for $\mathcal{K}_n(A, b)$. It is a sort of Gram–Schmidt procedure.

Equation $(*_2)$ shows that if K_n is the $m \times n$ matrix whose columns are the vectors $(b, Ab, \ldots, A^{n-1}b)$, then there is a $n \times n$ upper triangular matrix R_n such that

$$K_n = U_n R_n. \tag{$*_4$}$$

The above is called a *reduced QR factorization* of K_n.

Since $(u_1, \ldots, u_n)$ is an orthonormal system, the matrix $U_n^* U_{n+1}$ is the $n \times (n+1)$ matrix consisting of the identity matrix I_n plus an extra column of 0's, so $U_n^* U_{n+1} \widetilde{H}_n = U_n^* A U_n$ is obtained by deleting the last row of $\widetilde{H}_n$, namely H_n, and so

$$U_n^* A U_n = H_n. \tag{$*_5$}$$

We summarize the above facts in the following proposition.

Proposition 18.5. *If Arnoldi iteration run on an $m \times m$ matrix A starting with a nonzero vector $b \in \mathbb{C}^m$ does not have a breakdown at stage $n \leq m$, then the following properties hold:*

(1) If K_n is the $m \times n$ Krylov matrix associated with the vectors $(b, Ab, \ldots, A^{n-1}b)$ and if U_n is the $m \times n$ matrix of orthogonal vectors produced by Arnoldi iteration, then there is a QR-factorization

$$K_n = U_n R_n,$$

for some $n \times n$ upper triangular matrix R_n.

(2) The $m \times n$ upper Hessenberg matrices H_n produced by Arnoldi iteration are the projection of A onto the Krylov space $\mathcal{K}_n(A, b)$, that is,

$$H_n = U_n^* A U_n.$$

(3) The successive iterates are related by the formula

$$A U_n = U_{n+1} \widetilde{H}_n.$$

Remark: If Arnoldi iteration has a breakdown at stage n, that is, $h_{n+1} = 0$, then we found the first unreduced block of the Hessenberg matrix H. It can be shown that the eigenvalues of H_n are eigenvalues of A. So a breakdown is actually a good thing. In this case, we can pick some new nonzero vector u_{n+1} orthogonal to the vectors $(u_1, \ldots, u_n)$ as a new starting vector and run Arnoldi iteration again. Such a vector exists since the $(n+1)$th column of U works. So repeated application of Arnoldi yields a full Hessenberg reduction of A. However, this is not what we are after, since m is very large an we are only interested in a "small" number of eigenvalues of A.

There is another aspect of Arnoldi iteration, which is that it solves an optimization problem involving polynomials of degree n. Let $\mathcal{P}^n$ denote the set of (complex) monic polynomials of degree n, that is, polynomials of the form

$$p(z) = z^n + c_{n-1}z^{n-1} + \cdots + c_1 z + c_0 \quad (c_i \in \mathbb{C}).$$

For any $m \times m$ matrix A, we write

$$p(A) = A^n + c_{n-1}A^{n-1} + \cdots + c_1 A + c_0 I.$$

The following result is proven in Trefethen and Bau [176] (Lecture 34, Theorem 34.1).

Theorem 18.6. *If Arnoldi iteration run on an $m \times m$ matrix A starting with a nonzero vector b does not have a breakdown at stage $n \leq m$, then there is a unique polynomial $p \in \mathcal{P}^n$ such that $\|p(A)b\|_2$ is minimum, namely the characteristic polynomial* $\det(zI - H_n)$ *of H_n.*

Theorem 18.6 can be viewed as the "justification" for a method to find some of the eigenvalues of A (say $n \ll m$ of them). Intuitively, the closer the roots of the characteristic polynomials of H_n are to the eigenvalues of A, the smaller $\|p(A)b\|_2$ should be, and conversely. In the extreme case where $m = n$, by the Cayley–Hamilton theorem, $p(A) = 0$ (where p is the characteristic polynomial of A), so this idea is plausible, but this is far from constituting a proof (also, b should have nonzero coordinates in all directions associated with the eigenvalues).

The method known as the *Rayleigh–Ritz method* is to run Arnoldi iteration on A and some $b \neq 0$ chosen at random for $n \ll m$ steps before or until a breakdown occurs. Then run the QR algorithm with shifts on H_n. The eigenvalues of the Hessenberg matrix H_n may then be considered as approximations of the eigenvalues of A. The eigenvalues of H_n are called *Arnoldi estimates* or *Ritz values*. One has to be cautious because H_n is a truncated version of the full Hessenberg matrix H, so not all of the Ritz values are necessarily close to eigenvalues of A. It has been observed that the eigenvalues that are found first are the *extreme* eigenvalues of A, namely those close to the boundary of the spectrum of A plotted in $\mathbb{C}$. So if A has real eigenvalues, the largest and the smallest eigenvalues appear first as Ritz values. In many problems where eigenvalues occur, the extreme eigenvalues are the one that need to be computed. Similarly, the eigenvectors of H_n may be considered as approximations of eigenvectors of A.

The `Matlab` function `eigs` is based on the computation of Ritz values. It computes the six eigenvalues of largest magnitude of a matrix A, and the call is `[V, D] = eigs(A)`. More generally, to get the top k eigenvalues, use `[V, D] = eigs(A, k)`.

In the absence of rigorous theorems about error estimates, it is hard to make the above statements more precise; see Trefethen and Bau [176] (Lecture 34) for more on this subject.

However, if A is a symmetric (or Hermitian) matrix, then H_n is a symmetric (resp. Hermitian) tridiagonal matrix and more precise results can be shown; see Demmel [48] (Chapter 7, especially Section 7.2). We will consider the symmetric (and Hermitan) case in the next section, but first we show how Arnoldi iteration can be used to find approximations for the solution of a linear system $Ax = b$ where A is invertible but of very large dimension m.

18.5 GMRES

Suppose A is an invertible $m \times m$ matrix and let b be a nonzero vector in $\mathbb{C}^m$. Let $x_0 = A^{-1}b$, the unique solution of $Ax = b$. It is not hard to show that $x_0 \in \mathcal{K}_n(A, b)$ for some $n \leq m$. In fact, there is a unique monic polynomial $p(z)$ of minimal degree $s \leq m$ such that $p(A)b = 0$, so $x_0 \in \mathcal{K}_s(A, b)$. Thus it makes sense to search for a solution of $Ax = b$ in Krylov spaces of dimension $m \leq s$. The idea is to find an approximation $x_n \in \mathcal{K}_n(A, b)$ of x_0 such that $r_n = b - Ax_n$ is minimized, that is, $\|r_n\|_2 = \|b - Ax_n\|_2$ is minimized over $x_n \in \mathcal{K}_n(A, b)$. This minimization problem can be stated as

$$\text{minimize} \quad \|r_n\|_2 = \|Ax_n - b\|_2\,, \quad x_n \in \mathcal{K}_n(A, b).$$

This is a least-squares problem, and we know how to solve it (see Section 23.1). The quantity r_n is known as the *residual* and the method which consists in minimizing $\|r_n\|_2$ is known as GMRES, for *generalized minimal residuals*.

Now since $(u_1, \ldots, u_n)$ is a basis of $\mathcal{K}_n(A, b)$ (since $n \leq s$, no breakdown occurs, except for $n = s$), we may write $x_n = U_n y$, so our minimization problem is

$$\text{minimize} \quad \|AU_n y - b\|_2\,, \quad y \in \mathbb{C}^n.$$

Since by $(*_1)$ of Section 18.4, we have $AU_n = U_{n+1}\widetilde{H}_n$, minimizing $\|AU_n y - b\|_2$ is equivalent to minimizing $\|U_{n+1}\widetilde{H}_n y - b\|_2$ over $\mathbb{C}^m$. Since $U_{n+1}\widetilde{H}_n y$ and b belong to the column space of U_{n+1}, minimizing $\|U_{n+1}\widetilde{H}_n y - b\|_2$ is equivalent to minimizing $\|\widetilde{H}_n y - U_{n+1}^* b\|_2$. However, by construction,

$$U_{n+1}^* b = \|b\|_2 e_1 \in \mathbb{C}^{n+1},$$

so our minimization problem can be stated as

$$\text{minimize} \quad \|\widetilde{H}_n y - \|b\|_2 e_1\|_2, \quad y \in \mathbb{C}^n.$$

The approximate solution of $Ax = b$ is then

$$x_n = U_n y.$$

Starting with $u_1 = b/\|b\|_2$ and with $n = 1$, the GMRES method runs $n \leq s$ Arnoldi iterations to find U_n and $\widetilde{H}_n$, and then runs a method to solve the least squares problem

$$\text{minimize} \quad \|\widetilde{H}_n y - \|b\|_2 e_1\|_2, \quad y \in \mathbb{C}^n.$$

When $\|r_n\|_2 = \|\widetilde{H}_n y - \|b\|_2 e_1\|_2$ is considered small enough, we stop and the approximate solution of $Ax = b$ is then

$$x_n = U_n y.$$

There are ways of improving efficiency of the "naive" version of GMRES that we just presented; see Trefethen and Bau [176] (Lecture 35). We now consider the case where A is a Hermitian (or symmetric) matrix.

18.6 The Hermitian Case; Lanczos Iteration

If A is an $m \times m$ symmetric or Hermitian matrix, then Arnoldi's method is simpler and much more efficient. Indeed, in this case, it is easy to see that the upper Hessenberg matrices H_n are also symmetric (Hermitian respectively), and thus tridiagonal. Also, the eigenvalues of A and H_n are real. It is convenient to write

$$H_n = \begin{pmatrix} \alpha_1 & \beta_1 & & & \\ \beta_1 & \alpha_2 & \beta_2 & & \\ & \beta_2 & \alpha_3 & \ddots & \\ & & \ddots & \ddots & \beta_{n-1} \\ & & & \beta_{n-1} & \alpha_n \end{pmatrix}.$$

The recurrence $(*_2)$ of Section 18.4 becomes the three-term recurrence

$$Au_n = \beta_{n-1} u_{n-1} + \alpha_n u_n + \beta_n u_{n+1}. \qquad (*_6)$$

We also have $\alpha_n = u_n^* A u_n$, so Arnoldi's algorithm becomes the following algorithm known as *Lanczos' algorithm* (or *Lanczos iteration*). The inner loop on j from 1 to n has been eliminated and replaced by a single assignment.

```
Given an arbitrary nonzero vector $b \in \mathbb{C}^m$, let $u_1 = b/\|b\|$;
for $n = 1, 2, 3, \ldots$ do
  $z := Au_n$;
    $\alpha_n := u_n^* z$;
    $z := z - \beta_{n-1} u_{n-1} - \alpha_n u_n$
  $\beta_n := \|z\|$;
  if $\beta_n = 0$ quit
  $u_{n+1} = z/\beta_n$
```

When $\beta_n = 0$, we say that we have a *breakdown* of the Lanczos iteration.

Versions of Proposition 18.5 and Theorem 18.6 apply to Lanczos iteration.

Besides being much more efficient than Arnoldi iteration, Lanczos iteration has the advantage that the *Rayleigh–Ritz method* for finding some of the eigenvalues of A as the eigenvalues of the symmetric (respectively Hermitian) tridiagonal matrix H_n applies, but there are more methods for finding the eigenvalues of symmetric (respectively Hermitian) tridiagonal matrices. Also theorems about error estimates exist. The version of Lanczos iteration given above may run into problems in floating point arithmetic. What happens is that the vectors u_j may lose the property of being orthogonal, so it may be necessary to reorthogonalize them. For more on all this, see Demmel [48] (Chapter 7, in particular Section 7.2-7.4). The version of GMRES using Lanczos iteration is called MINRES.

We close our brief survey of methods for computing the eigenvalues and the eigenvectors of a matrix with a quick discussion of two methods known as power methods.

18.7 Power Methods

Let A be an $m \times m$ complex or real matrix. There are two power methods, both of which yield one eigenvalue and one eigenvector associated with this vector:

(1) *Power iteration.*

(2) *Inverse (power) iteration.*

Power iteration only works if the matrix A has an eigenvalue λ of largest modulus, which means that if $\lambda_1, \ldots, \lambda_m$ are the eigenvalues of A, then

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_m| \geq 0.$$

In particular, if A is a real matrix, then λ_1 must be real (since otherwise there are two complex conjugate eigenvalues of the same largest modulus). If the above condition is satisfied, then power iteration yields λ_1 and some eigenvector associated with it. The method is simple enough:

Pick some initial unit vector x^0 and compute the following sequence (x^k), where

$$x^{k+1} = \frac{Ax^k}{\|Ax^k\|}, \quad k \geq 0.$$

We would expect that (x^k) converges to an eigenvector associated with λ_1, but this is not quite correct. The following results are proven in Serre [156] (Section 13.5.1). First assume that $\lambda_1 \neq 0$.

We have

$$\lim_{k \mapsto \infty} \left\|Ax^k\right\| = |\lambda_1|.$$

If A is a complex matrix which has a unique complex eigenvalue λ_1 of largest modulus, then

$$v = \lim_{k \mapsto \infty} \left(\frac{\overline{\lambda_1}}{|\lambda_1|}\right)^k x^k$$

is a unit eigenvector of A associated with λ_1. If λ_1 is real, then

$$v = \lim_{k \mapsto \infty} x^k$$

is a unit eigenvector of A associated with λ_1. Actually some condition on x^0 is needed: x^0 must have a nonzero component in the eigenspace E associated with λ_1 (in any direct sum of $\mathbb{C}^m$ in which E is a summand).

The eigenvalue λ_1 is found as follows. If λ_1 is complex, and if $v_j \neq 0$ is any nonzero coordinate of v, then

$$\lambda_1 = \lim_{k \mapsto \infty} \frac{(Ax^k)_j}{x^k_j}.$$

If λ_1 is real, then we can define the sequence $(\lambda^{(k)})$ by

$$\lambda^{(k+1)} = (x^{k+1})^* A x^{k+1}, \quad k \geq 0,$$

and we have

$$\lambda_1 = \lim_{k \mapsto \infty} \lambda^{(k)}.$$

Indeed, in this case, since $v = \lim_{k \mapsto \infty} x^k$ and v is a unit eigenvector for λ_1, we have

$$\lim_{k \mapsto \infty} \lambda^{(k)} = \lim_{k \mapsto \infty} (x^{k+1})^* A x^{k+1} = v^* A v = \lambda_1 v^* v = \lambda_1.$$

Note that since x^{k+1} is a unit vector, $(x^{k+1})^* A x^{k+1}$ is a Rayleigh ratio.

If A is a Hermitian matrix, then the eigenvalues are real and we can say more about the rate of convergence, which is not great (only linear). For details, see Trefethen and Bau [176] (Lecture 27).

If $\lambda_1 = 0$, then there is some power $\ell < m$ such that $Ax^\ell = 0$.

The *inverse iteration method* is designed to find an eigenvector associated with an eigenvalue λ of A for which we know a good approximation μ.

Pick some initial unit vector x^0 and compute the following sequences (w^k) and (x^k), where w^{k+1} is the solution of the system

$$(A - \mu I) w^{k+1} = x^k \quad \text{equivalently} \quad w^{k+1} = (A - \mu I)^{-1} x^k, \quad k \geq 0,$$

and

$$x^{k+1} = \frac{w^{k+1}}{\|w^{k+1}\|}, \quad k \geq 0.$$

The following result is proven in Ciarlet [41] (Theorem 6.4.1).

Proposition 18.7. *Let A be an $m \times m$ diagonalizable (complex or real) matrix with eigenvalues $\lambda_1, \ldots, \lambda_m$, and let $\lambda = \lambda_\ell$ be an arbitrary eigenvalue of A (not necessary simple). For any μ such that*

$$\mu \neq \lambda \quad \text{and} \quad |\mu - \lambda| < |\mu - \lambda_j| \quad \text{for all } j \neq \ell,$$

if x^0 does not belong to the subspace spanned by the eigenvectors associated with the eigenvalues λ_j with $j \neq \ell$, then

$$\lim_{k \mapsto \infty} \left(\frac{(\lambda - \mu)^k}{|\lambda - \mu|^k} \right) x^k = v,$$

where v is an eigenvector associated with λ. Furthermore, if both λ and μ are real, we have

$$\begin{aligned} \lim_{k \mapsto \infty} x^k &= v & \quad & \text{if } \mu < \lambda, \\ \lim_{k \mapsto \infty} (-1)^k x^k &= v & \quad & \text{if } \mu > \lambda. \end{aligned}$$

Also, if we define the sequence $(\lambda^{(k)})$ by

$$\lambda^{(k+1)} = (x^{k+1})^* A x^{k+1},$$

then

$$\lim_{k \mapsto \infty} \lambda^{(k+1)} = \lambda.$$

The condition of x^0 may seem quite stringent, but in practice, a vector x^0 chosen at random usually satisfies it.

If A is a Hermitian matrix, then we can say more. In particular, the inverse iteration algorithm can be modified to make use of the newly computed $\lambda^{(k+1)}$ instead of μ, and an even faster convergence is achieved. Such a method is called the *Rayleigh quotient iteration*. When it converges (which is for almost all x^0), this method eventually achieves cubic convergence, which is remarkable. Essentially, this means that the number of correct digits is tripled at every iteration. For more details, see Trefethen and Bau [176] (Lecture 27) and Demmel [48] (Section 5.3.2).

18.8 Summary

The main concepts and results of this chapter are listed below:

- QR iteration, QR algorithm.
- Upper Hessenberg matrices.
- Householder matrix.

- Unreduced and reduced Hessenberg matrices.
- Deflation.
- Shift.
- Wilkinson shift.
- Double shift.
- Francis shift.
- Implicit shifting.
- Implicit Q-theorem.
- Arnoldi iteration.
- Breakdown of Arnoldi iteration.
- Krylov subspace.
- Rayleigh–Ritz method.
- Ritz values, Arnoldi estimates.
- Residual.
- GMRES
- Lanczos iteration.
- Power iteration.
- Inverse power iteration.
- Rayleigh ratio.

18.9 Problems

Problem 18.1. Prove Theorem 18.2; see Problem 13.7.

Problem 18.2. Prove that if a matrix A is Hermitian (or real symmetric), then any Hessenberg matrix H similar to A is Hermitian tridiagonal (real symmetric tridiagonal).

Problem 18.3. For any matrix (real or complex) A, if $A = QR$ is a QR-decomposition of A using Householder reflections, prove that if A is upper Hessenberg then so is Q.