Numerical Analysis Hilary Term 2018
Lectures 8–9: The Symmetric QR Algorithm

We consider only the case where A is symmetric.

Recall: a symmetric matrix A is similar to B if there is a nonsingular matrix P for which $A = P^{-1}BP$. Similar matrices have the same eigenvalues, since if $A = P^{-1}BP$,

$$0 = \det(A - \lambda I) = \det(P^{-1}(B - \lambda I)P) = \det(P^{-1})\det(P)\det(B - \lambda I),$$

so $\det(A - \lambda I) = 0$ if, and only if, $\det(B - \lambda I) = 0$.

The basic **QR algorithm** is:

```
Set A_1 = A.
for k = 1, 2, ...
  form the QR factorization A_k = Q_k R_k
  and set A_{k+1} = R_k Q_k
end
```

Proposition. The symmetric matrices $A_1, A_2, \ldots, A_k, \ldots$ are all similar and thus have the same eigenvalues.

Proof. Since

$$A_{k+1} = R_k Q_k = (Q_k^{\mathrm{T}} Q_k) R_k Q_k = Q_k^{\mathrm{T}} (Q_k R_k) Q_k = Q_k^{\mathrm{T}} A_k Q_k = Q_k^{-1} A_k Q_k,$$

A_{k+1} is symmetric if A_k is, and is similar to A_k. □

At least when A has distinct eigenvalues, this basic QR algorithm can be shown to work (A_k converges to a diagonal matrix as $k \to \infty$, the diagonal entries of which are the eigenvalues). However, a really practical, fast algorithm is based on some refinements.

Reduction to tridiagonal form: the idea is to apply explicit similarity transformations $QAQ^{-1} = QAQ^{\mathrm{T}}$, with Q orthogonal, so that QAQ^{T} is tridiagonal.
Note: direct reduction to triangular form would reveal the eigenvalues, but is not possible. If

$$H(w)A = \begin{bmatrix} \times & \times & \cdots & \times \\ 0 & \times & \cdots & \times \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \times & \cdots & \times \end{bmatrix}$$

then $H(w)AH(w)^{\mathrm{T}}$ is generally full, i.e., all zeros created by pre-multiplication are destroyed by the post-multiplication. However, if

$$A = \begin{bmatrix} \gamma & u^{\mathrm{T}} \\ u & C \end{bmatrix}$$

(as $A = A^{\mathrm{T}}$) and

$$w = \begin{bmatrix} 0 \\ \hat{w} \end{bmatrix} \quad \text{where} \quad H(\hat{w})u = \begin{bmatrix} \alpha \\ 0 \\ \vdots \\ 0 \end{bmatrix},$$

it follows that

$$H(w)A = \begin{bmatrix} \gamma & & u^{\mathrm{T}} & \\ \alpha & \times & \vdots & \times \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \times & \vdots & \times \end{bmatrix},$$

i.e., the u^{T} part of the first row of A is unchanged. However, then

$$H(w)AH(w)^{-1} = H(w)AH(w)^{\mathrm{T}} = H(w)AH(w) = \left[\begin{array}{c|c} \gamma & \alpha \ 0 \ \cdots \ 0 \\ \hline \alpha & \\ 0 & \\ \vdots & B \\ 0 & \end{array}\right],$$

where $B = H(\hat{w})CH^{\mathrm{T}}(\hat{w})$, as $u^{\mathrm{T}}H(\hat{w})^{\mathrm{T}} = (\alpha, \quad 0, \quad \cdots, \quad 0)$; note that $H(w)AH(w)^{\mathrm{T}}$ is symmetric as A is.
Now we inductively apply this to the smaller matrix B, as described for the QR factorization but using post- as well as pre-multiplications. The result of $n-2$ such Householder similarity transformations is the matrix

$$H(w_{n-2}) \cdots H(w_2)H(w)AH(w)H(w_2) \cdots H(w_{n-2}),$$

which is tridiagonal.
The QR factorization of a tridiagonal matrix can now easily be achieved with $n-1$ Givens rotations: if A is tridiagonal

$$\underbrace{J(n-1,n) \cdots J(2,3)J(1,2)}_{Q^{\mathrm{T}}}A = R, \quad \text{upper triangular.}$$

Precisely, R has a diagonal and 2 super-diagonals,

$$R = \begin{bmatrix} \times & \times & \times & 0 & 0 & 0 & \cdots & 0 \\ 0 & \times & \times & \times & 0 & 0 & \cdots & 0 \\ 0 & 0 & \times & \times & \times & 0 & \cdots & 0 \\ \vdots & \vdots & & & & & & \vdots \\ 0 & 0 & 0 & 0 & \times & \times & \times & 0 \\ 0 & 0 & 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

(exercise: check!). In the QR algorithm, the next matrix in the sequence is RQ.

Lemma. In the QR algorithm applied to a symmetric tridiagonal matrix, the symmetry and tridiagonal form are preserved when Givens rotations are used.

Proof. We have already shown that if $A_k = QR$ is symmetric, then so is $A_{k+1} = RQ$. If $A_k = QR = J(1,2)^{\mathrm{T}}J(2,3)^{\mathrm{T}} \cdots J(n-1,n)^{\mathrm{T}}R$ is tridiagonal, then $A_{k+1} = RQ =$

$RJ(1,2)^{\mathrm{T}}J(2,3)^{\mathrm{T}}\cdots J(n{-}1,n)^{\mathrm{T}}$. Recall that post-multiplication of a matrix by $J(i,i{+}1)^{\mathrm{T}}$ replaces columns i and $i+1$ by linear combinations of the pair of columns, while leaving columns $j = 1, 2, \ldots, i-1, i+2, \ldots, n$ alone. Thus, since R is upper triangular, the only subdiagonal entry in $RJ(1,2)^{\mathrm{T}}$ is in position $(2,1)$. Similarly, the only subdiagonal entries in $RJ(1,2)^{\mathrm{T}}J(2,3)^{\mathrm{T}} = (RJ(1,2)^{\mathrm{T}})J(2,3)^{\mathrm{T}}$ are in positions $(2,1)$ and $(3,2)$. Inductively, the only subdiagonal entries in

$$\begin{aligned} &RJ(1,2)^{\mathrm{T}}J(2,3)^{\mathrm{T}}\cdots J(i-2,i-1)^{\mathrm{T}}J(i-1,i)^{\mathrm{T}} \\ &\quad = (RJ(1,2)^{\mathrm{T}}J(2,3)^{\mathrm{T}}\cdots J(i-2,i-1)^{\mathrm{T}})J(i-1,i)^{\mathrm{T}} \end{aligned}$$

are in positions $(j, j-1)$, $j = 2, \ldots i$. So, the lower triangular part of A_{k+1} only has nonzeros on its first subdiagonal. However, then since A_{k+1} is symmetric, it must be tridiagonal. □

Using shifts. One further and final step in making an efficient algorithm is the use of **shifts**:

```
for k = 1, 2, ...
  form the QR factorization of A_k − μ_k I = Q_k R_k
  and set A_{k+1} = R_k Q_k + μ_k I
end
```

For any chosen sequence of values of $\mu_k \in \mathbb{R}$, $\{A_k\}_{k=1}^{\infty}$ are symmetric and tridiagonal if A_1 has these properties, and similar to A_1.

The simplest shift to use is $a_{n,n}$, which leads rapidly in almost all cases to

$$A_k = \left[\begin{array}{c|c} T_k & 0 \\ \hline 0^{\mathrm{T}} & \lambda \end{array}\right],$$

where T_k is $n-1$ by $n-1$ and tridiagonal, and λ is an eigenvalue of A_1. Inductively, once this form has been found, the QR algorithm with shift $a_{n-1,n-1}$ can be concentrated only on the $n-1$ by $n-1$ leading submatrix T_k. This process is called **deflation**.

The overall algorithm for calculating the eigenvalues of an n by n symmetric matrix:

```
reduce A to tridiagonal form by orthogonal
(Householder) similarity transformations.

for m = n, n − 1, ... 2
  while a_{m−1,m} > tol
    [Q, R] = qr(A − a_{m,m} * I)
    A = R * Q + a_{m,m} * I
  end while

  record eigenvalue λ_m = a_{m,m}
  A ← leading m − 1 by m − 1 submatrix of A
end
record eigenvalue λ_1 = a_{1,1}
```