

# Lecture 17

## The QR Method\*

The Power Method and Inverse Power Method each give us only one  $\mathbf{ew}$ - $\mathbf{ev}$  pair. While both of these methods can be modified to give more  $\mathbf{ew}$ 's and  $\mathbf{ev}$ 's, there is a better method for obtaining all the  $\mathbf{ew}$ 's called the *QR method*. This is the basis of all modern  $\mathbf{ew}$  software, including MATLAB, so we summarize it briefly here.

The QR method uses the fact that any square matrix has a *QR decomposition*. That is, for any  $A$  there are matrices  $Q$  and  $R$  such that  $A = QR$  where  $Q$  has the property

$$Q^{-1} = Q'$$

and  $R$  is upper triangular. A matrix  $Q$  with the property that its transpose equals its inverse is called an *orthogonal* matrix, because its column vectors are mutually orthogonal.

The QR method consists of iterating following steps:

- Transform  $A$  into a tridiagonal matrix  $H$ .
- decompose  $H$  in  $QR$ .
- multiply  $Q$  and  $R$  together in reverse order to form a new  $H$ .

The diagonal of  $H$  will converge to the eigenvalues.

We details of what makes this method converge are beyond the scope of the this book. However, we note the following theory behind it for those with more familiarity with linear algebra. First the Hessian matrix  $H$  is obtained from  $A$  by a series of similarity transformation, thus it has the same  $\mathbf{ew}$ 's as  $A$ . Secondly, if we denote by  $H_0, H_1, H_2, \dots$ , the sequence of matrices produced by the iteration, then

$$H_{i+1} = R_i Q_i = Q_i^{-1} Q_i R_i Q_i = Q_i' H_i Q_i.$$

Thus each  $H_{i+1}$  is related to  $H_i$  by an (orthogonal) similarity transformation and so they have the same  $\mathbf{ew}$ 's as  $A$ .

There is a built-in QR decomposition in MATLAB which is called with the command: `[Q R] = qr(A)`. Thus the following program implements QR method until it converges:

```

function E = myqrmetho(A)
[m n] = size(A);
if m ~= n
    warning('The input matrix is not square.')
    return
end
H = hess(A);
E = diag(H);
change = 1;
steps = 0;
while change > 0
    Eold = E;
    [Q R] = qr(H);
    H = R*Q;
    E = diag(H);
    change = norm(E - Eold);
    steps = steps + 1;
end
steps

```

As you can see the main steps of the program are very simple. The really hard calculations are contained in the built-in command `qr(A)`.

Run this program and compare the results with MATLAB's built in command:

```

> format long
> format compact
> A = hilb(5)
> Eqr = myqrmetho(A)
> Eml = eig(A)

```

## Exercises

- 17.1 Modify `myqrmetho` to stop after a maximum number of iterations. Use the modified program, with the maximum iterations set to 1000, on the matrix  $A = \text{hilb}(n)$  with  $n$  equal to 10, 50, and 200. Use the norm to compare the results to the `ew`'s obtained from MATLAB's built-in program `eig`. Turn in a printout of your program and a brief report on the experiment.