Lecture 17

The QR Method*

The Power Method and Inverse Power Method each give us only one eigenvalue-eigenvector pair. While both of these methods can be modified to give more eigenvalues and eigenvectors, there is a better method for obtaining all the eigenvalues called the *QR method*. This is the basis of all modern eigenvalue 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 the $A = QR$ 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.

The 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 eigenvalues as A. Secondly, if we denote by H_0, H_1, H_2, ..., 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 a related to H_i by an (orthogonal) similarity transformation and so they have the same eigenvalues 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,steps] = myqrmethod(A)
    % Computes all the eigenvalues of a matrix using the QR method.
    % Input: A -- square matrix
    % Outputs: E -- vector of eigenvalues
    %          steps -- the number of iterations it took
    [m n] = size(A);
    if m ~= n
       warning('The input matrix is not square.')
       return
    end
    % Set up initial estimate
    H = hess(A);
    E = diag(H);
    change = 1;
    steps = 0;
    % loop while estimate changes
    while change > 0
        Eold = E;
        % apply QR method
        [Q R] = qr(H);
        H = R*Q;
        E = diag(H);
        % test change
        change = norm(E - Eold);
        steps = steps +1;
    end
end
```

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

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

```
>> format long
>> format compact
>> A = hilb(5)
>> [Eqr,steps] = myqrmethod(A)
>> Eml = eig(A)
>> diff = norm(Eml - flipud(Eqr))
```

Exercises

17.1 Modify `myqrmethod` to stop after 1000 iterations. Use the modified program on the matrix `A = hilb(n)` with `n` equal to 10, 50, and 200. Use the norm to compare the results to the eigenvalues obtained from MATLAB's built-in program `eig`. Turn in your program and a brief report on the experiment.