Lecture 14

Eigenvalues and Eigenvectors

Suppose that A is a square $(n \times n)$ matrix. We say that a nonzero vector $\mathbf{v}$ is an eigenvector and a number λ is its eigenvalue if

$$A\mathbf{v} = \lambda\mathbf{v}. \tag{14.1}$$

Geometrically this means that $A\mathbf{v}$ is in the same direction as $\mathbf{v}$, since multiplying a vector by a number changes its length, but not its direction.

Matlab has a built-in routine for finding eigenvalues and eigenvectors:

```
A = pascal(4)
[v e] = eig(A)
```

The results are a matrix `v` that contains eigenvectors as columns and a diagonal matrix `e` that contains eigenvalues on the diagonal. We can check this by

```
v1 = v(:,1)
A*v1
e(1,1)*v1
```

Finding Eigenvalues for 2×2 and 3×3

If A is 2×2 or 3×3 then we can find its eigenvalues and eigenvectors by hand. Notice that Equation (14.1) can be rewritten as

$$A\mathbf{v} - \lambda\mathbf{v} = \mathbf{0}.$$

It would be nice to factor out the $\mathbf{v}$ from the right-hand side of this equation, but we can't because A is a matrix and λ is a number. However, since $I\mathbf{v} = \mathbf{v}$, we can do the following:

$$\begin{aligned} A\mathbf{v} - \lambda\mathbf{v} &= A\mathbf{v} - \lambda I\mathbf{v} \\ &= (A - \lambda I)\mathbf{v} \\ &= \mathbf{0} \end{aligned}$$

If $\mathbf{v}$ is nonzero, then by Theorem 3 in Lecture 10 the matrix $(A - \lambda I)$ must be singular. By the same theorem, we must have

$$\det(A - \lambda I) = 0.$$

This is called the *characteristic equation.*

For a 2×2 matrix, $A - \lambda I$ is calculated as in the following example:

$$\begin{aligned} A - \lambda I &= \begin{pmatrix} 1 & 4 \\ 3 & 5 \end{pmatrix} - \lambda \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 4 \\ 3 & 5 \end{pmatrix} - \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} \\ &= \begin{pmatrix} 1-\lambda & 4 \\ 3 & 5-\lambda \end{pmatrix}. \end{aligned}$$

The determinant of $A - \lambda I$ is then

$$\begin{aligned} \det(A - \lambda I) &= (1-\lambda)(5-\lambda) - 4 \cdot 3 \\ &= -7 - 6\lambda + \lambda^2. \end{aligned}$$

The characteristic equation $\det(A - \lambda I) = 0$ is simply a quadratic equation:

$$\lambda^2 - 6\lambda - 7 = 0.$$

The roots of this equation are $\lambda_1 = 7$ and $\lambda_2 = -1$. These are the eigenvalues of the matrix A. Now to find the corresponding eigenvectors we return to the equation $(A - \lambda I)\mathbf{v} = \mathbf{0}$. For $\lambda_1 = 7$, the equation for the eigenvector $(A - \lambda I)\mathbf{v} = \mathbf{0}$ is equivalent to the augmented matrix

$$\left(\begin{array}{cc|c} -6 & 4 & 0 \\ 3 & -2 & 0 \end{array} \right). \tag{14.2}$$

Notice that the first and second rows of this matrix are multiples of one another. Thus Gaussian elimination would produce all zeros on the bottom row. Thus this equation has infinitely many solutions, i.e. infinitely many eigenvectors. Since only the direction of the eigenvector matters, this is okay, we only need to find one of the eigenvectors. Since the second row of the augmented matrix represents the equation

$$3x - 2y = 0,$$

we can let

$$\mathbf{v}_1 = \begin{pmatrix} 2 \\ 3 \end{pmatrix}.$$

This comes from noticing that $(x, y) = (2, 3)$ is a solution of $3x - 2y = 0$.

For $\lambda_2 = -1$, $(A - \lambda I)\mathbf{v} = \mathbf{0}$ is equivalent to the augmented matrix

$$\left(\begin{array}{cc|c} 2 & 4 & 0 \\ 3 & 6 & 0 \end{array} \right).$$

Once again the first and second rows of this matrix are multiples of one another. For simplicity we can let

$$\mathbf{v}_2 = \begin{pmatrix} -2 \\ 1 \end{pmatrix}.$$

One can always check an eigenvector and eigenvalue by multiplying:

$$\begin{aligned} A\mathbf{v}_1 &= \begin{pmatrix} 1 & 4 \\ 3 & 5 \end{pmatrix} \begin{pmatrix} 2 \\ 3 \end{pmatrix} = \begin{pmatrix} 14 \\ 21 \end{pmatrix} = 7 \begin{pmatrix} 2 \\ 3 \end{pmatrix} = 7\mathbf{v}_1 \quad \text{and} \\ A\mathbf{v}_2 &= \begin{pmatrix} 1 & 4 \\ 3 & 5 \end{pmatrix} \begin{pmatrix} -2 \\ 1 \end{pmatrix} = \begin{pmatrix} 2 \\ -1 \end{pmatrix} = -1 \begin{pmatrix} -2 \\ 1 \end{pmatrix} = -1\mathbf{v}_2\,. \end{aligned}$$

For a 3×3 matrix we could complete the same process. The $\det(A - \lambda I) = 0$ would be a cubic polynomial and we would expect to usually get 3 roots, which are the eigenvalues.

Larger Matrices

For a $n \times n$ matrix with $n \geq 4$ this process is too long and cumbersome to complete by hand. Further, this process is not well suited even to implementation on a computer program since it involves determinants and solving a n-degree polynomial. For $n \geq 4$ we need more ingenious methods. These methods rely on the geometric meaning of eigenvectors and eigenvalues rather than solving algebraic equations. We will overview these methods in Lecture 16.

Complex Eigenvalues

It turns out that the eigenvalues of some matrices are complex numbers, even when the matrix only contains real numbers. When this happens the complex eigenvalues must occur in conjugate pairs, i.e.

$$\lambda_{1,2} = \alpha \pm i\beta.$$

The corresponding eigenvectors must also come in conjugate pairs:

$$\mathbf{w} = \mathbf{u} \pm i\mathbf{v}.$$

In applications, the imaginary part of the eigenvalue, β, often is related to the frequency of an oscillation. This is because of Euler's formula

$$e^{\alpha+i\beta} = e^{\alpha}(\cos\beta + i\sin\beta).$$

Certain kinds of matrices that arise in applications can only have real eigenvalues and eigenvectors. The most common such type of matrix is the symmetric matrix. A matrix is symmetric if it is equal to its own transpose, i.e. it is symmetric across the diagonal. For example,

$$\begin{pmatrix} 1 & 3 \\ 3 & -5 \end{pmatrix}$$

is symmetric and so we know beforehand that its eigenvalues will be real, not complex.

Exercises

14.1 Find the eigenvalues and eigenvectors of the following matrix by hand:

$$A = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}.$$

14.2 Find the eigenvalues and eigenvectors of the following matrix by hand:

$$B = \begin{pmatrix} 1 & -2 \\ 2 & 1 \end{pmatrix}.$$

Can you guess the eigenvalues of the matrix

$$C = \begin{pmatrix} a & -b \\ b & a \end{pmatrix}?$$

Lecture 15

An Application of Eigenvectors: Vibrational Modes and Frequencies

One application of eigenvalues and eigenvectors is in the analysis of vibration problems. A simple nontrivial vibration problem is the motion of two objects with equal masses m attached to each other and fixed outer walls by equal springs with spring constants k, as shown in Figure 15.1.

Let x_1 denote the displacement of the first mass and x_2 the displacement of the second, and note the displacement of the walls is zero. Each mass experiences forces from the adjacent springs proportional to the stretch or compression of the spring. Ignoring any friction, Newton's law of motion $ma = F$, leads to

$$\begin{array}{llllll} m\ddot{x}_1 & = -k(x_1 - 0) & +k(x_2 - x_1) & & = -2kx_1 + kx_2 & \text{and} \\ m\ddot{x}_2 & = & -k(x_2 - x_1) & +k(0 - x_2) & = kx_1 - 2kx_2 & . \end{array} \tag{15.1}$$

Dividing both sides by m we can write these equations in matrix form

$$\ddot{\mathbf{x}} = -A\mathbf{x}, \tag{15.2}$$

where

$$A = \frac{k}{m}B = \frac{k}{m}\begin{pmatrix} 2 & -1 \\ -1 & 2 \end{pmatrix}. \tag{15.3}$$

For this type of equation, the general solution is

$$\mathbf{x}(t) = c_1\mathbf{v}_1 \sin\left(\sqrt{\frac{k\lambda_1}{m}}\, t + \phi_1\right) + c_2\mathbf{v}_2 \sin\left(\sqrt{\frac{k\lambda_2}{m}}\, t + \phi_2\right) \tag{15.4}$$

where λ_1 and λ_2 are eigenvalues of B with corresponding eigenvectors $\mathbf{v}_1$ and $\mathbf{v}_2$. One can check that this is a solution by substituting it into the equation (15.2).

We can interpret the eigenvalues as the squares of the frequencies of oscillation. We can find the eigenvalues and eigenvectors of B using Matlab:

```
B = [2 -1 ; -1 2]
[v e] = eig(B)
```

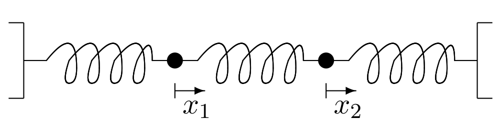

Figure 15.1: Two equal masses attached to each other and fixed walls by equal springs.

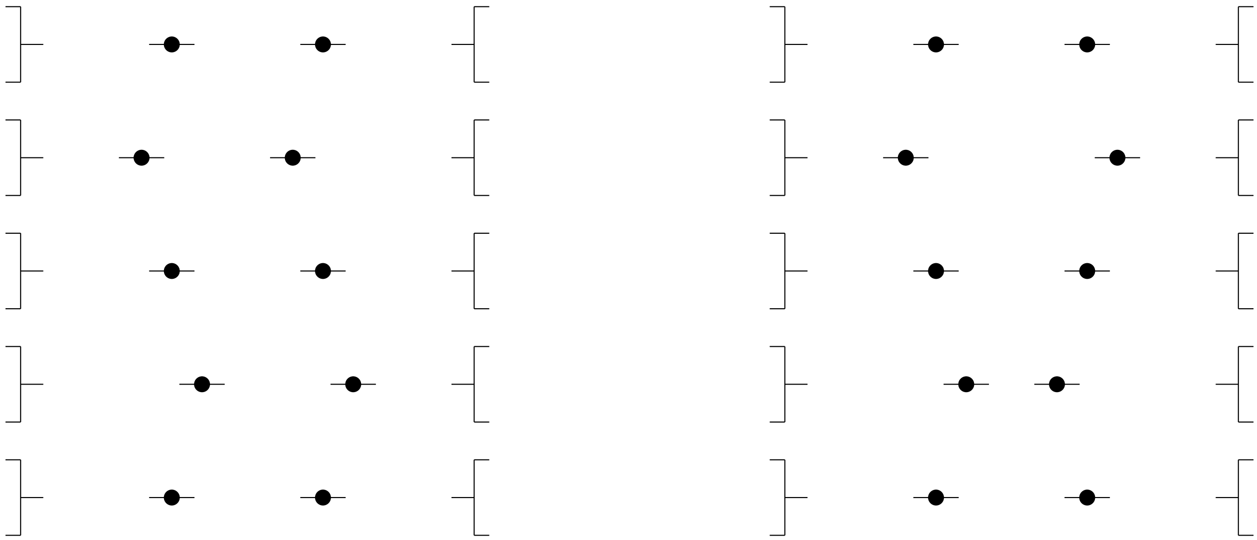

Figure 15.2: Two vibrational modes of a simple oscillating system. In the left mode the weights move together and in the right mode they move opposite. Note that the two modes actually move at different speeds.

This should produce a matrix $\mathbf{v}$ whose columns are the eigenvectors of B and a diagonal matrix e whose entries are the eigenvalues of B. In the first eigenvector, $\mathbf{v}_1$, the two entries are equal. This represents the mode of oscillation where the two masses move in sync with each other. The second eigenvector, $\mathbf{v}_2$, has the same entries but opposite signs. This represents the mode where the two masses oscillate in anti-synchronization. Notice that the frequency for anti-sync motion is $\sqrt{3}$ times that of synchronous motion.

Which of the two modes is the most dangerous for a structure or machine? It is the one with the *lowest frequency* because that mode can have the largest displacement. Sometimes this mode is called the *fundamental mode*.

To get the frequencies for the matrix $A = k/mB$, notice that if $\mathbf{v}_i$ is one of the eigenvectors for B then

$$A\mathbf{v}_i = \frac{k}{m}B\mathbf{v}_i = \frac{k}{m}\lambda_i\mathbf{v}_i.$$

Thus we can conclude that A has the same eigenvectors as B, but the eigenvalues are multiplied by the factor k/m. Thus the two frequencies are

$$\sqrt{\frac{k}{m}} \quad \text{and} \quad \sqrt{\frac{3k}{m}}.$$

We can do the same for three equal masses. The corresponding matrix B would be

$$B = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{pmatrix}.$$

Find the eigenvectors and eigenvalues as above. There are three different modes. Interpret them from the eigenvectors.

Exercises

15.1 Find the frequencies and modes for 4 equal masses with equal springs with $k = 1$. Interpret the modes.

15.2 Find the frequencies and modes for non-identical masses with equal springs with $k = 3$ in the following cases. How does unequal masses affect the modes?

(a) Two masses with $m_1 = 1$ and $m_2 = 2$.

(b) Three masses with $m_1 = 1$, $m_2 = 2$ and $m_3 = 3$.

Lecture 16

Numerical Methods for Eigenvalues

As mentioned above, the eigenvalues and eigenvectors of an $n \times n$ matrix where $n \geq 4$ must be found numerically instead of by hand. The numerical methods that are used in practice depend on the geometric meaning of eigenvalues and eigenvectors which is equation (14.1). The essence of all these methods is captured in the Power method, which we now introduce.

The Power Method

In the command window of MATLAB enter

```
A = hilb(5)
x = ones(5,1)
x = A*x
el = max(x)
x = x/el
```

Compare the new value of `x` with the original. Repeat the last three lines (you can use the scroll up button). Compare the newest value of `x` with the previous one and the original. Notice that there is less change between the second two. Repeat the last three commands over and over until the values stop changing. You have completed what is known as the *Power Method*. Now try the command

```
[v e] = eig(A)
```

The last entry in `e` should be the final `el` we computed. The last column in `v` is `x/norm(x)`. Below we explain why our commands gave this eigenvalue and eigenvector.

For illustration consider a 2×2 matrix whose eigenvalues are $1/3$ and 2 and whose corresponding eigenvectors are $\mathbf{v}_1$ and $\mathbf{v}_2$. Let $\mathbf{x}_0$ be any vector which is a combination of $\mathbf{v}_1$ and $\mathbf{v}_2$, e.g.,

$$\mathbf{x}_0 = \mathbf{v}_1 + \mathbf{v}_2.$$

Now let $\mathbf{x}_1$ be A times $\mathbf{x}_0$. It follows from (14.1) that

$$\begin{aligned} \mathbf{x}_1 &= A\mathbf{v}_1 + A\mathbf{v}_2 \\ &= \frac{1}{3}\mathbf{v}_1 + 2\mathbf{v}_2. \end{aligned} \tag{16.1}$$

Thus the $\mathbf{v}_1$ part is shrunk while the $\mathbf{v}_2$ is stretched. If we repeat this process k times then

$$\begin{aligned}\mathbf{x}_k &= A\mathbf{x}_{k-1}\\ &= A^k\mathbf{x}_0\\ &= \left(\frac{1}{3}\right)^k \mathbf{v}_1 + 2^k\mathbf{v}_2.\end{aligned} \tag{16.2}$$

Clearly, $\mathbf{x}_k$ grows in the direction of $\mathbf{v}_2$ and shrinks in the direction of $\mathbf{v}_1$. This is the principle of the Power Method, vectors multiplied by A are stretched most in the direction of the eigenvector whose eigenvalue has the largest absolute value.

The eigenvalue with the largest absolute value is called the *dominant* eigenvalue. In many applications this quantity will necessarily be positive for physical reasons. When this is the case, the MATLAB code above will work since `max(v)` will be the element with the largest absolute value. In applications where the dominant eigenvalue may be negative, the program must use flow control to determine the correct number.

Summarizing the Power Method:

- Repeatedly multiply $\mathbf{x}$ by A and divide by the element with the largest absolute value.
- The element of largest absolute value converges to largest absolute eigenvalue.
- The vector converges to the corresponding eigenvector.

Note that this logic only works when the eigenvalue largest in magnitude is real. If the matrix and starting vector are real then the power method can never give a result with an imaginary part. Eigenvalues with imaginary part mean the matrix has a rotational component, so the eigenvector would not settle down either.

Try

```
A = rand(15,15);
e = eig(A)
```

You can see that for a random square matrix, many of the **ew's** are complex.

However, matrices in applications are not just random. They have structure, and this can lead to real eigenvalues as seen in the next section.

The residual of an approximate eigenvector-eigenvalue pair

If $\mathbf{v}$ and λ are an eigenvector-eigenvalue pair for A, then they are supposed to satisfy the equations: $A\mathbf{v} = \lambda\mathbf{v}$. Thus a scalar residual for approximate $\mathbf{v}$ and λ would be:

$$r = \|A\mathbf{v} - \lambda\mathbf{v}\|.$$

Symmetric, Positive-Definite Matrices

As noted in the previous paragraph, the power method can fail if A has complex eigenvalues. One class of matrices that appear often in applications and for which the eigenvalues are always real are called the symmetric matrices. A matrix is *symmetric* if

$$A' = A,$$

i.e. A is symmetric with respect to reflections about its diagonal.

Try

```
A = rand(5,5)
C = A'*A
e = eig(C)
```

You can see that the eigenvalues of these symmetric matrices are real.

Next we consider an even more specialized class for which the eigenvalues are not only real, but positive. A symmetric matrix is called *positive definite* if for all vectors $\mathbf{v} \neq \mathbf{0}$ the following holds:

$$A\mathbf{v} \cdot \mathbf{v} > 0.$$

Geometrically, A does not rotate any vector by more than $\pi/2$. In summary:

- If A is symmetric then its eigenvalues are real.
- If A is symmetric positive definite, then its eigenvalues are positive numbers.

Notice that the B matrices in the previous section were symmetric and the **ew's** were all real. Notice that the Hilbert and Pascal matrices are symmetric.

The Inverse Power Method

In the application of vibration analysis, the mode (eigenvector) with the lowest frequency (eigenvalue) is the most dangerous for the machine or structure. The Power Method gives us instead the largest eigenvalue, which is the least important frequency. In this section we introduce a method, the *Inverse Power Method* which produces exactly what is needed.

The following facts are at the heart of the Inverse Power Method:

- If λ is an eigenvalue of A then $1/\lambda$ is an eigenvalue for A^{-1}.
- The eigenvectors for A and A^{-1} are the same.

Thus if we apply the Power Method to A^{-1} we will obtain the largest absolute eigenvalue of A^{-1}, which is exactly the reciprocal of the smallest absolute eigenvalue of A. We will also obtain the corresponding eigenvector, which is an eigenvector for both A^{-1} and A. Recall that in the application of vibration mode analysis, the smallest eigenvalue and its eigenvector correspond exactly to the frequency and mode that we are most interested in, i.e. the one that can do the most damage.

Here as always, we do not really want to calculate the inverse of A directly if we can help it. Fortunately, multiplying $\mathbf{x}_i$ by A^{-1} to get $\mathbf{x}_{i+1}$ is equivalent to solving the system

$$A\mathbf{x}_{i+1} = \mathbf{x}_i \,,$$

which can be done efficiently and accurately. Since iterating this process involves solving a linear system with the same A but many different right hand sides, it is a perfect time to use the LU decomposition to save computations. The following function program does n steps of the Inverse Power Method.

```
function [v e] = myipm(A,n)
    % Performs the inverse power method.
    % Inputs: A -- a square matrix.
    %         n -- the number of iterations to perform.
    % Outputs: v -- the estimated eigenvector.
    %          e -- the estimated eigenvalue.
    [L U P] = lu(A);  % LU decomposition of A with pivoting
    m = size(A,1); % determine the size of A
    v = ones(m,1); % make an initial vector with ones
    for i = 1:n
        pv = P*v;  % Apply pivot
        y = L\pv;  % solve via LU
        v = U\y;
        % figure out the maximum entry in absolute value, retaining its sign
        M = max(v);
        m = min(v);
        if abs(M) >= abs(m)
           el = M;
        else
           el = m;
        end
        v = v/el; % divide by the estimated eigenvalue of the inverse of A
    end
    e = 1/el; % reciprocate to get an eigenvalue of A
end
```

Exercises

16.1 For each of the following matrices, perform two interations of the power method by hand starting with a vector of all ones. State the resulting approximations of the eigenvalue and eigenvector.

$$A = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \tag{16.3}$$

$$B = \begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -3 \end{bmatrix} \tag{16.4}$$

16.2 (a) Write a well-commented MATLAB **function** program `mypm` that inputs a matrix and a tolerance, applies the power method until the residual is less than the tolerance, and outputs the estimated eigenvalue and eigenvector, the number of steps and the scalar residual.

(b) Test your program on the matrices A and B in the previous exercise.

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 a printout of your program and a brief report on the experiment.