Eigenvalues and Eigenvectors

In structural dynamics we have to solve eigenfuction of The following form:

$$\mathbf{k}\ \hat{\mathbf{v}}_n = \omega_n^2\ \mathbf{m}\ \hat{\mathbf{v}}_n$$

Symmetric Matrix Eigenvalue Problems

Find λ for which nontrivial solutions of $\mathbf{Ax} = \lambda\mathbf{x}$ exist

The *standard form* of the matrix eigenvalue problem is

$$\mathbf{Ax} = \lambda\mathbf{x} \tag{9.1}$$

where $\mathbf{A}$ is a given $n \times n$ matrix. The problem is to find the scalar λ and the vector $\mathbf{x}$. Rewriting Eq. (9.1) in the form

$$(\mathbf{A} - \lambda\mathbf{I})\,\mathbf{x} = \mathbf{0} \tag{9.2}$$

Expansion of the determinant leads to the polynomial equation known as the *characteristic equation*

$$a_1\lambda^n + a_2\lambda^{n-1} + \cdots + a_n\lambda + a_{n+1} = 0$$

which has the roots λ_i, $i = 1, 2, \ldots, n$, called the *eigenvalues* of the matrix $\mathbf{A}$. The solutions $\mathbf{x}_i$ of $(\mathbf{A} - \lambda_i\mathbf{I})\,\mathbf{x} = \mathbf{0}$ are known as the eigenvectors.

As an example, consider the matrix

$$\mathbf{A} = \begin{bmatrix} 1 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}$$

The characteristic equation is

$$|\mathbf{A} - \lambda\mathbf{I}| = \begin{vmatrix} 1-\lambda & -1 & 0 \\ -1 & 2-\lambda & -1 \\ 0 & -1 & 1-\lambda \end{vmatrix} = -3\lambda + 4\lambda^2 - \lambda^3 = 0 \qquad \text{(b)}$$

The roots of this equation are $\lambda_1 = 0, \lambda_2 = 1, \lambda_3 = 3$. To compute the eigenvector corresponding the λ_3, we substitute $\lambda = \lambda_3$ into Eq. (9.2), obtaining

$$\begin{bmatrix} -2 & -1 & 0 \\ -1 & -1 & -1 \\ 0 & -1 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} \qquad \text{(c)}$$

We know that the determinant of the coefficient matrix is zero, so that the equations are not linearly independent. Therefore, we can assign an arbitrary value to any one component of $\mathbf{x}$ and use two of the equations to compute the other two components. Choosing $x_1 = 1$, the first equation of Eq. (c) yields $x_2 = -2$ and from the third equation we get $x_3 = 1$. Thus the eigenvector associated with λ_3 is

$$\mathbf{x}_3 = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix}$$

The other two eigenvectors

$$\mathbf{x}_2 = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} \qquad \mathbf{x}_1 = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}$$

can be obtained in the same manner.

It is sometimes convenient to display the eigenvectors as columns of a matrix $\mathbf{X}$. For the problem at hand, this matrix is

$$\mathbf{X} = \begin{bmatrix} \mathbf{x}_1 & \mathbf{x}_2 & \mathbf{x}_3 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & -2 \\ 1 & -1 & 1 \end{bmatrix}$$

It is clear from the above example that the magnitude of an eigenvector is indeterminate; only its direction can be computed from Eq. (9.2). It is customary to *normalize* the eigenvectors by assigning a unit magnitude to each vector. Thus the normalized eigenvectors in our example are

$$\mathbf{X} = \begin{bmatrix} 1/\sqrt{3} & 1/\sqrt{2} & 1/\sqrt{6} \\ 1/\sqrt{3} & 0 & -2/\sqrt{6} \\ 1/\sqrt{3} & -1/\sqrt{2} & 1/\sqrt{6} \end{bmatrix}$$

Here are some useful properties of eigenvalues and eigenvectors, given without proof:

- All the eigenvalues of a symmetric matrix are real.
- All eigenvalues of a symmetric, positive-definite matrix are real and positive.
- The eigenvectors of a symmetric matrix are orthonormal; that is, $\mathbf{X}^T\mathbf{X} = \mathbf{I}$.
- If the eigenvalues of $\mathbf{A}$ are λ_i, then the eigenvalues of $\mathbf{A}^{-1}$ are λ_i^{-1}.

Eigenvalue problems that originate from physical problems often end up with a symmetric **A**. This is fortunate, because symmetric eigenvalue problems are much easier to solve than their nonsymmetric counterparts. In this chapter, we largely restrict our discussion to eigenvalues and eigenvectors of symmetric matrices.

Common sources of eigenvalue problems are the analysis of vibrations and stability. These problems often have the following characteristics:

- The matrices are large and sparse (e.g., have a banded structure).
- We need to know only the eigenvalues; if eigenvectors are required, only a few of them are of interest.

There are various methods like Schur's factorization, Jacobi's method etc for solving These problems. They are not too difficult but kinda involved. We will focus more on Simple approaches, and for more complicated matrices use the standard Matlab eigen solver.

Physical problems often give rise to eigenvalue problems of the form

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{B}\mathbf{x} \tag{9.21}$$

where $\mathbf{A}$ and $\mathbf{B}$ are symmetric $n \times n$ matrices. We assume that $\mathbf{B}$ is also positive defi-

MATLAB Functions

MATLAB's function for solving eigenvalue problems is `eig`. Its usage for the standard eigenvalue problem $\mathbf{Ax} = \lambda\mathbf{x}$ is

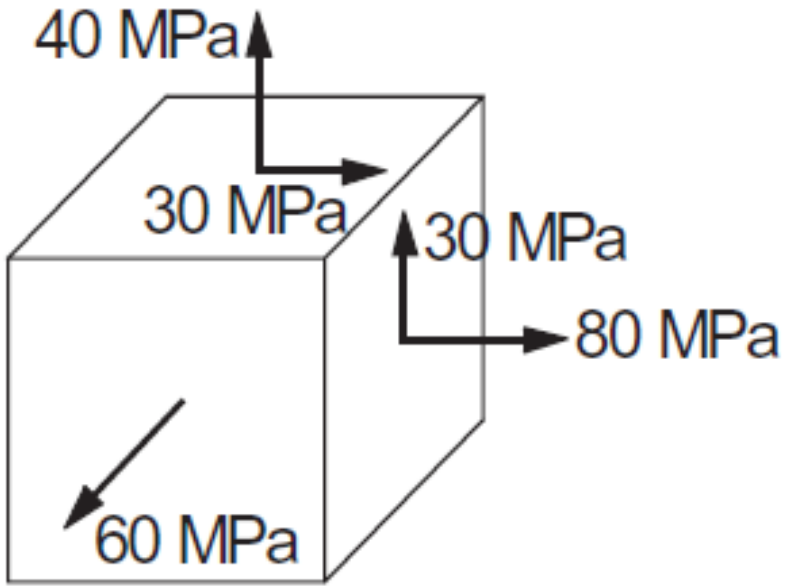
`eVals = eig(A)` returns the eigenvalues of the matrix `A` (`A` can be unsymmetric).

`[X,D] = eig(A)` returns the eigenvector matrix `X` and the diagonal matrix `D` that contains the eigenvalues on its diagonal; that is, `eVals = diag(D)`.

For the nonstandard form $\mathbf{Ax} = \lambda\mathbf{Bx}$, the calls are

```
eVals = eig(A,B)
[X,D] = eig(A,B)
```

EXAMPLE 9.1

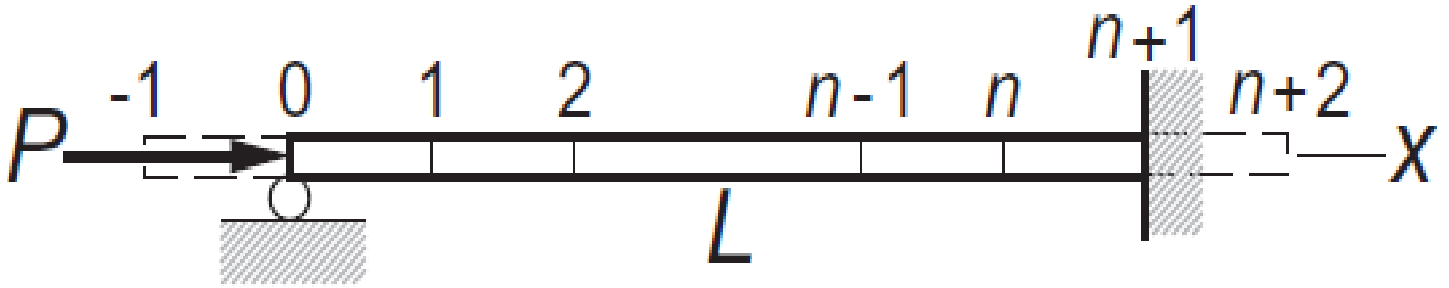


The stress matrix (tensor) corresponding to the state of stress shown is

$$\mathbf{S} = \begin{bmatrix} 80 & 30 & 0 \\ 30 & 40 & 0 \\ 0 & 0 & 60 \end{bmatrix} \text{MPa}$$

(each row of the matrix consists of the three stress components acting on a coordinate plane). It can be shown that the eigenvalues of **S** are the *principal stresses* and the eigenvectors are normal to the *principal planes*. (1) Determine the principal stresses by diagonalizing **S** with a Jacobi rotation and (2) compute the eigenvectors.

EXAMPLE 9.3



The propped cantilever beam carries a compressive axial load P. The lateral displacement $u(x)$ of the beam can be shown to satisfy the differential equation

$$u^{(4)} + \frac{P}{EI}u'' = 0 \tag{a}$$

where EI is the bending rigidity. The boundary conditions are

$$u(0) = u''(0) = 0 \qquad u(L) = u'(L) = 0 \tag{b}$$

(1) Show that buckling analysis of the beam results in a matrix eigenvalue problem if the derivatives are approximated by finite differences. (2) Use Jacobi method to compute the lowest three buckling loads and the corresponding eigenvectors.

Solution of Part (1) We divide the beam into $n+1$ segments of length $L/(n+1)$ each as shown and enforce the differential equation at nodes 1 to n. Replacing the derivatives of u in Eq. (a) by central finite differences of $\mathcal{O}(h^2)$ at the interior nodes (nodes 1 to n), we obtain

$$\frac{u_{i-2} - 4u_{i-1} + 6u_i - 4u_{i+1} + u_{i+2}}{h^4}$$

$$= \frac{P}{EI} \frac{-u_{i-1} + 2u_i - u_{i-1}}{h^2}, i = 1, 2, \ldots, n$$

After multiplication by h^4, the equations become

$$
\begin{aligned}
u_{-1} - 4u_0 + 6u_1 - 4u_2 + u_3 &= \lambda(-u_0 + 2u_1 - u_2) \\
u_0 - 4u_1 + 6u_2 - 4u_3 + u_4 &= \lambda(-u_1 + 2u_2 - u_3) \\
&\vdots \\
u_{n-3} - 4u_{n-2} + 6u_{n-1} - 4u_n + u_{n+1} &= \lambda(-u_{n-2} + 2u_{n-1} - u_n) \\
u_{n-2} - 4u_{n-1} + 6u_n - 4u_{n+1} + u_{n+2} &= \lambda(-u_{n-1} + 2u_n - u_{n+1})
\end{aligned}
$$

where

$$
\lambda = \frac{Ph^2}{EI} = \frac{PL^2}{(n+1)^2 EI}
$$

The displacements u_{-1}, u_0, u_{n+1}, and u_{n+2} can be eliminated by using the prescribed boundary conditions. Referring to Table 8.1, the finite difference approximations to the boundary conditions in Eqs. (b) yield

$$u_0 = 0 \qquad u_{-1} = -u_1 \qquad u_{n+1} = 0 \qquad u_{n+2} = u_n$$

Substitution into Eqs. (c) yields the matrix eigenvalue problem $\mathbf{Ax} = \lambda\mathbf{Bx}$, where

$$\mathbf{A} = \begin{bmatrix} 5 & -4 & 1 & 0 & 0 & \cdots & 0 \\ -4 & 6 & -4 & 1 & 0 & \cdots & 0 \\ 1 & -4 & 6 & -4 & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 1 & -4 & 6 & -4 & 1 \\ 0 & \cdots & 0 & 1 & -4 & 6 & -4 \\ 0 & \cdots & 0 & 0 & 1 & -4 & 7 \end{bmatrix}$$

$$\mathbf{B} = \begin{bmatrix} 2 & -1 & 0 & 0 & 0 & \cdots & 0 \\ -1 & 2 & -1 & 0 & 0 & \cdots & 0 \\ 0 & -1 & 2 & -1 & 0 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & 0 & -1 & 2 & -1 & 0 \\ 0 & \cdots & 0 & 0 & -1 & 2 & -1 \\ 0 & \cdots & 0 & 0 & 0 & -1 & 2 \end{bmatrix}$$

The buckling loads are given by $P_i = (n+1)^2 \lambda_i EI/L^2$. Thus

$$P_1 = \frac{(11)^2 (0.1641)\, EI}{L^2} = 19.86 \frac{EI}{L^2}$$

$$P_2 = \frac{(11)^2 (0.4720)\, EI}{L^2} = 57.11 \frac{EI}{L^2}$$

$$P_3 = \frac{(11)^2 (0.9022)\, EI}{L^2} = 109.2 \frac{EI}{L^2}$$

The analytical values are $P_1 = 20.19 EI/L^2$, $P_2 = 59.68 EI/L^2$, and $P_3 = 118.9 EI/L^2$.

Inverse Power and Power Methods

Inverse Power Method

The inverse power method is a simple iterative procedure for finding the smallest eigenvalue λ_1 and the corresponding eigenvector $\mathbf{x}_1$ of

$$\mathbf{A}\mathbf{x} = \lambda\mathbf{x} \tag{9.27}$$

The method works like this:

1. Let $\mathbf{v}$ be an approximation to $\mathbf{x}_1$ (a random vector of unit magnitude will do).
2. Solve

$$\mathbf{A}\mathbf{z} = \mathbf{v} \tag{9.28}$$

   for the vector $\mathbf{z}$.
3. Compute $|\mathbf{z}|$.
4. Let $\mathbf{v} = \mathbf{z}/|\mathbf{z}|$ and repeat steps 2–4 until the change in $\mathbf{v}$ is negligible.

At the conclusion of the procedure, $|\mathbf{z}| = \pm 1/\lambda_1$ and $\mathbf{v} = \mathbf{x}_1$. The sign of λ_1 is determined as follows: if $\mathbf{z}$ changes sign between successive iterations, λ_1 is negative; otherwise, use the plus sign.

The inverse power method also works with the nonstandard eigenvalue problem

$$\mathbf{Ax} = \lambda \mathbf{Bx} \tag{9.30}$$

provided that Eq. (9.28) is replaced by

$$\mathbf{Az} = \mathbf{Bv} \tag{9.31}$$

Eigenvalue Shifting

By inspection of Eq. (9.29) we see that the rate of convergence is determined by the strength of the inequality $|\lambda_1/\lambda_2| < 1$ (the second term in the equation). If $|\lambda_2|$ is well separated from $|\lambda_1|$, the inequality is strong and the convergence is rapid. On the other hand, close proximity of these two eigenvalues results in very slow convergence.

The rate of convergence can be improved by a technique called *eigenvalue shifting*. Letting

$$\lambda = \lambda^* + s \tag{9.32}$$

where s is a predetermined "shift," the eigenvalue problem in Eq. (9.27) is transformed to

$$\mathbf{A}\mathbf{x} = (\lambda^* + s)\mathbf{x}$$

$$\mathbf{A}^*\mathbf{x} = \lambda^*\mathbf{x}$$

$$\mathbf{A}^* = \mathbf{A} - s\mathbf{I}$$

Solving the transformed problem in Eq. (9.33) by the inverse power method yields λ_1^* and $\mathbf{x}_1$, where λ_1^* is the smallest eigenvalue of $\mathbf{A}^*$. The corresponding eigenvalue of the original problem, $\lambda = \lambda_1^* + s$, is thus the *eigenvalue closest to* s.

Eigenvalue shifting has two applications. An obvious one is the determination of the eigenvalue closest to a certain value s. For example, if the working speed of a shaft is s rev/min, it is imperative to assure that there are no natural frequencies (which are related to the eigenvalues) close to that speed.

Eigenvalue shifting is also used to speed up convergence. Suppose that we are computing the smallest eigenvalue λ_1 of the matrix $\mathbf{A}$. The idea is to introduce a shift s that makes λ_1^*/λ_2^* as small as possible. Since $\lambda_1^* = \lambda_1 - s$, we should choose $s \approx \lambda_1$ ($s = \lambda_1$ should be avoided to prevent division by zero). Of course, this method works only if we have a prior estimate of λ_1.

The inverse power method with eigenvalue shifting is a particularly powerful tool for finding eigenvectors if the eigenvalues are known. By shifting very close to an eigenvalue, the corresponding eigenvector can be computed in one or two iterations.

$$\mathbf{z} = \sum_{i=1}^{n} \frac{v_i}{\lambda_i}\mathbf{x}_i = \frac{1}{\lambda_1}\sum_{i=1}^{n} v_i \frac{\lambda_1}{\lambda_i}\mathbf{x}_i$$

$$= \frac{1}{\lambda_1}\left(v_1\mathbf{x}_1 + v_2\frac{\lambda_1}{\lambda_2}\mathbf{x}_2 + v_3\frac{\lambda_1}{\lambda_3}\mathbf{x}_3 + \cdots\right)$$

Power Method

The power method converges to the eigenvalue *furthest from zero* and the associated eigenvector. It is very similar to the inverse power method; the only difference between the two methods is the interchange of $\mathbf{v}$ and $\mathbf{z}$ in Eq. (9.28). The outline of the procedure is:

1. Let $\mathbf{v}$ be an approximation to $\mathbf{x}_n$ (a random vector of unit magnitude will do).
2. Compute the vector

$$\mathbf{z} = \mathbf{A}\mathbf{v} \tag{9.35}$$

3. Compute $|\mathbf{z}|$.
4. Let $\mathbf{v} = \mathbf{z}/|\mathbf{z}|$ and repeat steps 2–4 until the change in $\mathbf{v}$ is negligible.

At the conclusion of the procedure, $|\mathbf{z}| = \pm\lambda_n$ and $\mathbf{v} = \mathbf{x}_n$ (the sign of λ_n is determined in the same way as in the inverse power method).

EXAMPLE 9.4

The stress matrix describing the state of stress at a point is

$$\mathbf{S} = \begin{bmatrix} -30 & 10 & 20 \\ 10 & 40 & -50 \\ 20 & -50 & -10 \end{bmatrix} \text{MPa}$$

Determine the largest principal stress (the eigenvalue of **S** furthest from zero) by the power method.

Solution First iteration:

Let $v = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T$ be the initial guess for the eigenvector. Then

$$\mathbf{z} = \mathbf{S}\mathbf{v} = \begin{bmatrix} -30 & 10 & 20 \\ 10 & 40 & -50 \\ 20 & -50 & -10 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -30.0 \\ 10.0 \\ 20.0 \end{bmatrix}$$

$$|\mathbf{z}| = \sqrt{30^2 + 10^2 + 20^2} = 37.417$$

$$\mathbf{v} = \frac{\mathbf{z}}{|\mathbf{z}|} = \begin{bmatrix} -30.0 \\ 10.0 \\ 20.0 \end{bmatrix} \frac{1}{37.417} = \begin{bmatrix} -0.801\,77 \\ 0.267\,26 \\ 0.534\,52 \end{bmatrix}$$

Second iteration:

$$\mathbf{z} = \mathbf{S}\mathbf{v} = \begin{bmatrix} -30 & 10 & 20 \\ 10 & 40 & -50 \\ 20 & -50 & -10 \end{bmatrix} \begin{bmatrix} -0.801\,77 \\ 0.267\,26 \\ 0.534\,52 \end{bmatrix} = \begin{bmatrix} 37.416 \\ -24.053 \\ -34.744 \end{bmatrix}$$

$$|\mathbf{z}| = \sqrt{37.416^2 + 24.053^2 + 34.744^2} = 56.\,442$$

$$\mathbf{v} = \frac{\mathbf{z}}{|\mathbf{z}|} = \begin{bmatrix} 37.416 \\ -24.053 \\ -34.744 \end{bmatrix} \frac{1}{56.\,442} = \begin{bmatrix} 0.662\,91 \\ -0.426\,15 \\ -0.615\,57 \end{bmatrix}$$

Third iteration:

$$\mathbf{z} = \mathbf{S}\mathbf{v} = \begin{bmatrix} -30 & 10 & 20 \\ 10 & 40 & -50 \\ 20 & -50 & -10 \end{bmatrix} \begin{bmatrix} 0.66291 \\ -0.42615 \\ -0.61557 \end{bmatrix} = \begin{bmatrix} -36.460 \\ 20.362 \\ 40.721 \end{bmatrix}$$

$$|\mathbf{z}| = \sqrt{36.460^2 + 20.362^2 + 40.721^2} = 58.328$$

$$\mathbf{v} = \frac{\mathbf{z}}{|\mathbf{z}|} = \begin{bmatrix} -36.460 \\ 20.362 \\ 40.721 \end{bmatrix} \frac{1}{58.328} = \begin{bmatrix} -0.62509 \\ 0.34909 \\ 0.69814 \end{bmatrix}$$

At this point the approximation of the eigenvalue we seek is $\lambda = -58.328$ MPa (the negative sign is determined by the sign reversal of $\mathbf{z}$ between iterations). This is actually close to the second-largest eigenvalue $\lambda_2 = -58.39$ MPa. By continuing the iterative process we would eventually end up with the largest eigenvalue $\lambda_3 = 70.94$ MPa. But since $|\lambda_2|$ and $|\lambda_3|$ are rather close, the convergence is too slow from this point on

EXAMPLE 9.5

Determine the smallest eigenvalue λ_1 and the corresponding eigenvector of

$$\mathbf{A} = \begin{bmatrix} 11 & 2 & 3 & 1 & 4 \\ 2 & 9 & 3 & 5 & 2 \\ 3 & 3 & 15 & 4 & 3 \\ 1 & 5 & 4 & 12 & 4 \\ 4 & 2 & 3 & 4 & 17 \end{bmatrix}$$

Use the inverse power method with eigenvalue shifting knowing that $\lambda_1 \approx 5$.

```
% Example 9.5 (Inverse power method)
s = 5;
A = [11  2  3  1  4;
      2  9  3  5  2;
      3  3 15  4  3;
      1  5  4 12  4;
      4  2  3  4 17];
[eVal,eVec] = invPower(A,s)
```

■ `invPower`

Given the matrix **A** and the scalar s, the function `invPower` returns the eigenvalue of **A** closest to s and the corresponding eigenvector. The matrix $\mathbf{A}^* = \mathbf{A} - s\mathbf{I}$ is decomposed as soon as it is formed, so that only the solution phase (forward and back substitution) is needed in the iterative loop. If **A** is banded, the efficiency of the program can be improved by replacing `LUdec` and `LUsol` by functions that specialize in banded matrices – see Example 9.6. The program line that forms $\mathbf{A}^*$ must also be modified to be compatible with the storage scheme used for **A**.