1
REVIEW OF BACKGROUND MATERIALS

We shall briefly review index notation, the Gauss elimination method to solve a system of linear equations, Jacobi's method for an eigenvalue problem, and variational methods. These comprise the minimum background required in order to understand the materials given in this book. Readers with a good grasp of these topics can skip this chapter and start with Chapter 2.

1.1 Index notation

The most covenient notation for the study of finite element methods is *index notation,* since equations written using it can be translated to FORTRAN statements directly. For example, let us consider the dot product of two vectors,

$$\mathbf{u} = \sum_{i=1}^{N} u_i \mathbf{i}_i = u_1 \mathbf{i}_1 + u_2 \mathbf{i}_2 + \cdots + u_N \mathbf{i}_N \tag{1.1}$$

and

$$\mathbf{v} = \sum_{i=1}^{N} v_i \mathbf{i}_i = v_1 \mathbf{i}_1 + v_2 \mathbf{i}_2 + \cdots + v_N \mathbf{i}_N \tag{1.2}$$

Here the numbers $\{u_i\}$ are the components of the vector in the $\mathbb{R}^N$ space. For simplicity, let $N = 3$; we are then in the three-dimensional space that we use in mechanics. In most cases, x-, y-, and z-coordinate axes are set up in the space $\mathbb{R}^3$, and the unit vectors $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$ are introduced along each axis x, y, and z, respectively (see Figure 1.1). In index notation, we change these as follows:

$$\begin{array}{cccccc} x & y & z & \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \downarrow \\ x_1 & x_2 & x_3 & \mathbf{i}_1 & \mathbf{i}_2 & \mathbf{i}_3 \end{array}$$

Then the position vector $\mathbf{r} = x\mathbf{i} + y\mathbf{j} + z\mathbf{k}$ is written as $\mathbf{r} = x_i \mathbf{i}_i$, instead of $\mathbf{r} = \sum_{i=1}^{3} x_i \mathbf{i}_i$. The rule involved here is that summation is taken over the index i, which is repeated exactly once (i.e. appears exactly twice) in a term. If clarity is necessary on the range of summation, we may write

$$\mathbf{r} = x_i \mathbf{i}_i, \quad i = 1, 2, \ldots, N \tag{1.3}$$

The unit vectors $\mathbf{i}_1$, $\mathbf{i}_2$, and $\mathbf{i}_3$ (i.e., $\mathbf{i}$, $\mathbf{j}$, and $\mathbf{k}$) are called the *base vectors* for the $\mathbb{R}^3$ space, and the numbers x_1, x_2, and x_3 are components of the vector $\mathbf{r}$ with respect to the base vectors $\mathbf{i}_1$, $\mathbf{i}_2$, and $\mathbf{i}_3$. Generalization to the $\mathbb{R}^N$ space is straightforward. Thus, the vectors $\mathbf{u}$ and $\mathbf{v}$ in (1.1) and (1.2) are represented by

$$\mathbf{u} = u_i \mathbf{i}_i \qquad \mathbf{v} = v_i \mathbf{i}_i, \quad i = 1, 2, \ldots, N \tag{1.4}$$

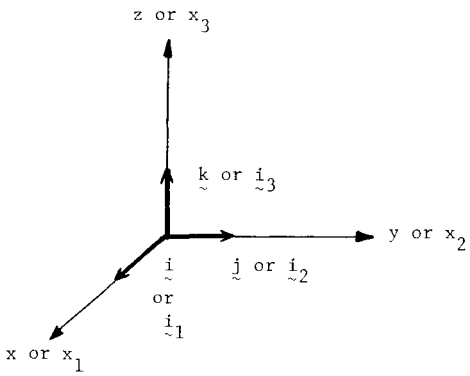

Figure 1.1 Coordinate system for $\mathbb{R}^3$.

and their dot product can be written as

$$\mathbf{u} \cdot \mathbf{v} = u_i v_i \left(\triangleq \sum_{i=1}^{N} u_i v_i \right) \tag{1.5}$$

For the dot product of two vectors we have the FORTRAN statements

```
      DOT = 0
      DO 100 I = 1,N
100 DOT = DOT + U(I) * V(I)
```
(1.6)

The index in index notation directly becomes the one that indicates the entry of the array in FORTRAN.

By the summation convention, we do not sum over i in the term $a_i b_i c_i$, since the index i is repeated three times; however, we take summation over j in $a_j c_i d_j$. A remark on the summation convention is that the letter used for the repeated index is immaterial in the sense that $a_j c_i d_j$ is exactly the same as the expression $a_k c_i d_k$ since summation is taken over j in the first and over k in the second. Thus, repeated indices are called *dummy indices*. The index i in the expression $a_j c_i d_j$ above is called a free index that takes any number from $i = 1, 2, \ldots, N$. A FORTRAN statement for this is given as

```
      SUM = 0
      DO 100 J = 1,N
100 SUM = SUM + A(J) * D(J)
      ACD(I) = SUM * C(I)
```
(1.7)

Similarly, let us introduce the basis for a matrix (or tensor) **T** defined in the space $\mathbb{R}^N\mathbb{R}^M$ as the set $\{\mathbf{i}_i\mathbf{e}_I\}$, $i = 1, 2, \ldots, N$, $I = 1, 2, \ldots, M$, where $\mathbf{i}_i\mathbf{e}_I$ is the *dyadic*, or outer product, and is best defined by the following two operations:

$$\mathbf{i}_i\mathbf{e}_I \circ \mathbf{v}_J = \mathbf{i}_i(\mathbf{e}_I \circ \mathbf{v}_J) \qquad \mathbf{v}_j \circ (\mathbf{i}_i\mathbf{e}_I) = (\mathbf{v}_j \circ \mathbf{i}_i)\mathbf{e}_I \tag{1.8}$$

where the product indicated by $\circ$ denotes ordinary, dot, or cross multiplication. Using the components T_{iI} of the tensor **T** with respect to $\mathbf{i}_i\mathbf{e}_I$, **T** is represented as

$$\mathbf{T} = T_{iI}\mathbf{i}_i\mathbf{e}_I \tag{1.9}$$

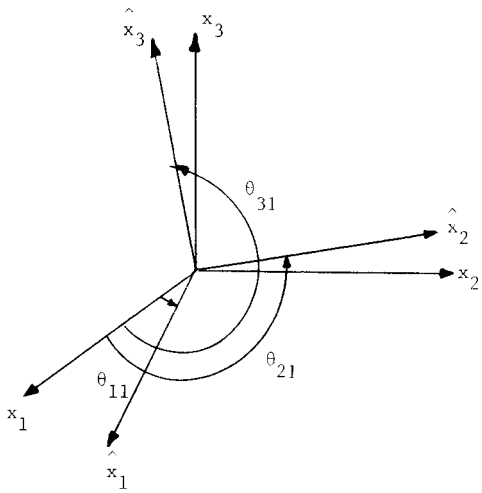

Figure 1.2 Rotation of the coordinate system.

We sometimes describe the tensor $\mathbf{T}$ by the matrix form $[T_{iI}]$ by arraying its components T_{iI}, $i = 1, 2, \ldots, N$ and $I = 1, 2, \ldots, M$. The operation of multiplying a matrix and a vector can be given as

$$\mathbf{v} = \mathbf{T} \cdot \mathbf{u} \qquad \text{or} \qquad v_i = T_{iJ} u_J \tag{1.10}$$

where $\mathbf{v} = v_i \mathbf{i}_i \in \mathbb{R}^N$ and $\mathbf{u} = u_J \mathbf{e}_J \in \mathbb{R}^M$. Note that a matrix transforms one vector into another vector. One very typical example of a matrix is the coordinate rotation matrix $\boldsymbol{\beta} = \beta_{Ii} \mathbf{e}_I \mathbf{i}_i$ (see Figure 1.2) defined by

$$\beta_{Ii} = \cos(\theta_{Ii}) \tag{1.11}$$

where θ_{Ii} is the angle between the I axis and the i axis. Then the unit base vectors $\mathbf{e}_I$ are related to the vectors $\{\mathbf{i}_i\}$:

$$\mathbf{e}_I = \beta_{Ii} \mathbf{i}_i \tag{1.12}$$

Using these, we can obtain the components of $\mathbf{v}$ in the rotated coordinate system $(\hat{x}_1, \hat{x}_2, \hat{x}_3)$:

$$\mathbf{v} = v_i \mathbf{i}_i = \hat{v}_I \mathbf{e}_I \tag{1.13a}$$

where

$$\begin{aligned} \hat{v}_I &\triangleq \mathbf{v} \cdot \mathbf{e}_I = (v_i \mathbf{i}_i) \cdot (\beta_{Ij} \mathbf{i}_j) \\ &= v_i \beta_{Ij} \mathbf{i}_i \cdot \mathbf{i}_j = v_i \beta_{Ij} \delta_{ij} = \beta_{Ii} v_i \end{aligned} \tag{1.13b}$$

Here we have used the fact that

$$\mathbf{i}_i \cdot \mathbf{i}_j = \delta_{ij} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{1.14}$$

by using the Kronecker delta. Similarly, we can find the new components of a matrix after a coordinate rotation has been performed. Indeed, for a given matrix

$$\mathbf{A} = A_{ij} \mathbf{i}_i \mathbf{i}_j \tag{1.15}$$

we also have

$$\mathbf{A} = \hat{A}_{IJ}\mathbf{e}_I\mathbf{e}_J \tag{1.16a}$$

where

$$\begin{aligned}\hat{A}_{IJ} \triangleq \mathbf{e}_I \cdot \mathbf{A}\mathbf{e}_J &= (\beta_{Ii}\mathbf{i}_i) \cdot (A_{jk}\mathbf{i}_j\mathbf{i}_k)(\beta_{Jl}\mathbf{i}_l) \\ &= \beta_{Ii}A_{jk}\beta_{Jl}\mathbf{i}_i \cdot (\mathbf{i}_j\mathbf{i}_k)\mathbf{i}_l \\ &= \beta_{Ii}A_{jk}\beta_{Jl}\underbrace{(\mathbf{i}_i \cdot \mathbf{i}_j)}_{\delta_{ij}}\underbrace{(\mathbf{i}_k \cdot \mathbf{i}_l)}_{\delta_{kl}} \\ &= \beta_{Ii}A_{ik}\beta_{Jk}\end{aligned} \tag{1.16b}$$

The transformation (1.16) is sometimes represented in *matrix* form as

$$\hat{\mathbf{A}} = \boldsymbol{\beta}\mathbf{A}\boldsymbol{\beta}^{\mathrm{T}} \tag{1.17}$$

by using the transpose of the matrix. A translation of equation (1.17) to FORTRAN is

```
      DO 100 IT = 1,3
      DO 100 JT = 1,3
      ATIJ = 0
      DO 102 I = 1,3
      DO 102 J = 1,3
  102 ATIJ = ATIJ + B(IT,I) * A(I,J) * B(JT,J)
  100 AT(IT,JT) = ATIJ
```
(1.18)

We shall now look at index notation for the gradient and divergence operators. In the (x, y, z) coordinate system, the gradient of a scalar function ϕ is given by

$$\nabla\phi = \text{grad } \phi \triangleq \mathbf{i}\frac{\partial\phi}{\partial x} + \mathbf{j}\frac{\partial\phi}{\partial y} + \mathbf{k}\frac{\partial\phi}{\partial z} \tag{1.19}$$

If we introduce the notation $\phi_{,i}$ for the partial derivative with respect to x_i, that is,

$$\phi_{,i} \triangleq \frac{\partial\phi}{\partial x_i} \tag{1.20}$$

then the gradient of ϕ becomes

$$\nabla\phi = \mathbf{i}_i\phi_{,i} \qquad \left(\text{i.e.,} \quad \nabla = \mathbf{i}_i\frac{\partial}{\partial x_i}\right) \tag{1.21}$$

The gradient of a vector $\mathbf{v} = v_i\mathbf{i}_i$ is a tensor $\nabla\mathbf{v}$ represented by

$$\nabla\mathbf{v} \triangleq v_{i,j}\mathbf{i}_j\mathbf{i}_i \tag{1.22}$$

The divergence of a vector $\mathbf{v} = v_x\mathbf{i} + v_y\mathbf{j} + v_z\mathbf{k}$ is defined as

$$\nabla \cdot \mathbf{v} = \text{div } \mathbf{v} = \frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} + \frac{\partial v_z}{\partial z} \tag{1.23}$$

Using index notation, we have

$$\begin{aligned}\nabla \cdot \mathbf{v} &= \left(\mathbf{i}_i \frac{\partial}{\partial x_i}\right) \cdot (v_j \mathbf{i}_j) \\ &= \frac{\partial v_j}{\partial x_i} \mathbf{i}_i \cdot \mathbf{i}_j = (v_{j,i})\delta_{ij} = v_{i,i}\end{aligned} \tag{1.24}$$

The divergence of a tensor $\mathbf{T} = T_{ij}\mathbf{i}_i\mathbf{i}_j$ is given by

$$\begin{aligned}\nabla \cdot \mathbf{T} = \operatorname{div} \mathbf{T} &= \left(\mathbf{i}_i \frac{\partial}{\partial x_i}\right) \cdot (T_{jk}\mathbf{i}_j\mathbf{i}_k) \\ &= T_{jk,i}\underbrace{\mathbf{i}_i \cdot (\mathbf{i}_j\mathbf{i}_k)}_{(\underbrace{\mathbf{i}_i \cdot \mathbf{i}_j}_{\delta_{ij}})\mathbf{i}_k} \\ &= T_{jk,j}\mathbf{i}_k\end{aligned} \tag{1.25}$$

In the usual notation, we have

$$\begin{aligned}\nabla \cdot \mathbf{T} = &\left(\frac{\partial T_{xx}}{\partial x} + \frac{\partial T_{yx}}{\partial y} + \frac{\partial T_{zx}}{\partial z}\right)\mathbf{i} \\ &+ \left(\frac{\partial T_{xy}}{\partial x} + \frac{\partial T_{yy}}{\partial y} + \frac{\partial T_{zy}}{\partial z}\right)\mathbf{j} \\ &+ \left(\frac{\partial T_{xz}}{\partial x} + \frac{\partial T_{yz}}{\partial y} + \frac{\partial T_{zz}}{\partial z}\right)\mathbf{k}\end{aligned} \tag{1.26}$$

Using the gradient and divergence operators, the Laplacian is given as

$$\begin{aligned}\Delta\phi \triangleq \nabla \cdot \nabla\phi &= \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2} \\ &= \left(\mathbf{i}_i \frac{\partial}{\partial x_i}\right) \cdot \left(\mathbf{i}_j \frac{\partial}{\partial x_j}\right)\phi \\ &= \phi_{,ij}\mathbf{i}_i \cdot \mathbf{i}_j = \phi_{,ii}\end{aligned} \tag{1.27}$$

More generally,

$$\operatorname{div}(\mathbf{k} \operatorname{grad} \phi) = \nabla \cdot (\mathbf{k}\, \nabla\phi) = (k_{ij}\phi_{,j})_{,i} \tag{1.28}$$

Another useful convention using index notation can be obtained from the permutation symbol

$$e_{ijk} = \begin{cases} 1 & \text{if } i, j, k \text{ is an even permutation of } 1, 2, 3 \\ -1 & \text{if } i, j, k \text{ is an odd permutation of } 1, 2, 3 \\ 0 & \text{otherwise} \end{cases} \tag{1.29}$$

More precisely, $e_{112} = 0$, $e_{231} = 1$, $e_{321} = -1$, $e_{132} = -1$, $e_{133} = 0$, $e_{312} = 1$, and others. If we use this symbol, the cross product $\mathbf{w} = \mathbf{u} \times \mathbf{v}$ of two vectors $\mathbf{u}$ and $\mathbf{v}$

can be written as

$$w_i = e_{ijk}u_j v_k \tag{1.30}$$

if $\mathbf{w} = w_i\mathbf{i}_i$, $\mathbf{u} = u_j\mathbf{i}_j$, and $\mathbf{v} = v_k\mathbf{i}_k$, since

$$\mathbf{w} = \mathbf{u} \times \mathbf{v} = \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ u_x & u_y & u_z \\ v_x & v_y & v_z \end{vmatrix} \tag{1.31}$$

Exercise 1.1: Suppose that the range of all indices is from 1 to 3 in the following.

1. Show that (a) $\delta_{ij}\delta_{ij} = 3$, (b) $e_{ijk}e_{kji} = -6$, (c) $e_{kki} = 0$, (d) $\delta_{ij}\delta_{jk} = \delta_{ik}$, and (e) $e_{ijk}A_jA_k = 0$.
2. If $b_i = a_i/\sqrt{a_j a_j}$, show that $\mathbf{b} = b_i\mathbf{i}_i$ is a unit vector.
3. Use index notation to prove that
$$\mathbf{a} \cdot (\mathbf{b} \times \mathbf{c}) = (\mathbf{a} \times \mathbf{b}) \cdot \mathbf{c}$$
4. Using the definition of a determinant, show that
$$\det \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = e_{ijk}a_{1i}a_{2j}a_{3k}$$
5. Show that (a) $e_{ijk}e_{imn} = \delta_{jm}\delta_{kn} - \delta_{jn}\delta_{km}$, (b) $e_{ijk}e_{ijn} = 2\delta_{kn}$, and
(c) $$\det \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix} = \tfrac{1}{6}e_{ijk}e_{lmn}a_{il}a_{jm}a_{kn} \tag{1.32}$$
6. Develop a FORTRAN program to normalize the vector $\mathbf{a} = a_i\mathbf{i}_i$ whose components are stored in the one-dimensional array $A(I)$, $I = 1, \ldots, N$.
7. Suppose that a 3×3 matrix array $A(I, J)$, $I, J = 1, 2, 3$, is given. Develop a FORTRAN program to compute its determinant.

1.2 Gauss elimination method for solving a system of linear equations

In the subsequent chapters, we shall solve the system of linear equations obtained by finite element approximations of problems in mechanics. Roughly speaking, a finite element method is a process by which a continuous problem in mechanics is reduced to a discrete problem, whose solution leads to a system of linear equations symbolically represented by

$$\mathbf{Ku} = \mathbf{f} \qquad \text{or} \qquad K_{ij}u_j = f_i \tag{1.33}$$

where $\mathbf{K}$ is the stiffness matrix, $\mathbf{u}$ the generalized displacement vector, and $\mathbf{f}$ the generalized load vector. Thus, in order to obtain the generalized displacement $\mathbf{u}$, we have to solve the system of linear equation (1.33).

One of the methods used to solve (1.33) is the Gauss elimination method discussed below. Suppose that we are given the system of linear equations

$$a_{ij}x_j = b_i, \quad i = 1, \ldots, N \tag{1.34a}$$

that is,

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1N}x_N &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2N}x_N &= b_2 \\ &\vdots \\ a_{N1}x_1 + a_{N2}x_2 + a_{N3}x_3 + \cdots + a_{NN}x_N &= b_N \end{aligned} \tag{1.34b}$$

If matrix notation is used, (1.34a) and (1.34b) are also expressed by the form

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} & \cdots & a_{1N} \\ a_{21} & a_{22} & a_{23} & \cdots & a_{2N} \\ & & \vdots & & \\ a_{N1} & a_{N2} & a_{N3} & \cdots & a_{NN} \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{Bmatrix} = \begin{Bmatrix} b_1 \\ b_2 \\ \vdots \\ b_N \end{Bmatrix} \tag{1.34c}$$

A standard Gaussian elimination process is divided into two parts, the forward elimination and the back substitution. We shall describe in detail the forward elimination for the first two steps and shall generalize the forward elimination process using index notation.

The first step is to eliminate the terms $a_{21}x_1, a_{31}x_1, \ldots, a_{N1}x_1$ from the system of linear equations (1.34) as indicated below:

$$\begin{aligned} & a_{11}x_1 + && a_{12}x_2 + && a_{13}x_3 + \cdots + && a_{1N}x_N = b_1 \\ & \left(a_{21} - \frac{a_{21}}{a_{11}}a_{11}\right)x_1 + && \left(a_{22} - \frac{a_{21}}{a_{11}}a_{12}\right)x_2 + && \left(a_{23} - \frac{a_{21}}{a_{11}}a_{13}\right)x_3 + \cdots + && \left(a_{2N} - \frac{a_{21}}{a_{11}}a_{1N}\right)x_N = b_2 - \frac{a_{21}}{a_{11}}b_1 \\ & \left(a_{31} - \frac{a_{31}}{a_{11}}a_{11}\right)x_1 + && \left(a_{32} - \frac{a_{31}}{a_{11}}a_{12}\right)x_2 + && \left(a_{33} - \frac{a_{31}}{a_{11}}a_{13}\right)x_3 + \cdots + && \left(a_{3N} - \frac{a_{31}}{a_{11}}a_{1N}\right)x_N = b_3 - \frac{a_{31}}{a_{11}}b_1 \\ & && && && \vdots \\ & \left(a_{N1} - \frac{a_{N1}}{a_{11}}a_{11}\right)x_1 + && \left(a_{N2} - \frac{a_{N1}}{a_{11}}a_{12}\right)x_2 + && \left(a_{N3} - \frac{a_{N1}}{a_{11}}a_{13}\right)x_3 + \cdots + && \left(a_{NN} - \frac{a_{N1}}{a_{11}}a_{1N}\right)x_N = b_N - \frac{a_{N1}}{a_{11}}b_1 \end{aligned}$$

Denoting the coefficient of the new equations by $\tilde{a}_{ij}$ where

$$\tilde{a}_{11} = a_{11}, \qquad \tilde{a}_{12} = a_{12}$$

$$\tilde{a}_{22} = a_{22} - \frac{a_{21}}{a_{11}}a_{12}, \qquad \tilde{a}_{23} = a_{23} - \frac{a_{21}}{a_{11}}a_{13}$$

$$\tilde{b}_2 = b_2 - \frac{a_{21}}{a_{11}}b_1, \text{ etc.}$$

we have

$$\begin{aligned} \tilde{a}_{11}x_1 + \tilde{a}_{12}x_2 + \tilde{a}_{13}x_3 + \cdots + \tilde{a}_{1N}x_N &= \tilde{b}_1 \\ \tilde{a}_{22}x_2 + \tilde{a}_{23}x_3 + \cdots + \tilde{a}_{2N}x_N &= \tilde{b}_2 \\ \tilde{a}_{32}x_2 + \tilde{a}_{33}x_3 + \cdots + \tilde{a}_{3N}x_N &= \tilde{b}_3 \\ &\vdots \\ \tilde{a}_{N2}x_2 + \tilde{a}_{N3}x_3 + \cdots + \tilde{a}_{NN}x_N &= \tilde{b}_N \end{aligned}$$

The second step is to eliminate the terms $\tilde{a}_{32}x_2, \tilde{a}_{42}x_2, \ldots, \tilde{a}_{N2}x_2$ using the second equation as follows:

$$\begin{aligned}
&\tilde{a}_{11}x_1 + \quad \tilde{a}_{12}x_2 + \quad \tilde{a}_{13}x_3 + \cdots + \quad \tilde{a}_{1N}x_N = \tilde{b}_1 \\
&\qquad\qquad \tilde{a}_{22}x_2 + \quad \tilde{a}_{23}x_3 + \cdots + \quad \tilde{a}_{2N}x_N = \tilde{b}_2 \\
&\left(\tilde{a}_{32} - \frac{\tilde{a}_{32}}{\tilde{a}_{22}}\tilde{a}_{22}\right)x_2 + \left(\tilde{a}_{33} - \frac{\tilde{a}_{32}}{\tilde{a}_{22}}\tilde{a}_{23}\right)x_3 + \cdots + \left(\tilde{a}_{3N} - \frac{\tilde{a}_{32}}{\tilde{a}_{22}}\tilde{a}_{2N}\right)x_N = \tilde{b}_3 - \frac{\tilde{a}_{32}}{\tilde{a}_{22}}\tilde{b}_2 \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\left(\tilde{a}_{N2} - \frac{\tilde{a}_{N2}}{\tilde{a}_{22}}\tilde{a}_{22}\right)x_2 + \left(\tilde{a}_{N3} - \frac{\tilde{a}_{N2}}{\tilde{a}_{22}}\tilde{a}_{23}\right)x_3 + \cdots + \left(\tilde{a}_{NN} - \frac{\tilde{a}_{N2}}{\tilde{a}_{22}}\tilde{a}_{2N}\right)x_N = \tilde{b}_N - \frac{\tilde{a}_{N2}}{\tilde{a}_{22}}\tilde{b}_2
\end{aligned}$$

Continuing the above two steps, we can generate the forward elimination procedure for the kth step:

$$\tilde{a}_{ij} = \tilde{a}_{ij} - \frac{\tilde{a}_{ik}}{\tilde{a}_{kk}}\tilde{a}_{kj} \quad \text{(no sum on } k\text{)}$$

$$i = k+1, k+2, \ldots, N \quad j = k+1, k+2, \ldots, N \tag{1.35}$$

$$\tilde{b}_i = \tilde{b}_i - \frac{\tilde{a}_{ik}}{\tilde{a}_{kk}}\tilde{b}_k \quad \text{(no sum on } k\text{)}$$

$$i = k+1, k+2, \ldots, N \tag{1.36}$$

for given $k = 1, 2, \ldots, N-1$. The above index expressions suggest a FORTRAN program for the forward elimination by the Gauss method:

```
C
C     (FORWARD ELIMINATION)
C
      N1 = N - 1
      DO 100 K = 1,N1
      K1 = K + 1
      DO 102 L = K,N
  102 C(L) = A(K,L)
      AKK = 1./C(K)
      BK = B(K)
      DO 108 I = K1,N
      AIK = A(I,K) * AKK
      B(I) = B(I) - AIK * BK
      DO 108 J = K,N
  108 A(I,J) = A(I,J) - AIK * C(J)
C
      WRITE(6,600) K
  600 FORMAT(///10X, 'STEP = ',I3,/)
      WRITE(6,602) ((A(I,J),J = 1,4),B(I),I = 1,N)
  602 FORMAT(4(E10.3,1X),5X,E10.3)
  100 CONTINUE
```

(1.37)

* These four steps of the program are prepared only for the purpose of checking if the program for the forward elimination is working correctly.

A routine for the back substitution can be obtained by

$$\tilde{a}_{kk}x_k + \tilde{a}_{kk+1}x_{k+1} + \cdots + \tilde{a}_{kN}x_N = \tilde{b}_k$$

that is,

$$x_k = \left(\tilde{b}_k - \sum_{j=k+1}^{N} \tilde{a}_{kj}x_j\right) \Big/ \tilde{a}_{kk} \quad \text{(no sum over } k\text{)} \tag{1.38}$$

This can be carried out by the program

```
C
C     (BACK SUBSTITUTION)
C
      K = N
      B(K) = B(K)/A(K,K)
  104 K = K - 1
      IF(K.LE.0) RETURN
      K1 = K + 1
      SUM = 0.
      DO 106 J = K1,N
  106 SUM = SUM + A(K,J) * B(J)
      B(K) = (B(K) - SUM)/A(K,K)
      GOTO 104
```
(1.39)

We now present an example illustrating the above two routines.

$$\begin{bmatrix} 0.200\text{E}+01 & 0.300\text{E}+01 & -0.100\text{E}+01 & 0.500\text{E}+01 \\ 0.400\text{E}+01 & 0.400\text{E}+01 & -0.300\text{E}+01 & 0.300\text{E}+01 \\ -0.200\text{E}+01 & 0.300\text{E}+01 & -0.100\text{E}+01 & 0.100\text{E}+01 \\ -0.300\text{E}+01 & 0.200\text{E}+01 & -0.100\text{E}+01 & 0.500\text{E}+01 \end{bmatrix} \begin{Bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{Bmatrix} = \begin{Bmatrix} 0.150\text{E}+02 \\ 0.100\text{E}+02 \\ -0.500\text{E}+01 \\ -0.100\text{E}+01 \end{Bmatrix}$$

Since the number of equations is 4, three steps are necessary in the forward elimination as shown in Table 1.1. Then the back substitution yields the solution

1	3.00000
2	1.00000
3	4.00000
4	2.00000

Exercise 1.2: It is inconvenient to transfer two-dimensional arrays, such as the coefficient matrix **A** in the above example, from one subroutine to another. Modify the above programs so that the coefficient matrix **A** is stored in a one-dimensional manner as

$$\mathbf{A} = (a_{11}, a_{12}, a_{13}, \ldots, a_{1n}, a_{21}, a_{22}, \ldots, a_{2n}, \ldots, a_{n1}, a_{n2}, \ldots, a_{nn})$$

Exercise 1.3: If the property of symmetry $a_{ij} = a_{ji}$ is assumed in a system of linear equations, it is possible to save storage space, that is, the array for the coefficient matrix **A**. Modify the programs to accomplish this.

Table 1.1

STEP = 1				
0.200E + 01	0.300E + 01	−0.100E + 01	0.500E + 01	0.150E + 02
0.0	−0.200E + 01	−0.100E + 01	−0.700E + 01	−0.200E + 02
0.0	0.600E + 01	−0.200E + 01	0.600E + 01	0.100E + 02
0.0	0.650E + 01	−0.250E + 01	0.125E + 02	0.215E + 02
STEP = 2				
0.200E + 01	0.300E + 01	−0.100E + 01	0.500E + 01	0.150E + 02
0.0	−0.200E + 01	−0.100E + 01	−0.700E + 01	−0.200E + 02
0.0	−0.0	−0.500E + 01	−0.150E + 02	−0.500E + 02
0.0	−0.0	−0.575E + 01	−0.103E + 02	−0.435E + 02
STEP = 3				
0.200E + 01	0.300E + 01	−0.100E + 01	0.500E + 01	0.150E + 02
0.0	−0.200E + 01	−0.100E + 01	−0.700E + 01	−0.200E + 02
0.0	−0.0	−0.500E + 01	−0.150E + 02	−0.500E + 02
0.0	−0.0	−0.191E − 05	0.700E + 01	0.140E + 02

Exercise 1.4: If the coefficient matrix is banded – that is, if

$$a_{ij} = \begin{cases} a_{ij} & \text{if } |j-i| \le M \\ 0 & \text{if } |j-i| > M \end{cases} \qquad M < n$$

we need not compute and store the zero elements. Modify the programs in order to exploit this property.

Exercise 1.5: Develop a BASIC program that is equivalent to the programs (1.37) and (1.39).

1.3 Jacobi's method for solving an eigenvalue problem

Another typical discrete form obtained by finite element approximations is the eigenvalue problem

$$\mathbf{Ku} = \lambda \mathbf{Mu} \quad \text{or} \quad K_{ij}u_j = \lambda M_{ij}u_j \tag{1.40}$$

where **M** is called the mass matrix in the area of finite element methods. In this case, the problem is to find λ and **u** satisfying (1.40). If the matrix is invertible, then (1.40) becomes

$$\mathbf{M}^{-1}\mathbf{Ku} = \lambda \mathbf{u} \qquad \text{or} \qquad M^{-1}_{ik}K_{kj}u_j = \lambda u_i \tag{1.41}$$

If the matrix $\mathbf{S} = \mathbf{M}^{-1}\mathbf{K}$ is a symmetric $N \times N$ matrix (in most vibration problems it is not!), there exist N pairs of solutions to the eigenvalue problem (1.41); that is, there exist the solutions $(\lambda_1, \mathbf{u}_1), (\lambda_2, \mathbf{u}_2), \ldots, (\lambda_n, \mathbf{u}_n)$ to (1.41). One method for finding the solutions to (1.41), that is,

$$\mathbf{Su} = \lambda \mathbf{u} \qquad \text{or} \qquad S_{ij}u_j = \lambda u_i \tag{1.42}$$

is the Jacobi method, which is based on the similarity transformation of a square matrix.

Suppose that $\mathbf{P}$ is an orthogonal matrix such that $\det \mathbf{P} = 1$ and $\mathbf{P}^{-1} = \mathbf{P}^{\mathrm{T}}$, where $\mathbf{P}^{\mathrm{T}}$ is the transpose of the matrix $\mathbf{P}$. Then the similarity transformation of $\mathbf{S}$ by $\mathbf{P}$ is given by

$$\hat{\mathbf{S}} = \mathbf{P}^{\mathrm{T}}\mathbf{S}\mathbf{P} \qquad \text{or} \qquad \hat{S}_{ij} = P_{ki}S_{kl}P_{lj} \tag{1.43}$$

Suppose that λ and $\mathbf{u}$ are an eigenvalue and the corresponding eigenvector of $\mathbf{S}$, that is, λ and $\mathbf{u}$ satisfy the system (1.42). Premultiplying by $\mathbf{P}^{\mathrm{T}}$ yields

$$\mathbf{P}^{\mathrm{T}}\mathbf{S}\mathbf{u} = \lambda\mathbf{P}^{\mathrm{T}}\mathbf{u}$$

Define

$$\mathbf{v} = \mathbf{P}^{\mathrm{T}}\mathbf{u}, \qquad \text{i.e.,} \qquad \mathbf{u} = \mathbf{P}\mathbf{v} \tag{1.44}$$

Then we have

$$\mathbf{P}^{\mathrm{T}}\mathbf{S}\mathbf{P}\mathbf{v} = \lambda\mathbf{v}, \qquad \text{i.e.,} \quad \hat{\mathbf{S}}\mathbf{v} = \lambda\mathbf{v} \tag{1.45}$$

This means that if $(\lambda, \mathbf{u})$ is a pair of eigenvalue and eigenvector of $\mathbf{S}$, then $(\lambda, \mathbf{v})$ is a pair of eigenvalue and eigenvector of $\hat{\mathbf{S}}$. The converse is also true. Therefore, solving problem (1.42) is equivalent to solving problem (1.45).

The Jacobi method is based on the above similarity transformation of a matrix and the idea that one of the off-diagonal terms of the new matrix can be forced to be zero after applying the transformation (as will be seen below). For example, if S_{ij}, $i \neq j$, is the element of $\mathbf{S}$ whose magnitude is the largest among S_{kl}, $k < l$, $l = 1, 2, \ldots, N$, the $\hat{S}_{ij}$ can be made zero by choosing properly the orthogonal matrix $\mathbf{P}$. Repeating this process until all off-diagonal terms become zero, we have

$$\mathbf{S}^*\mathbf{u}^* = \lambda\mathbf{u}^* \tag{1.46}$$

where

$$\begin{aligned} \mathbf{S}^* &= \cdots \mathbf{P}_3^{\mathrm{T}}\mathbf{P}_2^{\mathrm{T}}\mathbf{P}_1^{\mathrm{T}}\mathbf{S}\mathbf{P}_1\mathbf{P}_2\mathbf{P}_3 \cdots \\ \mathbf{u}^* &= \cdots \mathbf{P}_3^{\mathrm{T}}\mathbf{P}_2^{\mathrm{T}}\mathbf{P}_1^{\mathrm{T}}\mathbf{u} \triangleq \mathbf{T}^{\mathrm{T}}\mathbf{u} \end{aligned} \tag{1.47}$$

and $\mathbf{P}_i^{\mathrm{T}}$ is the orthogonal matrix used at the ith process of forcing an off-diagonal term to be zero. Since $\mathbf{S}^*$ is supposed to be a diagonal matrix, its diagonal elements are then the eigenvalues of $\mathbf{S}$. Using (1.47), the corresponding eigenvectors are obtained from

$$\mathbf{u} = \mathbf{P}_1\mathbf{P}_2\mathbf{P}_3 \cdots \mathbf{u}^* = \mathbf{T}\mathbf{u}^* \tag{1.48}$$

In general, there is no guarantee that the matrix $\mathbf{S}^*$ will be obtained after a finite number of applications of the similarity transformation. However, we shall assume this is obtainable for the matrix $\mathbf{S}$ considered here.

Let us denote by $\mathbf{P}^{(ij)}$ the orthogonal matrix that yields $\hat{S}_{ij} = 0$ by the similarity transformation of the matrix $\mathbf{S}$, where the indices i and j are defined so that the absolute value of S_{ij} is the largest among the off-diagonal terms S_{kl}, $k < l$, $l = 1, 2, \ldots, N$. To do this, we seek the angle ϕ to be used in the orthogonal matrix

$\mathbf{P}^{(ij)}$, whose elements are given by

$$[P_{kl}^{(ij)}] = \begin{bmatrix} & & & i & & j & \\ 1 & & & & & & \\ & 1 & & & & & \\ \cdots & & \cdots & \cos\phi & \cdots & \sin\phi & \cdots \; i \\ & & & \vdots & & \vdots & \\ \cdots & & \cdots & -\sin\phi & \cdots & \cos\phi & \cdots \; j \\ & & & & & & 1 \end{bmatrix} \tag{1.49}$$

that is,

$$\begin{aligned} &P_{ii}^{(ij)} = P_{jj}^{(ij)} = \cos\phi, && P_{ij}^{(ij)} = -P_{ji}^{(ij)} = \sin\phi \quad \text{(no sum)} \\ &P_{kl}^{(ij)} = 0, \quad k \neq i, \quad l \neq j, && P_{kk}^{(ij)} = 1, \quad k \neq i, \quad k \neq j \end{aligned} \tag{1.50}$$

Using this rotation matrix, the matrix $[S_{ij}]$ becomes $[\hat{S}_{kl}]$;

$$\hat{S}_{kl} = P_{mk}^{(ij)} S_{mn} P_{nl}^{(ij)} \tag{1.51}$$

As a result of the rotation, we want

$$\hat{S}_{ij} = 0 \tag{1.52}$$

Hence, the angle ϕ is determined by $\hat{S}_{ij} = 0$.

$$\hat{S}_{ij} = \hat{S}_{ji} = \tfrac{1}{2}(S_{ii} - S_{jj}) \sin 2\phi + S_{ij} \cos 2\phi = 0$$

that is,

$$\tan 2\phi = \frac{-2S_{ij}}{S_{ii} - S_{jj}}, \qquad S_{ii} \neq S_{jj} \quad \text{(no sum)} \tag{1.53}$$

Once the angle ϕ has been determined, $\hat{S}_{kl}$ in (1.51) is obtained as

$$\begin{aligned} &\hat{S}_{kl} = S_{kl}, \quad k, l \neq i, j \\ &\hat{S}_{ik} = \hat{S}_{ki} = S_{ik} \cos\phi - S_{jk} \sin\phi, \quad k \neq i, j \\ &\hat{S}_{jk} = \hat{S}_{kj} = S_{ik} \sin\phi + S_{jk} \cos\phi, \quad k \neq i, j \\ &\hat{S}_{ii} = \tfrac{1}{2}(S_{ii} + S_{jj}) + \tfrac{1}{2}(S_{ii} - S_{jj}) \cos 2\phi - S_{ij} \sin 2\phi \\ &\hat{S}_{jj} = \tfrac{1}{2}(S_{ii} + S_{jj}) + \tfrac{1}{2}(S_{ii} - S_{jj}) \cos 2\phi + S_{ij} \sin 2\phi \\ &\qquad \text{(no sum)} \end{aligned} \tag{1.54}$$

Convergence of the iterations may be checked by the quantity $|S_{ij}|$; that is, if

$$|S_{ij}| < \text{TOLE} \tag{1.55}$$

is satisfied for a given tolerance TOLE, the process of applying similarity transformations will be terminated. Thus, after a certain number of applications of the transformation, the matrix **S** is transformed to a diagonal matrix **S*** within errors whose order of magnitude is less than the given tolerance.

Let us now write a program of Jacobi's method to find the eigenvalues and eigenvectors for a given symmetric matrix. A flowchart for the program is shown in Figure 1.3. The program shown below is written in the BASIC language of the system of IBM PC-XT.

```
1000 PRINT "********************************"
1010 PRINT "   JACOBI'S METHOD     (S)   "
1020 PRINT "********************************"
1030 PRINT : PRINT :PRINT
1040 INPUT "Size of the symmetric matrix       NX = ";NX
1050 DIM S(NX,NX),T(NX,NX)
1060 PRINT : PRINT "SYMMETRIC MATRIX  S " : PRINT
1070 FOR I=1 TO NX : FOR J=I TO NX : PRINT "   S(";I;",";J;") = ";: INPUT S(I,J) :
     S(J,I)=S(I,J) : NEXT J : NEXT I
1080 GOSUB 1100
1090 GOTO 1580
1100 REM  Subroutine Jacobi's Method  ( Standard Eigenvalue Problems  )
1110 REM ------------------------------------------------------------------------
1120 PRINT
1130 INPUT "Tolerance   EP = ";EP
1140 PRINT :
     PRINT "ITERATION PROCESS ..........................................." : PRINT
1150 FOR I=1 TO NX : FOR J=1 TO NX : T(I,J)=0 : T(J,I)=0 : NEXT J : T(I,I)=1 : NEXT I
1160 N=0 : NP=1
1170 N=N+1
1180 REM  < FIND THE MAXIMUM ABSOLUTE VALUE  >
1190 SM=0
1200 FOR I=1 TO NX-1 : FOR J=I+1 TO NX : IF ABS(S(I,J))<SM THEN GOTO 1220
1210 SM=ABS(S(I,J)) : IM=I : JM=J
1220 NEXT J  : NEXT I
1230 REM  < FIND THE TWICE VALUE OF THE ANGLE  >
1240 SI=S(IM,IM) : SJ=S(JM,JM) : SK=S(IM,JM)
1250 IF SI=SJ THEN GOTO 1280
1260 SL=-2*SK/(SI-SJ)   :  T2=ATN(SL)
1270 GOTO 1310
1280 IF SK>0 THEN T2=-3.141592654#/2
1290 IF SK=0 THEN T2=0
1300 IF SK<0 THEN T2=3.141592654#/2
1310 T1=.5*T2
1320 C1=COS(T1)   :  SI= SIN(T1)   :  C2=COS(T2)   :  S2=SIN(T2)
1330 REM  <  MODIFY THE MATRIX S  >
1340 FOR K=1 TO NX  :  SY=S(IM,K)   :  SZ=S(JM,K)
1350 S3=SY*C1-SZ*SI   :  S(IM,K)=S3   :  S(K,IM)=S3
1360 S4=SY*SI+SZ*C1   :  S(JM,K)=S4   :  S(K,JM)=S4
1370 NEXT K
1380 S(IM,IM)=.5*((SI+SJ)+(SI-SJ)*C2)-SK*S2
1390 S(JM,JM)=.5*((SI+SJ)-(SI-SJ)*C2)+SK*S2
1400 S(IM,JM)=0   :  S(JM,IM)=0
1410 REM   <  MODIFY  T  >
1420 FOR K=1 TO NX  :  T3=T(K,IM)   :  T4=T(K,JM)
1430 T(K,IM)=T3*C1-T4*SI   :  T(K,JM)=T3*SI+T4*C1
1440 NEXT K
1450 IF N<NP THEN GOTO 1480
1460 PRINT N;TAB(10);SM
1470 NP=NP+10
1480 IF SM>EP THEN GOTO 1170
1490 PRINT :
     PRINT "RESULTS/EIGENVALUES & EIGENVECTORS ........................" : PRINT
1500 PRINT "NUMBER OF ITERATION = ";N
1510 PRINT "TOLERANCE  = ";SM
1520 PRINT :
     INPUT "Which eigenvalue and eigenvector will be output? (1,....,NX) = ";I
1530 IF I<=0 OR I>NX THEN GOTO 1570
1540 PRINT : PRINT " EIGENVALUE = ";S(I,I) : PRINT : PRINT "EIGENVECTOR" :  PRINT
1550 FOR J=1 TO NX : PRINT J;TAB(10);T(J,I) : NEXT J
1560 GOTO 1520
1570 RETURN
1580 END
```

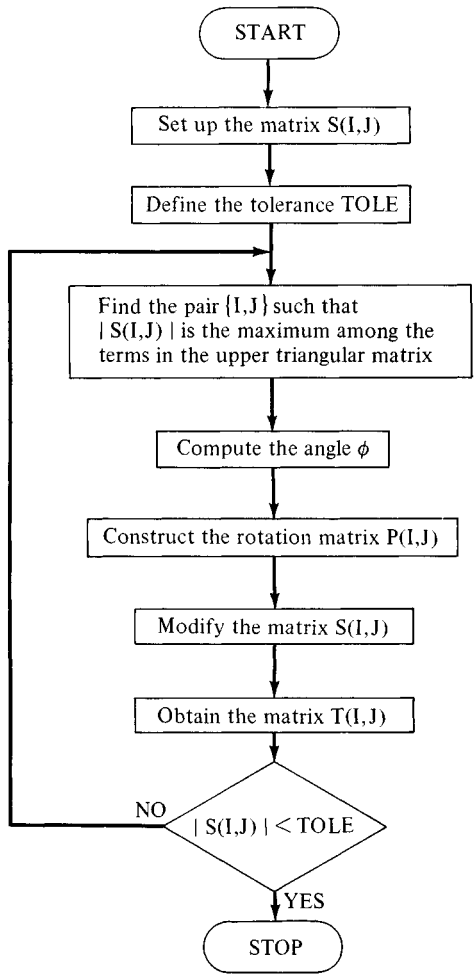

Figure 1.3 Flowchart of JACOBI.BAS.

Exercise 1.6: Write a program of Jacobi's method to obtain eigenvalues and eigenvectors by using the FORTRAN language.

1.4 Variational methods

We shall briefly review variational methods to solve an algebraic and a boundary value problem. For simplicity, we first study the problem: Find a vector $\mathbf{u} = u_i\mathbf{i}_i$ satisfying a system of linear equations

$$\mathbf{Ku} = \mathbf{f} \quad \text{(i.e.,} \quad K_{ij}u_j = f_i) \tag{1.56}$$

for a given vector $\mathbf{f} = f_i\mathbf{i}_i$ and a symmetric matrix $\mathbf{K} = K_{ij}\mathbf{i}_i\mathbf{i}_j$ such that

$$K_{ij} = K_{ji}, \qquad i, j = 1, 2, \ldots, N \tag{1.57}$$

Using the symmetry of $\mathbf{K}$, it can be shown that if $\mathbf{u}$ is a *minimizer* of a functional

$$F(\mathbf{v}) = \tfrac{1}{2}\mathbf{v} \cdot \mathbf{K}\mathbf{v} - \mathbf{v} \cdot \mathbf{f} \tag{1.58}$$

among all vectors $\mathbf{v} \in \mathbb{R}^n$, that is, if $\mathbf{u}$ satisfies

$$F(\mathbf{v}) \geq F(\mathbf{u}), \quad \forall \mathbf{v} \in \mathbb{R}^n \tag{1.59}$$

then $\mathbf{u}$ is also a solution of equation (1.56). Here $\mathbf{v} \in \mathbb{R}^n$ means "a vector $\mathbf{v}$ is an element of an N-dimensional Euclidean space $\mathbb{R}^n$," and $\forall$ means "for every." A functional is a special function whose range is a scalar field such as a real line $\mathbb{R}$. We now show that (1.59) yields (1.56). Taking $\mathbf{v} = \mathbf{u} + \varepsilon\mathbf{w}$ in (1.59) for an arbitrary vector $\mathbf{w}$ and a positive number $\varepsilon > 0$, we have

$$F(\mathbf{u} + \varepsilon\mathbf{w}) \geq F(\mathbf{u}), \quad \forall \mathbf{w} \in \mathbb{R}^n$$

Expanding the left side using (1.58) yields

$$\varepsilon\{\tfrac{1}{2}(\mathbf{w} \cdot \mathbf{K}\mathbf{u} + \mathbf{u} \cdot \mathbf{K}\mathbf{w}) - \mathbf{w} \cdot \mathbf{f}\} + \tfrac{1}{2}\varepsilon^2 \mathbf{w} \cdot \mathbf{K}\mathbf{w} \geq 0$$

Dividing by $\varepsilon > 0$ and passing through the limit $\varepsilon \to 0$, we have

$$\tfrac{1}{2}(\mathbf{w} \cdot \mathbf{K}\mathbf{u} + \mathbf{u} \cdot \mathbf{K}\mathbf{w}) - \mathbf{w} \cdot \mathbf{f} \geq 0, \quad \forall \mathbf{w} \in \mathbb{R}^n$$

Using the symmetry of $\mathbf{K}$, this implies

$$\mathbf{w} \cdot (\mathbf{K}\mathbf{u} - \mathbf{f}) \geq 0, \quad \forall \mathbf{w} \in \mathbb{R}^n \tag{1.60}$$

Since $\mathbf{w}$ is an arbitrary vector in $\mathbb{R}^n$, (1.60) has to be satisfied for the choice $-\mathbf{w}$ instead of $\mathbf{w}$. That is,

$$-\mathbf{w} \cdot (\mathbf{K}\mathbf{u} - \mathbf{f}) \geq 0, \quad \forall \mathbf{w} \in \mathbb{R}^n \tag{1.61}$$

Inequalities (1.60) and (1.61) yield

$$\mathbf{w} \cdot (\mathbf{K}\mathbf{u} - \mathbf{f}) = 0, \quad \forall \mathbf{w} \in \mathbb{R}^n \tag{1.62}$$

That is, $\mathbf{u}$ has to be a solution of (1.56):

$$\mathbf{K}\mathbf{u} - \mathbf{f} = 0$$

On the other hand, if an additional condition

$$\mathbf{w} \cdot \mathbf{K}\mathbf{w} \geq 0, \quad \forall \mathbf{w} \in \mathbb{R}^n \tag{1.63}$$

is imposed, the inverse relation to the above can be verified. Indeed, suppose that $\mathbf{u}$ is a solution of (1.56). Then applying the relation

$$F(\mathbf{v}) - F(\mathbf{u}) = (\mathbf{v} - \mathbf{u}) \cdot (\mathbf{K}\mathbf{u} - \mathbf{f}) + \tfrac{1}{2}(\mathbf{v} - \mathbf{u}) \cdot \mathbf{K}(\mathbf{v} - \mathbf{u})$$

the nonnegativeness condition (1.63) of the matrix $\mathbf{K}$ yields inequality (1.59):

$$F(\mathbf{v}) - F(\mathbf{u}) \geq 0, \quad \forall \mathbf{v} \in \mathbb{R}^n$$

Therefore, under the condition

$$K_{ij} = K_{ji} \qquad \text{and} \qquad \mathbf{w} \cdot \mathbf{K}\mathbf{w} \geq 0, \quad \forall \mathbf{w} \in \mathbb{R}^n \tag{1.64}$$

the following three are *equivalent* to each other:

$$\begin{aligned} &\text{(P1)} && \mathbf{Ku} - \mathbf{f} = 0 \\ &\text{(P2)} && \mathbf{w} \cdot (\mathbf{Ku} - \mathbf{f}) = 0, \quad \forall \mathbf{w} \in \mathbb{R}^n \\ &\text{(P3)} && F(\mathbf{v}) - F(\mathbf{u}) \geq 0, \quad \forall \mathbf{v} \in \mathbb{R}^n \end{aligned} \tag{1.65}$$

where F is the functional defined by (1.58). Very roughly speaking, (P1), (P2), and (P3) correspond to the equilibrium equation, the principle of virtual work, and the principle of minimum potential energy, respectively, in mechanics.

Let us extend the above to the boundary value problem:

$$-\frac{d}{dx} k \frac{du}{dx} = f \text{ in } (0, 1), \qquad u(0) = g, \qquad k \frac{du}{dx}(1) = h \tag{1.66}$$

Since the differential equation is defined on an interval (0, 1), we can obtain a functional F through integrating some quantity over the interval,

$$F(v) = \tfrac{1}{2} \int_0^1 k \left(\frac{dv}{dx} \right)^2 dx - \int_0^1 fv \, dx - hv(1) \tag{1.67}$$

The manner of defining a functional F for a given boundary value problem (1.66) is that the boundary value problem (1.66) is obtained as the *Euler equation* of the functional F as follows. Suppose that u is a minimizer of F such that

$$u(0) = g, \qquad F(v) \geq F(u), \quad \forall v \text{ with } v(0) = g \tag{1.68}$$

In other words, u is a minimizer of F among the functions defined on the interval (0, 1) that satisfy the "essential" boundary condition $v(0) = g$ at $x = 0$. Then, for every w such that $w(0) = 0$, taking $v = u \pm \varepsilon w$ in (1.68) yields

$$F(u \pm \varepsilon w) - F(u) \geq 0$$

which reduces to

$$\pm \varepsilon \left\{ \int_0^1 \left(k \frac{du}{dx} \frac{dw}{dx} - fw \right) dx - hw(1) \right\} + \tfrac{1}{2} \varepsilon^2 \int_0^1 k \left(\frac{dw}{dx} \right)^2 dx \geq 0$$

Dividing by $\varepsilon > 0$ and passing to the limit $\varepsilon \to 0$, we have

$$\int_0^1 \left(k \frac{du}{dx} \frac{dw}{dx} - fw \right) dx - hw(1) = 0 \tag{1.69}$$

for every w such that $w(0) = 0$. It it clear that (1.69) is exactly the same as the first variation of the functional F by the variation $\delta u = w$ from the minimizer u.* Indeed, the first variation of F is

$$\delta F(u) = \int_0^1 k \frac{du}{dx} \frac{d(\delta u)}{dx} dx - \int_0^1 f \, \delta u \, dx - h \, \delta u(1)$$

* The reader should note that for contact problems involving inequality constraints the current approach is applicable, whereas the traditional "See approach" is not.

Now, the Euler equation follows from the application of integration by parts in (1.69):

$$\int_0^1 \left[-\frac{d}{dx}\left(k\frac{du}{dx}\right) - f \right] w\,dx + k\frac{du}{dx}(1)w(1) - hw(1) = 0 \tag{1.70}$$

since $w(0) = 0$. Noting that w and $w(1)$ are arbitrary, equation (1.70) yields

$$-\frac{d}{dx}\left(k\frac{du}{dx}\right) = f \text{ in } (0, 1), \qquad k\frac{du}{dx}(1) = h \tag{1.71}$$

Since the boundary condition at $x = 1$ for the first derivative of the minimizer u is obtained as a part of the Euler equation, this is called a "natural" boundary condition. If the integrand of a functional F consists of the function v and its derivatives up to mth order, applying integration by parts m times yields boundary terms involving derivatives of $m, \ldots, (2m-1)$th order. Then we can define natural boundary conditions for such boundary terms derived from the process of integration by parts. On the other hand, if boundary conditions are written in terms of derivatives of $0, \ldots, (m-1)$th order, these are essential boundary conditions.

In the above, we have shown that the minimizer u of the minimization problem (1.68) is also a solution of the boundary value problem (1.66). As for the discrete system (1.56), it can be shown that a solution u of (1.66) is also a minimizer of the functional F under the condition

$$k \geq 0 \text{ in } (0, 1) \tag{1.72}$$

To see this, it suffices to note the following relation:

$$\begin{aligned} F(v) - F(u) = &\int_0^1 \left[k\frac{du}{dx}\frac{d}{dx}(v-u) - f(v-u) \right] dx \\ &- h(v-u)(1) + \tfrac{1}{2}\int_0^1 k\left[\frac{d}{dx}(v-u)\right]^2 dx \end{aligned} \tag{1.73}$$

for every v and u. Therefore, including the intermediate step (1.69) of (1.68) and (1.71), we have the equivalent relation among the following three forms:

(P1) $$-\frac{d}{dx}\left(k\frac{du}{dx}\right) = f \text{ in } (0, 1), \qquad k\frac{du}{dx}(1) = h, \qquad u(0) = g$$

(P2) $$\int_0^1 \left(k\frac{du}{dx}\frac{dw}{dx} - fw\right) dz - hw(1) = 0, \quad \forall w \text{ with } w(0) = 0$$

(P3) $$u(0) = g, \qquad F(v) \geq F(u), \quad \forall v \text{ with } v(0) = g$$

We shall call (P1), (P2), and (P3) the *local*, *weak*, and *variational forms*, respectively.

In the above, the functional form F is chosen so that (1.66) can be obtained as the Euler equation. However, for a given boundary value problem, it might be difficult to find the corresponding functional for the variational formulation (P3). To avoid this difficulty, the weak form (P2), an intermediate step of the originally given boundary value problem and the variational form, could be a basis for the *variational method* to solve (P1) instead of using (P3), since the form

(P2) is easily obtained from the local form (P1). Indeed, multiplying an arbitrary function w to the differential equation of (1.66) and applying integration by parts after integrating them over the domain (0, 1), we have

$$\int_0^1 \left(-\frac{d}{dx} k \frac{du}{dx} - f \right) w \, dx = 0$$

and

$$\int_0^1 \left(k \frac{du}{dx} \frac{dw}{dx} - fw \right) dx - \left[k \frac{du}{dx} w \right]_0^1 = 0$$

Assuming $w(0) = 0$ and using the boundary condition of (1.66) yields

$$\int_0^1 \left(k \frac{du}{dx} \frac{dw}{dx} - fw \right) dx - hw(1) = 0$$

which is nothing but the form (P2). The derivation of (P2) from (P1) is very straightforward, as shown. Thus, if it is possible to use the weak form (P2) for approximation methods to solve the original boundary value problem (P1), we need not go further up to the variational form (P3), which requires the deduction of a functional.

1.4.1 Approximation (direct methods)

Before discussing finite element methods, we shall briefly review approximation methods based on (P2) and (P3) that are precursors to the finite element methods developed during the 1960s.

Let the minimizer u of F be assumed to be a polynomial

$$u(x) = g + \sum_{j=1}^{N} u_j x^j, \quad u_j \in \mathbb{R}, \quad j = 1, \ldots, N \tag{1.74}$$

that satisfies the essential boundary condition at $x = 0$. Substitution of (1.74) into the functional F yields

$$F(u) = \tfrac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} k \frac{ij}{i+j-1} u_i u_j - f_0 g - \sum_{i=1}^{N} b_i u_i - h\left(g + \sum_{i=1}^{N} u_i \right) \tag{1.75}$$

where $b_i = \int_0^1 f x^i \, dx$, $f_0 = \int_0^1 f \, dx$, and k is assumed to be a constant. If u is a minimizer of F, we have to satisfy

$$\frac{\partial}{\partial u_i} F(u) = 0, \quad i = 1, \ldots, N \tag{1.76}$$

that is,

$$\sum_{j=1}^{N} k \frac{ij}{i+j-1} u_j - b_i - h = 0, \quad i = 1, \ldots, N$$

Defining

$$K_{ij} = \frac{kij}{i+j-1} \qquad \text{and} \qquad f_i = b_i + h \tag{1.77}$$

we have the system of linear equations

$$K_{ij}u_j = f_i, \quad i = 1, \ldots, N \tag{1.78}$$

Solving (1.78) yields an approximation to the minimizer u of F. By taking $N \to \infty$, we may have a minimizer u. This procedure to obtain u is called the *Ritz method*.

(P2) can also be used to find an approximation of u. Suppose that

$$w(x) = \sum_{i=1}^{N} w_i x^i, \quad w(0) = 0 \tag{1.79}$$

Substitution of (1.74) and (1.79) yields

$$\sum_{i=1}^{N}\sum_{j=1}^{N} k\frac{ij}{i+j-1} w_i u_j - \sum_{i=1}^{N} b_i w_i - \sum_{i=1}^{N} h w_i = 0$$

for every w, that is, for every w_i, $i = 1, \ldots, N$. Using (1.77), we have

$$w_i K_{ij} u_j = w_i f_i, \quad \forall w_i, \quad i = 1, \ldots, N \tag{1.80}$$

that is,

$$K_{ij}u_j = f_i, \quad i = 1, \ldots, N$$

Thus, the same system of linear equations as (1.78) is obtained from (P2). We shall call this *Galerkin's method* to find an approximation of a solution to (P1).

If the function w in (P2) is assumed to be

$$w(x) = \sum_{i=1}^{N} w_i \sin\left(\frac{i\pi}{2}x\right)$$

a similar system of linear equations to (1.78) can be obtained. In this case, since different representations to u and w are assumed in (P2), we shall call this the *generalized Galerkin* (or *weighted residual*) *method*.

Exercise 1.7: Solve (1.78) for $N = 3$, and $f(x) = 1 + x + x^2$.

Exercise 1.8: Assume that

$$u(x) = g + \sum_{i=1}^{N} u_i \sin\left(\frac{i\pi}{2}x\right), \qquad w(x) = \sum_{i=1}^{N} w_i \sin\left(\frac{i\pi}{2}x\right)$$

and apply the Ritz and Galerkin methods to solve (1.66).

Exercise 1.9: Find the (P2) and (P3) forms corresponding to the local formulation.

$$\text{(P1)} \quad \begin{cases} -\dfrac{d}{dx}\left(k\dfrac{du}{dx}\right) + \lambda u = f & \text{in } (0, 1) \\ k\dfrac{du}{dx} = -k_0(u - g) + h & \text{at } x = 0 \text{ and } 1 \end{cases}$$

Exercise 1.10: Suppose that a functional F is defined as

$$F(v) = \int_a^b f(v, v^{(1)}, \ldots, v^{(m)})\, dx$$

where $v^{(i)}$ is the ith derivative of a function v defined on the interval $(0, 1)$. Assuming that the integrand f is infinitely many times differentiable in its arguments, derive the Euler equation by taking the *first variation* of F and by applying integration by parts m times. Note that Taylor's expansion of the integrand f is given as follows:

$$\begin{aligned} f(u + \Delta u, u^{(1)} + \Delta u^{(1)}, \ldots, u^{(m)} + \Delta u^{(m)}) \\ = f(u, u^{(1)}, \ldots, u^{(m)}) + \frac{\partial f}{\partial u}(u, u^{(1)}, \ldots, u^{(m)})\,\Delta u \\ + \cdots + \frac{\partial f}{\partial u^{(m)}}(u, u^{(1)}, \ldots, u^{(m)})\,\Delta u^{(m)} \\ + \frac{1}{2}\frac{\partial^2 f}{\partial u^2}(u, u^{(1)}, \ldots, u^{(m)})\,\Delta u^2 + \cdots \end{aligned}$$

Exercise 1.11: Find the Euler equation for the functional

$$F(w) = \tfrac{1}{2}\int_0^1 EI\left(\frac{d^2w}{dx^2}\right)^2 dx - \int_0^1 fw\,dx + \left[M\frac{dw}{dx}\right]_0^1 - \left[Pw\right]_0^1$$

where $[g]_0^1 = g(1) - g(0)$.

1.4.2 Lagrange multiplier methods

In the above, the minimization problem has the constraint $u = g$ on the boundary point $x = 0$. This yields the condition $w(0) = 0$ for a variation w from the solution u. Now, if a Lagrange multiplier p is introduced to release the constraint $u = g$ at $x = 0$, the original minimization problem can be reformulated as a saddle point problem of a corresponding Lagrangian

$$L(v, q) = F(v) - qv(0) - g \tag{1.81}$$

where q is an arbitrary *admissible* Lagrange multiplier. That is, we shall seek a *saddle point* (u, p) such that

$$L(u, q) \le L(u, p) \le L(v, p), \quad \forall (v, q) \tag{1.82}$$

where v does not have any restriction. The first inequality of (1.82) yields the minimization problem

$$qu(0) - g \ge pu(0) - g, \quad \forall q \tag{1.83}$$

Taking $q = p \pm \varepsilon r$, $\varepsilon > 0$, $\forall r$, in (1.83) and dividing by ε, we have

$$ru(0) - g = 0, \quad \forall r, \qquad \text{i.e.,} \quad u = g \text{ at } x = 0 \tag{1.84}$$

On the other hand, the second inequality in (1.82) yields the minimization problem

$$F(v) - pv(0) \ge F(u) - pu(0) \tag{1.85}$$

Taking $v = u \pm \varepsilon w$, $\varepsilon > 0$ in (1.85), that is, taking the first variation, we have

$$\int_0^1 k\frac{du}{dx}\frac{dw}{dx}\,dx - pw(0) = \int_0^1 fw\,dx, \quad \forall w \tag{1.86}$$

Applying integration by parts, we have Euler's equation

$$-\frac{d}{dx}\left(k\frac{du}{dx}\right) = f \text{ in } (0, 1)$$

$$-k\frac{du}{dx} = p \text{ at } x = 0, \qquad k\frac{du}{dx} = 0 \text{ at } x = 1$$

Thus, the Lagrange multiplier p is the *heat flux* at the boundary $x = 0$.

If an inequality constraint $u - \bar{u} \leq 0$ is assumed in $(0, 1)$ for a given function $\bar{u}$ such that $g \leq \bar{u}$ at $x = 0$, the Lagrange multiplier p has to be restricted by $p < 0$ and the Lagrangian is defined by

$$L(v, q) = F(v) - \int_0^1 q(v - \bar{u})\, dx, \quad \forall q \leq 0$$

Note that v and u are free from the constraint $v - \bar{u} \leq 0$ and $u - \bar{u} \leq 0$. Thus, there are no restrictions on the variation with respect to u, although the Lagrange multiplier is restricted by $p \leq 0$. The first inequality of (1.82) yields

$$\int_0^1 (q - p)(u - \bar{u})\, dx \geq 0, \quad q \leq 0$$

that is,

$$p \leq 0, \quad u - \bar{u} \leq 0 \qquad p(u - \bar{u}) = 0 \quad \text{in } (0, 1) \tag{1.87}$$

The second inequality of (1.82) implies

$$\int_0^1 \left(k\frac{du}{dx}\frac{dw}{dx} - pw\right) dx = \int_0^1 fw\, dx, \quad \forall w, w(0) = 0$$

The local form of this integral identity can be obtained as

$$-\frac{d}{dx}\left(k\frac{du}{dx}\right) - p = f \quad \text{in } (0, 1)$$

$$u = g \quad \text{at } x = 0, \quad k\frac{du}{dx} = 0 \quad \text{at } x = 1 \tag{1.88}$$

It follows from (1.87) that $p = 0$ if $u < \bar{u}$ and p needs not be zero if $u = \bar{u}$. That is, if the solution reaches to the upper bound $\bar{u}$ at a point x, the Lagrange multiplier p becomes *active*. If the strict inequality $u < \bar{u}$ is satisfied, p is *nonactive*.

Exercise 1.12: Let us consider a minimization problem for a functional

$$F(w) = \tfrac{1}{2}\int_0^1 \left[EI\left(\frac{d^2w}{dx^2}\right)^2 + kw^2\right] dx - \int_0^1 fw\, dx$$

If a new function M is defined by

$$M = EI\frac{d^2w}{dx^2} \tag{1.89}$$

the original functional $F(w)$ is written as

$$\hat{F}(w, M) = \tfrac{1}{2}\int_0^1 \left(\frac{M^2}{EI} + kw^2\right) dx - \int_0^1 fw\, dx$$

and the minimization problem becomes a problem with respect to two variables w and M under a subsidiary condition (1.89). If a Lagrange multiplier p is introduced to make an unconstrained problem, a corresponding Lagrangian becomes

$$L(w, M, p) = \hat{F}(w, M) - \int_0^1 p\left(\frac{M}{EI} - \frac{d^2w}{dx^2}\right) dx$$

Assuming the boundary condition on the Lagrange multiplier p such that

$$p(0) = p(1) = 0$$

integration by parts on the last term of the Lagrangian yields

$$L(w, M, p) = \hat{F}(w, M) - \int_0^1 \left(p\frac{M}{EI} + \frac{dp}{dx}\frac{dw}{dx}\right) dx$$

Obtain similar expressions to (1.84) and (1.86) for the Lagrangian $L(w, M, p)$ starting from a saddle point problem similar to (1.82):

$$L(w, M, q) \leq L(w, M, p) \leq L(v, N, p), \quad \forall (v, N, q)$$

Using the relation $p = M$, which represents one of three equations obtained by the saddle point problem, rewrite the other two equations in terms of w and M.

1.4.3 Penalty methods

There are other ways to derive an unconstrained minimization problem to a constrained problem, which is subject to subsidiary conditions such as essential boundary conditions. The exterior penalty method is such a method. We shall briefly explain this using the minimization problem (1.68), which is constrained by the boundary condition $v(0) = g$.

The first step is to introduce a penalty functional $P(v)$ such that:

i. $P(v) = 0$ if and only if subsidiary conditions are exactly satisfied;
ii. $P(v) \geq 0$ and P is a continuous convex functional.

If a functional P satisfies inequality $P((1 - \theta)u + \theta v) \leq (1 - \theta)P(u) + \theta P(v)$ for every $\theta \in [0, 1]$, P is said to be convex. Now, if the condition $v(0) = g$ is concerned, a functional defined by

$$P(v) = \tfrac{1}{2}[v(0) - g]^2 \tag{1.90}$$

is clearly a penalty functional.

The second step is to define an unconstrained minimization problem to find u_ε such that

$$F_\varepsilon(u_\varepsilon) \leq F_\varepsilon(v), \quad \forall v \tag{1.91}$$

where

$$F_\varepsilon(v) = F(v) + \frac{1}{\varepsilon} P(v) \tag{1.92}$$

for a sufficiently small positive number ε. The weak form derived from (1.91) is

$$\int_0^1 \left(k \frac{du_\varepsilon}{dx} \frac{dw}{dx} - fw \right) dx - hw(1) + \frac{1}{\varepsilon} [u(0) - g] w(0) = 0, \quad \forall w \tag{1.93}$$

This then yields Euler's equation

$$-\frac{d}{dx}\left(k \frac{du_\varepsilon}{dx} \right) = f \text{ in } (0, 1)$$

$$k \frac{du_\varepsilon}{dx}(1) = h, \qquad k \frac{du_\varepsilon}{dx}(0) = \frac{1}{\varepsilon} [u_\varepsilon(0) - g] \tag{1.94}$$

Since the heat flux at the boundary is finite, it can be expected by the third equation in (1.94) that

$$u_\varepsilon(0) - g = \varepsilon \left(k \frac{du_\varepsilon}{dx}(0) \right) \to 0 \text{ as } \varepsilon \to 0$$

That is, $u_\varepsilon(0) - g = O(\varepsilon)$. If a constant ε is sufficiently small, the boundary condition $u(0) = g$ is approximately satisfied in the unconstrained minimization problem (1.91). A formal proof for the above consequence is as follows:

$$F_\varepsilon(u_\varepsilon) = \min_v F_\varepsilon(v) \le \min_{v, v(0) = g} F_\varepsilon(v) = F(u) \tag{1.95}$$

that is,

$$P(u_\varepsilon) \le \varepsilon [F(u) - F(u_\varepsilon)]$$

Since $F(u)$ and $F(u_\varepsilon)$ are finite values for any ε, the right side goes to zero as $\varepsilon \to 0$. Because of $P(v) \ge 0$, we have

$$\lim_{\varepsilon \to 0} P(u_\varepsilon) = 0$$

If a sequence $\{u_\varepsilon\}$ converges to a function $\hat{u}$, then $P(\hat{u}) = 0$; that is, $\hat{u}(0) = g$, since P is assumed to be continuous. The remaining question is to show that $\hat{u} = u$, that is, that $\hat{u}$ is a minimizer of the original problem. Noting that $P(v) \ge 0$ and $\varepsilon \ge 0$, (1.95) yields

$$F(u_\varepsilon) \le F(u)$$

Since u_ε is assumed to converge to $\hat{u}$, we have $F(\hat{u}) \le F(u)$; that is, $\hat{u}$ is also a minimizer of a functional F under the condition $\hat{u}(0) = g$.

The last remark is a relation between the Lagrange multiplier and penalty methods. If

$$p_\varepsilon = -\frac{1}{\varepsilon} [u_\varepsilon(0) - g] \tag{1.96}$$

is defined, the weak form (1.93) suggests that p_ε is an approximation of the Lagrange multiplier. Indeed, if p_ε converges to a function $\hat{p}$ as $\varepsilon \to 0$, it can be shown that $\hat{p}$ is the Lagrange multiplier to the constraint $u(0) - g = 0$.

Exercise 1.13: For a functional

$$F(w) = \tfrac{1}{2}\int_0^L \left[EI\left(\frac{d^2w}{dx^2}\right)^2 + kw^2 \right] dx - \int_0^L fw\, dx$$

Let us consider that the relation

$$\theta = \frac{dw}{dx}$$

is a subsidiary condition. Then the original functional becomes

$$\hat{F}(w, \theta) = \tfrac{1}{2}\int_0^L \left[EI\left(\frac{d\theta}{dx}\right)^2 + kw^2 \right] dx - \int_0^L fw\, dx$$

Applying the penalty method for a penalty functional

$$P(w, \theta) = \tfrac{1}{2}\int_0^L \left(\theta - \frac{dw}{dx}\right)^2 dx$$

derive the weak form and Euler's equation.