

## Chapter 2

# Truss Analysis: A Structural Mechanics Tour of the FEM—Nodal Equilibrium and Compatibility

**Abstract** Linear elastic behavior and small displacements and rotations (Spencer, Continuum mechanics. Longman, London, 1980) are assumed throughout. This model of linear systems allows us to obtain the nodal forces in terms of the bar stiffness matrices and nodal displacements that are the primary variables. The equilibrium equations can then be constructed in terms of displacements at nodes. The given forces and displacements are assumed to be prescribed as nodal quantities. The resulting system equation, which is in the matrix form, can be obtained in terms of nodal displacements and solved by employing the *Mathematica* function `Solve`.

A very important area of numerical analysis, i.e., solving positive definite simultaneous equation systems, will not be addressed in this textbook.

This chapter introduces *Mathematica* codes. To get started, a summary introduction is provided in Appendix A that should be studied before reading the current chapter.

After the first reading, Appendix B should be reviewed for additional examples and theoretical analysis.

## 2.1 Truss Structures

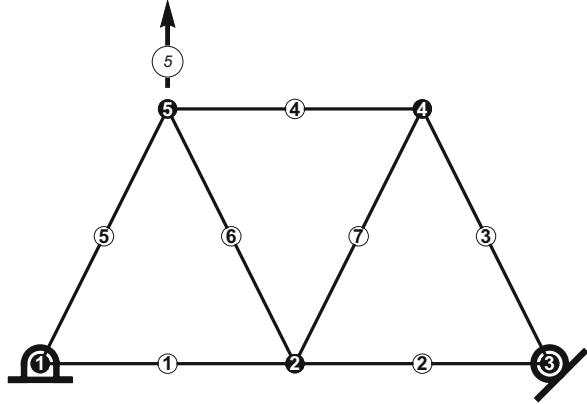
A truss is a pin-connected assembly of bars. As shown in Fig. 2.1, the structure is subjected to axial loading *only*, without bending [1, 6].

In a truss structure, *vide* Fig. 2.1, loads are applied only at joints.

In the finite element literature, a point where many elements meet or where a boundary condition is prescribed is called a *node*. Rollers and hinges provide the boundary conditions. The objective is to calculate all member forces and joint displacements.

In Fig. 2.1, the roller is at node 3. A roller is always placed on a *frictionless* platform. Then it can roll freely on the platform and the displacement along that direction is unknown and orthogonal to that the displacement is zero. Consequently,

**Fig. 2.1** Truss:  
pin-connected bar elements  
with hinge and roller supports



the roller reaction develops only along the latter direction, i.e., perpendicular to the platform.<sup>1</sup> Furthermore, we idealize the roller support to act as an anchor that permits the structure to be pulled.

The hinge, at node 1 in Fig. 2.1, does not permit any movement whatsoever. As a result, the hinge reaction can develop in any direction. The direction of the platform, indicated under the hinge, is inconsequential.

The truss in Fig. 2.1 has five nodes. Horizontal and vertical displacements at each node constitute the ten nodal degrees-of-freedom.

From Eq. (1.86), we obtain element stiffness matrices:  $[k]^{(1)} \dots [k]^{(7)}$  :

$$[k]^{(i)} : \text{stiffness matrix for the } i\text{th element} \quad (2.1a)$$

from nodal equilibrium and compatibility

$$[k]^{(S)} : \text{system (or global) stiffness matrix} \quad (2.1b)$$

*superscript encased within parentheses*

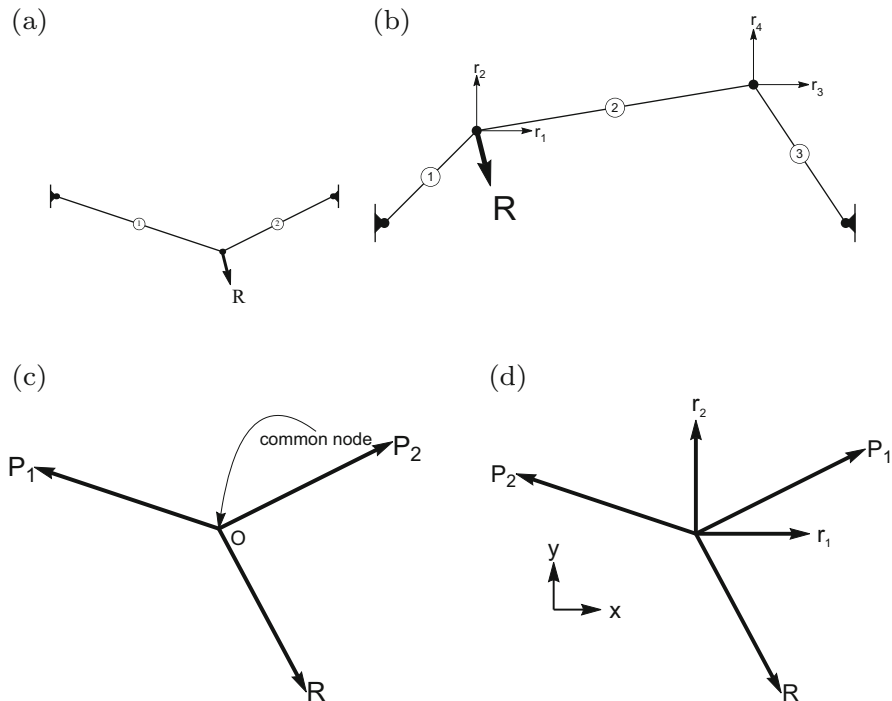
In Fig. 2.2, bar forces  $P_1, P_2$  and nodal displacements  $r_1, r_2$ <sup>2</sup> are unknown.

Figure 2.2b is unstable, the load can freely move on a circular arc!

We now add one extra member (also called a *redundant member* since it is not essential for stability) to Fig. 2.2 and generate Fig. 2.3. There are now three unknown bar forces  $P_1, P_2$  and  $P_3$ , but two unknown nodal displacements  $r_1, r_2$ , as before.

<sup>1</sup> Details are in Sect. B.4, vide Fig. B.10.

<sup>2</sup> The fonts in figures and text *do not* match. This issue is described in Sect. B.4.1.



**Fig. 2.2** A two bar system and a mechanism with a single load. (a) A *stable* structure. (b) A mechanism: *not* a structure. (c) Forces at O. (d) All forces and displacements

### 2.1.1 Equilibrium Is Satisfied Explicitly at Joints

In Figs. 2.2d and 2.3b, as in all truss problems, at each joint,  $i$ , equilibrium is satisfied by *balancing*<sup>3</sup> the externally applied forces and *support reactions*  $\mathbf{R}_j$  and the member forces  $\mathbf{P}^{(m)}$ :

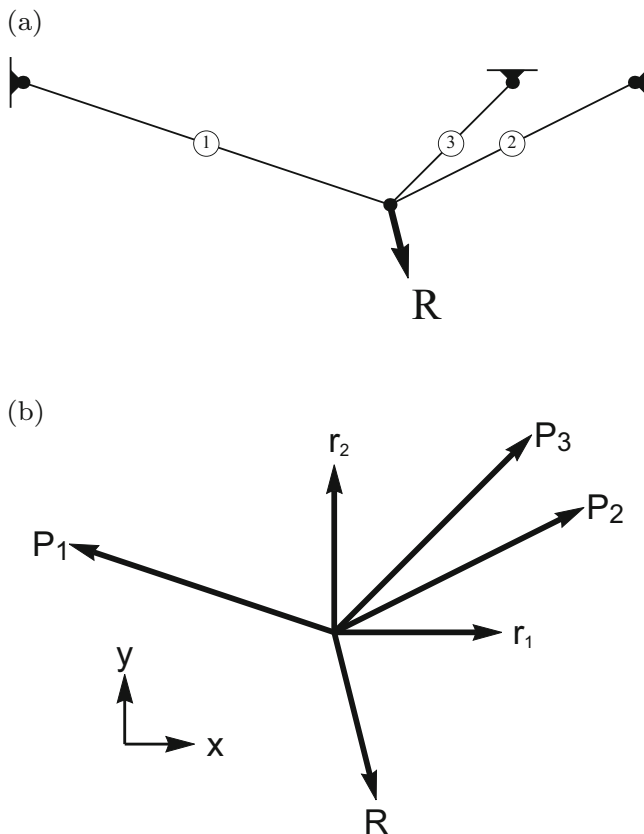
$$\sum \mathbf{R}_j + \sum \mathbf{P}^{(m)} = 0; \text{ bold } \mathbf{P}, \mathbf{R} \text{ represent vectors} \quad (2.2)$$

$j$  : external forces at node  $i$ ; this index is a subscript

$m$  : members meeting at node  $i$ ; superscript index in parentheses

In Figs. 2.2 and 2.3 since all forces are acting at a point, Eq.(2.2) denotes equilibrium of the entire plane structures. It then suffices to satisfy equilibrium *individually* in the  $x$ - and  $y$ - (or in any two orthogonal) directions. The zero net moment condition is trivially satisfied.

<sup>3</sup>'Equating to zero the sum in the *vector sense*.' Components of all forces, which are meeting at a point, in any two orthogonal directions should add up to zero.



**Fig. 2.3** Three bar forces and two displacements are unknown. (a) A redundant system. (b) Nodal displacements are positive along coordinates; in the members tensile forces are positive

### 2.1.2 Matrix Notation

In Figs. 2.2 and 2.3  $R$  is a vector. On the other hand, in Figs. 2.2d and 2.3b, the *displacement components*, not the displacement vectors, are clearly indicated by  $r_1, r_2$ . The matrix notation allows component-wise representation of external forces and nodal displacements, at all degrees-of-freedom.

### 2.1.3 An Argument for the Displacement Method

We can add any number of bars at the joint O of Fig. 2.2 without increasing the number of displacement unknowns. There are only two equilibrium equations for a

joint. Then *only* from equations of statics, even for Fig. 2.3, member forces (and support reactions) cannot be evaluated. The method must involve  $r_1$  and  $r_2$  that explicitly satisfy compatibility.

Overwhelmingly, the number of displacement unknowns is (significantly) less than that of the forces. Also, the solution demands less numerical efforts when the displacements are the primary variables,<sup>4</sup> as compared to the force counterpart, i.e., the *flexibility method*. Therefore we select the displacement and the stiffness formulation for all truss problems, and use these in the finite element method.

### 2.1.4 Notations: Superscripts and Subscripts

In Figs. 2.2 and 2.3, the compatibility requirement dictates that at joint O, the  $(x, y)$  displacement components,  $r_1$  and  $r_2$  should be the same for all members. Let us consider the equilibrium equation:

$$\begin{Bmatrix} R_x \\ R_y \end{Bmatrix} + \sum_{i: \text{ members meeting at node } O} \begin{Bmatrix} P_x^{(i)} \\ P_y^{(i)} \end{Bmatrix} = 0 \quad (2.3)$$

To avoid cluttering the symbols in the textbook, the element number will be indicated with a superscript. The subscripts are reserved to indicate components that could be the coordinates or the degree-of-freedom number designator. In Figs. 2.2 and 2.3, the member forces  $\mathbf{P}^{(i)}$  will be finally obtained from the element stiffness matrices with  $(r_1, r_2)$  as the primary unknowns:

$$\begin{Bmatrix} r_1 \\ r_2 \end{Bmatrix} = \{r\} : \text{displacement vector (a column matrix)} \quad (2.4)$$

In displacement-based finite element method, which is the focus of this textbook, nodal displacements will always be treated as the target variables to be calculated. Subsequently, all physical quantities are evaluated in terms of the displacements at the degrees-of-freedom.

### 2.1.5 Truss System Stiffness Matrix Equation

In Fig. 2.1, there are five nodes. *Before* the application of the displacement constraints provided by the roller and the hinge, each node can independently freely

<sup>4</sup> The same principle applies to all complicated structures. Once the element stiffness matrices  $[k]^{(i)}$  are obtained then at each node compatibility and equilibrium conditions lead to the system equations, via the system stiffness matrix  $[k]^{(S)}$ , relating all unknowns.

move in two orthogonal directions. Hence there are ten degrees-of-freedom in the truss system of Fig. 2.1. Then, in the spirit of footnote 4 we can define the  $10 \times 10$  truss stiffness<sup>5</sup> or the system stiffness matrix  $[k]^{(S)}$ :

$$\{R\}_{10 \times 1}^{(S)} = [k]_{10 \times 10}^{(S)} \{r\}_{10 \times 1}^{(S)} ; \text{ superscript } (S) : \text{ system} \quad (2.5)$$

The virtual displacement counterpart of Eq. (2.5) will have  $\delta \{R\}, \delta \{r\}$ :

$$\delta \{R\}^{(S)} = [k]^{(S)} \delta \{r\}^{(S)} \quad (2.6)$$

The components of  $\delta \{r\}^{(S)}$  must be zero at the degrees-of-freedom where the displacement boundary conditions are prescribed. This is formalized next.

### 2.1.6 Unit Virtual Displacement in a System

A procedure to ‘pull out’ the  $i, j$  element  $k_{ij}$  of any stiffness matrix (local or global, singular or non-singular)  $[k]$  (a *general* matrix is encased in [ ]) is:

$$\begin{bmatrix} \dots\dots\dots \\ \dots\dots\dots \\ \dots \vdots \dots \\ \dots k_{ij} \dots \\ \dots \vdots \dots \\ \dots\dots\dots \end{bmatrix} \text{ and define } \underbrace{\{\delta r^{(i)}\} = \begin{Bmatrix} 0 \\ 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{Bmatrix}}_{\substack{\text{the unit virtual displacement} \\ \text{1 at row } j, \text{ other values are zero}}} \text{ then:} \quad (2.7)$$

$$\{\delta r^{(i)}\}^T [k] \{\delta r^{(i)}\} = k_{ij} : \text{ clearly a virtual work quantity} \quad (2.8)$$

This idea is generalized in the continuum sense with finite element shape functions that are the analogs of  $\{\delta r^{(i)}\}$ —the displacement field due to a unit virtual nodal displacement.<sup>6</sup>

<sup>5</sup>Roller and hinge boundary conditions are not considered in the system stiffness matrix  $[k]^{(S)}$ . There are three zero eigenvalues, for two rigid body translations in  $(x, y)$ , and one rigid body rotation on the truss plane.

<sup>6</sup> Vide  $n_i(x, y)$  of Sect. 3.1 for an illustration with plane problems.

### 2.1.6.1 Reduced Stiffness Matrix of a Truss $[k]$ from the Global Stiffness Matrix $[k]^{(S)}$

The global displacements are housed in a row matrix  $\langle r \rangle^{(S)}$  (a row matrix is encased in  $\langle \rangle$  but a column matrix in  $\{ \}$ ):

$$\langle r \rangle^{(S)} = \left\langle \overbrace{\langle r \rangle}^{\text{arbitrary}}, \overbrace{\langle r_o \rangle}^{\text{prescribed}} \right\rangle : \text{partitioned} \quad (2.9)$$

$$\text{then: } \delta \langle r \rangle^{(S)} = \left\langle \overbrace{\delta \langle r \rangle}^{\text{to be calculated}}, \overbrace{\delta \langle r_o \rangle}^{\text{zero}} \right\rangle \quad (2.10)$$

The prescribed displacement boundary conditions  $\{r_o\}$  *cannot* be altered:

$$\text{hence: } \delta \langle r \rangle^{(S)} = \langle \delta \langle r \rangle, \langle 0 \rangle \rangle \quad (2.11)$$

whose imposition reduces  $[k]^{(S)}$ —the global matrix—to the reduced matrix  $[k]$ . Associated with the three zero eigenvalues of  $[k]^{(S)}$ , there must be at least three *prescribed* nodal displacements to “*anchor down*” the truss.

### 2.1.6.2 Stiffness Elements as Virtual Work Quantities

For a generic stiffness matrix  $[k]$  we shall *temporarily* state, without resorting to the concept of virtual work or virtual displacement, that:

“ $k_{ij}$  is the force needed at the  $i$ th degree-of-freedom, when a unit displacement is *gradually* applied at the  $j$ th degree-of-freedom with *all other degrees-of-freedom held* (rigid or) *locked*.”

From the unit nodal virtual displacement concept, we state:

“ $k_{ij}$  is the virtual work due to the application of the unit virtual displacement at the  $i$ th degree-of-freedom, when a unit displacement is imposed at the  $j$ th degree-of-freedom<sup>7</sup> while *all other degrees-of-freedom are held locked*.”

As explained in Sect. 2.1.6, we can polish up the definition of  $k_{ij}$  to:

“ $k_{ij}$  is the virtual work necessary to impose a unit virtual displacement at the  $i$ th degree-of-freedom, on the virtual nodal forces introduced by a unit virtual displacement applied at the  $j$ th degree-of-freedom.”

Recall, the unit virtual nodal displacement at the  $j$ th degree-of-freedom necessarily implies that all *other* degrees-of-freedom are held fixed.

<sup>7</sup>There is a slight anomaly, depending on whether the displacements at  $i$ th and  $j$ th degree-of-freedom be zero or unity.

### 2.1.6.3 Equations To Be Solved

The  $n$  displacement boundary conditions reduce the global stiffness matrix  $[k]^{(S)}$  to the non-singular<sup>8</sup> *reduced stiffness matrix*  $[k]$ . Now, the number of  $m$  unknowns in  $\{r\}$  gets reduced by  $n$  due to the displacement constraints:

$$\text{global : } [k]_{m \times m}^{(S)} \rightarrow [k]_{(m-n) \times (m-n)} : \text{reduced} \quad (2.12)$$

For degrees-of-freedom where the displacements are unknown:

$$\{R\} = [k] \{r\}; \text{ all displacements in } \{r\} \text{ to be calculated} \quad (2.13)$$

In the summation convention we have to solve:

$$R_j = \sum_i k_{ji} r_i = \sum_i k_{ij} r_i \quad (\text{due to symmetry of } [k]) \quad (2.14)$$

However, due to the symmetry of stiffness matrices we shall always utilize symmetric equation solvers to drastically reduce computing time and storage.

The bar problems in Figs. 2.2 and 2.3 capture the essence of the finite element method. In fact, Eqs. (2.3) through (2.6) and Eqs. (2.12) and (2.14) encapsulate the formulation in the  $(x, y)$  frame.

## 2.2 Displacement Formulations: Statically Determinate and Indeterminate Trusses

Trusses are classified into two groups: statically determinate and indeterminate, [1, 6]. In statically determinate systems, e.g. Fig. 2.2a, equations of statics *alone* suffice to complete the analysis.

### 2.2.1 Statically Determinate Trusses

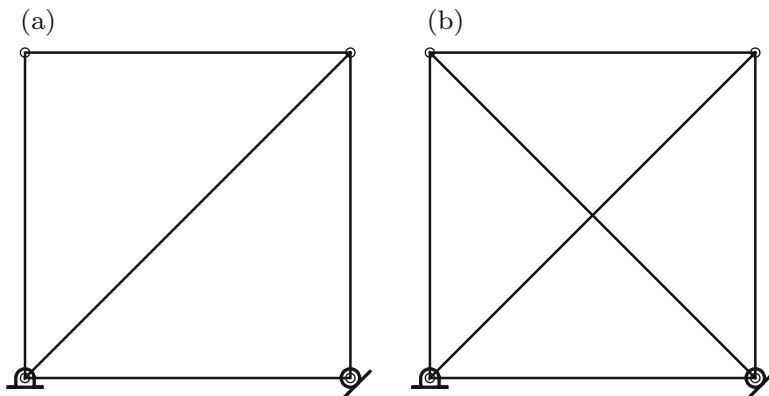
In Fig. 2.4a, an important implication is that the member areas and material properties do not enter into the equations. Also, there is no *redundancy*, i.e., if any member is knocked-off the entire system will collapse as a mechanism. That is to say there are just enough members laid out in a fashion that geometrical stability is guaranteed.

We consider equilibrium of selected sections and generate enough equations to solve all unknowns. Obviously, the stiffness properties of the elements play no role.

---

<sup>8</sup>Here, a square matrix with non-zero determinant.





**Fig. 2.4** Truss problems. (a) Statically determinate. (b) Statically indeterminate

Any standard structural analysis book, such as Chap. 2 in [1], addresses this in detail. The material properties need to be considered only when the joint displacements<sup>9</sup> are to be determined [1, 3].

### 2.2.2 *Statically Indeterminate Trusses*

It is often necessary to strengthen the structure by adding additional members beyond the element arrangement that ensures stability, *vide* Fig. 2.4b. We must satisfy compatibility at joints. This statement implies that all members meeting at a joint should undergo the same displacement. The displacements depend on member stiffnesses that involve cross-sectional areas and elastic moduli.

In general, the number of displacements at a joint is less than the number of member forces, *vide* Fig. 2.3. One vertical and one horizontal joint displacement help determine three member forces. Thus, for structural analysis it is more convenient to solve all trusses with the same computer program based on the displacement formulation with matrix notation.

### 2.2.3 *Importance of the Truss Problem*

In a plane truss, (at the minimum) the *triangular bar structure* attributes stability. Thus, statically determinate trusses, with the minimum number of members, can be solved for all its member forces without consideration of any displacement variables. Since the equations of statics suffice, the member properties, such as the

<sup>9</sup>From Williot Mohr's diagrams we can determine the nodal displacements from bar (positive or negative) extensions.

Young's moduli and bar cross-sectional areas, do not enter into the picture of force computations. In statically indeterminate trusses additional members (also called *redundancy*) secure a higher strength requirement, and need additional equations (beyond static equilibrium) that come from *displacement* compatibility conditions. For practical structures with a high redundancy number, it is convenient to carry out a *displacement formulation* where element stiffness matrices are used where equilibrium of forces and displacement compatibility *only at joints* suffices.

### 2.2.4 Symbolic Computation for Finite Element Systems

Systems other than the two-dimensional trusses will require different element stiffness matrices than those discussed in this chapter. But the way the nodal displacement unknowns are solved remains practically intact.

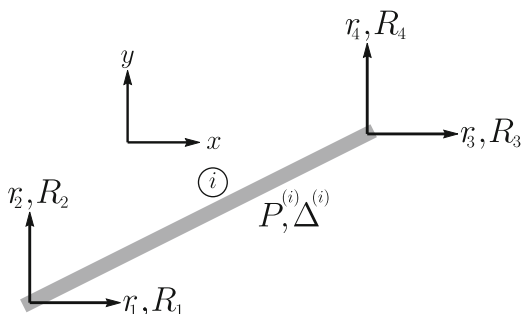
*Mathematica* programming facilitates code development irrespective of spatial dimensionality. If each nodal coordinate `List` has 1 entry, then the system must be a connected bar system, as in Fig. 1.12. If the `Length` of each nodal coordinate `List` is two or three, then the appropriate element stiffness matrices will have two and three degrees-of-freedom per node.

Equilibrium and compatibility is enforced only at the nodal degrees-of-freedom; the equation generator Listing 2.2 does not need any modification.

## 2.3 A Mathematica Formulation to Evaluate Unknowns: Input: nodes, connections, and Member Properties

In any problem, we first identify the reference coordinates and the associated degrees-of-freedom,<sup>10</sup> vide Fig. 2.5 where an isolated member is shown.

**Fig. 2.5** A truss element; nodal forces and displacements with subscript are global



<sup>10</sup>In Fig. 2.5, the MaTeX fonts, vide Sect. B.4.1, match the symbols in the text and figures.

In general, e.g. in Fig. 2.1, the two lists: `nodes` and `connections` furnish the geometrical description and they will be used throughout:

$$\text{nodes} = \{\{x_1, y_1\}, \dots, \{x_j, y_j\}, \dots, \{x_{n\text{Nodes}}, y_{n\text{Nodes}}\}\}; \quad (2.15a)$$

`nNodes` : total number of nodes, in Fig. 2.1: `nNodes` = 5

$$\text{connections} = \{\{i^{(1)}, j^{(1)}\}, \dots, \{i^{(n\text{El})}, j^{(n\text{El})}\}\} \quad (2.15b)$$

`nEl` : total number of elements, in Fig. 2.1: `nEl` = 7

Note, all *Mathematica* lists are encased in curly braces.

In Fig. 2.5,  $R, r$  are associated with degrees-of-freedom; they are indicated with subscripts. For the member force and extension,  $P, \Delta$  the element number, which is within a circle, is encased within parentheses as superscripts. For the  $i$ th element, the stiffness matrix  $[k]^{(i)}$  transforms the independent nodal displacements  $\{r_1, r_2, r_3, r_4\}$  into the nodal forces  $\{R_1^{(i)}, R_2^{(i)}, R_3^{(i)}, R_4^{(i)}\}$ :

$$\langle R_1^{(i)}, R_2^{(i)}, R_3^{(i)}, R_4^{(i)} \rangle = \langle r_1, r_2, r_3, r_4 \rangle [k]^{(i)} : \text{by balancing forces} \quad (2.16)$$

A displacement,  $r_i$ , is typeset in L<sup>A</sup>T<sub>E</sub>X as:

$$\{\text{\ensuremath{\{r\hspace{-0.03in}\}}}_{\_i}\}$$

*Mathematica* modules to generate element stiffness matrices are merely coded equation (1.86) in Listing 2.3. The boundary conditions are expressed as a list of rules in Listing 2.4. This is a better choice than setting up equations with `==` or the `Equal` construct of *Mathematica*. The rollers and hinge boundary conditions are transformed into rules.

In a truss problem, e.g. in Fig. 2.1, resulting member forces  $P^{(i)}$  and axial stretches  $\Delta^{(i)}$ , as in Fig. 2.5, are needed for design-analysis. All such  $P^{(i)}$  and  $\Delta^{(i)}$  are calculated from the nodal quantities,  $R_j$  and  $r_j$ . These are obtained by the stiffness formulation summarized below.

A list<sup>11</sup>  $\{kS\}$  houses all element stiffness matrices,  $[k]^{(i)}$  of Eq. (2.1a):

$$\{kS\} = \{[k]^{(1)}, [k]^{(2)}, \dots, [k]^{(n\text{El})}\}; n\text{El} : \text{number of elements} \quad (2.17)$$

In Fig. 2.1, `nEl` equals seven. Then  $\{kS\}$  is a list<sup>12</sup> consisting of seven  $4 \times 4$  element stiffness matrices. Compare this  $\{kS\}$  with the global stiffness matrix  $[k]^{(S)}$  that is  $10 \times 10$  associated with the five nodal displacements.

The global displacements, e.g.  $\{r_1, r_2, r_3, r_4\}$  in Eq. (2.16), pertaining to the element number  $i\text{El}$  can be retrieved from the list `connections`, via the nodal numbers  $\{i^{(i\text{El})}, j^{(i\text{El})}\}$ : `connections[[iEl]]`; the associated four nodal displacements correspond to the degrees-of-freedom numbered:

$$(2i^{(i\text{El})} - 1), 2i^{(i\text{El})}, (2j^{(i\text{El})} - 1), 2j^{(i\text{El})} \quad (2.18)$$

<sup>11</sup>It is very different from the global stiffness matrix  $[k]^{(S)}$  of Eq. (2.1b).

<sup>12</sup>In *Mathematica*,  $\{kS\}$  is `kS`;  $[k]^{(i)}$  can be extracted as: `ki = Part[kS, i]` or `ki = kS[[i]]`.

### 2.3.1 The Balance Principles of Physics

In finite elements, equilibrium of forces prevails *only in a global sense*.<sup>13</sup> The global balance of energy *is equivalent* to satisfying the global<sup>14</sup> equilibrium, *vide* Sect. 1.5.1. The Eulerian formulation averages out the local errors. Ritz in [4] minimized the Eulerian  $\mathfrak{E}$  of Eq. (1.8), which is the foundation of the finite element method. He devised the best possible adherence to the global (not *point-wise*) balance principle (derived as an integral) associated with assumed approximate response fields.

Akin to the Ritz *variational* formulation, the *force-displacement relations*:  $[k]^{(i)}$ ,  $[k]^{(S)}$ , and  $\{kS\}$  are *virtual work quantities* that originate from the global balance of energy. These two concepts share the same notation,  $\delta$ , as elaborated in Sect. 1.5.1.1.

### 2.3.2 Satisfying Equilibrium at Each Joint

The compatibility at each joint has been assured by using global numbering for each member nodal displacements. This is the advantage of the displacement formulation on which the finite element method is anchored.

In Fig. 2.6, the generic joint shows two degrees-of-freedom labeled  $I$  and  $J$  where several members  $j$  applying forces  $P^{(j)}$ . At the  $J$ th degree-of-freedom,  $R_J, r_J$ <sup>15</sup> are the respective force and displacement.

```
locationP = {{-1.45, -0.83},{-0.48`, -1.8}, {0.78`, -1.64}};
names = {"1", "i", "j"};
iTexts = Table[Text[ StringJoin["( ", ToString[names[[i]]], "
    )"], {i,3}]
```

**Listing 2.1** Code for  $P^{(i)}$

In Sect. B.4.1 proper typesetting in  $\text{\LaTeX}$  figures is described. In particular, the  $\text{\LaTeX}$  package is used in *Mathematica*.

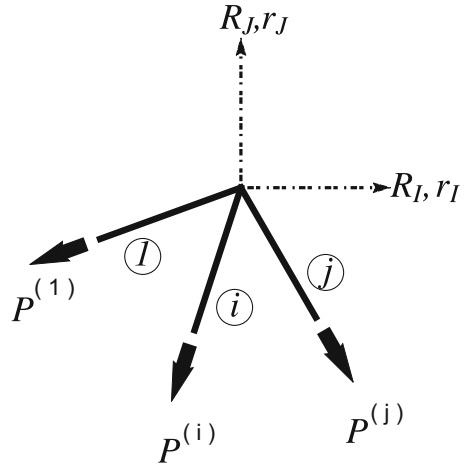
From the list connections in Eq. (2.15b), the *Mathematica* built-in function `Position` yields the element numbers and the order the degree-of-freedom occurs in the member. Suppose for a given degree-of-freedom  $I$  an element number  $p$  has its  $q$ th local degree-of-freedom. `Part[kS,p,q]` pulls out the stiffness row that contributes the force for the  $I$ th degree-of-freedom.

<sup>13</sup>For a given distributed skin force  $f(x)$ , *vide* Sect. 1.5, we still approximate the displacement field to be a linear function  $u(x)$ . By considering a constant bar stress, we do *not* satisfy equilibrium at individual sections.

<sup>14</sup>Also for members, *vide* Eq. (2.16).

<sup>15</sup>Fonts of graphics from Listing 2.1 do not match those in the text, e.g. Eq. (2.16).

**Fig. 2.6** Equilibrium at degrees-of-freedom



### 2.3.3 Generation of Finite Element System Equations

From Eq. (2.18) we get all the degrees-of-freedom that pertain to the member number  $p$ . This facilitates programming, *vide* Listing 2.2. The equilibrium statement in Eq. (2.14) is coded in Listing 2.2:

```
systemEquations::usage = "systemEquations[{r,R},connections, kS]
                           returns the form: R[i]=a[i,j] r[j]; sum
                           on j
                           R[i], r[i]: force and displacement at degree-of-freedom
                           number i
                           connections: array containing element node numbers
                           kS: houses all element stiffness matrices (same order of
                           connections)"

systemEquations[{r_, R_}, connections_, kS_] :=
  Module[{elementDofs, nDim},
    nDim = Length[kS[[1]]]/Length[connections[[1]]];
    elementDofs = Flatten[Range[(# - 1)*nDim + 1, #* nDim] &
      /@ connections;

    Table[R[iDofForce] == Plus @@ ((Part[
      kS, #[[1]], #[[2]]].(r[#] & /@ Part[elementDofs,
        #[[1]]])) & /@
      Position[elementDofs, iDofForce]), {iDofForce,
        (Length[kS[[1]])/
          Length[connections[[1]]]*Max[Flatten
            [connections]]}]]]
```

**Listing 2.2** Code for the system equations

### 2.3.4 An Example from Timoshenko and Young [6]: Fig. 2.7

There are a number of excellent problems in [6]. Listing 2.3 generates element stiffness matrices. The loads and support conditions are input as rules as in Listing 2.4:

```
strussElementStiffnessMatrix::usage="strussElementStiffnessMatrix[
  {node1, node2}, ae_] returns the 4 by 4 matrix for an
  element with nodal coordinates node1 and node2.
  The bar stiffness ae = area * Young's modulus has a
  default value 1"

strussElementStiffnessMatrix[{node1_, node2_}, ae_:1] :=
  Module[{c, s, L},
    L = Sqrt[(node2 - node1). (node2 - node1)]; {c, s} =
      (node2 - node1)/L;
    (ae/L) * {{c^2, c s, -c^2, -c s}, {c s, s^2, -c s, -s^2},
      {-c^2, -c s, c^2,
        c s}, {-c s, -s^2, c s, s^2}} // N]
```

**Listing 2.3** Code for 2-D truss element stiffness matrices from Eq. (1.86)

Prescribed boundary conditions from Eq. (2.20), are organized as *Mathematica* rules (using the syntax form:  $a \rightarrow b$ ):

```
boundaryCondition = Flatten[{
  {r[1] -> 0, r[2] -> 0, r[4] -> 0 },
  {R[5] -> 2, R[11] -> -1, R[12] -> 1},
  Thread[{R[3], R[6], R[7], R[8], R[9], R[10]} -> 0]}]
```

**Listing 2.4** Boundary conditions in Eq. (2.20) as a set of rules

In Fig. 2.7, from [6], a member number is shown within a circle. The connections are described by:

$$\begin{array}{l|cccccccc}
 \text{member number:} & 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \\
 \text{first node number:} & 1 & 2 & 3 & 4 & 5 & 6 & 1 & 2 & 3 \\
 \text{other node number:} & 2 & 3 & 1 & 5 & 6 & 4 & 4 & 6 & 5
 \end{array} \tag{2.19}$$

The problem description is:

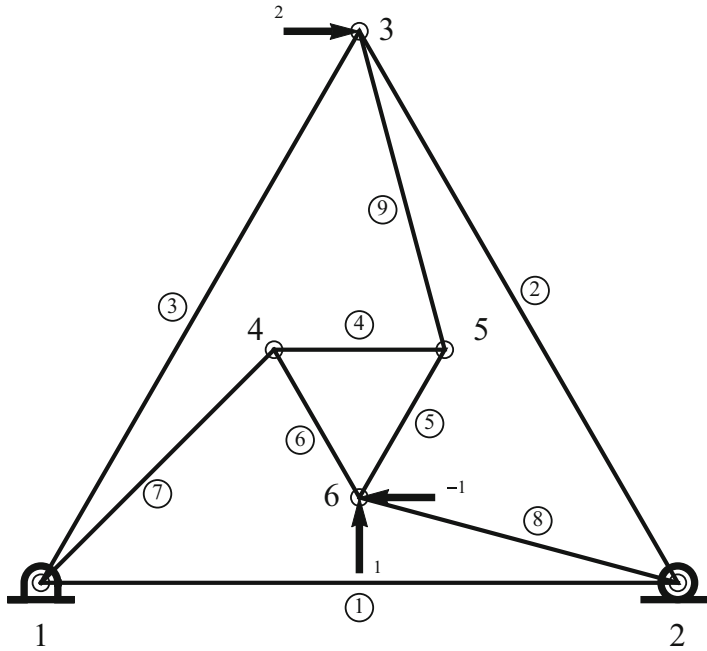


Fig. 2.7 From Timoshenko and Young [6, p. 84, Fig. 2.43]

$$\begin{pmatrix}
 \text{node} & \{x, y\} \\
 1 & \{-\sqrt{3}, -1\} \\
 2 & \{\sqrt{3}, -1\} \\
 3 & \{0, 2\} \\
 4 & \left\{ -\frac{\sqrt{3}(-1 + \sqrt{3})}{1 + \sqrt{3}}, \frac{-1 + \sqrt{3}}{1 + \sqrt{3}} \right\} \\
 5 & \left\{ \frac{\sqrt{3}(-1 + \sqrt{3})}{1 + \sqrt{3}}, \frac{-1 + \sqrt{3}}{1 + \sqrt{3}} \right\} \\
 6 & \left\{ 0, -\frac{2(-1 + \sqrt{3})}{1 + \sqrt{3}} \right\}
 \end{pmatrix}$$

Numbering degrees-of-freedom :

at a node  $i$

$2i - 1$  along  $x$

$2i$  along  $y$

Prescribed quantities :

Loading :

$$R_5 = 2; \quad R_{11} = -1; \quad R_{12} = 1$$

Displacements :

$$r_1 = 0; \quad r_2 = 0; \quad r_4 = 0$$

Quantities to be calculated :

$$R_1, R_2, \text{ and } R_4$$

$$r_3, r_5, r_6 \dots r_{12}$$

(2.20)

The geometrical description from Eq.(2.20) and Fig.2.7 is in nodes and connections in Listing 2.5:

```
nodes = {{-10 Sqrt[3], -10}, {10 Sqrt[3], -10}, {0, 20},
         {30 - 20 Sqrt[3], -10 (-2 + Sqrt[3])}, {-30 + 20
         Sqrt[3], -10 (-2 + Sqrt[3])}, {0, 20 (-2 + Sqrt[3])}};
connections = {{1, 2}, {2, 3}, {3, 1}, {4, 5}, {5, 6}, {6, 4},
              {1, 4},
              {2, 6}, {3, 5}};
memberForces = {-0.314459, -1.6151, 2.14088, 0.3849, -0.525783,
               -0.281766,
               0.345092, 1.16159,
               -0.471405}
```

**Listing 2.5** Input and output data; result with all  $AE = 1$

```
tensileForce[kS_, i_, connections_, r_, allValues_, nodes_] :=
Module[{f1, f2, f3, f4, elementDofs, nDim, c, s, n1, n2, n},
  {n1, n2} = nodes[[#]] & /@ connections[[i]];
  n = n2 - n1; {c, s} = n/Sqrt[n.n];

  nDim = Length[kS[[1]]]/Length[connections[[1]]];
  elementDofs = Flatten[Range[(# - 1)*nDim + 1, #*nDim]] & /@
    connections;
  {f1, f2, f3, f4} = kS[[i]]. ((r[#] & /@ elementDofs[[i]])
    /. allValues); -(f1 *c + f2 *s)]
```

**Listing 2.6** Calculating the tensile force in a member from Eq.(1.82b)

```
memberForces = {-0.314459, -1.6151, 2.14088, 0.3849, -0.525783,
               -0.281766,
               0.345092, 1.16159,
               -0.471405}
```

**Listing 2.7** Calculated tensile force in a member from Eq.(1.82b) in memberForces

In Fig.2.8, the external forces are indicated in black and the bar tensile forces, obtained from Eq.(1.82b) and Listings 2.6 and 2.7, are in white on black background.

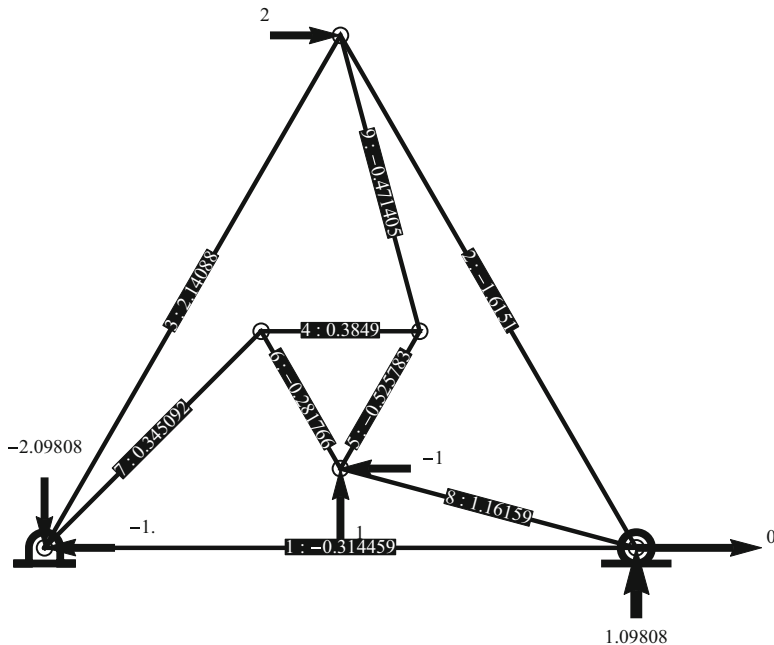
In all cases, we shall indicate the tensile member force with a positive number. The nodal forces and displacements are positive along coordinate axes.

All graphics can be generated from the geometry and boundary data. The lists, nodes and connections, are always included in all function calls that draw on different components.

More elaborate graphics and calculations are presented in Appendix B.

Listing 2.8 evaluates all nodal displacements  $\{r\}$  and forces  $\{R\}$ :





**Fig. 2.8** Calculated forces—figure and forces displayed *algorithmically*

```
(* Solve all forces and displacement at all degrees-of-freedom *)

(* generate all element stiffness matrices in kS *)

kS = Table[strussElementStiffnessMatrix[nodes[[#]] & /@
connections[[iElm]]],
{ iElm, Length[connections] }];

(* indicate nodal displacements and forces by r and R *)

eqs = systemEquations[{r, R}, connections, kS];
nDof = (Length[kS[[1]]]/Length[connections[[1]])*
Max[Flatten[connections]]];

allDOF = Flatten[{Array[r, nDof], Array[R, nDof]}];
variablesToBeSolved = Complement[allDOF, (First /@
boundaryCondition)]

sol = Flatten[Solve[eqs /. boundaryCondition,
variablesToBeSolved]];

(* values of all nodal displacements and forces *)

allValues = Union[sol, boundaryCondition]
```

**Listing 2.8** Solving all nodal displacements and forces

### 2.3.5 A Note on Numerical Efficiency

A finite element system equation, indicated by `eqs` in Listing 2.8, is always symmetric, positive definite, and banded. In Listing 2.8, `Solve[]` does not take advantage of these mathematical properties; hence, it employs a robust but inefficient algorithm to yield the solutions in `sol`. In concept development and homework problems, `Solve[]` is adequate. *Mathematica* has a number of ‘under the hood’ enhancements to speed up calculations.

For production runs `C++` libraries can be called from *Mathematica*; such programming details, e.g. [2], are not within the scope in this textbook.

Finite element system equations are not only symmetric but sparse in general, and mostly banded. These features enhance numerics significantly. *Mathematica* functions such as:

`SparseArray\[RawBackquote]KrylovLinearSolve`

can be employed in production runs. In this textbook, we shall not go into any such detail.

## 2.4 Extensions to General Finite Element Systems

For general finite elements we can keep most of the computer codes from truss problems.<sup>16</sup> For example, in plane strain analysis, we replace the bar stiffness matrices by the two-dimensional plane strain stiffness matrices. The assembly and solution procedures, hence the *Mathematica* routines, remain intact.

### 2.4.1 Spatial Discretization

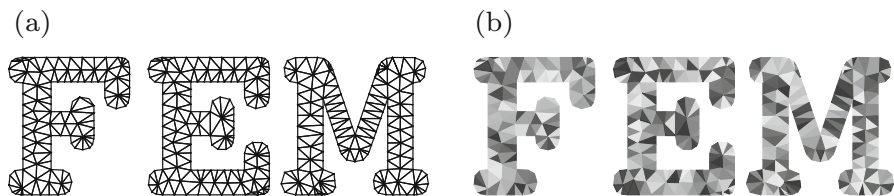
Finite elements,  $\Omega^{(i)}$  with a boundary  $\Gamma^{(i)}$ , are non-intersecting geometrical entities whose collection fully covers the body  $\Omega$  with a boundary  $\Gamma$  :

$$\cup \Omega^{(i)} = \Omega; \quad \text{for } i \neq j, \quad \Omega^{(i)} \cap \Omega^{(j)} = \emptyset : \text{the empty set} \quad (2.21)$$

Mesh generation, with a predetermined spatial discretization by design, is an active field of research. There are modules (also included in *Mathematica*) that are widely available for all computer systems.

---

<sup>16</sup>Except for the incompressible case where the set of degrees-of-freedom contains isochoric shape functions and a constant pressure for each finite element.



**Fig. 2.9** Covering arbitrary 2-D shapes with simplex elements. (a) Covering with truss elements. (b) Covering with triangular elements

### 2.4.2 *Triangulation: Simplex Elements*

Figure 2.9a furnishes a generic example where truss-like triangles cover an arbitrary domain. In Fig. 2.9b, we have filled in the space between the truss members with thin plane sheets and encountered a plane stress problem. The elements of the figures have the simplest geometrical shapes, and are thus termed simplex (in three-dimensions, a simplex element is a tetrahedron).

### 2.4.3 *The Scope of Truss Problems*

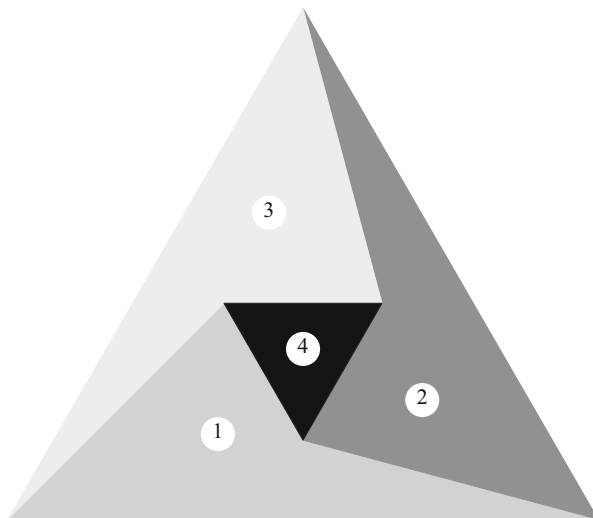
It is of the utmost importance to treat distinctly the scalar<sup>17</sup> problems from their vector field counterparts, e.g. continuum elasticity.<sup>18</sup> The latter is converted into the former by idealizing the axial displacement to be the only independent variable in individual members, in trusses.

In Fig. 2.10, we have non-simplex tessellation. For elasticity problems in two- and three-dimensions, the shape functions themselves must be vectors coupled via the Poisson's ratio, even though the triangulation scalar shape functions suffice. These issues are addressed in the following chapters of this textbook.

For two-dimensional plane strain cases, incompressibility, i.e., for the Poisson's ratio  $\nu = \frac{1}{2}$ , poses additional difficulties. The shape functions are kinematically constrained and a constant element pressure  $p_0$  needs to be addressed. The notion of the pseudoinverse, described in Sects. 1.8 and 1.9, provides the computational concept and the tool to resolve the 'zero dilatation' situation.

<sup>17</sup>For example, temperature distributions and the torsion in non-circular prismatic shafts.

<sup>18</sup>For triangles and tetrahedra the interpolants are identical for scalar and vector problems.



**Fig. 2.10** 2D system with same nodes as in Fig. 2.7

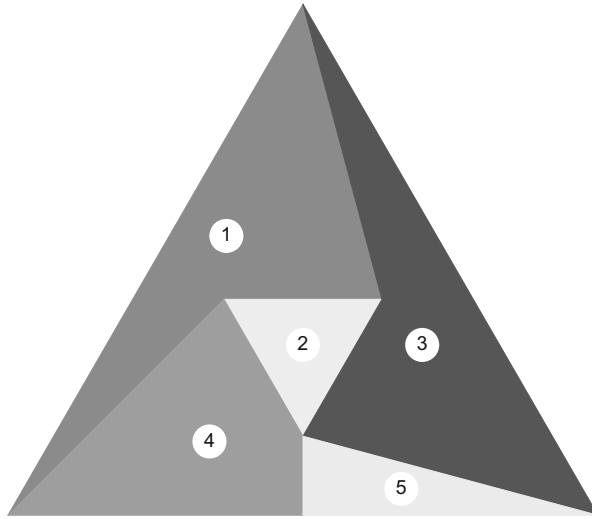
## 2.5 Comments on Problems

Linear elastic behavior and small displacements and rotations [5] are assumed throughout. Truss analyses, especially those pertaining to statically indeterminate cases, encompass all crucial aspects of *linear elastostatic* modeling by finite elements. This is the reason why the entire Appendix B is developed to demonstrate a number of truss problems using *Mathematica*. By following these examples, we can grasp the details of the analytical and computational steps. In all finite element approximations, the shape functions are the basic objects. They are (simply) linear in the axial direction, for truss problems.

In the finite element method, equilibrium and compatibility conditions are enforced *only at nodes*. The (equivalent) nodal forces are obtained as *virtual work quantities* by conjugating the distributed forces with assumed shape functions.

The advantage of *Mathematica* programs is that the reader can design their own examples. This is indeed very strongly suggested. An adequate set of problems, specified by the reader, will provide the skill normally earned through a fixed number of exercises. The graphic codes of Appendix B are indispensable for generating meaningful results. These aspects can be fully comprehended with basic truss examples.

No problems (exercises) are assigned in this chapter. The reader is encouraged to follow Appendix B for self-study. For example, the reader can change all the necessary data to solve truss problems related to Fig. 2.11.



**Fig. 2.11** A modification of Fig. 2.10 in two- and three-dimensions

## References

1. Connor JJ, Faraji S (2013) Fundamentals of structural engineering. Springer, New York
2. Fritzson P (2004) Principles of object-oriented modeling and simulation with Modelica 2.1. Wiley, London
3. Lubliner J, Papadopoulos P (2016) Introduction to solid mechanics, 2nd edn. Springer, New York
4. Ritz W (1908) Über eine neue methode zur lösung gewisser variationalprobleme der mathematischen physik. J Reine Angew Math 135:1–61
5. Spencer AJM (1980) Continuum mechanics. Longman, London (also 1990 Dover, New York)
6. Timoshenko SP, Young DH (1965) Theory of structures, 2nd edn. McGraw-Hill, New York

Finite Element Concepts

A Closed-Form Algebraic Development

Dasgupta, G.

2018, XXXVI, 333 p. 45 illus., Hardcover

ISBN: 978-1-4939-7421-4