Chapter 7

NETWORKS AND TRUSSES *

7.1 Energy Networks

In the previous chapters we saw that governing principles based on common differential equations can often be cast into an exact equivalent form based on a governing integral principal that is frequently based on a stational energy relation, or 'energy balance'. Here we will begin a review of such principals where the primary nodal unknowns are a scalar quantity and then introduce other considerations that come into play when they are vector quantities. The emphasis here is on the network concept where an essentially one-dimensional problem can be conceptually (or actually) extended to two- or three-dimensions because of available connectivity data that describes how basic one-dimensional components (elements) in the system are interacting in the energy balance process overall and at a connecting node of the network. Here we refer to balance equations as some discipline specific governing physical equations that are converted to a system of linear algebraic equations of the form **SD** = **C**. Examples from common engineering disciplines include: Heat Transfer - Fourier's Law, Electrical - Ohm's and Kirchoff's Laws, Chemical - Flick's Law, Mechanics -Newton's Laws, and Structures - Minimum Total Potential Energy.

Many balance equations involve two quantities, **D** and **C**, in the above system equations whose product is an energy measure. If one of the quantities is given at a point, then the second must be computed as a reaction. The balance laws place one of the variables in **D** and the other in **C**. For example, in an electrical resistive circuit network if the voltage, V, at a point is given, then the current, j, necessary to maintain that voltage is a reaction that can be computed. The reverse situation is also true. Their product is the energy, $E = jV$. Similar related pairs used in common applications are: temperature and heat flow in thermal studies; voltage and current in electrical studies; displacement and force in stress analysis; and velocity and pressure in fluid flow models. Here we will utilize the finite element concepts to represent in an energy form the basis laws that engineers are taught to employ on a more localized basis so they can develop equations that are suitable to hand solution. We will be solving the same governing concepts but in a process automated by finite element analysis. For example, our results still satisfy the basis laws:

Thermal Equilibrium Networks: The algebraic sum of the heat flows into a joint, including external sources, is zero. The temperature distribution in an assembly of components minimizes the rate of entropy production.

Electrical Resistive Networks: The algebraic sum of the currents flowing into a joint, including external sources, is zero. The voltage (and resulting current) distribution minimizes the energy in a circuit network.

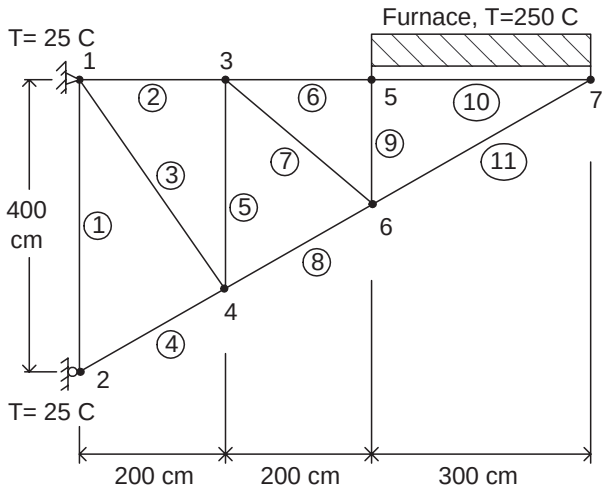
Elastic Structures: The algebraic sum of the force components at a joint, including external sources, is zero. The equilibrium displacements of a structure in equilibrium minimize the total potential energy.

Recall from Chapter 2 that the principle of minimum total potential energy states that the unique set of displacements that occur at equilibrium will satisfy the essential displacement boundary conditions and minimize the total potential energy of the system. The total potential energy is the energy stored in the material minus the mechanical work of the external forces. For the linear spring, the total potential energy, Π, is $\Pi = kd^2/2 - Fd$ for the relative displacement, d of a force, F, in stretching a spring of stiffness, k. Our alternate equilibrium balance statement was the minimization operation: $\partial\Pi/\partial d = 0$, which gives $0 = kd - F$, or simply $F = kd$ which is the well known force-displacement equilibrium covered in physics. Likewise, we could represent the balance of an electrical resistance network as an energy minimization procedure. For a single resistor, having a resistance R, with a voltage drop of V across it due to a current flow, j, we recall that the electrical energy is $W = GV^2/2 - jV$ where $G = 1/R$ is the conductance of the element in the network. The voltage that corresponds to a minimum energy state is governed by $\partial W/\partial V = 0$, so $GV = j$ which we re-arrange to its more familiar form $V = jR$ which is known as Ohm's law. Since energy is a scalar quantity, we can write the energy contribution from each component in a network and then sum (or assemble) those to form the total system energy (or rate of energy production) which is to be minimized. This type of approach usually leads to a system of symmetric linear algebraic equations for the primary unknown at each node or junction in the network. In the current discussion we will put off considering transient effects occuring over time. We will often employ the thermal-electrical-mechanical analogy:

Thermal	Electrical	Mechanical
k = conductivity	G = conductance	K = stiffness
T = temperature	V = voltage	U = displacement
q = heat flow	j = current	F = force

7.2 Thermal Networks

There are many applications where a thermal equilibrium problem can be accurately represented as a two- or three-dimensional assembly of one-dimensional thermal elements. Two such examples would be the space frame of a aircraft or rocket, and the copper or gold path printed on a circuit board. In both cases the primary purpose for the existance of the network (carrying loads, and current, respectively) also gives rise to the ability, or need, to serve as a thermal network. While such thermal systems can be modeled in a continuum sense (as we will do later) it is much more cost effective to employ a network of line components. For a thermal network we can directly utilize the



Elem	Area	Nodes	
1	4.0	1	2
2	4.0	1	3
3	4.0	1	4
4	6.0	2	4
5	4.0	4	3
6	4.0	3	5
7	4.0	3	6
8	6.0	4	6
9	4.0	5	6
10	4.0	5	7
11	6.0	6	7

Steel

Figure 7.2.1 Thermal environment for a truss

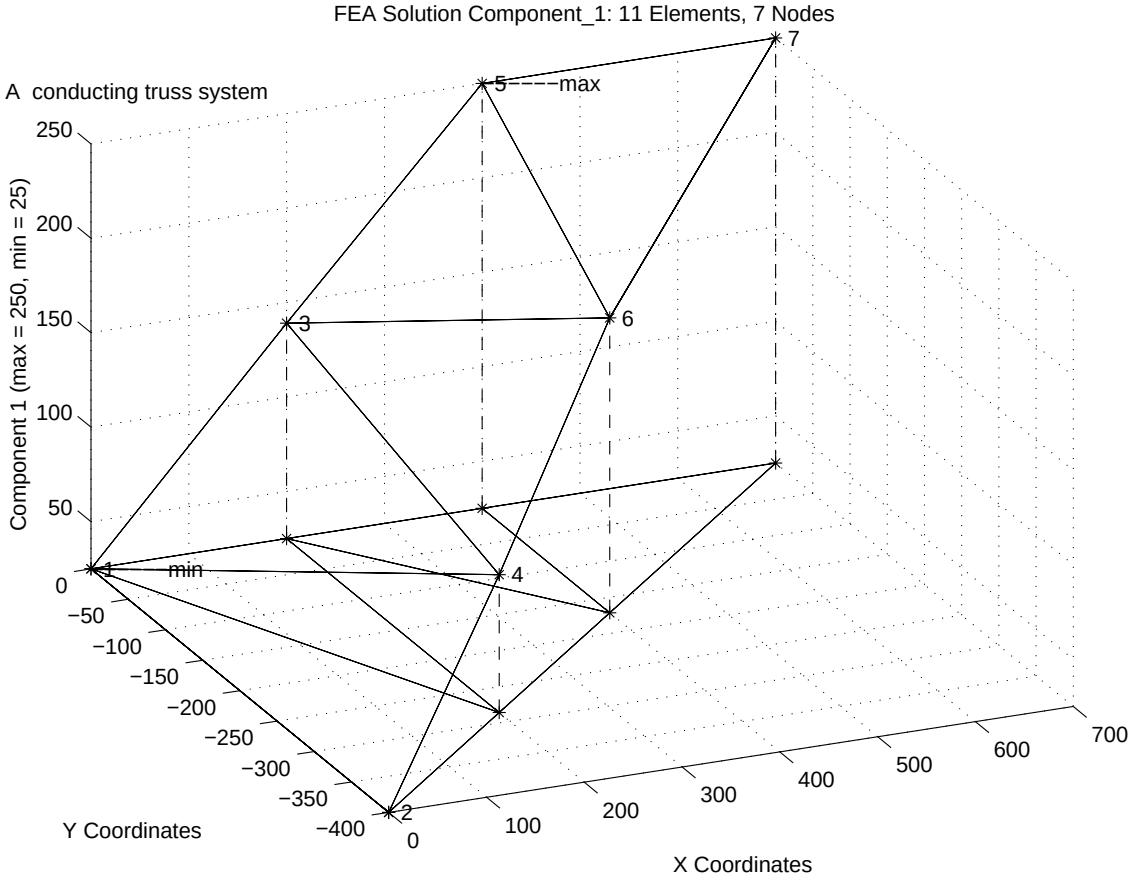


Figure 7.2.2 Truss temperatures and mesh

conduction and convection element matrices developed earlier for a pure one-dimensional formulation. Only the calculation of the physical length of the element needs to be generalized. That is, we simply allow 2-D or 3-D coordinates to be input. The element connectivity data still govern where contributions get scattered into the system solution.

Consider the two-dimensional thermal network shown in Fig. 7.2.1. The far end supports a furnace that keeps the end nodes at a high temperature compared to the wall supports at room temperature. In this case the two-dimensional coordinates are mainly for later use in a structural analysis and are used only to compute the effective thermal lengths in this example. The resulting temperature distributions are shown in Fig. 7.2.2. Here we have limited the element to a linear interpolation between two nodes and no numerical integration is required. The element matrices are so simple they are hand coded as given in Fig. 7.2.3. Note that the comments show that we could allow

```
! begin file my_el_sq_inc                                              ! 1
! ..................................................................... ! 2
!    ***  ELEM_SQ_MATRIX PROBLEM DEPENDENT STATEMENTS FOLLOW ***       ! 3
!      For required REAL (DP) :: S (LT_FREE, LT_FREE)                  ! 4
!      and optional REAL (DP) :: C (LT_FREE)                           ! 5
! ..................................................................... ! 6
! Define any new array or variable types, then give statements         ! 7
                                                                       ! 8
! This file always need for a user defined application                 ! 9
                                                                       !10
! A conducting, convection truss member with heat generation           !11
! Equation: K*A U,xx - H*P (U - U_e) + Q_e = 0,                        !12
! U = temperature, K = conductivity, A = area, h = convection ,        !13
! P = perimeter, Q_e = heat source per unit length                     !14
!    1 *---(K_e, A_e, h_e, P_e, U_e, Q_e)---* 2, Element in xyz        !15
  REAL(DP)  :: L_BAR                          ! Length                 !16
  REAL(DP)  :: K_e, A_e, h_e, P_e, U_e, Q_e  ! properties              !17
                                                                       !18
  IF ( debug_el_sq .or. debug_include )     &                          !19
    WRITE (N_BUG, *) 'Entering my_el_sq_inc'                           !20
                                                                       !21
  L_BAR = SQRT( SUM( (COORD (2, 1:N_SPACE)       &                     !22
                    - COORD (1, 1:N_SPACE)) **2) ) ! Length            !23
  K_e = GET_REAL_LP (1) ! thermal conductivity                         !24
  A_e = GET_REAL_LP (2) ! area of bar                                  !25
  h_e = GET_REAL_LP (3) ! convection coefficent on perimeter           !26
  P_e = GET_REAL_LP (4) ! perimeter of area A_e                        !27
  Q_e = GET_REAL_LP (5) ! source per unit length, BTU/ hr ft           !28
  U_e = GET_REAL_LP (6) ! convecting temperature, F                    !29
                                                                       !30
  S (1, 1) =  K_e * A_e / L_BAR  + h_e * P_e * L_BAR / 3.d0            !31
  S (2, 1) = -K_e * A_e / L_BAR  + h_e * P_e * L_BAR / 6.d0            !32
  S (1, 2) = -K_e * A_e / L_BAR  + h_e * P_e * L_BAR / 6.d0            !33
  S (2, 2) =  K_e * A_e / L_BAR  + h_e * P_e * L_BAR / 3.d0            !34
                                                                       !35
  C (1) = (h_e * P_e * U_e  + Q_e) * L_BAR / 2.d0                      !36
  C (2) = (h_e * P_e * U_e  + Q_e) * L_BAR / 2.d0                      !37
! end file: my_el_sq_inc                                              !38
```

Figure 7.2.3 General matrices for a two-node bar

convection losses as well, and have the ability to give a heat source per unit length, such as an electric motor generating heat at a member. Recall that if we do include convection then the true solution along a member is not linear, but a hyperbolic function. It is not practical to consider an error estimator here for such a network and one would have consider refining the mesh. Convection is usually not important in trusses.

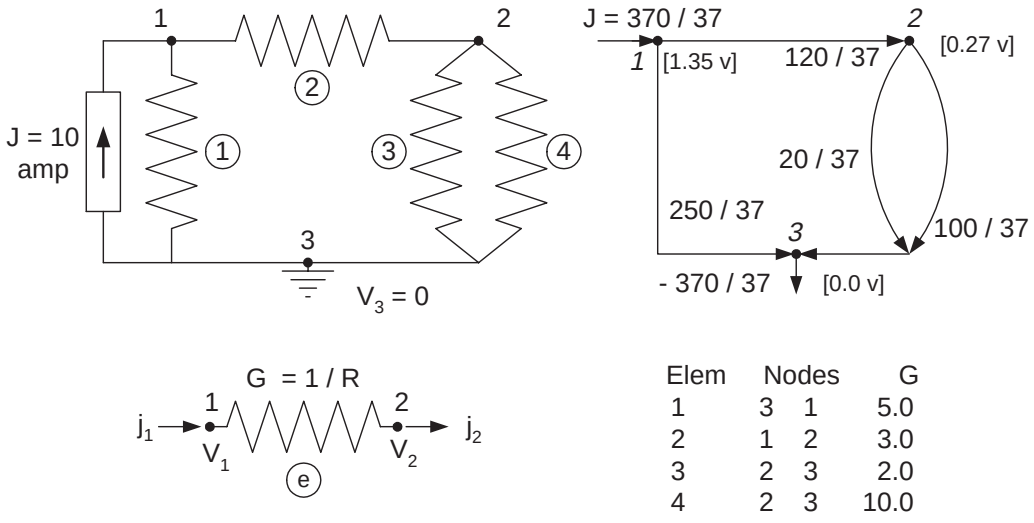
7.3 Electrical Resistance Networks

Consider an electric resistance element connecting two nodes in a DC circuit network. Ohm's law gives the relation between the direct current, j, enterine the element at node 1, the voltage drop $E_2 - E_1$ over its length, and the resistance, R, of the material. Specifically $j = (E_2 - E_1)/R$ so noting that the current flowing in one end is the negative of that flowing out at the other end (node 2) we can express this relation in matrix form:

$$\frac{1}{R^e}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}\begin{Bmatrix} E^e{}_1 \\ E^e{}_2 \end{Bmatrix} = \begin{Bmatrix} j^e{}_1 \\ j^e{}_2 \end{Bmatrix}.$$

Symbolically this element relation is $\mathbf{G}^e\,\mathbf{E}^e = \mathbf{j}^e$ and the corresponding system network balance relationship is $\mathbf{G}\,\mathbf{E} = \mathbf{J}$ (or our previous notation $\mathbf{S}\,\mathbf{D} = \mathbf{C}$). Here vector $\mathbf{J}$ is the resultant of all nodal currents. That is, at each node $\mathbf{J}$ equals the sum of the currents, j^e, from all the connecting elements at that node, plus the external current which is usually zero (except for reactions to applied voltages).

As an example, consider the DC current circuit network, illustrated in Fig. 7.3.1. Assembling the system shown gives the system balance equations:



Elem	Nodes		G
1	3	1	5.0
2	1	2	3.0
3	2	3	2.0
4	2	3	10.0

Figure 7.3.1 A simple current driven network

$$\begin{bmatrix} \frac{1}{20} & -\frac{1}{20} & 0 & 0 \\ -\frac{1}{20} & \left(\frac{1}{20}+\frac{1}{5}+\frac{1}{6}\right) & -\frac{1}{5} & -\frac{1}{6} \\ 0 & -\frac{1}{5} & \frac{1}{5} & 0 \\ 0 & -\frac{1}{6} & 0 & \frac{1}{6} \end{bmatrix} \begin{Bmatrix} E_1 = 140 \\ E_2 \\ E_3 = 90 \\ E_4 = 0 \end{Bmatrix} = \begin{Bmatrix} J_1 \\ 0 \\ J_3 \\ J_4 \end{Bmatrix}$$

Solving the network system equilibrium yields the single unknown voltage: $(1/20 + 1/5 + 1/6)\, E_2 = 0 + 140/20 + 90/5$ or simply $E_2 = 60$ volts. Substituting all the voltages to determine the 'reaction' currents gives $J_1 = 4, J_3 = 4$, and $J_4 = -10$ amps, respectively. That is, external current entered the system at nodes 1 and 3 and was removed at node 4. Post-processing the results gives the current in each element. The results, for the first node in the topology, are

$j_1 = (E_2 - E_1)/R_1 = (60 - 140)/20 = -4$, entering
$j_2 = (E_3 - E_2)/R_2 = (90 - 60)/5 = +6$, exiting
$j_3 = (E_4 - E_2)/R_3 = (0 - 60)/6 = -10$, entering.

In a similar manner the electrical power $P = EJ = J^2R$ can be computed at the system level as a matrix product or by summing similar products at the element level. At the system level $P = \mathbf{E}^T\mathbf{J}$ or $P = [140\ \ 60\ \ 90\ \ 0]^T[4\ \ 0\ \ 6\ \ -10] = 1100$ watts, and this network internal power should match the sum of the power in the elements. That is,

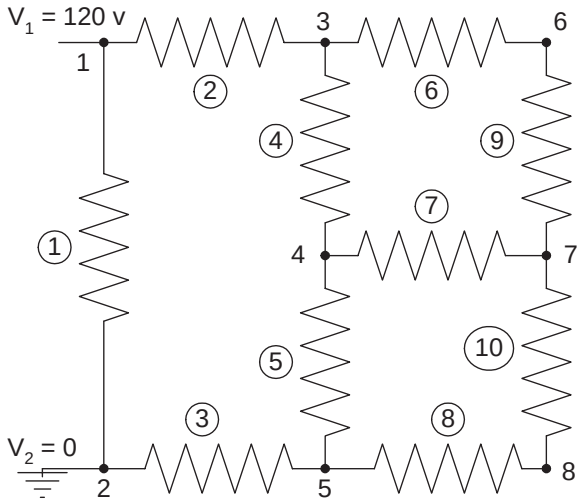
$$P = \sum_e \mathbf{E}^{e^t}\, \mathbf{S}^e\, \mathbf{E}^e$$

$$P_1 = [140 \quad 60] \begin{bmatrix} \frac{1}{20} & -\frac{1}{20} \\ -\frac{1}{20} & \frac{1}{20} \end{bmatrix} \begin{Bmatrix} 140 \\ 60 \end{Bmatrix} = 320$$

$$P_2 = [60 \quad 90] \begin{bmatrix} \frac{1}{5} & -\frac{1}{5} \\ -\frac{1}{5} & \frac{1}{5} \end{bmatrix} \begin{Bmatrix} 60 \\ 90 \end{Bmatrix} = 180$$

$$P_3 = [60 \quad 0] \begin{bmatrix} \frac{1}{6} & -\frac{1}{6} \\ -\frac{1}{6} & \frac{1}{6} \end{bmatrix} \begin{Bmatrix} 60 \\ 0 \end{Bmatrix} = 600$$

and the total power is $P = \sum_e P^e = (320 + 180 + 600) = 110$ watts, as expected. These procedures can be used on DC systems in general, but it can become difficult to clarify the topology and the boundary conditions. A typical implementation of a static DC circuit is given in Fig. 7.3.3. When it is applied to the network in Fig. 7.3.2 the computed voltages are shown in Fig. 7.3.4. They are listed with the element current



Elem	Nodes		R
1	1	2	36.
2	1	3	1.
3	2	5	2.
4	3	4	10.
5	4	5	20.
6	3	6	3.
7	4	7	5.
8	5	8	4.
9	6	7	7.
10	7	8	6.

Figure 7.3.2 A voltage driven network

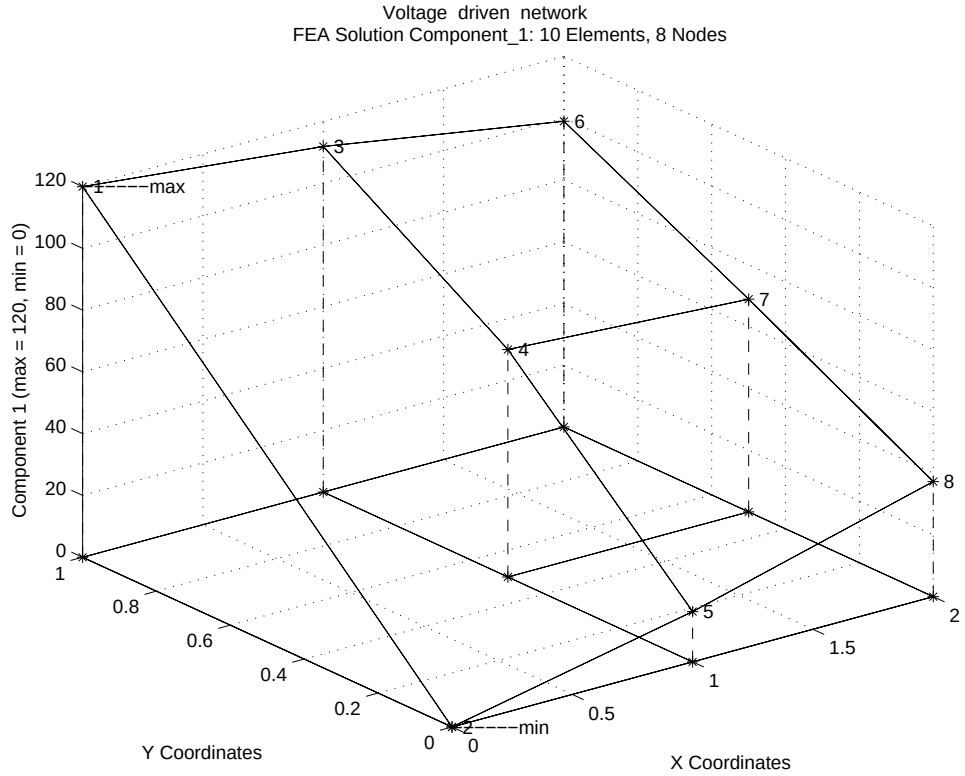


Figure 7.3.4 Computed nodal voltage levels


```
! ..................................................... ! 1
!  ** ELEM_SQ_MATRIX PROBLEM DEPENDENT STATEMENTS FOLLOW *** ! 2
! ..................................................... ! 3
! Define new array or variable types, then give statements  ! 4
! NOTE: ELEMENT REACTIONS ARE THE IN & OUT CURRENT FLOWS    ! 5
  REAL(DP) :: resistance, conductance ! R, 1/R              ! 6
                                                            ! 7
!  Get resistance                                           ! 8
  resistance = GET_REAL_LP (1) ! real element property      ! 9
  IF ( resistance > 0.d0 ) THEN                            !10
    conductance = 1.d0 / resistance                        !11
  ELSE                                                     !12
    PRINT *, 'WARNING: Invalid resistance, element ', IE   !13
    N_WARN = N_WARN + 1 ; conductance = 1.d0               !14
  END IF  ! valid data                                     !15
                                                           !16
!  Conductance matrix                                      !17
  S (1,1) =  conductance  ; S (2,1) = -conductance         !18
  S (1,2) = -conductance  ; S (2,2) =  conductance         !19
! *** END ELEM_SQ_MATRIX PROBLEM DEPENDENT STATEMENTS ***   !20
```

Figure 7.3.3 A typical DC electric network element

```
*** REACTION RECOVERY ***                          ! 1
   NODE, PARAMETER,     REACTION, EQUATION         ! 2
      1,  DOF_1,       1.1461E+01     1           ! 3
      2,  DOF_1,      -1.1461E+01     2           ! 4
                                                   ! 5
***  OUTPUT OF RESULTS IN NODAL ORDER  ***         ! 6
    NODE,  X-Coord,    Y-Coord,    DOF_1,         ! 7
       1  0.0000E+00  1.0000E+00  1.2000E+02      ! 8
       2  0.0000E+00  0.0000E+00  0.0000E+00      ! 9
       3  1.0000E+00  1.0000E+00  1.1187E+02      !10
       4  1.0000E+00  5.0000E-01  7.3625E+01      !11
       5  1.0000E+00  0.0000E+00  1.6255E+01      !12
       6  2.0000E+00  1.0000E+00  9.8964E+01      !13
       7  2.0000E+00  5.0000E-01  6.8845E+01      !14
       8  2.0000E+00  0.0000E+00  3.7291E+01      !15
                                                   !16
** ELEMENT REACTION, AND INTERNAL SOURCES **      !17
ELEMENT  NODE  DOF   REACTION      ELEM_SOURCE     !18
    1     1    1    3.33333E+00    0.00000E+00    !19
    2     1    1    8.12749E+00    0.00000E+00    !20
    3     2    1   -8.12749E+00    0.00000E+00    !21
    4     3    1    3.82470E+00    0.00000E+00    !22
    5     4    1    2.86853E+00    0.00000E+00    !23
    6     3    1    4.30279E+00    0.00000E+00    !24
    7     4    1    9.56175E-01    0.00000E+00    !25
    8     5    1   -5.25896E+00    0.00000E+00    !26
    9     6    1    4.30279E+00    0.00000E+00    !27
   10     7    1    5.25896E+00    0.00000E+00    !28
```

Figure 7.3.5 Network voltage and current results

(element reactions) in Fig. 7.3.5. There we see that the external reaction currents are 11.46 amps entering at node 1 and an equal amount leaving at ground node 2. The element reactions show how the current splits among the elements. For example at node 1 we see that elements 1 and 2 take 3.33 and 8.13 amps (29 % and 71%), respectively of

the incoming current.

7.4 Trusses as a Network

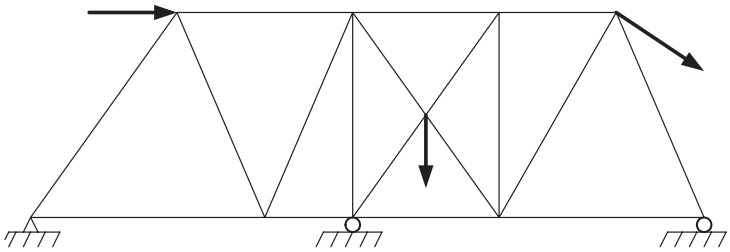
The truss element is a very common structural member. Recall that a truss element is a "two force member". That is, it is loaded by two equal and opposite collinear forces. These two forces act along the line through the two connection points of the member. In elementary statics we compute the forces in truss elements as if they were rigid bodies. However, there was a class of problems, called statically indeterminant, that could not be solved by treating the members as rigid bodies. With the finite element approach we will be able to solve both classes of problems. In Sec. 3.2 the equilibrium equation for an elastic bar was developed. Clearly, the elastic bar is a special form of a truss member. To extend the previous work to include trusses in two- or three-dimensions basically requires some review of analytic geometry. Thus, we begin by reviewing that subject.

7.4.1 Direction Cosines

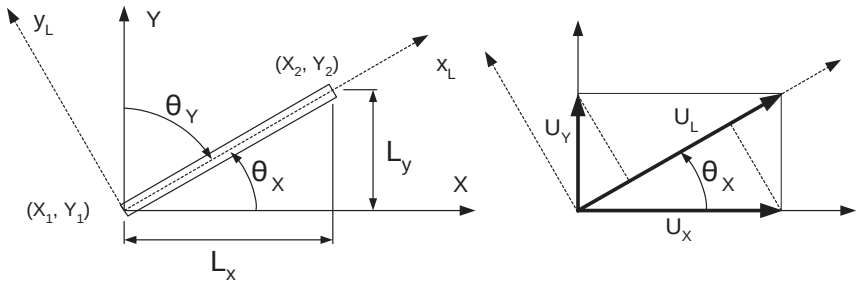
Consider a directed line segment in global space going from point 1 at (x_1, y_1, z_1) to point 2 at (x_2, y_2, z_2). Then the length of the line between the two points has components parallel to the axes of $L_x = x_2 - x_1$, $L_y = y_2 - y_1$, $L_z = z_2 - z_1$ and the total length is $L^2 = (L_x^2 + L_y^2 + L_z^2)$. Specifying the end points of a line is a common way of locating its direction in space. Another common way to describe the direction is to give the *direction angles* or the corresponding *direction cosines*. Let the direction angles from the x-, y-, and z-axes to the line segment be denoted by ϕ_x, ϕ_y, and ϕ_z, respectively. Recall the relation between the total magnitude of a vector and its components, i.e., $L_x = L \operatorname{Cos} \phi_x$, etc. We generally will find the inverse geometric relation more useful. Specifically the direction cosines become: $\operatorname{Cos} \phi_x = L_x / L$, $\operatorname{Cos} \phi_y = L_y / L$, and $\operatorname{Cos} \phi_z = L_z / L$.

For two-dimensional problems we will assume that the structure lies in the global x-y plane so that $L_z = 0$, $\operatorname{Cos} \phi_z = 0$, and $\phi_z = 90$. In that special case only one angle is required to describe the direction rather than the usual three. It is common then to select ϕ_x as the required angle and to omit reference to $\phi_y = 90 - \phi_x$ and to replace the second direction cosine with the relation $\operatorname{Cos} \phi_y = \operatorname{Sin} \phi_x$ (for $\phi_z = 90$). This is illustrated in Fig. 7.4.1. For two-dimensional problems one can utilize the simplicity of referring to a single angle. However, if one wants to automate the analysis for two- and three-dimensional problems then it is best in the long run to refer to the direction cosines.

To extend the bar element to a general truss element we need to consider the relations between a local coordinate system that is parallel and perpendicular to the element and the fixed global coordinate directions. Let the local x-axis lie along the member, that is, it passes through the two end points of the member. This means that the direction cosines of the local x-axis are the same as those for the line segment. The bar element had a single displacement, u, at any point. That local displacement vector will have components in the global space. Let the global displacements of a point be denoted by u_x, u_y, and u_z. To be consistent with this, one could also define three local components of the displacement. For a bar element the local y- and z-components are identically zero. Later we will consider members that have no zero local components.



a) Typical truss structure



b) Coordinates and displacement components

Figure 7.4.1 A truss structure and components

Thus, we will consider the general case of transformation of local displacement components. Referring to Fig. 7.4.1 again, one finds from geometry that the local x displacement is related to the two-dimensional global displacements by

$$u_{xL} = u_{xg} \text{ Cos } \theta_x + u_{yg} \text{ Cos } \theta_y .$$

Similarly, if there was a local y-component of displacement it would be related to the global components by $u_{yL} = -u_{xg} \text{ Cos } \theta_y + u_{yg} \text{ Cos } \theta_x$. Writing these identities in a matrix form in terms of $\theta_x = \theta$

$$\left\{ \begin{array}{c} u_x \\ u_y \end{array} \right\}_L = \left[\begin{array}{cc} \text{Cos } \theta & \text{Sin } \theta \\ -\text{Sin } \theta & \text{Cos } \theta \end{array} \right] \left\{ \begin{array}{c} u_x \\ u_y \end{array} \right\}_g \tag{7.4.1}$$

or symbolically this *transformation* is $\mathbf{u}_L = \mathbf{t}(\theta)\, \mathbf{u}_g$ where $\mathbf{t}$ is a nodal *transformation matrix* and $\mathbf{u}_g$ and $\mathbf{u}_L$ denote the global and local displacement components, respectively, at a point. If this relation is written at each node of the element it defines the element dof transformation matrix, $\mathbf{T}$. Specifically,

$$\left\{\begin{array}{c} u_{1x} \\ u_{1y} \\ \hline\hline u_{2x} \\ u_{2y} \end{array}\right\}^e_L = \left[\begin{array}{cc|cc} \text{Cos}\,\theta & \text{Sin}\,\theta & 0 & 0 \\ -\text{Sin}\,\theta & \text{Cos}\,\theta & 0 & 0 \\ \hline\hline 0 & 0 & \text{Cos}\,\theta & \text{Sin}\,\theta \\ 0 & 0 & -\text{Sin}\,\theta & \text{Cos}\,\theta \end{array}\right] \left\{\begin{array}{c} u_{1x} \\ u_{1y} \\ \hline\hline u_{2x} \\ u_{2y} \end{array}\right\}^e_g$$

or

$$\mathbf{u}^e_L \;=\; \mathbf{T}(\theta)\,\mathbf{u}^e_g\,. \tag{7.4.2}$$

The same type of coordinate transformation will apply to components of the element force vector, $\mathbf{P}^e$, namely:

$$\mathbf{P}^e_L \;=\; \mathbf{T}(\theta)\,\mathbf{P}^e_g\,. \tag{7.4.3}$$

Notice that the transformation matrix is square. Thus, the inverse transformation can be found by inverting the matrix T. Therefore,

$$\mathbf{u}_g \;=\; \mathbf{T}^{-1}\,\mathbf{u}_L\,, \qquad \mathbf{P}_g \;=\; \mathbf{T}^{-1}\,\mathbf{P}_L\,. \tag{7.4.4}$$

If one carries out the inversion process, an interesting result is obtained. Specifically, we find that the inverse of the transformation is the same as its transpose. This is always true, and it makes our calculations much easier since we can write

$$\mathbf{T}^{-1} \;=\; \mathbf{T}^T. \tag{7.4.5}$$

A matrix with this property is called an *orthogonal* matrix. Therefore, the simple way to write the inverse transformation is

$$\mathbf{u}_g \;=\; \mathbf{T}^T\,\mathbf{u}_L\,. \tag{7.4.6}$$

7.4.2 Transformation of Element Matrices

Our ultimate goal is to solve the global equilibrium equations. This requires that all elements be referred to a single global coordinate system, and that the assembly of element contributions be relative to that system. Therefore, before we can assemble the element stiffness and load matrices they must be written relative to the global axes. This means that we need to define global versions of the element matrices, say $\mathbf{S}^e_g$ and $\mathbf{C}^e_g$. Clearly, they are somehow related to the corresponding local element matrices, $\mathbf{S}^e_L$ and $\mathbf{C}^e_L$. To gain some insight into the relation between the two systems recall that the element behavior was defined in terms of the total potential energy, Π^e, of the element. Since that quantity is a scalar, its value must be the same regardless of whether it is computed in element coordinates or global coordinates. If we compute Π^e using Eq. (4.7) in local coordinates the result is

$$\Pi^e \;=\; \tfrac{1}{2}\,\mathbf{u}^{e^T}_L\,\mathbf{S}^e_L\,\mathbf{u}^e_L - \mathbf{u}^{e^T}_L\,\mathbf{C}^e_L\,. \tag{7.4.7}$$

By way of comparison, if it is calculated in global coordinates

$$\Pi^e \;=\; \tfrac{1}{2}\,\mathbf{u}^{e^T}_g\,\mathbf{S}^e_g\,\mathbf{u}^e_g - \mathbf{u}^{e^T}_g\,\mathbf{C}^e_g\,. \tag{7.4.8}$$

The two forms can be more easily compared if Eq. (7.4.7) is also written in terms of the global components of the displacements of the element. Before doing that, let us recall the form of the element stiffness and load matrices for a bar parallel to the x-axis:

$$\mathbf{S}_L^e = \frac{E^e A^e}{L^e}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}, \qquad \mathbf{C}_L^e = \begin{Bmatrix} C_{1x} \\ C_{2x} \end{Bmatrix}$$

where C_1 and C_2 represent the resultant loads along the local x-axis. Since the global structure will have two displacements per node it will be useful to rewrite the element equations in terms of two local displacements per node. Specifically, the expanded element equations for the equilibrium of a single element are

$$\frac{E^e A^e}{L}\begin{bmatrix} 1 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}\begin{Bmatrix} u_{1x} \\ u_{1y} \\ u_{2x} \\ u_{2y} \end{Bmatrix}_L^e = \begin{Bmatrix} C_{1x}^e \\ 0 \\ C_{2x}^e \\ 0 \end{Bmatrix}_L .$$

Note that the stiffness matrix has been expanded by adding rows and columns of zeros to correspond to the local y displacement. That was done because the element cannot resist loads in the local y direction. The above expanded element matrices would be substituted into Eq. (7.4.7). Next, substituting the transformation identity of Eq. (7.4.2) into Eq. (7.4.7) yields $\Pi^e = \frac{1}{2}\,\mathbf{u}_g^{e^T}(\mathbf{T}^{e^T}\,\mathbf{S}_L^e\,\mathbf{T}^e)\,\mathbf{u}_g^e - \mathbf{u}_g^e(\mathbf{T}^{e^T}\,\mathbf{C}_g^e)$. Comparing this scalar with the same quantity in Eq. (7.4.8) gives the desired identities

$$\mathbf{S}_g^e = \mathbf{T}^{e^T}\,\mathbf{S}_L^e\,\mathbf{T}^e \text{, and } \quad \mathbf{C}_g^e = \mathbf{T}^{e^T}\,\mathbf{C}_L^e . \tag{7.4.9}$$

Of major importance here is that Eq. (7.4.9) is not restricted to truss elements. For certain types of elements it would be simpler to form the global element matrices numerically by matrix multiplication. For the truss element in two dimensions the products in these transformations are easily written out. The results are

$$\mathbf{S}^e = \frac{E^e A^e}{L^e}\begin{bmatrix} \lambda\lambda & \lambda\mu & -\lambda\mu & -\lambda\mu \\ \lambda\mu & \mu\mu & -\lambda\mu & -\mu\mu \\ -\lambda\mu & -\lambda\mu & \lambda\lambda & \lambda\mu \\ -\lambda\mu & -\mu\mu & \lambda\mu & \mu\mu \end{bmatrix}^e \tag{7.4.10}$$

and

$$\mathbf{C}^{eT} = \lfloor\, \lambda C_{1x} \quad -\mu C_{1x} \quad \lambda C_{2x} \quad -\mu C_{2x} \,\rfloor^e \tag{7.4.11}$$

where $\lambda = \text{Cos}\,\phi_x = L_x/L$, $\mu = \text{Cos}\,\phi_y = L_y/L = \text{Sin}\,\phi_x$. A similar set of transformed global stiffness and force vectors can be obtained for a truss element located in three-dimensional space.

7.4.3 Direct Energy Approach

The above transformation process is valid in many structural applications and is the usually way to see truss, frame, plate, and shell elements developed. The truss element can be expressed directly from the energy approach and this leads to a simpler program. In Chapter 3 on variational methods we saw that we could define the work and energy terms in a general integral form. For an axially loaded bar the stiffness and load matrices are given in Eqs. (3.20-22) and the initial thermal strain effects are in Eq. (3.32). Those equations need the strain-displacement matrix $\mathbf{B}^e$ expressed in the global coordinate system. Here we need to rotate the bar and express its behavior in terms in terms of four

displacement components, instead of two as before. The two axial displacements are related to the four truss member displacements by a sub-set of Eq (7.4.2), namely

$$\mathbf{u}_{axial} = \boldsymbol{\tau}^e \mathbf{u}^e$$

$$\left\{ \begin{matrix} u_{1x} \\ u_{2x} \end{matrix} \right\}^e_{axial} = \left[\begin{matrix} \text{Cos}\,\theta & \text{Sin}\,\theta & 0 & 0 \\ 0 & 0 & \text{Cos}\,\theta & \text{Sin}\,\theta \end{matrix} \right] \left\{ \begin{matrix} u_{1x} \\ u_{1y} \\ u_{2x} \\ u_{2y} \end{matrix} \right\}^e_g$$

Thus we can get the global $\mathbf{B}^e$ matrix as $\boldsymbol{\varepsilon} = \mathbf{B}_{axial}\mathbf{u}_{axial} = (\mathbf{B}_{axial}\boldsymbol{\tau}^e)\mathbf{u}^e = \mathbf{B}^e\mathbf{u}^e$ which simplifies to

$$\mathbf{B}^e = [-\text{Cos}\,\theta \quad -\text{Sin}\,\theta \quad \text{Cos}\,\theta \quad \text{Sin}\,\theta]/L^e$$

Assuming constant properties we can write the stiffness matrix by inspection as the matrix product

$$\mathbf{S}^e = \mathbf{B}^{e^T} E^e \mathbf{B}^e A^e L^e$$

and the truss member load vector due to any temperature rise (from stress free) is

$$\mathbf{C}^e_t = \mathbf{B}^{e^T} E^e \alpha^e \Delta t \; A^e \; L^e \, .$$

Another common loading condition is the member weight. Here we just need to put half the weight at each end node and assume that Y is vertical so there are no X components of this load. We get the weight from the weight density, γ, times the member volume. This form is much simpler to program than the previous one. A typical implementation is shown in Fig. 7.5.1.

The calculation of the stiffness, thermal and weight effects should be clear. In this case $\mathbf{B}^e$ has only one row but it usually has one per spatial dimension. The last action, in line 46, is to save data necessary to recover the strains and stresses in each element. It gets activated if the data keyword "post_el" is present. If it is present then each element gets post-processed as shown in Fig. 7.5.2. There the same data (modulus of elasticity, $\mathbf{B}^e$ matrix, and initial strains) are recovered from sequential storage. Multiplying $\mathbf{B}^e$ by the gathered displacements yields the mechanical strains, in line 25. We invoke Hooke's law, generalized to include initial strains, to get the one stress component, in line 29, and then present the three items for output.

The text by Logan [6] gives a detailed hand calculation of a two bar truss with one heated member and no external loads considered. The input data are shown in Fig. 7.5.3 and the computed result shown in Fig. 7.5.4 was exact, as summarized in the output of Fig. 7.5.5.

7.4.4 Example Structure Calculations

Consider the example three bar truss shown in Fig. 7.4.2. Assume that all three members have the same area and modulus of elasticity. The structure is described by


```
! .........................................................   ! 1
! ***  ELEM_SQ_MATRIX PROBLEM DEPENDENT STATEMENTS FOLLOW ***   ! 2
! .........................................................   ! 3
! Define any new local array or variable types, then statements ! 4
                                                               ! 5
!  A TWO-DIMENSIONAL TRUSS BY DIRECT ENERGY APPROACH           ! 6
!  ELEMENT REAL PROPERTIES: (1) = AREA, (2) = ELASTIC MODULUS   ! 7
!  (3) = TEMP RISE, (4) = COEFF EXPANSION, (5) = WEIGHT DENSITY ! 8
                                                               ! 9
  REAL(DP) :: X_I, X_J, Y_I, Y_J     ! coordinates             !10
  REAL(DP) :: D_X, D_Y, BAR_L        ! lengths                 !11
  REAL(DP) :: DELTA_T, ALPHA         ! temp rise, expansion    !12
  REAL(DP) :: AREA, GAMMA            ! area, wt. density       !13
  REAL(DP) :: M_E, THERMAL           ! modulus, thermal strain !14
  REAL(DP) :: C_X, C_Y               ! direction cosines       !15
                                                               !16
! Get geometry                                                 !17
    X_I = COORD (1, 1) ;        X_J = COORD (2, 1)             !18
    Y_I = COORD (1, 2) ;        Y_J = COORD (2, 2)             !19
                                                               !20
! Get properties for this element                             !21
    AREA    = GET_REAL_LP (1); M_E   = GET_REAL_LP (2)         !22
    DELTA_T = GET_REAL_LP (3); ALPHA = GET_REAL_LP (4)         !23
    GAMMA   = GET_REAL_LP (5)                                  !24
                                                               !25
! Find bar length and direction cosines                       !26
    D_X = X_J - X_I  ; D_Y = Y_J - Y_I    ! lengths            !27
    BAR_L = SQRT (D_X * D_X + D_Y * D_Y)  ! total length       !28
    C_X = D_X / BAR_L ; C_Y = D_Y / BAR_L ! cosines            !29
                                                               !30
! Form global strain-displacement matrix                      !31
    B (1, :) = (/ - C_X, - C_Y,  C_X, C_Y /) / BAR_L           !32
                                                               !33
! Form global stiffness, S = B' EAL B                         !34
    S = M_E * AREA * BAR_L * MATMUL ( TRANSPOSE (B), B )       !35
                                                               !36
! Initial (thermal) strain loading                            !37
    THERMAL = ALPHA * DELTA_T                        ! strain  !38
    C       = B (1, :) * M_E * THERMAL * AREA * BAR_L ! force  !39
                                                               !40
! Weight load, in negative Y-direction (wt density * volume)  !41
    C = C + (/ 0.d0, -0.5d0, 0.d0, -0.5d0 /) &  ! components  !42
          * GAMMA * AREA * BAR_L                ! total weight !43
                                                               !44
! Save for stress post-processing (set post_el in keywords)   !45
    IF ( N_TAPE1 > 0 ) WRITE (N_TAPE1) M_E, B, THERMAL         !46
!  End of application dependent code                          !47
```

Figure 7.5.1 A truss element stiffness and loads


```
! ..................................................................... ! 1
!  *** POST_PROCESS_ELEM PROBLEM DEPENDENT STATEMENTS FOLLOW *** ! 2
! ..................................................................... ! 3
! Define any new array or variable types, then give statements   ! 4
                                                                       ! 5
!  A TWO-DIMENSIONAL TRUSS BY DIRECT ENERGY APPROACH              ! 6
!  ELEMENT REAL PROPERTIES: (1) = AREA, (2) = ELASTIC MODULUS     ! 7
!  (3) = TEMP RISE, (4) = COEFF EXPANSION, (5) = WEIGHT DENSITY   ! 8
                                                                       ! 9
!    STRESS = M_E * (MECHANICAL STRAIN - INITIAL STRAIN)          !10
                                                                       !11
  REAL(DP) :: THERMAL              ! initial strain               !12
  REAL(DP) :: M_E                  ! modulus of elasticity        !13
  LOGICAL, SAVE :: FIRST = .TRUE.  ! printing                     !14
                                                                       !15
   IF ( FIRST ) THEN                   ! first call               !16
     FIRST = .FALSE. ; WRITE (6, 5)    ! print headings           !17
     5 FORMAT (’        E L E M E N T    S T R E S S E S’, /,  & !18
      & ’ ELEMENT      STRESS   MECH. STRAIN    THERMAL STRAIN’) !19
   END IF ! first call                                             !20
                                                                       !21
!-->  Read stress strain data from N_TAPE1 (set by post_el)       !22
   READ (N_TAPE1) M_E, B, STRAIN_0 (1) ! THERMAL = STRAIN_0       !23
                                                                       !24
!--> Calculate mechanical strain, STRAIN = B * D                  !25
   STRAIN (1) = DOT_PRODUCT ( B(1, :), D )                         !26
                                                                       !27
!-->  Generalized Hooke’s Law                                     !28
   STRESS (1) = M_E * (STRAIN (1) - STRAIN_0 (1))                  !29
                                                                       !30
   WRITE (6, 1) IE, STRESS (1), STRAIN (1), STRAIN_0 (1)           !31
   1 FORMAT (I5, 3ES15.5)                                          !32
!  *** END POST_PROCESS_ELEM PROBLEM DEPENDENT STATEMENTS ***     !33
```

Figure 7.5.2 Post-processing the truss


```
title "Logan 3rd Ed. thermal loaded truss"                 ! 1
nodes    3 ! Number of nodes in the mesh                   ! 2
elems    2 ! Number of elements in the system              ! 3
el_real  5 ! Number of real properties per element         ! 4
dof      2 ! Number of unknowns per node                   ! 5
el_nodes 2 ! Maximum number of nodes per element           ! 6
space    2 ! Solution space dimension                      ! 7
b_rows   1 ! Number of rows in the B (operator) matrix     ! 8
shape    1 ! Element shape, 1=line, 2=tri, 3=quad, 4=hex   ! 9
post_el    ! Require post-processing, create n_tape1       !10
remarks 12 ! Number of user remarks                        !11
quit ! keyword input, begin remarks                        !12
Logan Example 15.3, 2-D Truss with temperature rise        !13
occuring in bar (1) only                                   !14
  1 o  Y_Roller          E = 30,000 ksi,  A = 2 in^2       !15
    | \                  8 ft high, 6 ft wide (96 by 72)   !16
   (1) (2)               alpha = 7e-6, rise = 75 F         !17
    |     \              Y_1 = 0.0333 inch                 !18
  2 *       * 3      Stress: -5,333, + 6,666 psi          !19
  Pin       Pin                                            !20
ELEMENT REAL PROPERTIES:                                   !21
(1) = AREA,      (2) = MODULUS OF ELASTICITY,              !22
(3) = TEMP RISE, (4) = COEF THERMAL EXPANSION,             !23
(5) = WEIGHT DENSITY                                       !24
 1    10  0.0       96.0   ! node, bc flag, x, y          !25
 2    11  0.0        0.0                                  !26
 3    11  72.0       0.0                                  !27
    1    1    2  ! element, two nodes                      !28
    2    1    3                                          !29
  1    1  0.0    ! node, direction, given displacement    !30
  2    1  0.0                                             !31
  2    2  0.0                                             !32
  3    1  0.0                                             !33
  3    2  0.0                                             !34
1 2. 30.e6 75. 7.e-6 0.  ! elem, properties                !35
2 2. 30.e6 0.0 7.e-6 0.  ! elem, properties                !36
```

Figure 7.5.3 A simple thermally loaded truss

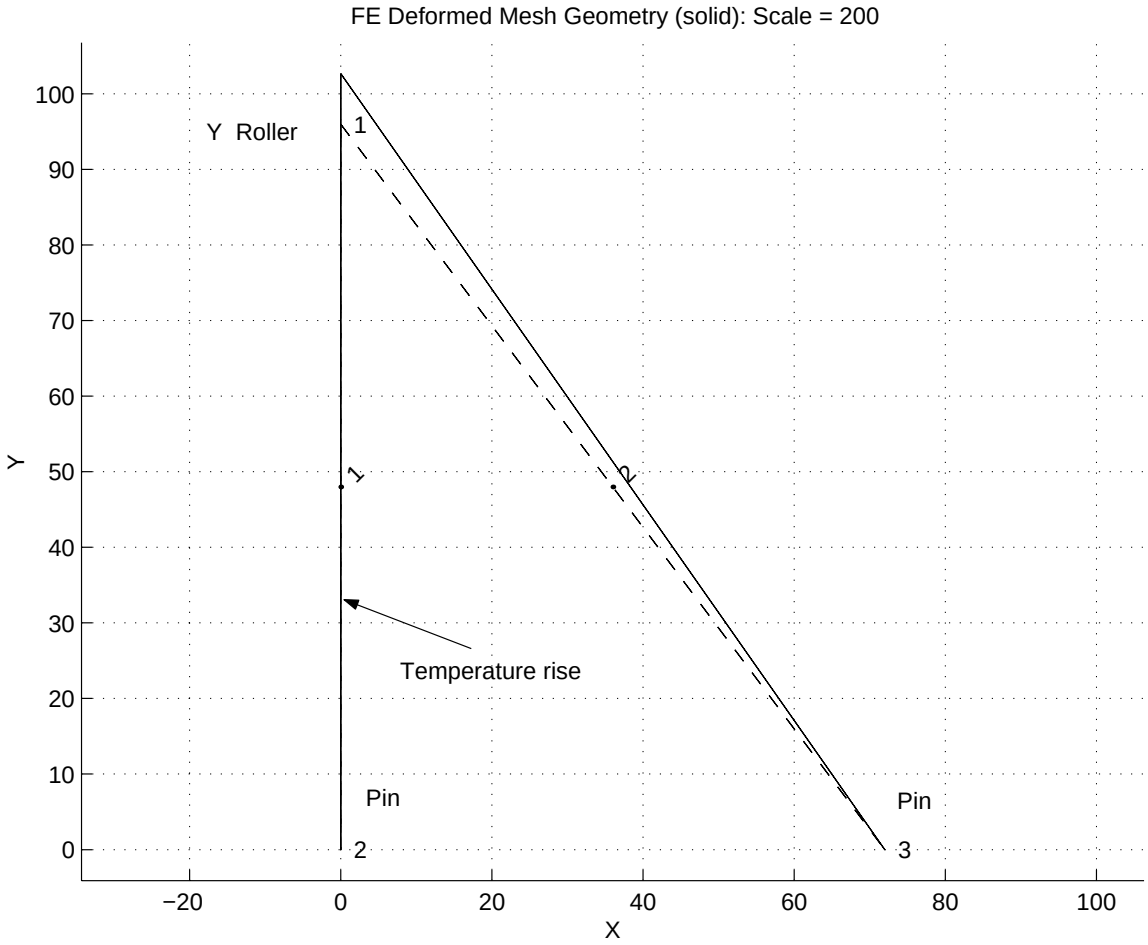


Figure 7.5.4 Deformation of the two bar truss

```
*** REACTION RECOVERY ***                                          ! 1
NODE, PARAMETER,     REACTION, EQUATION                            ! 2
   1,  DOF_1,       -8.0000E+03     1                              ! 3
   2,  DOF_1,        0.0000E+00     3                              ! 4
   2,  DOF_2,        1.0667E+04     4                              ! 5
   3,  DOF_1,        8.0000E+03     5                              ! 6
   3,  DOF_2,       -1.0667E+04     6                              ! 7
                                                                   ! 8
***  OUTPUT OF RESULTS IN NODAL ORDER  ***                         ! 9
NODE,   X-Coord,    Y-Coord,    DOF_1,      DOF_2,                !10
   1  0.0000E+00  9.6000E+01  0.0000E+00  3.3333E-02 !11
   2  0.0000E+00  0.0000E+00  0.0000E+00  0.0000E+00 !12
   3  7.2000E+01  0.0000E+00  0.0000E+00  0.0000E+00 !13
                                                                   !14
        E L E M E N T    S T R E S S E S                           !15
ELEMENT      STRESS   MECH. STRAIN    THERMAL STRAIN               !16
   1   -5.33333E+03    3.47222E-04    5.25000E-04                  !17
   2    6.66667E+03    2.22222E-04    0.00000E+00                  !18
```

Figure 7.5.5 Output summary for the two bar truss

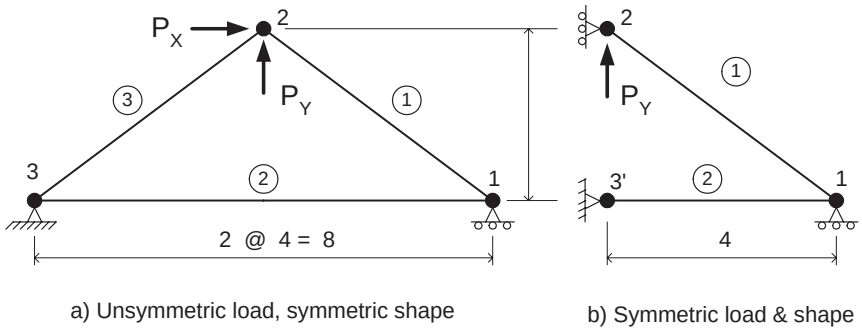


Figure 7.4.2 A three-bar truss structure

Element	L_x	L_y	L	Topology		EA
1	4	3	5	1	2	1000
2	8	0	8	1	3	1000
3	4	−3	5	2	3	1000

The structure is pinned at node 1 and on a horizontal roller at node 3. No distributed loads or thermal loads are considered on the bars. Thus, for each element $\mathbf{C}^e = \mathbf{0}$. Only nodal loads are externally applied. Their values, at node 2 are $P_x = 10$, and $P_y = -20$. From Eq. (7.4.11) the element stiffness matrices, when transformed to the global axes, have the values of

$$e = 1: \qquad \text{Global}$$

$$S^e = \frac{1000}{125}\begin{bmatrix} 16 & 12 & -16 & -12 \\ 12 & 9 & -12 & -9 \\ -16 & -12 & 16 & 12 \\ -12 & -9 & 12 & 9 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix}$$

$$e = 2: \qquad \text{Global}$$

$$S^e = \frac{1000}{512}\begin{bmatrix} 64 & 0 & -64 & 0 \\ 0 & 0 & 0 & 0 \\ -64 & 0 & 64 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix} \begin{matrix} 1 \\ 2 \\ 5 \\ 6 \end{matrix}$$

$$e = 3: \qquad \text{Global}$$

$$S^e = \frac{1000}{125}\begin{bmatrix} 16 & -12 & -16 & 12 \\ -12 & 9 & 12 & -9 \\ -16 & 12 & 16 & -12 \\ 12 & -9 & -12 & 9 \end{bmatrix} \begin{matrix} 3 \\ 4 \\ 5 \\ 6 \end{matrix}.$$

The assembled system equilibrium equations are

$$\begin{bmatrix} (128+125) & 96 & -128 & -96 & -125 & 0 \\ & 72 & -96 & -72 & 0 & 0 \\ & & (128+128) & (96-96) & -128 & 96 \\ & & & (72+72) & 96 & -72 \\ & & & & (125+128) & -96 \\ \text{symmetric} & & & & & 72 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_3 \\ v_3 \end{Bmatrix} = \begin{Bmatrix} 0 \\ 0 \\ P_x \\ P_y \\ 0 \\ 0 \end{Bmatrix} + \begin{Bmatrix} R_1 \\ R_2 \\ 0 \\ 0 \\ 0 \\ R_3 \end{Bmatrix}.$$

However, three displacements (u_1, v_1, and $v_{3)}$ are prescribed to be zero. Modifying the above equations to include the boundary conditions reduces them to

$$\begin{bmatrix} 256 & 0 & -128 \\ & 144 & 96 \\ \text{Sym.} & & 253 \end{bmatrix} \begin{Bmatrix} u_2 \\ v_2 \\ u_3 \end{Bmatrix} = \begin{Bmatrix} P_x \\ P_y \\ 0 \end{Bmatrix}.$$

These equations can be inverted by hand yielding:

$$\begin{Bmatrix} u_2 \\ v_2 \\ u_3 \end{Bmatrix} = \frac{1}{4.608 \times 10^6} \begin{bmatrix} 27216 & & \text{Sym.} \\ -12288 & 48384 & \\ 18432 & -24576 & 36864 \end{bmatrix} \begin{Bmatrix} P_x \\ P_y \\ 0 \end{Bmatrix}.$$

Substituting the given load values yields $[u_2 \;\; v_2 \;\; u_3] = [0.1124 \;\; -0.2367 \;\; 0.1467]$. The reader should verify that substituting these three displacements into the original equilibrium assembly yields reaction values of $R_1 = -10.00$, $R_2 = 6.25$ and $R_3 = 13.75$. Thus, the resultant values are equal and opposite to the applied loads as expected. Of course, they also satisfy moment equilibrium at all points in the plane. Note that if P_x had been zero, then $u_2 = u_3/2 = 0.0533$, and $v_2 = -0.21$. That is, the deformation of the structure would have been symmetric with respect to the center of the truss.

The concepts of symmetry and anti-symmetry are often useful in finite element analysis. It is common to find half, quarter, or one-eighth order symmetry conditions that can reduce the analysis cost to the square of the corresponding fractional part of a total analysis cost. Half symmetry was employed in the previous chapter in Sec. 6.4. For a truss we have no rotational degrees of freedom so we only have to consider the displacement components tangent or normal to the symmetry plane. Here we will apply symmetry to the above truss. First, we view the loads, members, and supports as viewed relative to a mirror placed at the symmetry section. The resulting partial model is shown in Fig. 7.4.2b. The applied loads and the stiffness of members lying in the symmetry plane are reduced by half. The nodes or member midpoints that lie in the symmetry plane are allowed to move only in that plane. Any supports that are not on the symmetry plane can be modified to support the structure in a consistent manner when viewed from the symmetry plane. This means that our simplified structure can be described as

Element	L_x	L_y	L	Topology		EA
1	4	3	5	1	2	1000
2	4	0	4	1	3	1000

The stiffness for the third element is no longer needed. The first member is unchanged. The length of the second member is cut in half so its stiffness doubles. The assembled elements give

$$\begin{bmatrix} (128+250) & 96 & -128 & -96 & -250 & 0 \\ & 72 & -96 & -72 & 0 & 0 \\ & & 128 & 96 & 0 & 0 \\ & & & 72 & 0 & 0 \\ & & & & 250 & 0 \\ \text{symmetric} & & & & & 0 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_1 \\ u_2 \\ v_2 \\ u_{3'} \\ v_{3'} \end{Bmatrix} = \begin{Bmatrix} 0 \\ R_1 \\ R_2 \\ P_y/2 \\ R_{3'} \\ 0 \end{Bmatrix}.$$

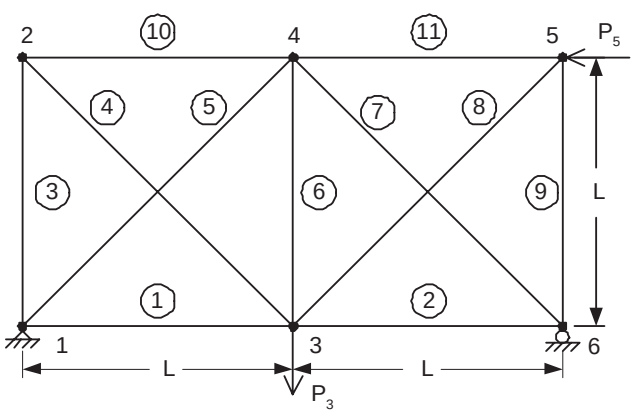
Points in the plane of symmetry must always move in that plane. Thus, $u_{3'} = 0$. Conversely, node 1 must be able to move normal to the plane of symmetry. Thus, $u_1 \neq 0$. Node 2 has an external load, P, applied tangent to the plane of symmetry. Thus, it must be allowed to move tangent to the plane ($v_2 \neq 0$) so that the force can do work on the structure. That is, in a given direction one can specify either the force or the displacement at a point, but not both. Clearly, node 1 has an unknown reaction that is parallel to the symmetry plane. Thus, it must be restrained in that direction, $v_1 = 0$. The restrained structural stiffness is

$$\begin{bmatrix} 378 & -96 & 0 \\ & 72 & 0 \\ \text{sym.} & & 0 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_2 \\ v_{3'} \end{Bmatrix} = \begin{Bmatrix} 0 \\ -20/2 \\ 0 \end{Bmatrix}.$$

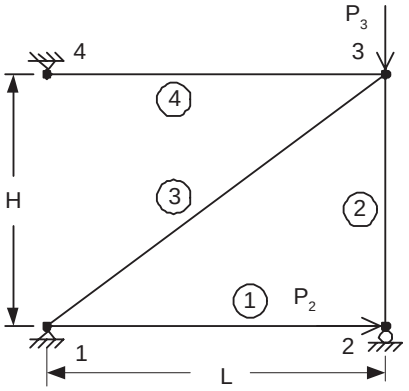
However, these equations are still singular after the application of the usually symmetric conditions. Note that the third row and column are zero. This means that there is no stiffness associated with the displacement $v_{3'}$. From the original structure in Fig. 7.3.2, we note that the center of member 2 must have a zero vertical deflection. Employing this additional physical insight, we can now also state that $v_{3'} = 0$. Therefore, for a symmetric structure with symmetric loads the equilibrium equations, relative to the plane of symmetry are

$$\begin{bmatrix} 378 & -96 \\ -96 & 72 \end{bmatrix} \begin{Bmatrix} u_1 \\ v_2 \end{Bmatrix} = \begin{Bmatrix} 0 \\ -10 \end{Bmatrix}, \qquad \begin{Bmatrix} u_1 \\ v_2 \end{Bmatrix} = \begin{Bmatrix} -0.05333 \\ -0.21 \end{Bmatrix}$$

as before, except for the sign change on u_1. This solution shows that for the above example there are only two degrees of freedom required when symmetry is available versus the three that were used before. For this simple example there was not much difference in the computational effort required in the symmetric and non-symmetric solutions. However, if there are hundreds or thousands of symmetric elements then the cost saving is very significant when a symmetric analysis can be utilized.



P 7.1



P 7.2

7.5 Exercises

1. Carry out an elastic bar patch test assuming $u(x) = a + b\,x$ and employing two co-axial truss elements over the domain $0 \le x \le L = 2$. Let $a = 5, b = -4$. Find the interior displacement, reactions $(40, -40)$, and element strains.

2. Employ two co-axial truss elements to model a bar that is fixed at both ends and undergoes a temperature increase. Assume numerical values of $E = 30,000\ ksi, A = 4\ in^2, L = 4\ ft = 48\ inches, \alpha = 7e-6\ 1/F$, and a temperature rise of $50\ F$. Obtain the displacements (null), reactions $42,000\ lb$, and stress.

3. The truss in Fig. P7.1 has two $L = 10\ inch$ bays with members made of steel, $E = 30,000\ ksi$, and constant cross-sections of $A = 1\ in^2$. The bottom center load is $P_3 = 10,000\ lb, P_5 = 0$, and the left and right corners are supported with a pin and roller, respectively. Compute the deflections (maximum vertical value is $-4.40e-3\ inches$) and the reactions ($5,000\ lb$ each).

4. Solve the previous problem for the loadcase of $P_3 = 0, P_5 = 10,000\ lb$.

5. The truss in Fig. P7.2 has a geometry of $L = 40, H = 30\ inch$, is made of steel with $E = 29.5e3\ ksi$, and all members have a constant cross-sectional area of $A = 1\ in^2$. If the joint loads are $P_2 = 20,000, P_3 = 25,000\ lb$ determine the deflections ($u_3 = 5.65e-3\ in.$), the structure reactions ($R_{1_x} = -15,833\ lb$), and the member stresses ($\sigma_4 = 4.167\ ksi$).

6. Solve the previous problem if the horizontal roller support at node 2 undergoes a vertical settlement of $v_2 = -0.12\ inch$, and the joint loads are still $P_2 = 20,000, P_3 = 25,000\ lb$ ($u_3 = 3.951e-3\ in., \sigma_4 = 23.833\ ksi$).

7. Solve for the deflections, reactions, and stresses for the truss in problem 5 assuming no joint loads, but having elements 2 and 3 subjected to a temperature increase of $\delta T = +50\ F$. Assume a coefficient of thermal expansion of $\alpha = 6.667e-6\ 1/F$

($u_3 = 1.222e-2\ in.$, $\sigma_4 = 2.914\ ksi$).

8. Implement Eqs. 7.4.10 and 11 as an explicit formulation of the stiffness and loads on a two-dimensional truss.

7.6 References

[1] Bathe, K.J., *Finite Element Procedures*, Englewood Cliffs: Prentice-Hall (1996).
[2] Chandrupatla, T.R. and Belegundu, A.D., *Introduction to Finite Elements in Engineering*, Englewood Cliffs: Prentice-Hall (1991).
[3] Cook, R.D., *Concepts and Applications of Finite Element Analysis*, New York: John Wiley (1974).
[4] Cook, R.D., Malkus, D.S., Plesha, N.E., and Witt, R.J., *Concepts and Applications of Finite Element Analysis*, New York: John Wiley (2002).
[5] Fenner, R.T., *Finite Element Methods for Engineers*, London: Macmillan (1975).
[6] Logan, D.L., *A First Course in the Finite Element Method*, 3nd Edition, Pacific Grove: Brooks Cole (2002).
[7] Martin, H.C. and Carey, G.F., *Introduction to Finite Element Analysis*, New York: McGraw-Hill (1974).
[8] Smith, I.M. and Griffiths, D.V., *Programming the Finite Element Method*, 3nd Edition, Chichester: John Wiley (1998).
[9] Ural, O., *Matrix Operations and Use of Computers in Structural Engineering*, International Textbook (1971).
[10] Weaver, W.F., Jr. and Johnston, P.R., *Finite Elements for Structural Analysis*, Englewood Cliffs: Prentice-Hall (1984).

