UNIVERSITY of SAN CARLOS

SCIENTIA • VIRTUS • DEVOTIO

TECHNOLOGICAL CENTER

SCHOOL OF ENGINEERING

DEPARTMENT OF CIVIL ENGINEERING

NASIPIT, TALAMBAN, CEBU CITY

CE 523A

PROJECT NO. 1

ANALYSIS OF TRUSSES

GROUP 1

Irinco, Jasmin Evans

Tabada, Earljohn Gabrielle

Tiu, Meryll Shaine

Academic integrity pledge: *"I swear on my honor, I have not given nor received any inappropriate aid."*

TABLE OF CONTENTS

ABSTRACT

Two methods are commonly used for structural analysis: the flexibility or force method, which is concerned with static indeterminacy, and the stiffness method, which is related to kinematic indeterminacy. The stiffness method is the preferred approach to computer-based solving. A size optimization study used a heuristic particle swarm optimizer (HPSO) on ten and fifteen-bar benchmark truss problems. The resulting optimal structure and applied loads and constraints of the study were used as the basis for solving for the forces, stresses, and nodal displacements of the benchmark truss problems using MATLAB software. The data shows that all the constraints were met by the optimal structures, except for the second load case of the ten-bar truss which violated both stress and displacement constraints.

INTRODUCTION

There are two main approaches to solving structural statically indeterminate systems: the flexibility method and the stiffness method. The flexibility method is concerned with static indeterminacy, which is defined as the number of actions or forces that need to be released in order to transform the structural system into a statically determinate system. These forces, such as shear force, axial force, and bending moment, are treated as the unknowns. The stiffness method, on the other hand, is related to kinematic determinacy, or the number of unrestrained components of translational and rotational nodal displacements required to satisfy equilibrium requirements prior to other responses. The displacements are the unknowns in this method. The flexibility method converts the structural system into a statically determinate structure, while the stiffness method converts the system into a kinematically determinate structure (Chandrasekaran, 2019).

The stiffness method is sometimes preferred over the flexibility method because unlike the latter, which has many different choices for the redundant that lead to different processes, the stiffness approach only has one set of standard procedures for a single possible restrained structure, making it more appropriate for digital methods. In this paper, matrices are utilized to solve truss problems using the stiffness method (Chandrasekaran, 2019).

Stiffness Matrix

Truss structures are assumed to have pinned connections, so there is axial force but no moment transfer between members. The member force *q* is directly proportional to the displacement *d* along its longitudinal axis through the spring constant, the product of the member cross-sectional area A and its Young's modulus E, divided by its length *L*. There are two possible independent displacements. When a positive displacement is imposed on the near end with the far end pinned, the forces developed are (Hibbeler, 2012):

$$q_N^{'} = \frac{AE}{L} d_N$$

$$q_F^{'} = -\frac{AE}{L} d_F$$

Where subscripts N and F refer to the near and far ends, respectively. The negative sign indicates that the force is in the negative x' direction. Similarly, when the near end is fixed and there is a positive displacement on the far end, the forces developed are (Hibbeler, 2012):

$$q_N^{''} = -\frac{AE}{L} d_F$$
$$q_F^{''} = \frac{AE}{L} d_N$$

By superposition, the resultant forces caused by both displacements are:

$$q_N = \frac{AE}{L} d_N - \frac{AE}{L} d_F$$
$$q_F = \frac{AE}{L} d_F - \frac{AE}{L} d_N$$

Where essentially, *q=k'd*

These equations can be written in matrix form as:

$$\begin{bmatrix} q_N \\ q_F \end{bmatrix} = \frac{AE}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \begin{bmatrix} d_N \\ d_F \end{bmatrix}$$

The stiffness matrix *k'* can then be extracted to be equal to:

$$k' = \frac{AE}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

A truss member must be translated from its arbitrary (local) axis to its reference (global) axis. This is done using the transformation equation:

$$\hat{V}_x = v_x \cos\theta + v_y \sin\theta$$
$$\hat{V}_y = -v_x \sin\theta + v_y \cos\theta$$

where $\hat{V}_x$ and $\hat{V}_y$ represent the x and y components, respectively, of vector V in the global axis while v_x and v_y symbolize those of the local coordinate system. These equations can be expressed in matrix form:

$$\begin{Bmatrix} \hat{V}_x \\ \hat{V}_y \end{Bmatrix} = \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{Bmatrix} v_x \\ v_y \end{Bmatrix}$$

When $cos\theta$ is denoted as *l* and $sin\theta$ as *m*, the matrix equation can be rewritten as:

$$\begin{Bmatrix} \hat{V}_x \\ \hat{V}_y \end{Bmatrix} = \begin{bmatrix} l & m \\ -m & l \end{bmatrix} \begin{Bmatrix} v_x \\ v_y \end{Bmatrix}$$

where *l* and *m* are calculated as:

$$l = \cos\theta = \frac{x_F - x_N}{L}$$

$$m = \sin\theta = \frac{y_F - y_N}{L}$$

Establishing matrix T*,

$$\mathrm{T}^* = \begin{bmatrix} l & m \\ -m & l \end{bmatrix},$$

The transformation matrix T is as follows:

$$T = \begin{bmatrix} \mathrm{T}^* & 0 \\ 0 & \mathrm{T}^* \end{bmatrix}$$

This can be rewritten as

$$T = \begin{bmatrix} l & m & 0 & 0 \\ -m & l & 0 & 0 \\ 0 & 0 & l & m \\ 0 & 0 & -m & l \end{bmatrix}$$

The global stiffness matrix k is the product of matrices T^T, $\hat{k}$, and T. The transformation matrix T is orthogonal; therefore, its transpose is equal to its inverse. This can be expressed in mathematical form as $\mathrm{T}^{\mathrm{T}} = \mathrm{T}^{-1}$.

$$\hat{k} = \begin{bmatrix} k & 0 & -k & 0 \\ 0 & 0 & 0 & 0 \\ -k & 0 & k & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

The global stiffness matrix k can then be evaluated as:

$$k = T^T \hat{k} T = \begin{bmatrix} l^2 & lm & -l^2 & -lm \\ lm & m^2 & -lm & -m^2 \\ -l^2 & -lm & l^2 & lm \\ -lm & -m^2 & lm & m^2 \end{bmatrix}$$

To solve a truss problem, a global stiffness matrix is created for each member. Since truss joints are assumed to be hinged, there is no rotation due to moment at the nodes. Therefore, for each of the members' two nodes, there are two degrees of freedom: possible displacements in the x and y directions. This totals to four degrees of freedom for each member, which translates into a 4x4 matrix.

Once the global stiffness matrix for each member has been created, all of their contents are added to their counterparts for the same degree of freedom, such that all the matrices are combined into a single matrix that accounts for the entire structure.

Next, boundary conditions are established, wherein the support nodes are assumed or specified to have zero displacements in the corresponding degree of freedom that it has been restrained. The degrees of freedom, then, can be classified into two categories: free or unknown, and specified or restrained. The stiffness matrix is also rearranged to group the free and restrained degrees of freedom together.

Having determined the required matrices, the member forces and stresses, and nodal displacements can then be calculated.

Truss Problems

Common benchmark problems used in truss optimization studies are the 10-bar, 6-joint truss, and the 15-bar, 8-joint truss. Li, Huang, and Liu (2009) performed size optimization on the trusses using a heuristic particle swarm optimizer (HSPO), based on the original particle swarm optimizer and harmony search scheme. The algorithm achieved better results, or a lighter structure, compared to results from other modified or hybrid particle swarm optimization studies.

Ten-bar truss

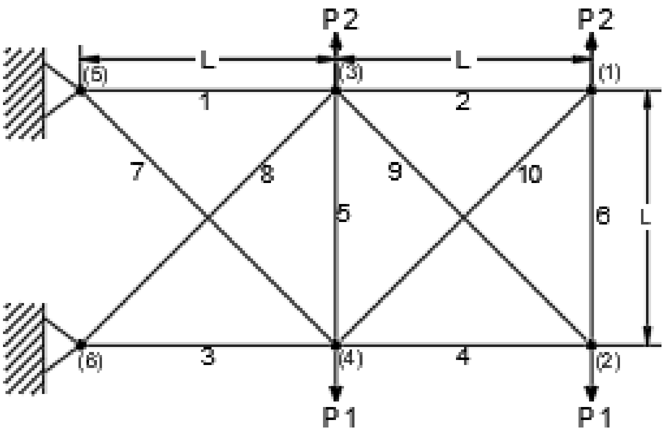

Figure 1. Standard 10-bar truss

The ten-bar benchmark truss design problem as illustrated in Figure 1 is classified into two cases that have different load forces. Case 1 consists of two load joints while Case 2 has four load joints. Table 1 shows the properties and loadings for both cases of the ten-bar truss.

Table 1. Data for Ground Structure of the Ten-Bar Truss

	Case 1	**Case 2**
Length (in)	360	360
Loading (kips)	$P_1 = 100$	$P_1 = 150$
	$P_2 = 0$	$P_2 = 150$
Allowable Tensile Stress (ksi)	25	25
Allowable Compressive Stress (ksi)	25	25
Allowable displacement (in)	2.0	2.0
Material Density (lb/in^3)	0.1	0.1
Young's Modulus (ksi)	10,000	10,000

Each load case also takes discrete variables, or choices of member cross-sectional areas, from different ranges. Case 1 is from the set D = {1.62, 1.80, 1.99, 2.13, 2.38, 2.62, 2.63, 2.88, 2.93, 3.09, 3.13, 3.38, 3.47, 3.55, 3.63, 3.84, 3.87, 3.88, 4.18, 4.22, 4.49, 4.59, 4.80, 4.97, 5.12, 5.74, 7.22, 7.97, 11.50, 13.50, 13.90, 14.20, 15.50, 16.00, 16.90, 18.80, 19.90, 22.00, 22.90, 26.50, 30.00, 33.50} (in.2); while case 2 discrete variables are selected from the set D = {0.1, 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5, 5.0, 5.5, 6.0, 6.5, 7.0, 7.5, 8.0, 8.5, 9.0, 9.5, 10.0, 10.5, 11.0, 11.5, 12.0, 12.5, 13.0, 13.5, 14.0, 14.5, 15.0, 15.5, 16.0, 16.5, 17.0, 17.5, 18.0, 18.5, 19.0, 19.5, 20.0, 20.5, 21.0, 21.5, 22.0, 22.5, 23.0, 23.5, 24.0, 24.5, 25.0, 25.5, 26.0, 26.5, 27.0, 27.5, 28.0, 28.5, 29.0, 29.5, 30.0, 30.5, 31.0, 31.5} (in^2).

The resulting optimal cross-sectional areas for each member are as shown in Table 2.

Table 2: Results for Ten-Bar Truss

Variables (in^2)	Case 1	Case 2
A_1	30.00	31.50
A_2	1.62	0.10
A_3	22.90	24.50
A_4	13.50	15.50
A_5	1.62	0.10
A_6	1.62	0.50
A_7	7.97	7.50
A_8	26.50	20.50
A_9	22.00	20.50
A_{10}	1.80	0.10
Weight (lb)	5593.44	5073.51

Fifteen-bar truss

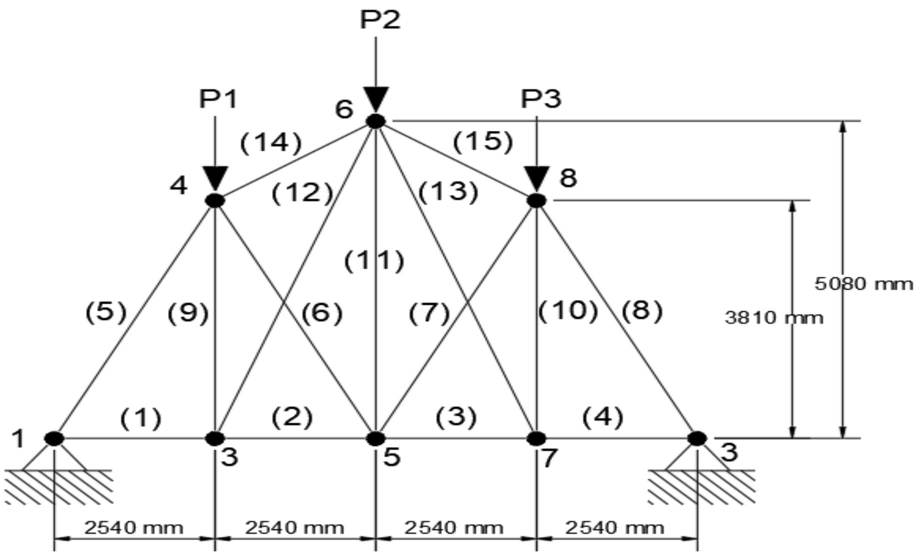

Figure 2. 15-bar truss

The fifteen-bar truss as illustrated in Figure 2 considers three cases of different load forces. Table 3 shows the data for three loading cases of the 15-bar truss.

Table 3. Data for Ground Structure of the 15-Bar Truss

	Case 1	Case 2	Case 3
	$P_1 = 35$	$P_1 = 35$	$P_1 = 35$
Loading (kN)	$P_2 = 35$	$P_2 = 0$	$P_2 = 35$
	$P_3 = 35$	$P_3 = 35$	$P_3 = 0$
Allowable Tensile Stress (MPa)	120	120	120
Allowable Compressive Stress (MPa)	120	120	120
Allowable displacement (mm)	10	10	10
Material Density (kg/m^3)	7800	7800	7800
Young's Modulus (GPa)	200	200	200

All three load cases obtain discrete variables from the set D = {113.2, 143.2, 145.9, 174.9, 185.9, 235.9, 265.9, 297.1, 308.6, 334.3, 338.2, 497.8, 507.6, 736.7, 791.2, 1063.7} (mm^2). The results obtained by the study are indicated in Table 4.

Table 4: Results for Fifteen-Bar Truss

Variables (in^2)	Area
A_1	113.2
A_2	113.2
A_3	113.2
A_4	113.2
A_5	736.7
A_6	113.2
A_7	113.2
A_8	736.7
A_9	113.2
A_{10}	113.2
A_{11}	113.2
A_{12}	113.2
A_{13}	113.2
A_{14}	334.3
A_{15}	334.3
Weight (lb)	105.735

The MATLAB codes for finding the displacement, forces, and stresses of the ten-bar truss cases 1 and 2 and the fifteen-bar truss cases 1, 2, and 3 are found in Appendices A to E, respectively.

RESULTS AND DISCUSSION

1. 10-bar truss

a. Case 1

ELEMENT	AREA (in^2)	MEMBER FORCES (k)	STRESSES (ksi)
1	30	222.30	7.41
2	1.62	1.79	1.11
3	22.9	-177.70	-7.76
4	13.5	-98.21	-7.27
5	1.62	24.09	14.87
6	1.62	1.79	1.11
7	7.97	109.89	13.79
8	26.5	-172.96	-6.53
9	22	138.89	6.31
10	1.8	-2.53	-1.41

NODES	NODAL DISPLACEMENT	
	X (in)	Y (in)
1	0.31	-1.96
2	-0.54	-2.00
3	0.27	-0.74
4	-0.28	-1.27
5	0.00	0.00
6	0.00	0.00

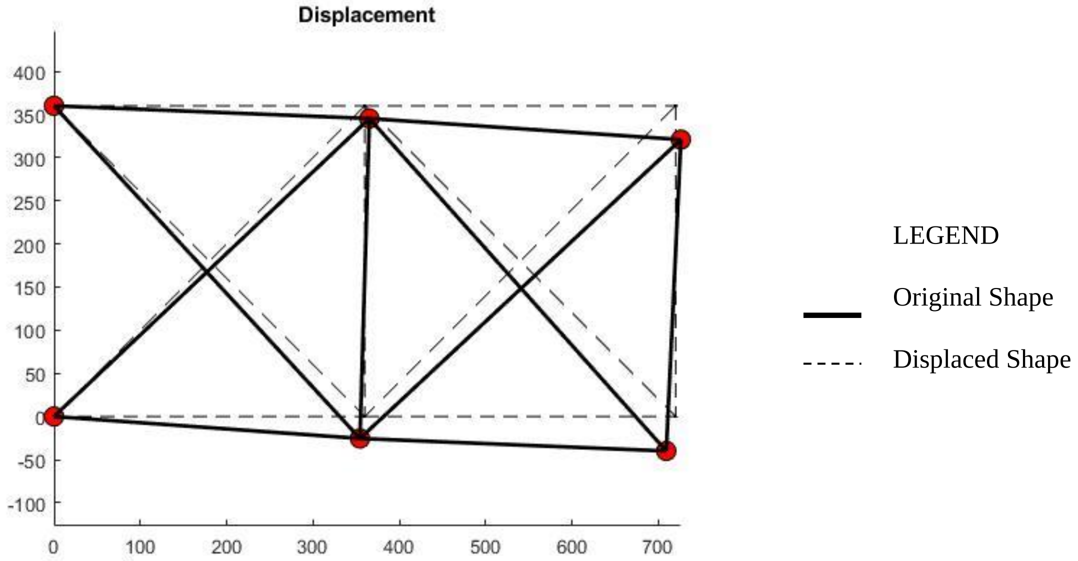

Figure 3. 10-Bar Truss Displacement: Case 1

b. Case 2

ELEMENT	AREA (in^2)	MEMBER FORCES (k)	STRESSES (ksi)
1	31.50	-132.85	-4.22
2	0.10	-9.34	-93.38
3	24.50	-132.85	-5.42
4	15.50	-9.34	-0.60
5	0.10	7.81	78.07
6	0.50	140.66	281.32
7	7.50	187.88	25.05
8	20.50	187.88	9.17
9	20.50	13.21	0.64
10	0.10	13.21	132.06

NODES	NODAL DISPLACEMENT	
	X (in)	Y (in)
1	-3.51	10.83
2	-0.22	0.70
3	-0.15	0.81
4	-0.20	-2.00
5	0.00	0.00
6	0.00	0.00

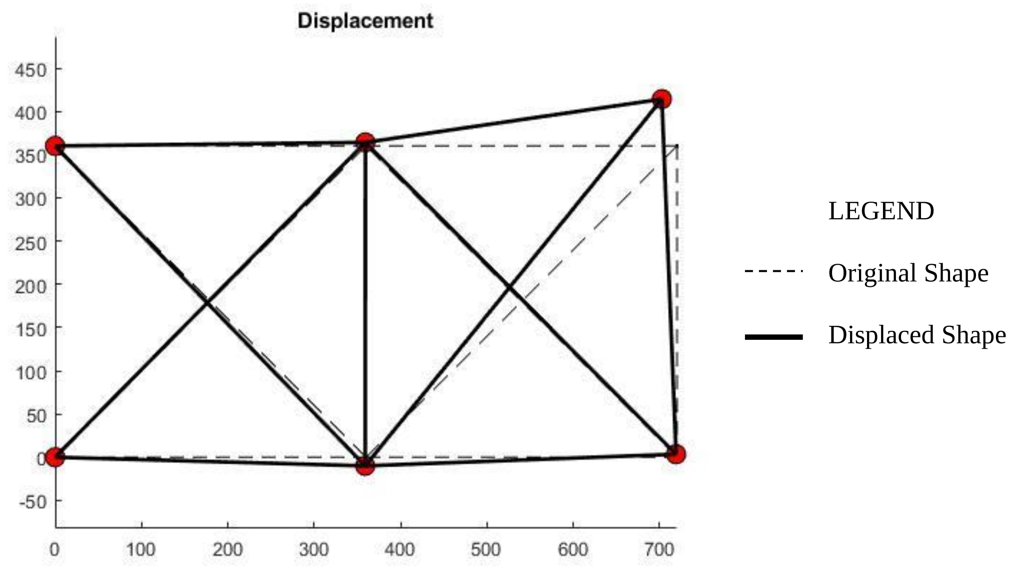

Figure 4. 10-Bar Truss Displacement: Case 2

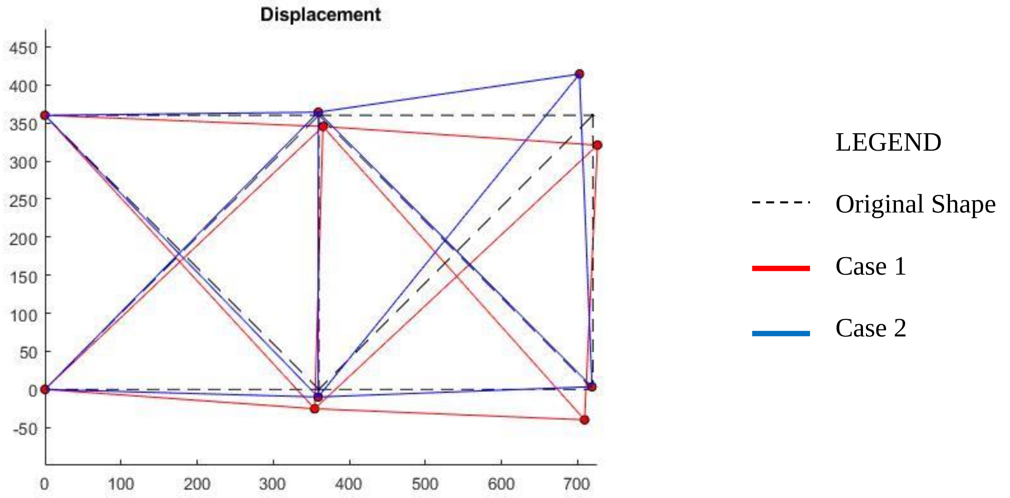

Figure 5. 10-Bar Truss Displacement: Case 1 and 2

2. 15-bar truss

ELEMENT	AREA (mm^2)	MEMBER FORCES (kN)			STRESSES (MPa)		
		Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
1	113.2	-250.86	1187.37	4988.79	-2.22	10.49	44.07
2	113.2	250.86	-1187.37	5045.62	2.22	-10.49	44.57
3	113.2	250.86	-1187.37	-3356.53	2.22	-10.49	-29.65
4	113.2	-250.86	1187.37	-6677.88	-2.22	10.49	-58.99
5	736.7	-63097.15	-42064.76	-52580.96	-85.65	-57.10	-71.37
6	113.2	-904.49	-6235.08	-5360.54	-7.99	-55.08	-47.35
7	113.2	-904.49	-6235.08	9786.64	-7.99	-55.08	86.45
8	736.7	-63097.15	-42064.76	-31548.57	-85.65	-57.10	-42.82
9	113.2	1003.44	-4749.46	113.65	8.86	-41.96	1.00
10	113.2	1003.44	-4749.46	6642.70	8.86	-41.96	58.68
11	113.2	1505.17	10375.81	-3682.74	13.30	91.66	-32.53
12	113.2	-1121.89	5310.06	-127.07	-9.91	46.91	-1.12
13	113.2	-1121.89	5310.06	-7426.76	-9.91	46.91	-65.61
14	334.3	-38570.25	-22220.63	-29284.86	-115.38	-66.47	-87.60
15	334.3	-38570.25	-22220.63	-25635.01	-115.38	-66.47	-76.68

NODES	NODAL DISPLACEMENT					
	x (mm)			y (mm)		
	Case 1	Case 2	Case 3	Case 1	Case 2	Case 3
1	0.00	0.00	0.00	0.00	0.00	0.00
2	0.00	0.00	0.00	0.00	0.00	0.00
3	-0.03	0.13	0.56	-3.58	-2.26	-3.39
4	1.58	2.23	2.12	-3.41	-3.06	-3.37
5	0.00	0.00	1.13	-4.25	-3.03	-2.73
6	0.00	0.00	0.82	-3.91	-0.70	-3.56
7	0.03	-0.13	0.75	-3.58	-2.26	-1.51
8	-1.58	-2.23	1.18	-3.41	-3.06	-0.39

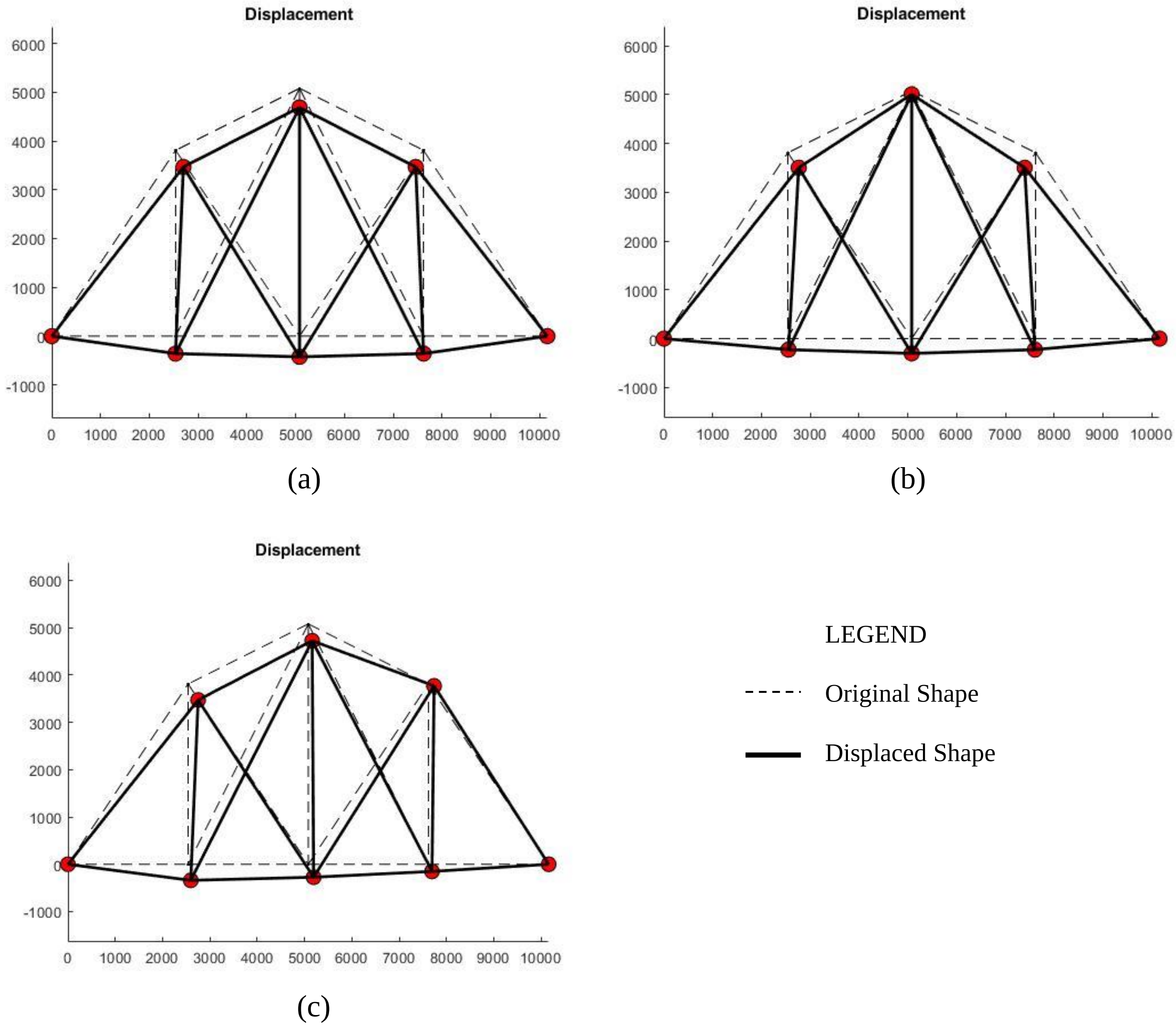

Figure 6. 15-Bar Truss Displacement: (a) Case 1; (b) Case 2; (c) Case 3

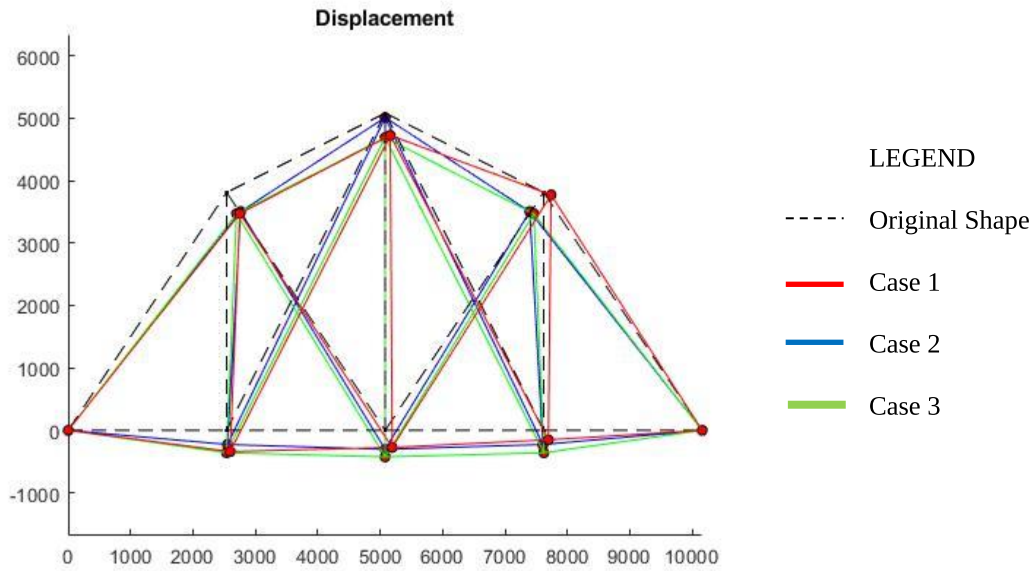

Figure 7. 15-Bar Truss Displacement: All Cases

The optimal structures illustrated are the results of a size optimization study by Li, Huang, and Liu (2009) using a heuristic particle swarm optimizer (HPSO). Given the optimal cross-sectional areas and properties, loadings, and constraints, the stiffness method was used to determine each member's force and stress, as well as all nodal displacements. Figures 3, 4, and 5 illustrate the original and displaced shapes for cases 1 and 2 of the ten-bar truss, while Figures 6a, 6b, 6c, and 7 show those of the three cases of the fifteen-bar truss.

Both load cases of the ten-bar benchmark truss problem have three constraints: allowable tensile and compressive stresses of 25 ksi each, and an allowable displacement of 2 inches. The optimal structure for the first load case satisfied all of these constraints; all the members did not exceed 25 ksi of tensile or compressive stresses, while none of the nodes were displaced more than 2 inches.

The second load case, however, violated the constraints. Element 2 has a compressive stress of 93.38 ksi, while elements 5, 6, 7, and 10 are all tensile members that exceeded 25 ksi. Node 1 also has a displacement of 3.51 inches in the *x* direction and 10.83 inches in the *y* direction, violating the displacement constraint in both axes.

All the cases of the fifteen-bar truss also have three constraints: allowable tensile and compressive stresses of 120 MPa and an allowable displacement of 10 mm. The results show that all of the elements in all three cases did not violate the stress constraints, and none of the nodes violated the displacement constraint.

CONCLUSIONS AND RECOMMENDATIONS

The results show that matrix analysis of the stiffness method using MATLAB is an effective way of determining a truss structure's member forces and stresses, and nodal displacements. Therefore, it is desirable as a quick and flexible approach to solving structurally indeterminate problems, and has advantages over manual calculation using the flexibility method.

Regarding the global stiffness matrix for the entire planar structure, the matrix size is equal to the total number of degrees of freedom. Since there are two degrees of freedom for each node or joint, the number of rows and columns for the matrix is two times the number of nodes in the structure.

The weight-optimal structures were obtained by Li et al. (2009) using a heuristic particle swarm optimizer. Overall, the algorithm was able to optimize the benchmark trusses within the set constraints, with the exception of the second load case of the ten-bar benchmark truss which had violations in both stress and displacement constraints. In order to satisfy these constraints, the member sizes should be increased. However, this would likely also increase the structure's weight to more than that of the other comparative studies. It may be recommended to the researchers to employ a better-fitting constraint-handling technique to find the optimal structure without violating any constraints, or at least minimizing the violations.

REFERENCES

Chandrasekaran, S. (2019). *Advanced Structural Analysis with MATLAB*. Boca Raton, FL: CRC Press.

Hibbeler, R.C. (2012). *Structural Analysis* (8^{th} ed.). New Jersey: Pearson Prentice Hall.

Khennane, A. (2013). *Introduction to Finite Element Analysis using MATLAB and Abaqus.* Boca Raton, FL: CRC Press.

Li, L.J., et al. (2009). A heuristic particle swarm optimization method for truss structures with discrete variables. *Computers and Structures 87:* 435-443**.**

APPENDIX A – 10-Bar Truss Case 1

```
clc,clear all;

% NODAL AND MEMBER DEFINITION

    nodes = [ 720 360 ; 720 0 ; 360 360 ; 360 0 ; 0 360 ; 0 0 ];
%coordinate
    elems = [ 5 3 ; 3 1 ; 6 4 ; 4 2 ; 3 4 ; 1 2 ; 5 4 ; 3 6 ; 3 2 ; 1 4 ];
%elements nodal connectivity
    E = 10e3+ zeros(1,size(elems,1)); %Youngs Modulus
    A = [ 30 1.62 22.90 13.50 1.62 1.62 7.97 26.50 22 1.8 ]; %Optimal Area
    bc = [ 5 1 0 ; 5 2 0 ; 6 1 0 ; 6 2 0 ]; %boundary conditions
    loads = [ 1 2 0 ; 2 2 -100 ; 3 2 0 ; 4 2 -100 ]; %applied loads

    Nnodes = size(nodes,1); %number of nodes
    Nel = size(elems,1); %number of elements

% DEFINE K , u , and f

    alldofs = 1:2*Nnodes;

    K = zeros(2*Nnodes,2*Nnodes);
    u = zeros(2*Nnodes,1);
    f = zeros(2*Nnodes,1);

% SPECIFIED NODAL DISPLACEMENTS

    dofspec = [] ;

    for i=1:size(bc,1);

        thisdof = 2*(bc(i,1)-1) + bc(i,2);
        dofspec = [dofspec thisdof];

    end

        doffree = alldofs ;
        doffree(dofspec) = [] ;

% SPECIFIED NODAL FORCES

    for i=1:size(loads,1);

        a = loads(i,1);
        b = loads(i,2);
        f( 2*(a-1) + b) = loads(i,3);

    end
```

```
% DEFINE GLOBAL STIFFNESS

    for i=1:Nel;

        elnodes = elems(i,:);
        % Transformation Matrix

        nodesxy = nodes(elnodes,:);

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        T= [c s 0 0 ; -s c 0 0 ; 0 0 c s ; 0 0 -s c] ;



        kk = (A(i)*E(i))/L ; % AE/L Per member
        k_loc=[kk 0 -kk 0; 0 0 0 0; -kk 0 kk 0; 0 0 0 0] ; %local stiffness
        Kel= T'*k_loc*T ;

        %Determine Corresponding Global DOF

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)];

        %Assemble Element to Global Stiffness matrix

        K(eldofs,eldofs) = K(eldofs,eldofs) + Kel;

    end

% Displacements and Member Forces

    u(doffree) = K(doffree,doffree)^-1 * (f(doffree) -
K(doffree,dofspec)*u(dofspec)) %displacements

    f(dofspec) = K(dofspec,doffree)*u(doffree) +
K(dofspec,dofspec)*u(dofspec); %nodal forces / reactions

% Member Forces

    for i=1:Nel

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);

        % Transformation Matrix
```

```
        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        kk = (A(i)*E(i))/L;
        la =[ c s -c -s ];

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)]; %currentdof

        Force(i,1)= la*u(eldofs)*kk

    end

% Stresses
    for i=1:Nel

       stress(i,1)= Force(i)/A(i)

    end

%plot old shape
figure (1); hold on;
plot(nodes(:,1), nodes(:,2), 'k.')
hold on; axis equal;
for iel=1:Nel
    elnodes=elems(iel, 1:2);
    nodesxy = nodes(elnodes, :);
    plot(nodesxy(:,1),nodesxy(:,2), 'k--')
end

%plot new shape
Magnification = 20;
nodesnew = nodes + Magnification*reshape(u,2,Nnodes)';
plot(nodesnew(:,1),nodesnew(:,2),'o', ...
    'MarkerEdgeColor' , 'k', 'MarkerFaceColor' , 'r', 'MarkerSize', 5)
hold on; axis equal;
for iel = 1:Nel
   elnodes =elems(iel, 1:2);
   nodesxy = nodesnew(elnodes, :);
   plot(nodesxy(:,1),nodesxy(:,2), 'r-', 'LineWidth', 0.5)
end

title('Displacement: 10-Bar Truss Case 1');
xlabel('(in)')
ylabel('(in)')
```

APPENDIX B – 10-Bar Truss Case 2

```
clc,clear all;

% NODAL AND MEMBER DEFINITION

    nodes = [ 720 360 ; 720 0 ; 360 360 ; 360 0 ; 0 360 ; 0 0 ];
%coordinate
    elems = [ 5 3 ; 3 1 ; 6 4 ; 4 2 ; 3 4 ; 1 2 ; 5 4 ; 3 6 ; 3 2 ; 1 4 ];
%elements nodal connectivity
    E = 10e3+ zeros(1,size(elems,1)); %Youngs Modulus
    A = [ 31.5 0.1 24.5 15.50 0.1 0.5 7.5 20.5 20.5 0.1 ]; %Optimal Area
    bc = [ 5 1 0 ; 5 2 0 ; 6 1 0 ; 6 2 0 ]; %boundary conditions
    loads = [ 1 2 150 ; 2 2 -150 ; 3 2 150 ; 4 2 -150 ]; %applied loads

    Nnodes = size(nodes,1); %number of nodes
    Nel = size(elems,1); %number of elements

% DEFINE K , u , and f

    alldofs = 1:2*Nnodes;

    K = zeros(2*Nnodes,2*Nnodes);
    u = zeros(2*Nnodes,1);
    f = zeros(2*Nnodes,1);

% SPECIFIED NODAL DISPLACEMENTS

    dofspec = [] ;

    for i=1:size(bc,1);

        thisdof = 2*(bc(i,1)-1) + bc(i,2);
        dofspec = [dofspec thisdof];

    end

        doffree = alldofs ;
        doffree(dofspec) = [] ;

% SPECIFIED NODAL FORCES

    for i=1:size(loads,1);

        a = loads(i,1);
        b = loads(i,2);
        f( 2*(a-1) + b) = loads(i,3);

    end

% DEFINE GLOBAL STIFFNESS

    for i=1:Nel;

        elnodes = elems(i,:);
        % Transformation Matrix
```

```
        nodesxy = nodes(elnodes,:);

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        T= [c s 0 0 ; -s c 0 0 ; 0 0 c s ; 0 0 -s c] ;


        kk = (A(i)*E(i))/L ; % AE/L Per member
        k_loc=[kk 0 -kk 0; 0 0 0 0; -kk 0 kk 0; 0 0 0 0] ; %local stiffness
        Kel= T'*k_loc*T ;

        %Determine Corresponding Global DOF

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)];

        %Assemble Element to Global Stiffness matrix

        K(eldofs,eldofs) = K(eldofs,eldofs) + Kel;

    end

% Displacements and Member Forces

    u(doffree) = K(doffree,doffree)^-1 * (f(doffree) -
K(doffree,dofspec)*u(dofspec)) %displacements

    f(dofspec) = K(dofspec,doffree)*u(doffree) +
K(dofspec,dofspec)*u(dofspec); %nodal forces / reactions

% Member Forces

    for i=1:Nel

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);

        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        kk = (A(i)*E(i))/L;
        la =[ c s -c -s ];

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)]; %currentdof

        Force(i,1)= la*u(eldofs)*kk

    end

% Stresses
    for i=1:Nel
```

```
        stress(i,1)= Force(i)/A(i)

    end

%plot old shape
figure (1); hold on;
plot(nodes(:,1), nodes(:,2), 'k.')
hold on; axis equal;
for iel=1:Nel
    elnodes=elems(iel, 1:2);
    nodesxy = nodes(elnodes, :);
    plot(nodesxy(:,1),nodesxy(:,2), 'k--')
end

%plot new shape
Magnification = 5;
nodesnew = nodes + Magnification*reshape(u,2,Nnodes)';
plot(nodesnew(:,1),nodesnew(:,2),'o', ...
    'MarkerEdgeColor' , 'k', 'MarkerFaceColor' , 'r', 'MarkerSize', 5)
hold on; axis equal;
for iel = 1:Nel
   elnodes =elems(iel, 1:2);
   nodesxy = nodesnew(elnodes, :);
   plot(nodesxy(:,1),nodesxy(:,2), 'b-', 'LineWidth', 0.5)
end
title('Displacement');
```

APPENDIX C – 15-Bar Truss Case 1

```
clc,clear all;

% NODAL AND MEMBER DEFINITION

    nodes = [ 0 0 ; 10160 0 ; 2540 0 ; 2540 3810 ; 5080 0 ; 5080 5080; 7620
0 ; 7620 3810 ]; %coordinate
    elems = [ 1 3 ; 3 5 ; 5 7 ; 7 2 ; 1 4; 4 5 ; 5 8 ; 8 2 ; 4 3 ; 8 7 ; 6
5 ; 6 3 ; 6 7 ; 6 4 ; 6 8 ]; %elements nodal connectivity
    E = 200e3 + zeros(1,size(elems,1)); %Youngs Modulus
    A = [ 113.2 113.2 113.2 113.2 736.7 113.2 113.2 736.7 113.2 113.2 113.2
113.2 113.2 334.3 334.3 ] ;
    bc = [ 1 1 0 ; 1 2 0 ; 2 1 0 ; 2 2 0 ]; %boundary conditions
    loads = [ 4 2 -35000 ; 6 2 -35000 ; 8 2 -35000 ]; %applied loads

    Nnodes = size(nodes,1); %number of nodes
    Nel = size(elems,1); %number of elements

% DEFINE K , u , and f

    alldofs = 1:2*Nnodes;

    K = zeros(2*Nnodes,2*Nnodes);
    u = zeros(2*Nnodes,1);
    f = zeros(2*Nnodes,1);

% SPECIFIED NODAL DISPLACEMENTS

    dofspec = [] ;

    for i=1:size(bc,1);

        thisdof = 2*(bc(i,1)-1) + bc(i,2);
        dofspec = [dofspec thisdof];

    end

        doffree = alldofs ;
        doffree(dofspec) = [] ;

% SPECIFIED NODAL FORCES

    for i=1:size(loads,1);

        a = loads(i,1);
        b = loads(i,2);
        f( 2*(a-1) + b) = loads(i,3);

    end

% DEFINE GLOBAL STIFFNESS

    for i=1:Nel;

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);
```

```
        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        T= [c s 0 0 ; -s c 0 0 ; 0 0 c s ; 0 0 -s c] ;


        kk = (A(i)*E(i))/L ; % AE/L Per member
        k_loc=[kk 0 -kk 0; 0 0 0 0; -kk 0 kk 0; 0 0 0 0] ; %local stiffness
        Kel= T'*k_loc*T ;

        %Determine Corresponding Global DOF

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)];

        %Assemble Element to Global Stiffness matrix

        K(eldofs,eldofs) = K(eldofs,eldofs) + Kel;

    end

% Displacements and Member Forces

    u(doffree) = K(doffree,doffree)^-1 * (f(doffree) -
K(doffree,dofspec)*u(dofspec)); %displacements

    f(dofspec) = K(dofspec,doffree)*u(doffree) +
K(dofspec,dofspec)*u(dofspec); %nodal forces / reactions

% Member Forces

    for i=1:Nel

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);

        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        kk = (A(i)*E(i))/L;
        la =[ c s -c -s ];

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)]; %currentdof

        Force(i,1)= la*u(eldofs)*kk ;

    end

% Stresses
```

```
for i=1:Nel
   stress(i,1) = Force(i)/A(i)
end

format long
disp(['Displacement vector:']); u
disp(['Reactions:']); f
disp(['Member Forces:']); Force

%plot old shape
figure (1); hold on;
plot(nodes(:,1), nodes(:,2), 'k.')
hold on; axis equal;
for iel=1:Nel
    elnodes=elems(iel, 1:2);
    nodesxy = nodes(elnodes, :);
    plot(nodesxy(:,1),nodesxy(:,2), 'k--')
end

%plot new shape
Magnification = 100;
nodesnew = nodes + Magnification*reshape(u,2,Nnodes)';
plot(nodesnew(:,1),nodesnew(:,2),'o', ...
    'MarkerEdgeColor' , 'k', 'MarkerFaceColor' , 'r', 'MarkerSize', 5)
hold on; axis equal;
for iel = 1:Nel
   elnodes =elems(iel, 1:2);
   nodesxy = nodesnew(elnodes, :);
   plot(nodesxy(:,1),nodesxy(:,2), 'g-', 'LineWidth', 0.5)
end
title('Displacement');
```

APPENDIX D – 15-Bar Truss Case 2

```
clc,clear all;

% NODAL AND MEMBER DEFINITION

    nodes = [ 0 0 ; 10160 0 ; 2540 0 ; 2540 3810 ; 5080 0 ; 5080 5080; 7620 
0 ; 7620 3810 ]; %coordinate
    elems = [ 1 3 ; 3 5 ; 5 7 ; 7 2 ; 1 4; 4 5 ; 5 8 ; 8 2 ; 4 3 ; 8 7 ; 6 
5 ; 6 3 ; 6 7 ; 6 4 ; 6 8 ]; %elements nodal connectivity
    E = 200e3 + zeros(1,size(elems,1)); %Youngs Modulus
    A = [ 113.2 113.2 113.2 113.2 736.7 113.2 113.2 736.7 113.2 113.2 113.2 
113.2 113.2 334.3 334.3 ] ;
    bc = [ 1 1 0 ; 1 2 0 ; 2 1 0 ; 2 2 0 ]; %boundary conditions
    loads = [ 4 2 -35000 ; 6 2 0 ; 8 2 -35000  ]; %applied loads

    Nnodes = size(nodes,1); %number of nodes
    Nel = size(elems,1); %number of elements

% DEFINE K , u , and f

    alldofs = 1:2*Nnodes;

    K = zeros(2*Nnodes,2*Nnodes);
    u = zeros(2*Nnodes,1);
    f = zeros(2*Nnodes,1);

% SPECIFIED NODAL DISPLACEMENTS

    dofspec = [] ;

    for i=1:size(bc,1);

        thisdof = 2*(bc(i,1)-1) + bc(i,2);
        dofspec = [dofspec thisdof];

    end

        doffree = alldofs ;
        doffree(dofspec) = [] ;

% SPECIFIED NODAL FORCES

    for i=1:size(loads,1);

        a = loads(i,1);
        b = loads(i,2);
        f( 2*(a-1) + b) = loads(i,3);

    end

% DEFINE GLOBAL STIFFNESS

    for i=1:Nel;

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);
```

```
        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        T= [c s 0 0 ; -s c 0 0 ; 0 0 c s ; 0 0 -s c] ;


        kk = (A(i)*E(i))/L ; % AE/L Per member
        k_loc=[kk 0 -kk 0; 0 0 0 0; -kk 0 kk 0; 0 0 0 0] ; %local stiffness
        Kel= T'*k_loc*T ;

        %Determine Corresponding Global DOF

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)];

        %Assemble Element to Global Stiffness matrix

        K(eldofs,eldofs) = K(eldofs,eldofs) + Kel;

    end

% Displacements and Member Forces

    u(doffree) = K(doffree,doffree)^-1 * (f(doffree) -
K(doffree,dofspec)*u(dofspec)); %displacements

    f(dofspec) = K(dofspec,doffree)*u(doffree) +
K(dofspec,dofspec)*u(dofspec); %nodal forces / reactions

% Member Forces

    for i=1:Nel

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);

        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        kk = (A(i)*E(i))/L;
        la =[ c s -c -s ];

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)]; %currentdof

        Force(i,1)= la*u(eldofs)*kk ;

    end

% Stresses
```

```
for i=1:Nel
   stress(i,1) = Force(i)/A(i)
end

format long
disp(['Displacement vector:']); u
disp(['Reactions:']); f
disp(['Member Forces:']); Force

%plot old shape
figure (1); hold on;
plot(nodes(:,1), nodes(:,2), 'k.')
hold on; axis equal;
for iel=1:Nel
    elnodes=elems(iel, 1:2);
    nodesxy = nodes(elnodes, :);
    plot(nodesxy(:,1),nodesxy(:,2), 'k--')
end

%plot new shape
Magnification = 100;
nodesnew = nodes + Magnification*reshape(u,2,Nnodes)';
plot(nodesnew(:,1),nodesnew(:,2),'o', ...
    'MarkerEdgeColor' , 'k', 'MarkerFaceColor' , 'r', 'MarkerSize', 5)
hold on; axis equal;
for iel = 1:Nel
   elnodes =elems(iel, 1:2);
   nodesxy = nodesnew(elnodes, :);
   plot(nodesxy(:,1),nodesxy(:,2), 'b-', 'LineWidth', 0.5)
end
title('Displacement');
```

APPENDIX E – 15-Bar Truss Case 3

```
clc,clear all;

% NODAL AND MEMBER DEFINITION

    nodes = [ 0 0 ; 10160 0 ; 2540 0 ; 2540 3810 ; 5080 0 ; 5080 5080; 7620
0 ; 7620 3810 ]; %coordinate
    elems = [ 1 3 ; 3 5 ; 5 7 ; 7 2 ; 1 4; 4 5 ; 5 8 ; 8 2 ; 4 3 ; 8 7 ; 6
5 ; 6 3 ; 6 7 ; 6 4 ; 6 8 ]; %elements nodal connectivity
    E = 200e3 + zeros(1,size(elems,1)); %Youngs Modulus
    A = [ 113.2 113.2 113.2 113.2 736.7 113.2 113.2 736.7 113.2 113.2 113.2
113.2 113.2 334.3 334.3 ] ;
    bc = [ 1 1 0 ; 1 2 0 ; 2 1 0 ; 2 2 0 ]; %boundary conditions
    loads = [ 4 2 -35000 ; 6 2 -35000 ; 8 2 0  ]; %applied loads

    Nnodes = size(nodes,1); %number of nodes
    Nel = size(elems,1); %number of elements

% DEFINE K , u , and f

    alldofs = 1:2*Nnodes;

    K = zeros(2*Nnodes,2*Nnodes);
    u = zeros(2*Nnodes,1);
    f = zeros(2*Nnodes,1);

% SPECIFIED NODAL DISPLACEMENTS

    dofspec = [] ;

    for i=1:size(bc,1);

        thisdof = 2*(bc(i,1)-1) + bc(i,2);
        dofspec = [dofspec thisdof];

    end

        doffree = alldofs ;
        doffree(dofspec) = [] ;

% SPECIFIED NODAL FORCES

    for i=1:size(loads,1);

        a = loads(i,1);
        b = loads(i,2);
        f( 2*(a-1) + b) = loads(i,3);

    end

% DEFINE GLOBAL STIFFNESS

    for i=1:Nel;

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);
```

```
        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        T= [c s 0 0 ; -s c 0 0 ; 0 0 c s ; 0 0 -s c] ;


        kk = (A(i)*E(i))/L ; % AE/L Per member
        k_loc=[kk 0 -kk 0; 0 0 0 0; -kk 0 kk 0; 0 0 0 0] ; %local stiffness
        Kel= T'*k_loc*T ;

        %Determine Corresponding Global DOF

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)];

        %Assemble Element to Global Stiffness matrix

        K(eldofs,eldofs) = K(eldofs,eldofs) + Kel;

    end

% Displacements and Member Forces

    u(doffree) = K(doffree,doffree)^-1 * (f(doffree) -
K(doffree,dofspec)*u(dofspec)); %displacements

    f(dofspec) = K(dofspec,doffree)*u(doffree) +
K(dofspec,dofspec)*u(dofspec); %nodal forces / reactions

% Member Forces

    for i=1:Nel

        elnodes = elems(i,:);
        nodesxy = nodes(elnodes,:);

        % Transformation Matrix

        L = sqrt((nodesxy(1,1)-nodesxy(2,1))^2 + (nodesxy(1,2)-
nodesxy(2,2))^2);
        c= (nodesxy(1,1) - nodesxy(2,1))/L ;
        s= (nodesxy(1,2) - nodesxy(2,2))/L ;
        kk = (A(i)*E(i))/L;
        la =[ c s -c -s ];

        eldofs=[2*elnodes(1,1)-1 2*elnodes(1,1) 2*elnodes(1,2)-1
2*elnodes(1,2)]; %currentdof

        Force(i,1)= la*u(eldofs)*kk ;

    end

% Stresses
```

```
for i=1:Nel
   stress(i,1) = Force(i)/A(i)
end

format long
disp(['Displacement vector:']); u
disp(['Reactions:']); f
disp(['Member Forces:']); Force

%plot old shape
figure (1); hold on;
plot(nodes(:,1), nodes(:,2), 'k.')
hold on; axis equal;
for iel=1:Nel
    elnodes=elems(iel, 1:2);
    nodesxy = nodes(elnodes, :);
    plot(nodesxy(:,1),nodesxy(:,2), 'k--')
end

%plot new shape
Magnification = 100;
nodesnew = nodes + Magnification*reshape(u,2,Nnodes)';
plot(nodesnew(:,1),nodesnew(:,2),'o', ...
    'MarkerEdgeColor' , 'k', 'MarkerFaceColor' , 'r', 'MarkerSize', 5)
hold on; axis equal;
for iel = 1:Nel
   elnodes =elems(iel, 1:2);
   nodesxy = nodesnew(elnodes, :);
   plot(nodesxy(:,1),nodesxy(:,2), 'r-', 'LineWidth', 0.5)
end
title('Displacement');
```