Istanbul Technical University
Graduate School of Science Engineering and Technology

MKC525E
Finite Element Analysis in Engineering

Homework 1

Erdem Çalışkan
503191531
01/07/2020

Question 1

Truss system in the question can be seen in Figure 1. The truss members have the cross-sectional area of $8\,\mathrm{cm}^2$ and made of steel having the elasticity modulus of $E = 200\,\mathrm{GPa}$. Nodal deflections, element stresses and reaction forces are to be found. The system is hyperstatic.
Matlab code for the solution can be found in Listing 1.

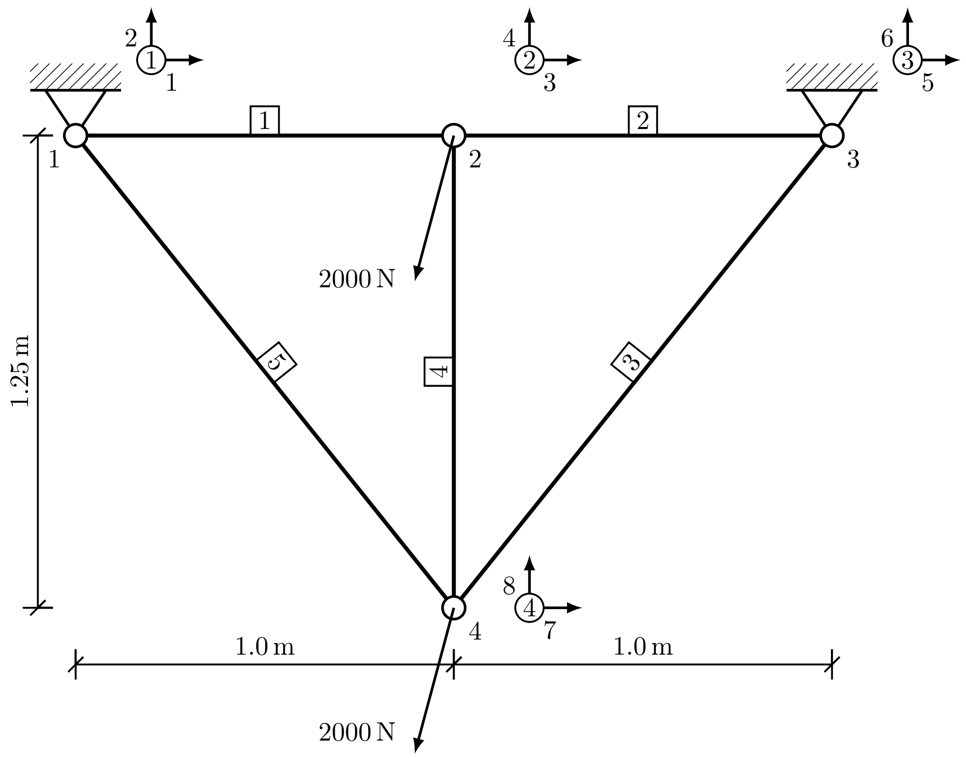


Figure 1: Truss system schematic for question 1

In the Figure 1, node numbers are given inside circle with the corresponding global DOF and element numbers are given inside squares. There are two forces acting on the system with $-15°$ between $y - axis$.
2D Truss element stiffness matrix is given by:

$$\mathbf{K} = \frac{EA}{L_e}\begin{bmatrix} l^2 & lm & -l^2 & -lm \\ lm & m^2 & -lm & m^2 \\ -l^2 & -lm & l^2 & lm \\ -lm & -m^2 & lm & m^2 \end{bmatrix} \tag{1}$$

Where $l = \cos\theta$ and $m = \sin\theta$.

Corresponding topology matrix for the system:

$$\text{Topology Matrix} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 5 & 6 \\ 7 & 8 & 5 & 6 \\ 7 & 8 & 3 & 4 \\ 7 & 8 & 1 & 2 \end{bmatrix} \tag{2}$$

Element Number	Stresses [MPa]
Element 1	-0.32
Element 2	0.32
Element 3	3.61
Element 4	-2.41
Element 5	2.57

Table 1: Element Stresses for Question 1.

Global DOF	Displacement [mm]
1	0
2	0
3	$-1.62E-03$
4	$-4.68E-02$
5	0
6	0
7	$-6.64E-03$
8	$-3.17E-02$

Table 2: Nodal displacements for Question 1.

Constrained DOF	Reaction Force [N]
1	-1028
2	1608
5	2063
6	2255

Table 3: Nodal Reaction forces in the constrained nodes for Question 1.

Question 2

Same formulation and similar code with Question 1 is used for Question 2.

In the Figure 2, node numbers are given inside circle with the corresponding global DOF and element numbers are given inside squares. There are two forces acting on the system with in lateral direction. The truss members have the cross-sectional area of 8 $in^2$and made of steel having the elasticity modulus of $E = 1.9 \cdot 10^4\ lb/in^2$. Nodal deflections, element stresses and reaction forces are to be found. The system is hyperstatic.
Matlab code for the solution can be found in Listing 2.
Corresponding topology matrix for the system:

$$\text{Topology Matrix} = \begin{bmatrix} 1 & 2 & 3 & 4 \\ 3 & 4 & 5 & 6 \\ 7 & 8 & 5 & 6 \\ 7 & 8 & 3 & 4 \\ 7 & 8 & 1 & 2 \\ 9 & 10 & 7 & 8 \end{bmatrix}$$

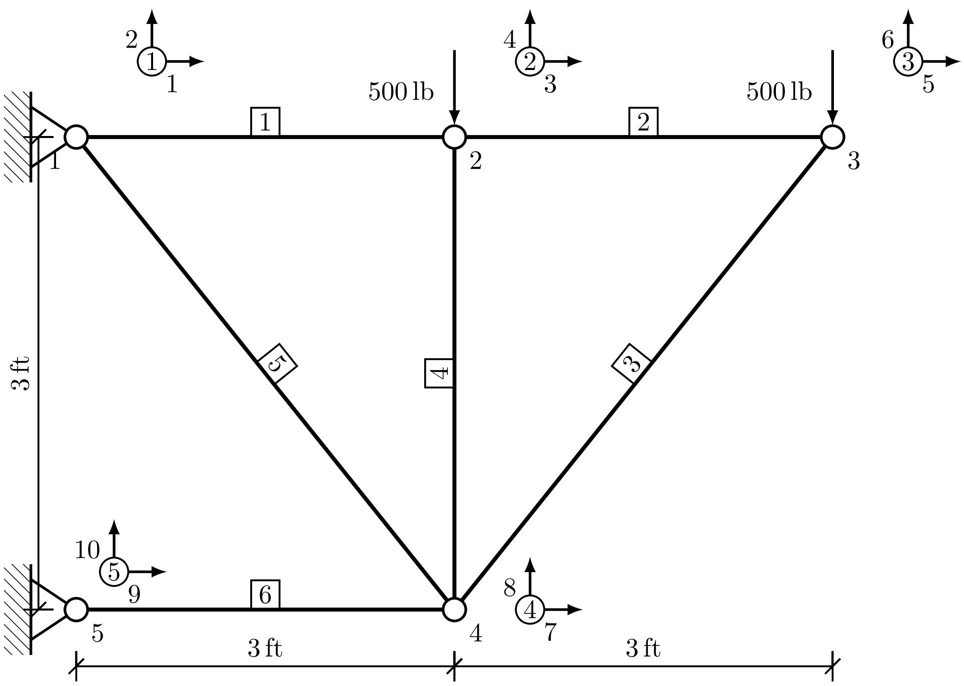


Figure 2: Truss system schematic for question 2

Element Number	Stresses [psi]
Element 1	62.5
Element 2	62.5
Element 3	−88.39
Element 4	−62.5
Element 5	176.78
Element 6	−187.5

Table 4: Element Stresses for Question 2.

Global DOF	Displacement [in]
1	0
2	0
3	0.12
4	−1.14
5	0.24
6	−1.95
7	−0.36
8	−1.03
9	0
10	0

Table 5: Nodal displacements for Question 2.

Constrained DOF	Reaction Force [lb]
1	−1500
2	1000
9	1500
10	0

Table 6: Nodal Reaction forces in the constrained nodes for Question 2.

Question 3

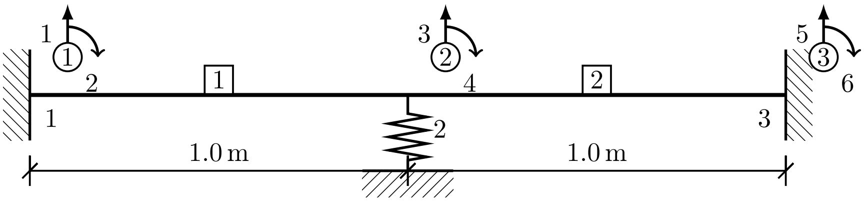


Figure 3: 2 element example for Euler-Bernoulli beam in Question 3, initial condition is $u_3 = 3\ mm$

A beam clamped at the both ends subjected to an initial deflection at the center with a spring also attached at the same point. Spring coefficient k = $100\,\mathrm{N \cdot mm^{-1}}$ and elastic modulus of the beam E = 210 GPa. The beam has the width of 8 mm, height of 2 mm and length of 1 m. Element stiffness matrix of Euler-Bernoulli beam is given by:

$$[k] = \int_0^L [B]^T EI[B]\,\mathrm{d}x = \frac{EI}{L^3}\begin{bmatrix} 12 & 6L & -12 & 6L \\ 6L & 4L^2 & -6L & 2L^2 \\ -12 & -6L & 12 & -6L \\ 6L & 2L^2 & -6L & 4L^2 \end{bmatrix} \tag{3}$$

$$[d^e] = \begin{bmatrix} w_1 \\ \theta_1 \\ w_2 \\ \theta_2 \end{bmatrix} \tag{4}$$

This question is solved using both Matlab and commercial program (Ansys). Results are compared using different element sizes. Matlab code for the solution can be found in Listing 3.

Commercial Program

A 3D model with linear elements are created. Body is divided into 4 from the midpoint to define a spring. Because of this, even though there are same number of elements throughout the length of the beam, total number of elements in this model is always 4 times more than the Matlab code. The model and the boundary conditions are given in Figure 4 and Figure 7.

Workflow in the analysis is to apply the initial deflection at the first time step, than deactivating the deflection. To ensure the system is in steady state, there is an additional time step after deactivation. So there are three time steps in the analysis.

Results

In this section, results of the Matlab code and the commercial code are going to be compared with same element size.

For 2 elements Deflection at the middle is little smaller than the initial value. Bending angle throughout the beam is zero because of the linear elements and no use of shape functions.

Number of Elements	Matlab Result [mm]	Ansys Result [mm]
2	2.9992	0.11175

Table 7: Maximum deflection compensation between Matlab code and Ansys for 2 elements.

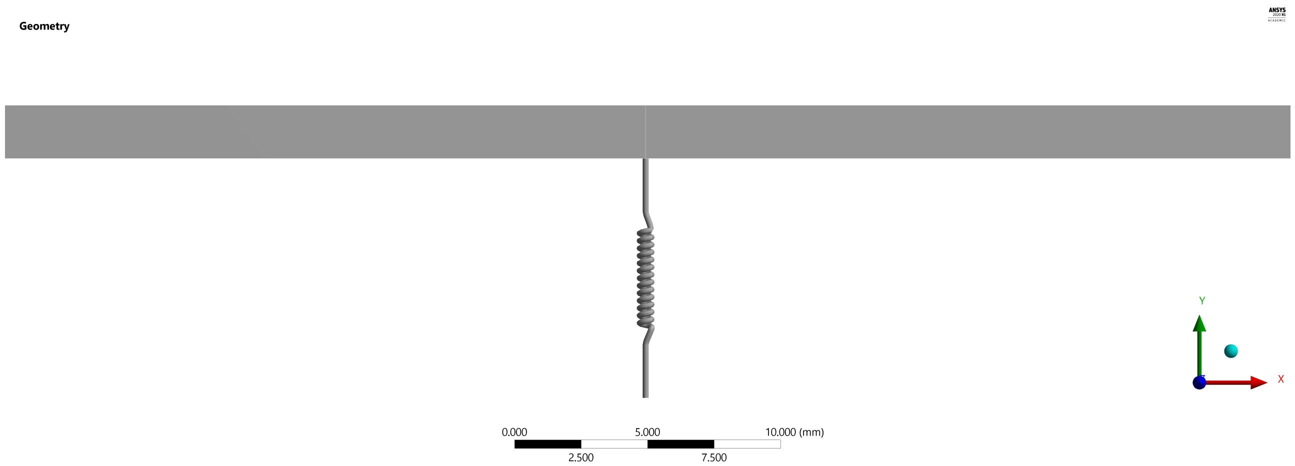


Figure 4: Middle portion of the modelled beam in Ansys.

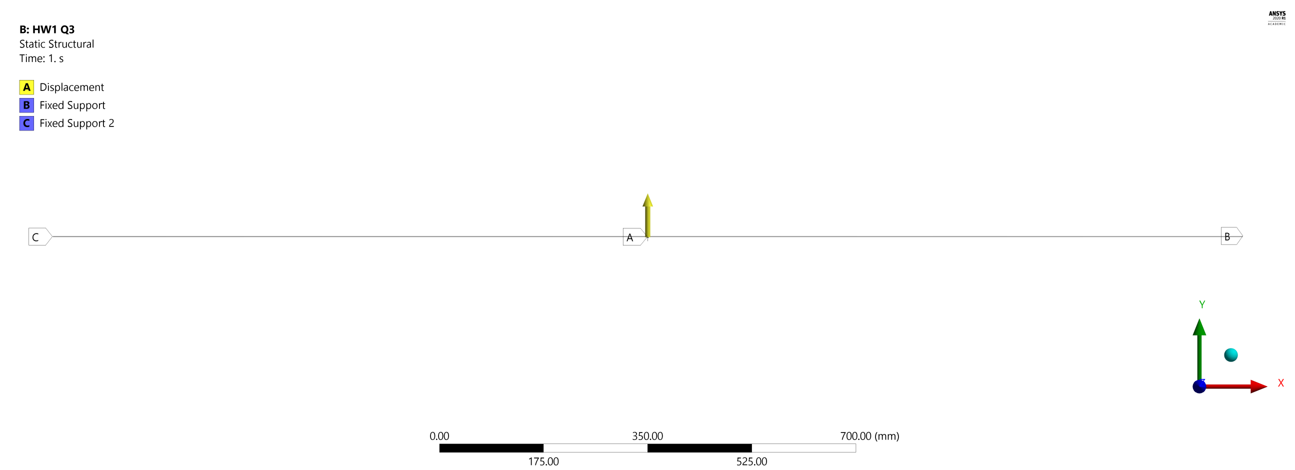
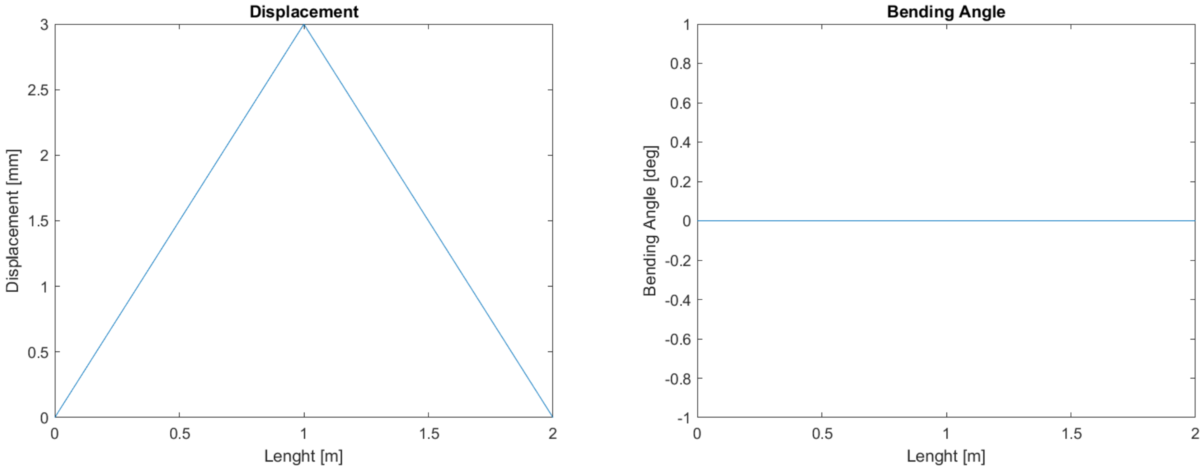


Figure 5: Boundary conditions in Ansys.



(a) Deflection [mm]

(b) Angle [°]

Figure 6: Results for 2 elements (Matlab)

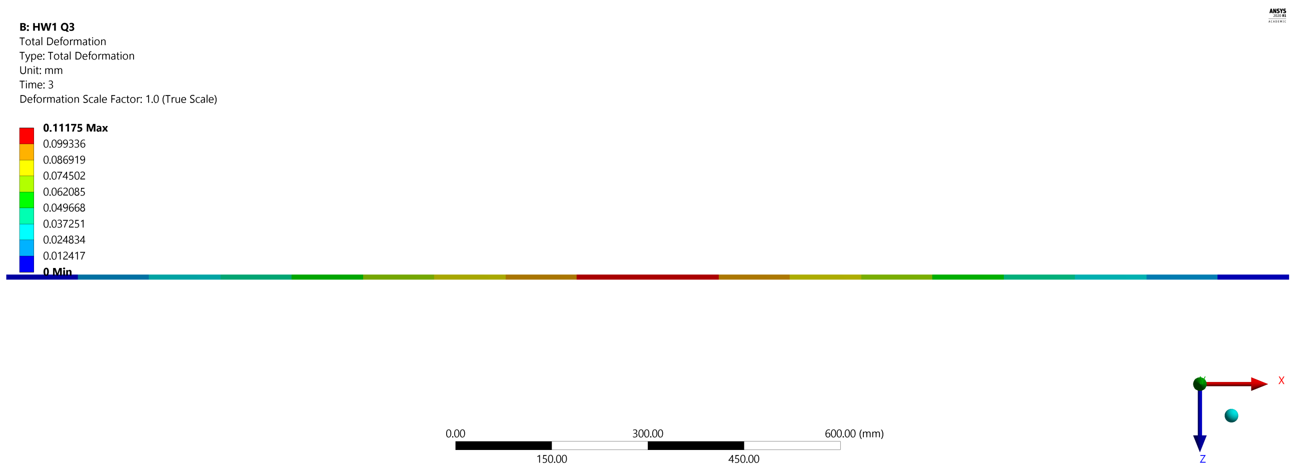
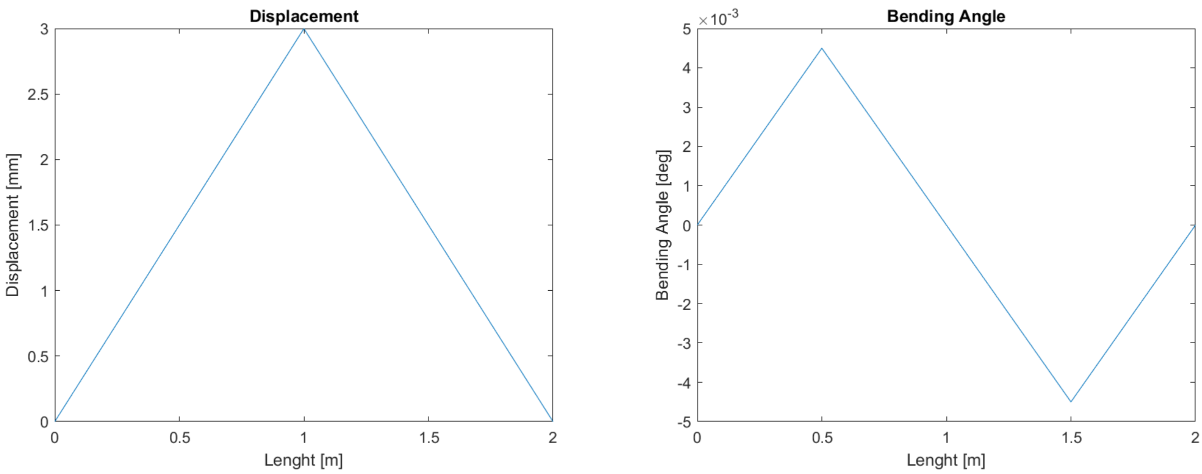


Figure 7: Results for 2 elements (Ansys)

For 4 elements Bending angle throughout the beam is not this time. Maximum deflection is same with 2 elements but the curves are more accurate. Results of both programs are similar from this point on wards.

Number of Elements	Matlab Result [mm]	Ansys Result [mm]
2	2.9992	2.9991

Table 8: Maximum deflection compensation between Matlab code and Ansys for 4 elements.



(a) Deflection [mm] (b) Angle [°]

Figure 8: Results for 4 elements (Matlab)

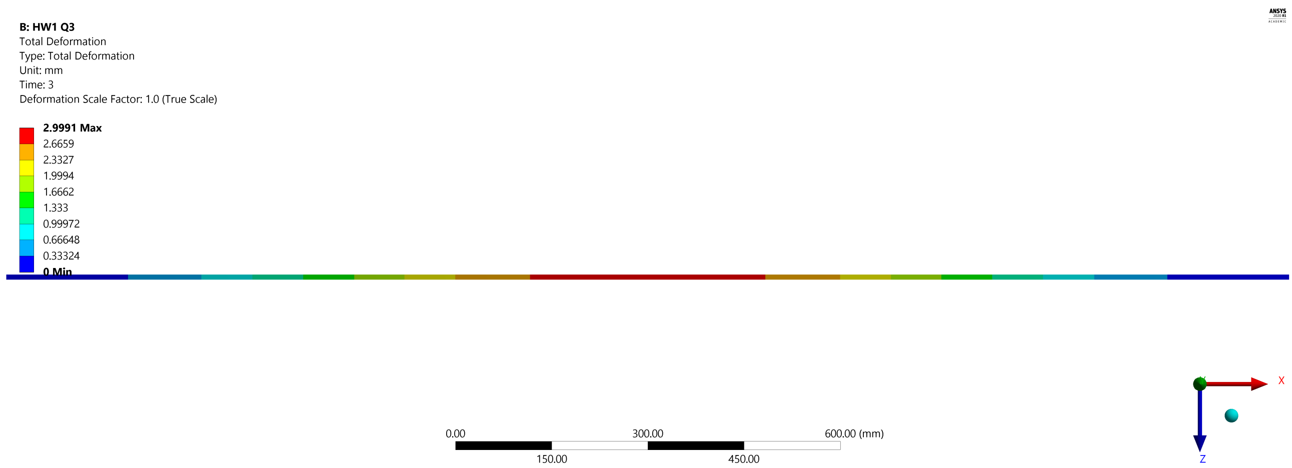


Figure 9: Results for 4 elements (Ansys)

Number of Elements	Matlab Result [mm]	Ansys Result [mm]
2	2.9992	2.9992

Table 9: Maximum deflection compensation between Matlab code and Ansys for 10 elements.

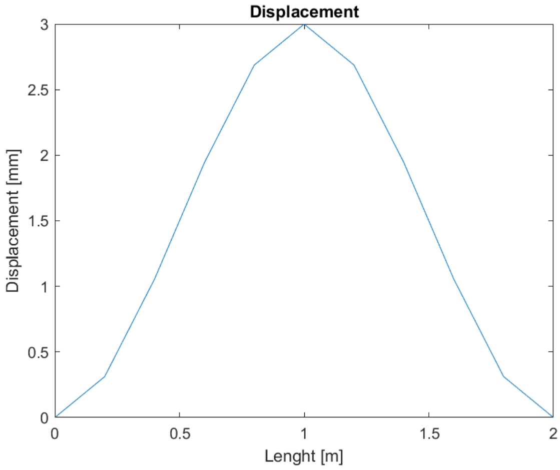
For 10 elements

Number of Elements	Matlab Result [mm]	Ansys Result [mm]
2	2.9992	2.9992

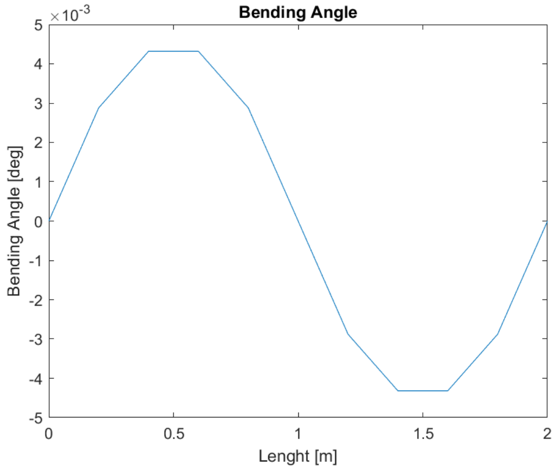
Table 10: Maximum deflection compensation between Matlab code and Ansys for 100 elements.

For 100 elements

For 1000 elements Bending angle and deflection is similar to the 100 elements.

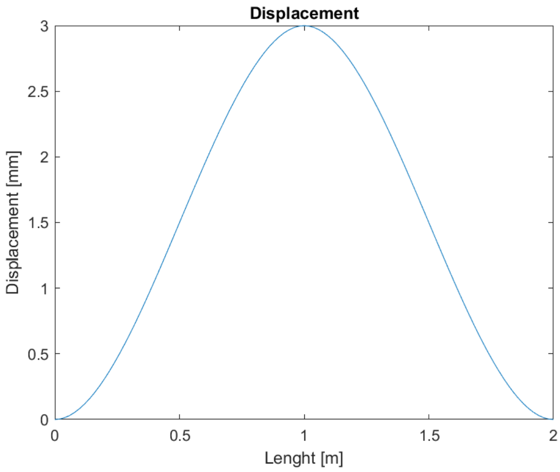


(a) Deflection [mm]

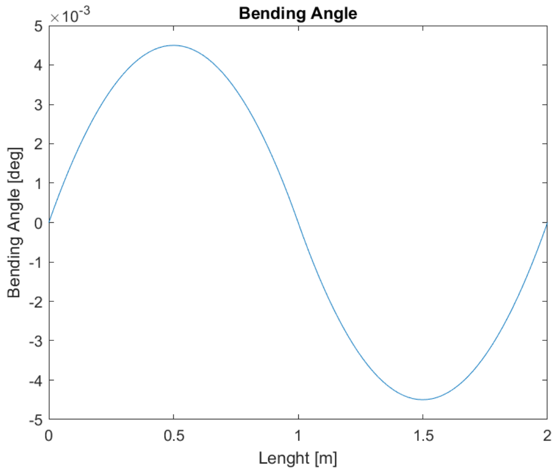


(b) Angle [°]

Figure 10: Results for 10 elements (Matlab)



(a) Deflection [mm]

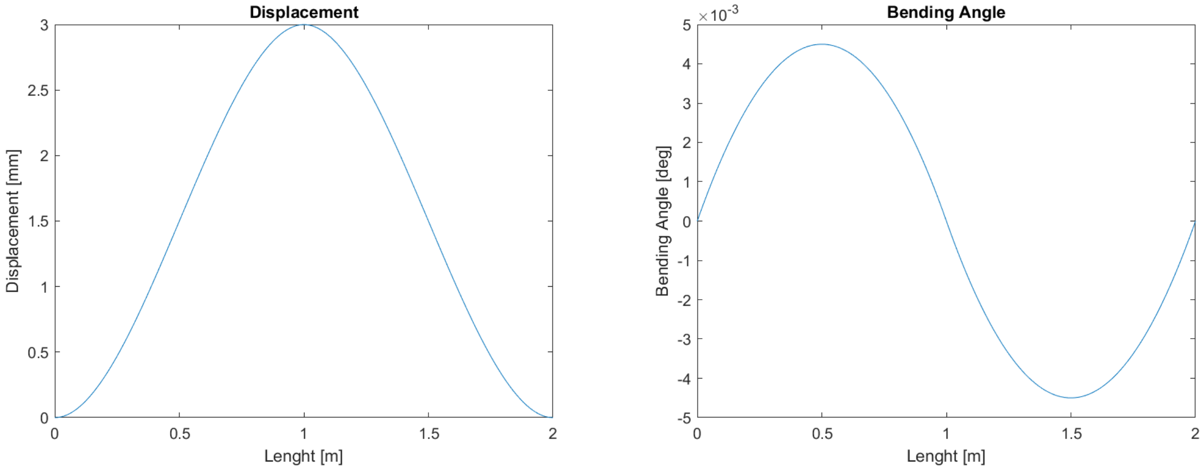


(b) Angle [°]

Figure 11: Results for 100 elements (Matlab)

Number of Elements	Matlab Result [mm]	Ansys Result [mm]
2	2.9992	2.9992

Table 11: Maximum deflection compensation between Matlab code and Ansys for 1000 elements.



(a) Deflection [mm]

(b) Angle [°]

Figure 12: Results for 1000 elements (Matlab)

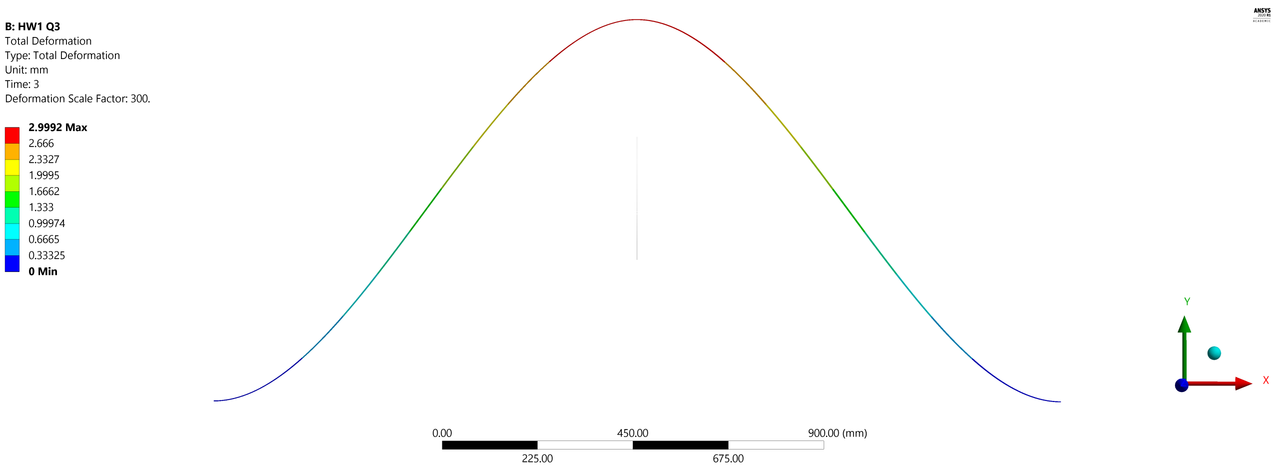


Figure 13: Results for 1000 elements (Ansys)

Question 4

In this question, temperature distribution through composite wall is to be found where convection heat loss occurs on the left surface. Cross-sectional area $A = 2\,\mathrm{m}^2$. Element stiffness matrices corresponding to heat convection and heat conduction are respectively given by:

$$\mathbf{K}^e = hA \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \text{ and } \mathbf{K}^e = \frac{kA}{L} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{5}$$

Composite wall schematic is given in 14. Matlab code used in this question is similar to the code in question 3 and can be found on Listing 4.
Here, 3 elements are used to solve the problem and they corresponds to the heat convection area, the wall with 2 cm thickness (k_1) and the wall with 6 cm thickness (k_2). Solution of the problem (temperatures at the nodes) is given in 16.

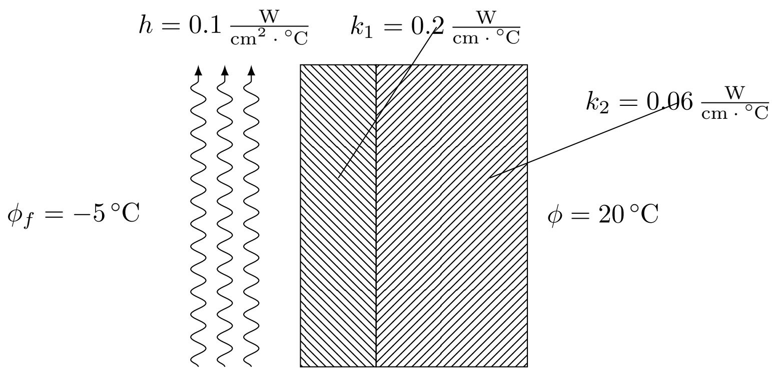


Figure 14: Compsite wall

Figure 15: Corresponding spring system of the problem.

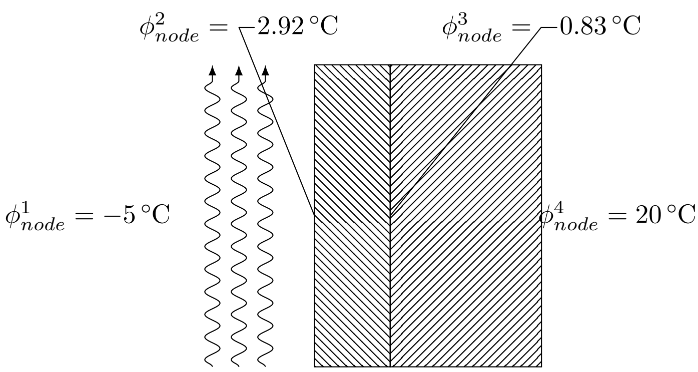


Figure 16: Temperatures at the nodes.

Matlab Scripts

Listing 1: Matlab code for question 1

```
%Q1 - Erdem Caliskan

clear all

e              = 200e9;              %Elasticity [Pa]
al             = 8e-4;               %Cross Section Area [m^2]
FORCE          = 2000;               %Forces [N]

nel            = 5;                  %Number of elements
nnode          = 4;                  %Number of nodes
NodalDOF       = 2*nnode;            %Total nodal DOF

elementNodes =    [    1  2;
                       2  3;
                       4  3;
                       4  2;
                       4  1];        %nodes

nodeCoords =      [    -1  1.25;
                       0  1.25;
                       1  1.25;
                       0  0];        %coordinates of nodes

xcoord = nodeCoords(:,1);
ycoord = nodeCoords(:,2);

%Define topology matrix

top = zeros(nel,4);
top(:,1) = (2*elementNodes(:,1))-1;
top(:,2) = (2*elementNodes(:,1));
top(:,3) = (2*elementNodes(:,2))-1;
top(:,4) = (2*elementNodes(:,2));

kglob = zeros(NodalDOF,NodalDOF);

disp('Topology Matrix')
disp(top)

%Defining Global Stiffness Matrix

for i=1:nel

    DOFe = top(i,:);

    nodesofelement = elementNodes(i,:);

    elementx = xcoord(nodesofelement(2)) - xcoord(nodesofelement(1));
    elementy = ycoord(nodesofelement(2)) - ycoord(nodesofelement(1));

    l = sqrt(elementx^2+elementy^2);

    C = elementx/l;
    S = elementy/l;

```

```
    ke=e*al/l*[ C^2 C*S -C^2 -C*S
                C*S S^2 -C*S -S*S
                -C^2 -C*S C^2 C*S
                -C*S -S^2 C*S S^2]; %element stiffness matrix

    kglob(DOFe,DOFe)=kglob(DOFe,DOFe)+ke;

end

kglob1 = kglob; % saves the kglob to calculate reaction forces
kglob2 = kglob; % saves the kglob to display

% Eleminate 1., 2., 5., 6. row and column for boundary conditions
% (Direct Method)

kglob(1,:) = [];
kglob(:,1) = [];
kglob(1,:) = [];
kglob(:,1) = [];
kglob(3,:) = [];
kglob(:,3) = [];
kglob(3,:) = [];
kglob(:,3) = [];

f = zeros(NodalDOF-4,1);
f(1) = -FORCE*sind(15);
f(2) = -FORCE*cosd(15);
f(3) = -FORCE*sind(15);
f(4) = -FORCE*cosd(15);

d = kglob\f; % solve the system

disp_total = [  0
                0
                d(1)
                d(2)
                0
                0
                d(3)
                d(4)]; % insert 0 displacements

stresses = zeros (nel,1);

for j = 1:nel % Defining elements stresses

    DOFe = top(j,:);

    nodesofelement = elementNodes(j,:);

    elementx = xcoord(nodesofelement(2)) - xcoord(nodesofelement(1));
    elementy = ycoord(nodesofelement(2)) - ycoord(nodesofelement(1));

    l = sqrt(elementx^2+elementy^2);

    C = elementx/l;
    S = elementy/l;

    elongation =((-disp_total(DOFe(1,1))+disp_total(DOFe(1,3)))*C)+...
```

```
                    ((-disp_total(DOFe(1,2))+disp_total(DOFe(1,4)))*S);

    strain = elongation/l;

    stresses(j,1)=strain*e;
end

kglob1(3,:) = [];
kglob1(3,:) = [];
kglob1(5,:) = [];
kglob1(5,:) = [];

nodalforces = kglob1*disp_total ;

%Check the force balance

fx = nodalforces(1) + nodalforces(3) - 2*FORCE*sind(15)
fy = nodalforces(2) + nodalforces(4) - 2*FORCE*cosd(15)

disp('Displacements')
disp(disp_total)
disp('Nodal Reaction Forces')
disp(nodalforces)
disp('Member Stresses')
disp(stresses)
```

Listing 2: Matlab code for question 2

```
%Q2 - Erdem Caliskan

clear all


e            = 1.9e4;             %Elasticity [Pa]
al           = 8;                 %Cross Section Area [m^2]
FORCE        = 500;               %Forces [N]

nel          = 6;                 %Number of elements
nnode        = 5;                 %Number of nodes
NodalDOF     = 2*nnode;           %Total nodal DOF

elementNodes =   [    1  2;
                      2  3;
                      4  3;
                      4  2;
                      4  1
                      5  4];      %nodes

nodeCoords =     [    0  0;
                      36  0;
                      72  0;
                      36  -36;
                      0  -36];    %coordinates of nodes

xcoord = nodeCoords(:,1);
ycoord = nodeCoords(:,2);

%Define topology matrix
top = zeros(nel,4);
top(:,1) = (2*elementNodes(:,1))-1;
top(:,2) = (2*elementNodes(:,1));
top(:,3) = (2*elementNodes(:,2))-1;
top(:,4) = (2*elementNodes(:,2));
kglob=zeros(NodalDOF,NodalDOF);

disp('Topology Matrix')
disp(top)

%Defining Global Stiffness Matrix

for i=1:nel

    DOFe = top(i,:);

    nodesofelement = elementNodes(i,:);

    elementx = xcoord(nodesofelement(2)) - xcoord(nodesofelement(1));
    elementy = ycoord(nodesofelement(2)) - ycoord(nodesofelement(1));

    l = sqrt(elementx^2+elementy^2);

    C = elementx/l;
    S = elementy/l;

    ke=e*al/l*[ C^2 C*S -C^2 -C*S
```

```
                    C*S  S^2 -C*S -S*S
                    -C^2 -C*S C^2 C*S
                    -C*S -S^2 C*S S^2]; %element stiffness matrix

     kglob(DOFe,DOFe)=kglob(DOFe,DOFe)+ke;


end

% Eleminate 1., 2., 9., 10. row and column for boundary conditions
% (Direct Method)

kglob1 = kglob;

kglob(1,:) = [];
kglob(:,1) = [];
kglob(1,:) = [];
kglob(:,1) = [];
kglob(7,:) = [];
kglob(:,7) = [];
kglob(7,:) = [];
kglob(:,7) = [];

f = zeros(NodalDOF-4,1);
f(2) = -FORCE;
f(4) = -FORCE;

d = kglob\f; %solve the system

disp_total = [0
     0
     d
     0
     0
     ];

stresses = zeros (nel,1);

for j = 1:nel %Defining elements stresses

     DOFe = top(j,:);

     nodesofelement = elementNodes(j,:);

     elementx = xcoord(nodesofelement(2)) - xcoord(nodesofelement(1));
     elementy = ycoord(nodesofelement(2)) - ycoord(nodesofelement(1));

     l = sqrt(elementx^2+elementy^2);

     C = elementx/l;
     S = elementy/l;

     elongation =((-disp_total(DOFe(1,1))+disp_total(DOFe(1,3)))*C)+...
                  ((-disp_total(DOFe(1,2))+disp_total(DOFe(1,4)))*S);

     strain = elongation/l;

     stresses(j,1)=strain*e;
```

```
end

kglob1(3,:) = [];
kglob1(3,:) = [];
kglob1(3,:) = [];
kglob1(3,:) = [];
kglob1(3,:) = [];
kglob1(3,:) = [];

nodalforces = kglob1*disp_total;

%Check the force balance

fx = nodalforces(1) + nodalforces(3)
fy = nodalforces(2) + nodalforces(4) - 2*FORCE

disp('Displacements')
disp(disp_total)
disp('Nodal Reaction Forces')
disp(nodalforces)
disp('Member Stresses')
disp(stresses)
```

Listing 3: Matlab code for question 3

```
%Q3 - Erdem Caliskan

clear all
close all
format compact
format short e
tic

nel = 1000;             % Number of elements
nnode = nel + 1;

L = 2000;               %lenght [mm]
e = 210e3;              %MPa
h = 5.333;              %cross-sectional inertia
l = L/nel;              %element lenght [mm]
al = 16;                %cross-sectional area [mm^2]

% Topology matrix
topsay = 0;
top1 = zeros(nel,4);

for i1 = 1:nel

    topsay = topsay + 1;
    top1(i1,1) = topsay;
    topsay = topsay + 1;
    top1(i1,2) = topsay;
    topsay = topsay + 1;
    top1(i1,3) = topsay;
    topsay = topsay + 1;
    top1(i1,4) = topsay;
    topsay = topsay - 2;
end

% Stiffness matrix
nbir = max(max(top1));
kglob = zeros(nbir,nbir);

for isay = 1:nel

    kel = e*h/(l*l*l)*[12 6*l -12 6*l
        6*l 4*l*l -6*l 2*l*l
        -12 -6*l 12 -6*l
        6*l 2*l*l -6*l 4*l*l]; % Element stiffness matrix
    for i1 = 1:4 % Global stiffness matrix
        for j1 = 1:4
            kglob(top1(isay,i1),top1(isay,j1)) = ...
            kglob(top1(isay,i1),top1(isay,j1)) + kel(i1,j1);
        end
    end
end

% Global stiffness matrix with BC's, both ends are clamped
% Direct method

for i2 = 1:nbir-4
    for j2 =1 :nbir-4
```

```
        kglob2(i2,j2) = kglob(i2+2,j2+2);
    end
end

[mg1,junk1] = size(kglob2);
ef = zeros(mg1,1);

% Define the force application point: Vertical load (upward) at the first
% node

FORCE = 300;
ef(mg1/2,1) = 1*FORCE;

% Boundary cond.: Spring at the middle of beam, spring stiffness added to
% the global stiffness

kglob2(mg1/2,mg1/2) = kglob2(mg1/2,mg1/2) + 1e2;

d1 = kglob2\ef; %solve

[s1,s2] = size(d1);

d = zeros(s1+4,1);

d(3:s1+2,1) = d1(:,1);  % add the clamped ends to the deflections

u = d(1:2:s1+4,1);      % vertical displacements
phi = d(2:2:s1+4,1);    % bending angle

f = kglob*d;            % calculate the reaction forces

R = f(1:2:s1+4,1);      % rection forces
M = f(2:2:s1+4,1);      % moments

lenght = (0:nel)/nel*2;

% plot the deflection and the bending angle throughout the beam

figure('Name','Displacement');
plot(lenght,u)
title('Displacement')
xlabel('Lenght [m]')
ylabel('Displacement [mm]')

filename = ['deflection' num2str(nel) '.png'];
saveas(1,filename)

figure('Name','Bending Angle');
plot(lenght,phi)
title('Bending Angle')
xlabel('Lenght [m]')
ylabel('Bending Angle [deg]')
filename = ['angle' num2str(nel) '.png'];
saveas(2,filename)
toc
```

Listing 4: Matlab code for question 4

```
%Q4 - Erdem Caliskan

clear all
close all
format compact
format short e
tic

nel = 3;
nnode = nel + 1;

l1    = 0.02;        %m
l2    = 0.06;        %m
h     = 1000;        %W/m^2*C
k1    = 20;          %W/m*C
k2    = 6;           %W/m*C
phiF  = -5;          %C
phi   = 20;          %C
al    = 2;           %m2

topsay = 0;
top1 = zeros(nel,4);

for i1 = 1:nel

    topsay = topsay + 1;
    top1(i1,1) = topsay;

    topsay = topsay + 1;
    top1(i1,2) = topsay;

    topsay = topsay + 1;
    top1(i1,3) = topsay;

    topsay = topsay + 1;
    top1(i1,4) = topsay;

    topsay = topsay - 2;
end

nbir = max(max(top1));
kglob = zeros(nbir,nbir);

kel_hava = h*al*[1 -1 0 0
                 -1 1 0 0
                 0 0 0 0
                 0 0 0 0];

kel_1 = k1*al/l1*[0 0 0 0
                  0 1 -1 0
                  0 -1 1 0
                  0 0 0 0];

kel_2 = k2*al/l2*[0 0 0 0
                  0 0 0 0
                  0 0 1 -1
                  0 0 -1 1];
```

```

kglob = kel_hava + kel_1 + kel_2;

kglob1 = kglob;
kglob1(1,:) = [1 0 0 0];
kglob1(4,:) = [0 0 0 1];

f = zeros(4,1);

f(1) = phiF;
f(4) = phi;

d = kglob1\f;

disp('Temperatures at nodes')
disp(d)

toc
```