Istanbul Technical University
Graduate School of Science Engineering and Technology

MKC525E
Finite Element Analysis in Engineering

Homework 5

Erdem Çalışkan
503191531
01/07/2020

1 Question 1

Domain in the problem: **A5+B3+C3+D2**

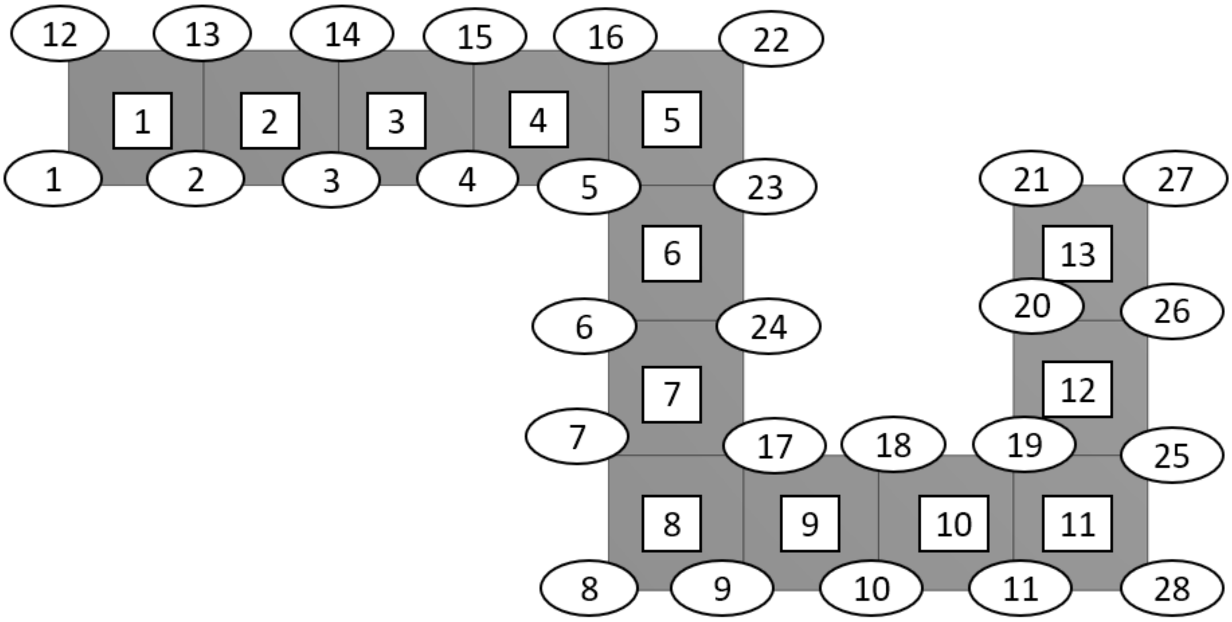


Figure 1: Domain; element numbers are inside square, node numbers are inside ellipse.

Topology matrix:

$$\begin{matrix} 1 & 2 & 3 & 4 & 25 & 26 & 23 & 24 \\ 3 & 4 & 5 & 6 & 27 & 28 & 25 & 26 \\ 5 & 6 & 7 & 8 & 29 & 30 & 27 & 28 \\ 7 & 8 & 9 & 10 & 31 & 32 & 29 & 30 \\ 9 & 10 & 45 & 46 & 43 & 44 & 31 & 32 \\ 11 & 12 & 47 & 48 & 45 & 46 & 9 & 10 \\ 13 & 14 & 33 & 34 & 47 & 48 & 11 & 12 \\ 15 & 16 & 17 & 18 & 33 & 34 & 13 & 14 \\ 17 & 18 & 19 & 20 & 35 & 36 & 33 & 34 \\ 19 & 20 & 21 & 22 & 37 & 38 & 35 & 36 \\ 21 & 22 & 55 & 56 & 49 & 50 & 37 & 38 \\ 37 & 38 & 49 & 50 & 51 & 52 & 39 & 40 \\ 39 & 40 & 51 & 52 & 53 & 54 & 41 & 42 \end{matrix} \tag{1}$$

$$[\text{ff}] = \begin{Bmatrix} 31100 \\ 8333 \\ 2233 \\ 8333 \end{Bmatrix} \tag{2}$$

$$\text{Maximum deflection} = 0.46\,\text{mm} \tag{3}$$

$$\text{Maximum deflection (Ansys)} = 0.44\,\text{mm} \tag{4}$$

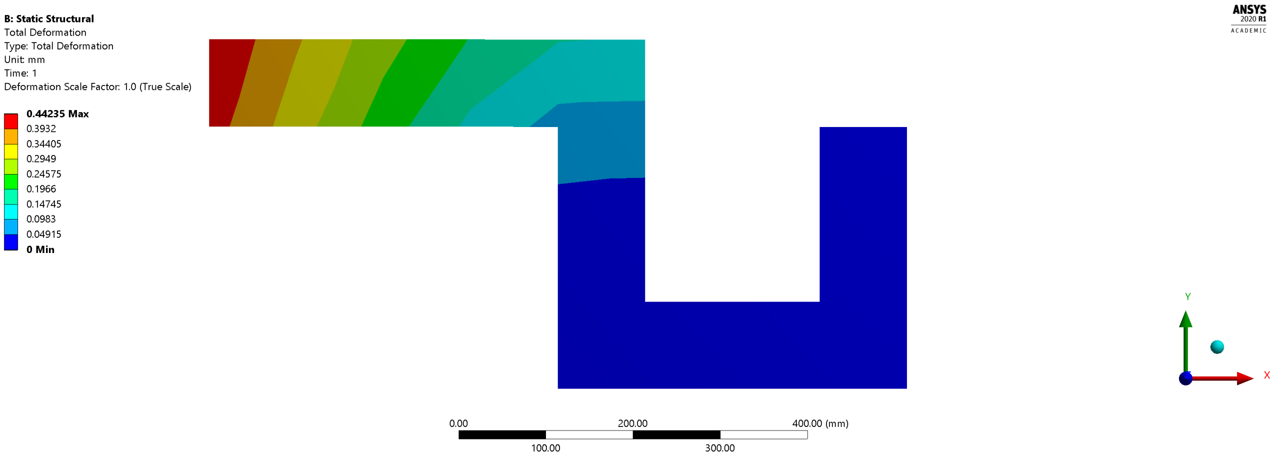


Figure 2: Total deformation in Ansys.

Mode	Natural frequency (Normal) [Hz]	Natural frequency (Constrained) [Hz]
1	3157.60 i	2522.95 i
2	2815.50 i	2452.48 i
3	2457.56 i	2210.45 i
4	1771.23 i	1736.88 i
5	1693.28 i	356.98
6	177.97	1419.05
7	399.60	1598.20
8	619.53	2326.74
9	1649.43	2705.62
10	2272.36	3198.49

Table 1: First 10 modes of the plate

Mode	Natural frequency (Normal) [Hz]	Natural frequency (Constrained) [Hz]
1	0.00	201.34
2	0.00	616.26
3	0.00	1319.40
4	386.46	2013.70
5	626.34	3185.60
6	1457.60	4324.90
7	2248.60	4633.20
8	3697.30	5291.00
9	4814.20	5912.20
10	5232.30	7182.50

Table 2: First 10 modes of the plate (Ansys)

1.1 Deflection at the uuper right edge of the plate when a sudden in-plane distributed loading of $0.5\,\mathrm{N/mm^2}$ is applied at $t = 0$

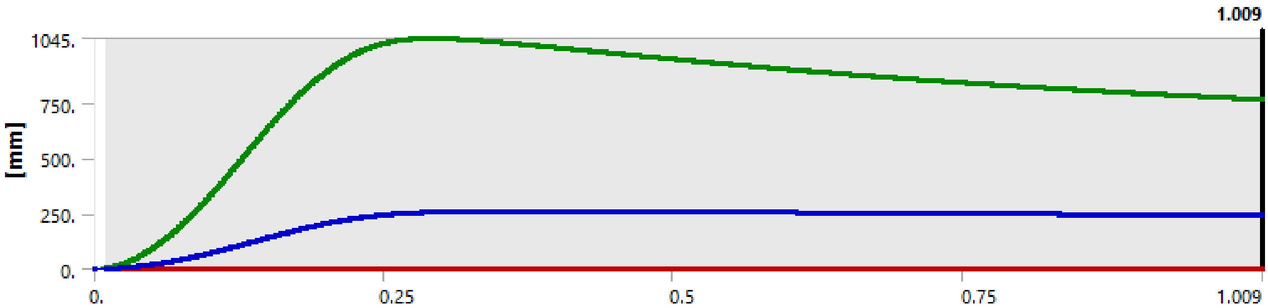
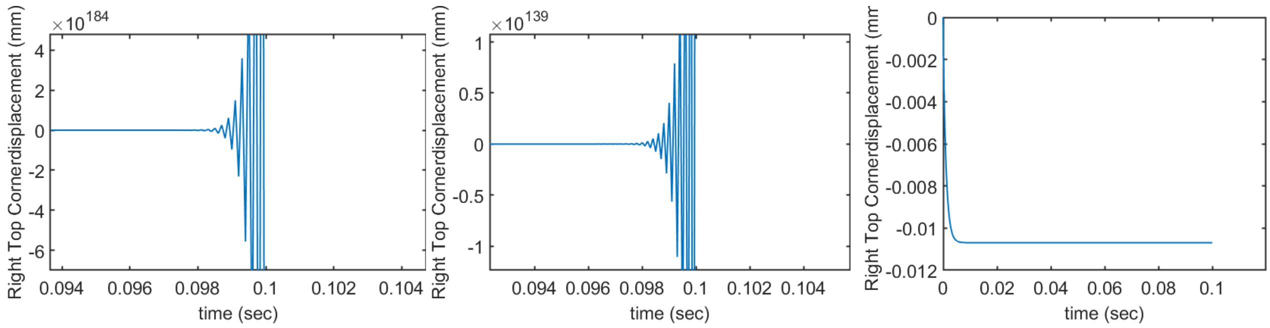


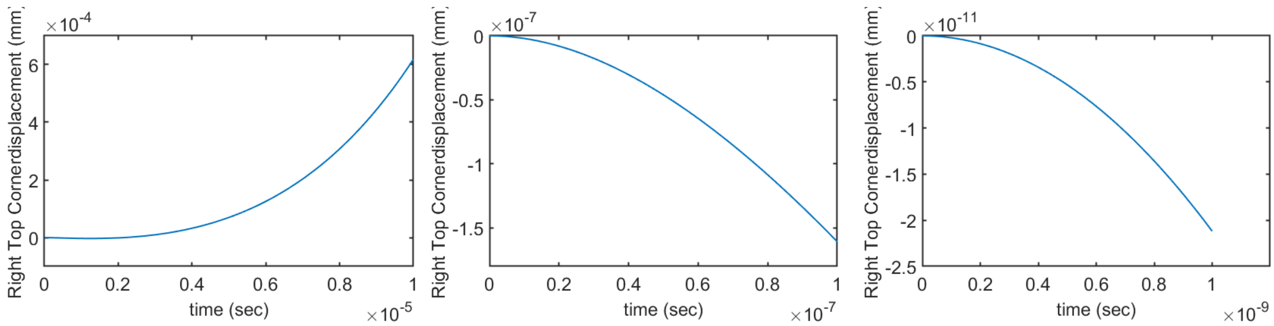
Figure 3: Total deformation in Ansys Explicit Dynamics (Plane strain).

In this section three different time steps are compared using Central Difference, Trapezoidal and Damped Newark methods. And different number of time steps are examined using Trapezoidal method.



(a) Central Difference (b) Trapezoidal (c) Damped Newark

Figure 4: Deflection at the upper right edge of the plate ($\Delta t = 1e-4$).



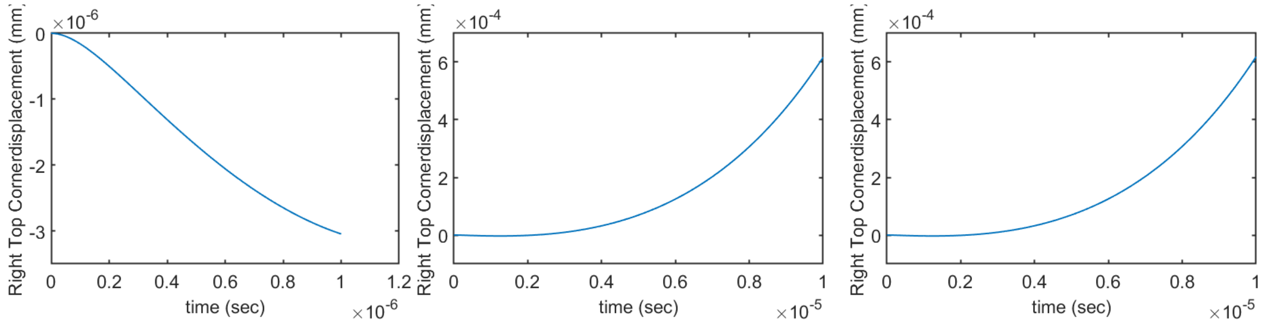
(a) ($\Delta t = 1 \times 10^{-8}$) (b) ($\Delta t = 1 \times 10^{-10}$) (c) ($\Delta t = 1 \times 10^{-12}$)

Figure 5: Deflection at the upper right edge of the plate for ($t = 1e4$) time steps.

1.2 Reaction forces

DOF	Node 8	Node 9	Node 10	Node 11	Node 28
u	9	-96	-134	185	171
v	-158	-477	-1054	-1104	-534

Table 3: Reaction forces (Matlab)



(a) 1×10^6 number of steps (b) $(1 \times 10^7$ number of steps (c) 1×10^8 number of steps

Figure 6: Deflection at the upper right edge of the plate for $(\delta t = 1e - 12)$ time steps.

DOF	Node 8	Node 9	Node 10	Node 11	Node 28
u	37	195	218	-11	-2
v	-393	-90	147	-50	-13

Table 4: Reaction forces (Ansys)

1.3 Element stresses

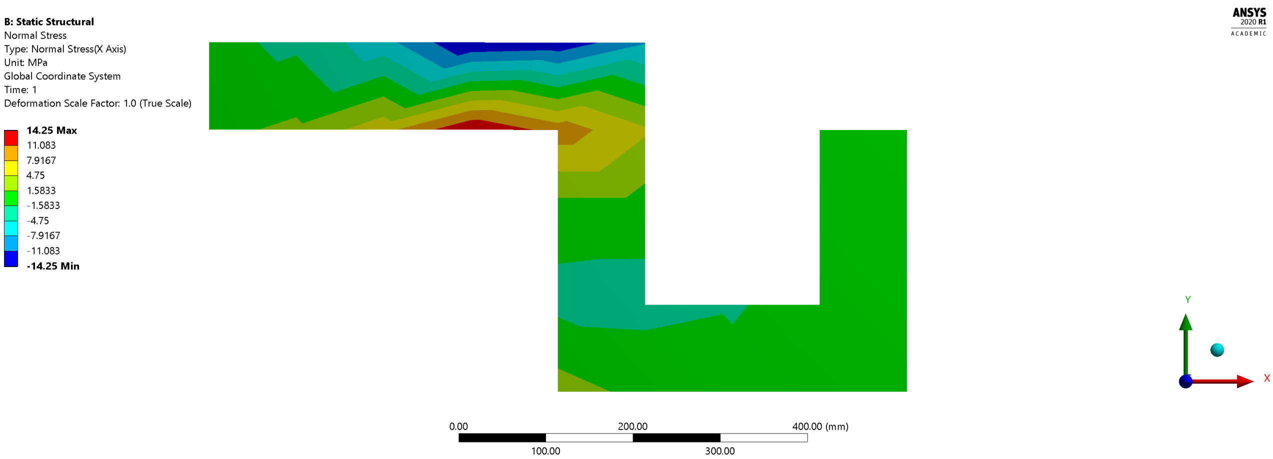


Figure 7: Normal stress (x-direction) in Ansys.

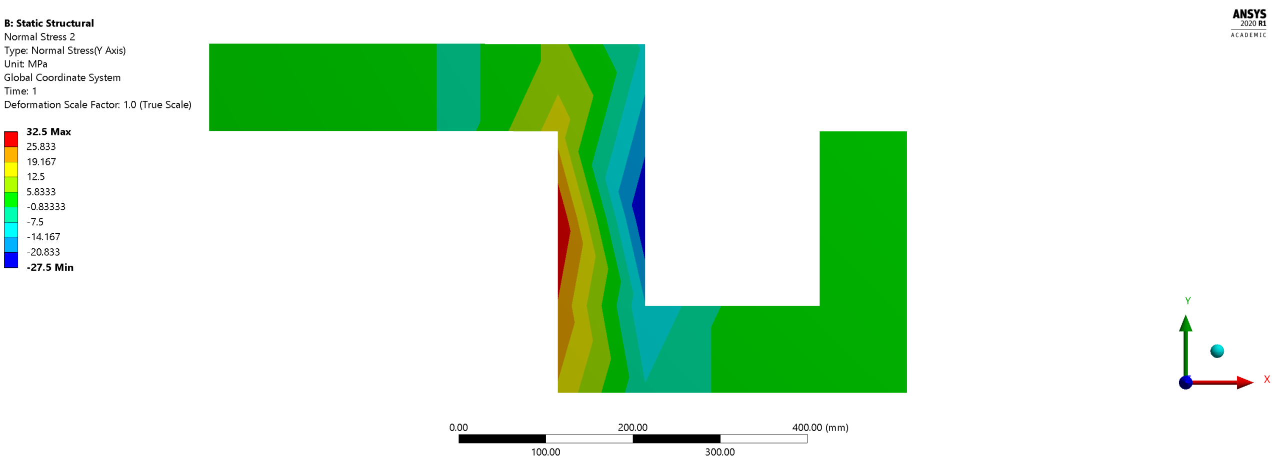


Figure 8: Normal stress (y-direction) in Ansys.

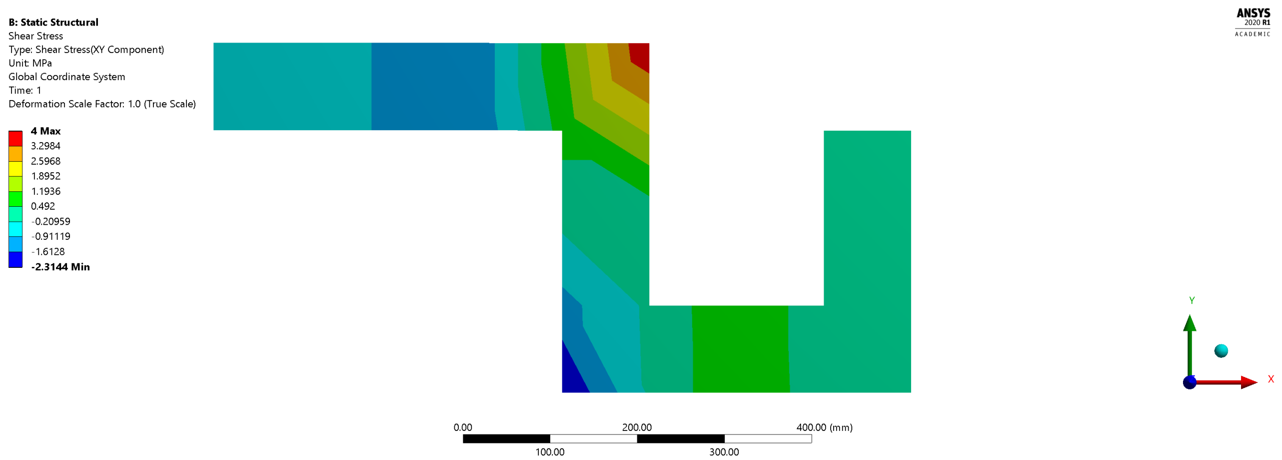


Figure 9: Shear stress in Ansys.

Element No	Element 1	Element 2	Element 3	Element 4	Element 5	Element 6	Element 7
σ_{11}	-2.20	0.90	3.47	-0.63	-2.58	-0.51	0.26
σ_{22}	3.46	3.20	0.91	-1.98	-0.10	2.25	1.22
τ	-0.48	-1.18	-0.58	0.95	0.39	-0.48	-0.42
Element No	Element 8	Element 9	Element 10	Element 11	Element 12	Element 13	
σ_{11}	0.05	0.00	0.09	0.08	-3.63	-2.63	
σ_{22}	0.16	0.00	0.28	0.25	2.98	2.04	
τ	-0.05	0.09	-0.01	-0.06	-0.48	-1.18	

Table 5: Element stresses

1	2398.07 i
2	2283.47 i
3	1910.13 i
4	1500.89 i
5	349.20
6	1206.37
7	1357.62
8	2057.53
9	2158.28
10	2402.52

Table 6: Natural frequencies, found using lumped mass matrices.

2 Question 2

Same domain is used for this problem but node numbering is different.

Boundary conditions:

1. Temperature at the left edge $T_L = 20\,°C$
2. Temperature at the right edge $T_L = 50\,°C$

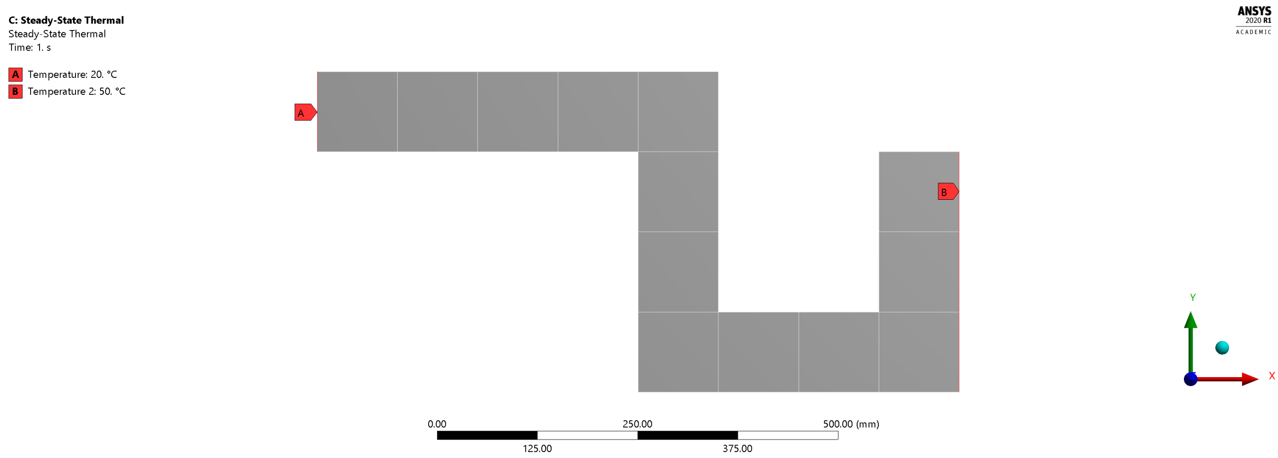


Figure 10: Boundary conditions.

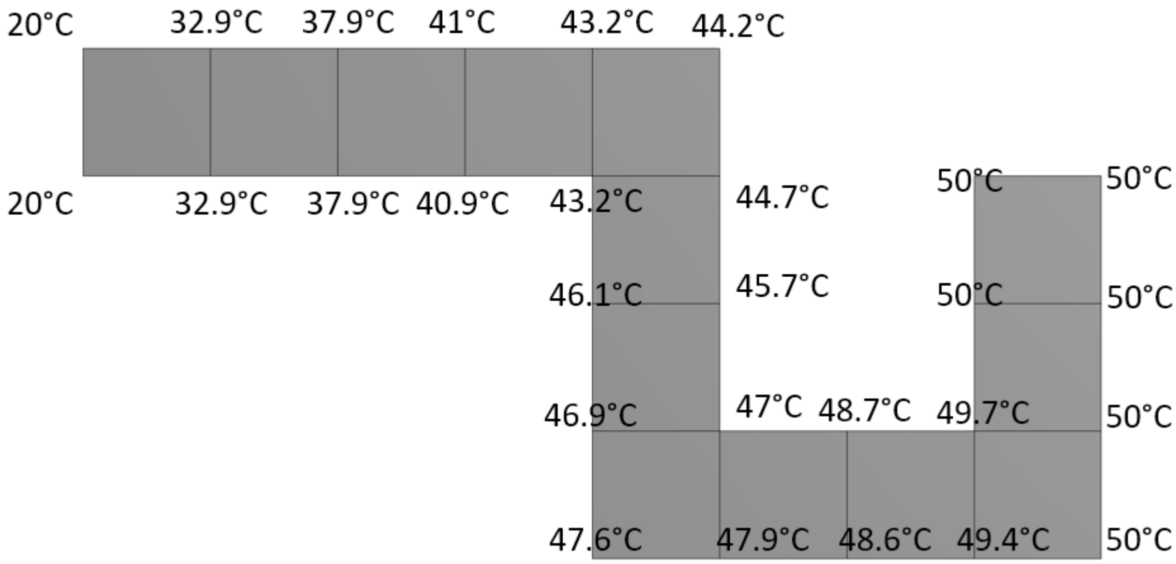


Figure 11: Temperature distribution.

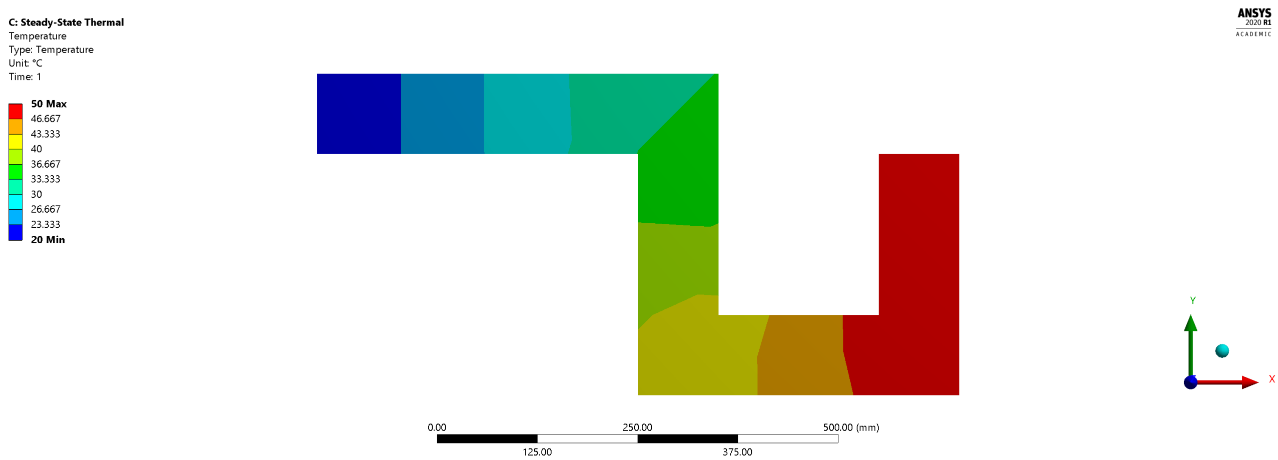


Figure 12: Temperature distribution.

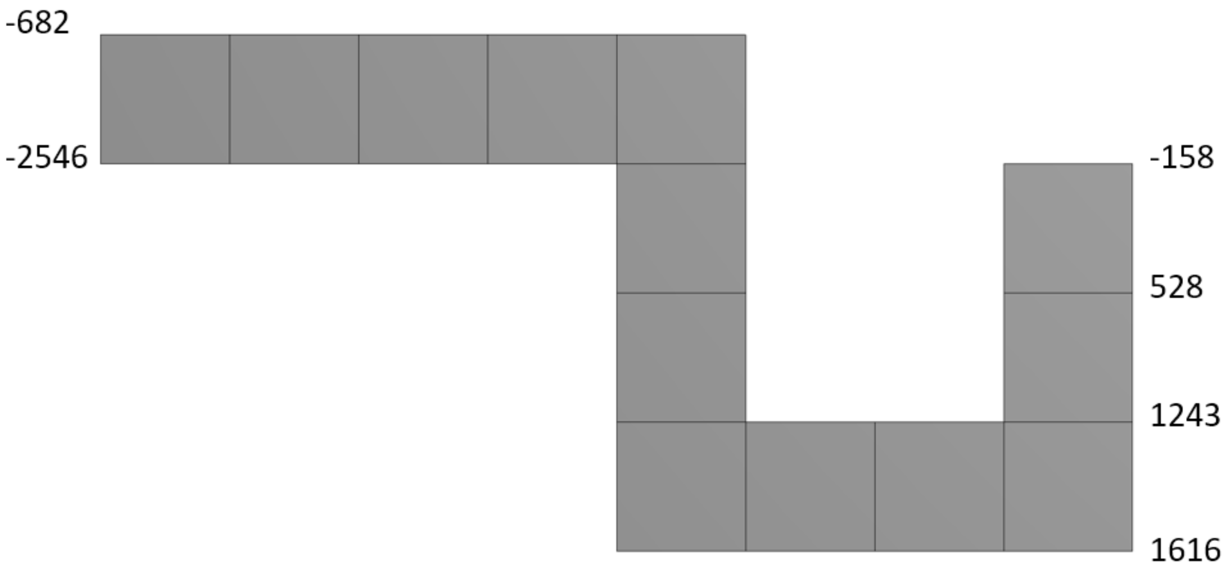


Figure 13: Heat flux.

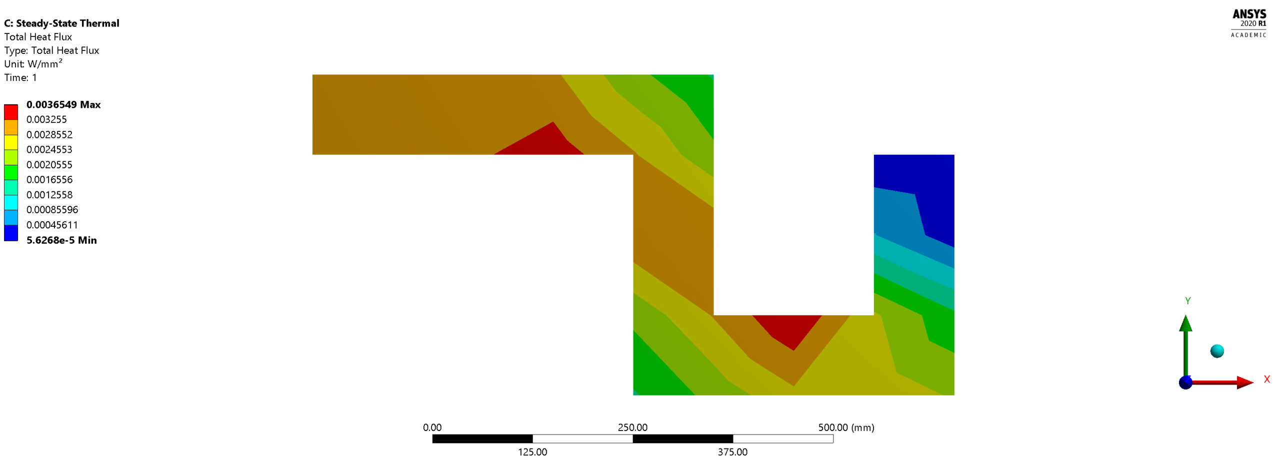


Figure 14: Heat flux.

Matlab Scripts

Listing 1: Matlab code for question 1

```
%HW5 Q1

clc
clear all
close all
format compact
format short e
tic

nel=13; %Total Number of Elements
nnode=28; %Total Number of Nodes

num_node=28;                    %number of total nodes
num_elem=13;                    %number of total elements

element_type=4;                 %number of nodes on each element
ndof=2;                         %degrees-of-freedom per node

num_eq=num_node*ndof;           %number of equations

h=2; %Thickness
E=2.1e5; %Young Modulus
v=0.33; %Poisson Ratio

a=100; %Length of 1 element
b=100; %Width of 1 element
ro=7.85e-9; %Density

pz=5; %Distributed load



Kelstiff = zeros(12,12,nel);

k11 = [ 8+3*(1-v)     3*(1+v)       -8+3*(1-v)      3*(3*v-1)
        3*(1+v)       8+3*(1-v)     -3*(3*v-1)      4-3*(1-v)
        -8+3*(1-v)    -3*(3*v-1)    8+3*(1-v)       -3*(1+v)
        3*(3*v-1)     4-3*(1-v)     -3*(1+v)        8+3*(1-v)];

k21 = [ -4-3*(1-v)    -3*(1+v)      4-3*(1-v)       -3*(3*v-1)
        -3*(1+v)      -4-3*(1-v)    3*(3*v-1)       -8-3*(1-v)
        4-3*(1-v)     3*(3*v-1)     -8+3*(1-v)      3*(1+v)
        -3*(3*v-1)    -8+3*(1-v)    3*(1+v)         -4-3*(1-v)];
k22=k11;

kel=E*h/24/(1-v^2)*[ k11 transpose(k21);
                     k21 k22];

me = ro*h*a*b/9*[ 4   0   2   0   1   0   2   0;
                  0   4   0   2   0   1   0   2;
                  2   0   4   0   2   0   1   0;
                  0   2   0   4   0   2   0   1;
                  1   0   2   0   4   0   2   0;
                  0   1   0   2   0   4   0   2;
                  2   0   1   0   2   0   4   0;
```

```
                         0    2    0    1    0    2    0    4];

topology=[  1    2    3    4    25   26   23   24
            3    4    5    6    27   28   25   26
            5    6    7    8    29   30   27   28
            7    8    9    10   31   32   29   30
            9    10   45   46   43   44   31   32
            11   12   47   48   45   46   9    10
            13   14   33   34   47   48   11   12
            15   16   17   18   33   34   13   14
            17   18   19   20   35   36   33   34
            19   20   21   22   37   38   35   36
            21   22   55   56   49   50   37   38
            37   38   49   50   51   52   39   40
            39   40   51   52   53   54   41   42];

kglobal=zeros(num_eq);

for e=1:1:13
    elem_top=[topology(e,:)];
    kglobal(elem_top,elem_top)=kglobal(elem_top,elem_top)+kel;

end

gpoints = [ -0.577350269189626               -0.577350269189626;
             0.577350269189626               -0.577350269189626;
             0.577350269189626                0.577350269189626;
            -0.577350269189626                0.577350269189626];

gweights = [1  1;
            1  1;
            1  1;
            1  1];


E1 = E/(1-v^2);

ff = zeros(8,1);

N = zeros(1,8);
n = sym('n',[8 2]);
ns = sym('ns',[8 2]);
nt = sym('nt',[8 2]);
ns2 = sym('ns2',[8 2]);
nt2 = sym('nt2',[8 2]);
nsnt = sym('nsnt',[8 2]);

syms s t
for i=1:length (gweights)
    for j=1:length(gweights)
        for node=1:4
            if node==1
            sc=-1;
            tc=-1;
            elseif node==2
            sc=1;
            tc=-1;
            elseif node==3
```

```
            sc=1;
            tc=1;
            else
            sc=-1;
            tc=1;
            end
            n(node,:)=[(1/4)*(1+sc*s)*(1+tc*t),...
                       (1/4)*(1+sc*s)*(1+tc*t)];
        end

        ns=diff(n,s);
        nt=diff(n,t);
        ns2=diff(n,s,2);
        nt2=diff(n,t,2);
        nsnt=diff(ns,t);

        N=[n(1,:) n(2,:) n(3,:) n(4,:)];
        Ns2=[ns(1,:) ns(2,:) ns(3,:) ns(4,:)];
        Nt2=[nt(1,:) nt(2,:) nt(3,:) nt(4,:)];
        Nsnt=[nsnt(1,:) nsnt(2,:) nsnt(3,:) nsnt(4,:)];

        B=[Ns2/a^2; Nt2/b^2; 2/(a*b)*Nsnt];

        D=[ E1 E1*v 0;...
            E1*v E1 0;...
            0 0 E/(2*(1+v))];

        %Element consistent load vector
        ff=a*b*N'*pz*gweights(i)*gweights(j);
        ff=subs(ff,s,gpoints(i));
        ff=subs(ff,t,gpoints(j));

    end
end


%Global mass matrix
Mglob=zeros(nnode*ndof);

for nel=1:13
    for i=1:8
        for j=1:8
            Mglob(topology(nel,i),topology(nel,j))= ...
            Mglob(topology(nel,i),topology(nel,j))+me(i,j);
        end
    end
end

%Lumped mass matrix using row sum method
MglobLum=zeros(nnode*ndof);
for i=1:nnode*ndof
    MglobLum(i,i)=sum(Mglob(i,:));
end

%Global force vector
Fglob=zeros(nnode*ndof,1);
for nel=1:5
    for i=[6 8]
```

```
            Fglob(topology(nel,i),1)= ...
            Fglob(topology(nel,i),1)+ff(i,1);
    end
end

for nel=8
    for i=6
            Fglob(topology(nel,i),1)= ...
            Fglob(topology(nel,i),1)+ff(i,1);
    end
end

for nel=8
    for i=6
            Fglob(topology(nel,i),1)= ...
            Fglob(topology(nel,i),1)+ff(i,1);
    end
end

for nel=[9 10 13]
    for i=[6 8]
            Fglob(topology(nel,i),1)= ...
            Fglob(topology(nel,i),1)+ff(i,1);
    end
end


%Boundary conditions (Bottom edge cantilever)
BCs=[15;16;17;18;19;20;21;22;55;56];
activeDof=setdiff(1:nnode*ndof',BCs);
Kglobactive=kglobal(activeDof,activeDof);
Mglobactive=Mglob(activeDof,activeDof);
MglobLumactive=MglobLum(activeDof,activeDof);
Fglobactive=Fglob(activeDof,1);

%Solve
d=Kglobactive\Fglobactive;
do(activeDof,1)=[d];

do = [  do
        0
        0];
%Obtaining only vertical displacement DOFs


count=1;
for i=1:28
    du(i)=do(count); %Nodal displacements
    count=count+2;
end

count=2;
for i=2:27
    dv(i)=do(count); %Nodal displacements
    count=count+2;
end

maxdu=max(du) %Maximum deflection
```

```
maxdv=max(dv)

%Reaction forces and moments at nodes
Freaction=kglobal*do-Fglob;

%Strain at nodes (eps x, eps y, gama xy)
B1=subs(B,s,-1);
B1=subs(B1,t,-1);
B1=double(B1);
strain=zeros(3,1,13);

e=0;
for i=[1:11] %Nodes [1:11]
    e=e+1;
    strain(:,:,i)=B1*do(topology(e,:));
end

B2=subs(B,s,-1);
B2=subs(B2,t,1);
B2=double(B2);

e=0;
for i=[12:16] %Nodes [12 13 14 15 16]
    e=e+1;
    strain(:,:,i)=B2*do(topology(e,:));
end

e=8;
for i=[17:21] %Nodes [17 18 19 20 21]
    e=e+1;
    strain(:,:,i)=B2*do(topology(e,:));
end

B3=subs(B,s,1);
B3=subs(B3,t,1);
B3=double(B3);

e=4;
for i=[22:24] %Nodes [22 23 24]
    e=e+1;
    strain(:,:,i)=B3*do(topology(e,:));
end

e=10;
for i=[25:27] %Nodes [25 26 27]
    e=e+1;
    strain(:,:,i)=B3*do(topology(e,:));
end

B4=subs(B,s,1);
B4=subs(B4,t,-1);
B4=double(B4);
strain(:,:,28)=B4*do(topology(11,:)); %Node 25

%Stress at nodes (sigma x, sigma y,tau xy)
stress=zeros(3,1,13);
for i=1:13
    stress(:,:,i)=D*strain(:,:,i)
```

```
end

%Normal modes and corresponding natural frequencies
[vecfreq, freq]=eig(kglobal,Mglob);
freq=diag(freq);
[freq,I1]=sort(freq, 'ascend');       %Sorted eigenvalues
vecfreq=vecfreq(I1,:);                %Sorted eigenvalues
freq=sqrt(freq);                      %UNITS : rad per sec
freqHz=freq/(2*pi);                   %UNITS : Hertz

%Constrained modes and corresponding natural frequencies
[vecfreqc,freqc]=eig(Kglobactive,Mglobactive);
freqc=diag(freqc);
[freqc,I2]=sort(freqc, 'ascend');     %Sorted eigenvalues
vecfreqc=vecfreqc(I2,:);              %Sorted eigenvalues
freqc=sqrt(freqc);                    %UNITS : rad per sec
freqcHz=freqc/(2*pi);                 %UNITS : Hertz

%Constrained modes and corresponding natural frequencies
%with lumped mass matrix
[vecfreqcLum,freqcLum]=eig(Kglobactive,MglobLumactive);
freqcLum=diag(freqcLum);
[freqcLum,I3]=sort(freqcLum, 'ascend'); %Sorted eigenvalues
vecfreqcLum=vecfreqcLum(I3,:);          %Sorted eigenvalues
freqcLum=sqrt(freqcLum);                %UNITS : rad per sec
freqcLumHz=freqcLum/(2*pi);             % UNITS : Hertz

%Rayleigh damping model
cglob1=0.001*Kglobactive+0.02*Mglobactive;
em=Mglobactive;
ka=Kglobactive;
fe=Fglobactive;
ce=cglob1;
[mg1,junk1]=size(Mglobactive);

%INTEGRATION USING NEWMARK METHOD

% %Central difference formula
% gama=0.5;
% beta=0;

% %trapezoidal rule
% gama=0.5;
% beta=0.25;

% Damped Newmark Method
gama=0.6;
beta=0.3025;

% % Linear Acceleration
% gama=0.5;
% beta=1/6;

% % Fox-Goodwin
% gama=0.5;
% beta=1/12;

ksi=(0.001/max(freqc)+0.02/max(freqc))/2;
```

```
om=(ksi*(gama-0.5)+sqrt(gama/2-beta+ksi^2*(gama-0.5)^2))/(gama/2-beta);
deltatcrit=om/max(freqc);
deltat=1e-4;
delta=deltat;

dold=zeros(mg1,1);
vold=zeros(mg1,1);
say=0;
say=say+1;

rubbish=inv(em+gama*delta*ce+beta*delta*delta*ka);
aold=em\(fe-ce*vold-ka*dold);
dtildanew=dold+delta*vold+0.5*delta*delta*(1-2*beta)*aold;
vtildanew=vold+(1-gama)*delta*aold;
anew=rubbish*(fe-ce*vtildanew-ka*dtildanew);
dnew=dtildanew+beta*delta*delta*anew;
vnew=vtildanew+gama*delta*anew;
aold=anew;
vold=vnew;
dold=dnew;
dtildaold=dtildanew;
vtildaold=vtildanew;
time1(say)=0;
RightTopCornerdisplacement(say)=0;
RightTopCornerdisplacement2(say)=0;
RightTopCornerdisplacement3(say)=0;
LeftTopCornerdisplacement(say)=0;
for i=1:1000
    say=say+1;
    time1(say)=time1(say-1)+delta;
    dtildanew=dold+delta*vold+0.5*delta*delta*(1-2*beta)*aold;
    vtildanew=vold+(1-gama)*delta*aold;
    anew=rubbish*(fe-ce*vtildanew-ka*dtildanew);
    dnew=dtildanew+beta*delta*delta*anew;
    vnew=vtildanew+gama*delta*anew;
    aold=anew;
    vold=vnew;
    dold=dnew;
    dtildaold=dtildanew;
    vtildaold=vtildanew;
    df(activeDof)=dold;
    RightTopCornerdisplacement(say)=df(54);
    RightTopCornerdisplacement2(say)=df(52);
    RightTopCornerdisplacement3(say)=df(50);
    LeftTopCornerdisplacement(say)=df(24);
end

toc

figure, plot(time1, RightTopCornerdisplacement)
xlabel('time (sec)')
ylabel('Right Top Cornerdisplacement (mm)' )
% figure, plot(time1, RightTopCornerdisplacement2)
% xlabel('time (sec)')
% ylabel('Right Top Cornerdisplacement 2 (mm)' )
% figure, plot(time1, RightTopCornerdisplacement3)
% xlabel('time (sec)')
% ylabel('Right Top Cornerdisplacement 3 (mm)' )
```

```
% figure, plot(time1, LeftTopCornerdisplacement)
% xlabel('time (sec)')
% ylabel('Left Top Cornerdisplacement (mm)' )
```

Listing 2: Matlab code for question 2

```
%Final Q2

clear all; close all; clc; format short;

% width=5; left=1.5; right=2.5; %dimensions (m)
%
% seed_x=6;
% seed_y=3;

num_node=28;                    %number of total nodes
num_elem=13;                    %number of total elements

element_type=4;                 %number of nodes on each element
ndof=1;                         %degrees-of-freedom per node

num_eq=num_node*ndof;           %number of equations

z=[1 7 13 19];
k=100;

%%--------------Coordinates--------------%%

x_coord=[0  10  20  30  40  50  0   10  20  31  40  50  40  50  40  50  40  50
    60  70  80  60  70  80  70  80  70  80];

y_coord=[0  0   0   0   0   0   10  10  10  10  10  10  -10 -10 -20 -20 -30 -30
    -30 -30 -30 -20 -20 -20 -10 -10 0   0];

Coords=[x_coord', y_coord'];

%%-------topology matrix-------%%

topology=[  1   2   8   7
            2   3   9   8
            3   4   10  9
            4   5   11  10
            5   6   12  11
            13  14  6   5
            15  16  14  13
            17  18  16  15
            18  19  22  16
            19  20  23  22
            20  21  24  23
            23  24  26  25
            25  26  28  27];

%%----Gauss Quadrature and Shape function ----%%

num_gauss_point=2;

if num_gauss_point==1
    gp=0;
    w=2;
```

```
    elseif num_gauss_point==2
        %if four gauss points are used (two in each direction)
    gp=[-0.57735027, 0.57735027]; %[eta psi]
    w= [1,1];
end

eta=gp(1);
psi=gp(2);

N=(1/4)*[(1-psi)*(1-eta) (1+psi)*(1-eta) (1+psi)*(1+eta) (1-psi)*(1+eta)];
%shape functions

dN=(1/4)*[eta-1 1-eta 1+eta -eta-1; % Gradient
          psi-1 -psi-1 1+psi 1-psi];

%%----Obtaining Element Stiffness Matrix----%%

ke=zeros(element_type,element_type);
kglobal=zeros(num_eq);

for e=1:1:num_elem

    elem_top=[topology(e,:)];
    J=dN*Coords(elem_top',:);                  % compute Jacobian matrix
    detJ=det(J);                               % Jacobian
    B=J\dN;                                    % compute the B matrix
    D=k*eye(2);
    for i=1:num_gauss_point
        for j=1:num_gauss_point
            ke=ke+w(i)*w(j)*B'*D*B*detJ;       % element conductance matrix
            kglobal(elem_top,elem_top)=kglobal(elem_top,elem_top)+ke;
        end
    end
end

% At node 1,2,3,4,5,6,7,12,13 and 18, the temperature is to be fixed
fix_temperature_nodes=[1 7 21 24 26 28];

% The temperature is to be fixed at a particular value ( C )
fix_temperature=[20 20 50 50 50 50];

% Nodal force(heat source) vector
nodal_force{1}=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0]';

% 4) LINEAR 2-D HEAT CONDUCTION ANALYSIS
% 4.1) Matching vector for obtaining global matrix

% All DoF
all_DoF_number=(1:num_node)';

% Restrained DoF
restrained_DoF_number_for_matching=fix_temperature_nodes';

% Unrestrained DoF
unrestrained_DoF_number_for_matching=all_DoF_number;
unrestrained_DoF_number_for_matching(restrained_DoF_number_for_matching)...
    =[];

```

```
matching_vector_first=[unrestrained_DoF_number_for_matching;
restrained_DoF_number_for_matching];        % Matching vector for reorganize

for i=1:length(matching_vector_first)
    matching_vector_end(i,1)=find(matching_vector_first==i);
    % Matching vector for orijinal position
end

% 4.2) Reorganize nodal force and stiffness matrix
nodal_force_reorganize=nodal_force{1}(matching_vector_first);

stiffness_of_system_reorganize=...
    kglobal(matching_vector_first,matching_vector_first);

% 4.3) Linear solution with finite element method
% a) Input

% a1) Forces
F_A=nodal_force_reorganize(1:length(unrestrained_DoF_number_for_matching));

% a2) Stiffness matrix partition
K_AA=stiffness_of_system_reorganize...
    ([1:length(unrestrained_DoF_number_for_matching)],...
    [1:length(unrestrained_DoF_number_for_matching)]);

K_AB=stiffness_of_system_reorganize...
    ([1:length(unrestrained_DoF_number_for_matching)],...
    [length(unrestrained_DoF_number_for_matching)+1:end]);

K_BA=stiffness_of_system_reorganize...
    ([length(unrestrained_DoF_number_for_matching)+1:end],...
    [1:length(unrestrained_DoF_number_for_matching)]);

K_BB=stiffness_of_system_reorganize...
    ([length(unrestrained_DoF_number_for_matching)+1:end],...
    [length(unrestrained_DoF_number_for_matching)+1:end]);

% a3) Fix tempreture ( F )
D_B=fix_temperature';

% b) Solution
% Specify nodal tempreture ( C )
D_A=K_AA\(F_A-K_AB*D_B);
% Specify nodal in-bound heat fluxes(Watts)
F_B_heat_fluxes=K_BA*D_A+K_BB*D_B;

% 4.4) Orijinal position tempreture and heat fluxes

temperature_of_system_reorganized=[D_A;D_B];

temperature_of_system_orijinal_position...
    =temperature_of_system_reorganized(matching_vector_end)
heat_fluxes_of_system_orijinal_position_{1}...
    =nodal_force{1};

heat_fluxes_of_system_orijinal_position_{1}...
(restrained_DoF_number_for_matching)...
    =heat_fluxes_of_system_orijinal_position_{1}...
```

```
    (restrained_DoF_number_for_matching)+F_B_heat_fluxes;

heat_fluxes_of_system_orijinal_position...
    =heat_fluxes_of_system_orijinal_position_{1}
```