INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

Department of Aerospace Engineering

AE675 – Introduction to Finite Element Method

Project Report On

"Temperature Distribution in a Heated Steel Plate Using FEM"

Submitted By:

Shreya Anand 241010405

Sujeet Kumar 241010407

Vivek Mourya 241010074

Submitted To:

Dr. Pritam Chakraborty

Table of Contents

Chapter -1

Introduction

This report presents the finite element analysis of a two-dimensional steady-state heat conduction problem in a square steel plate with 1 m × 1 m dimensions. The plate is subjected to a localized heat source of 1 kW at its center. The analysis is done using linear triangular elements with varying levels of mesh refinement to examine the effect of discretization on the accuracy of the temperature distribution.

Two different cases of boundary conditions are considered:

Case 1: Three sides of the plate are insulated, while the fourth is maintained at a constant temperature of 25°C.

Case 2: Three sides of the plate are insulated, and the fourth side is subjected to convective heat transfer with a fluid at 25°C.

The temperature distribution is calculated for different mesh configurations, and contour plots, along with temperature profiles, are used to compare the solutions' accuracy and behavior.

Chapter-2

Problem Description

Boundary Conditions:-

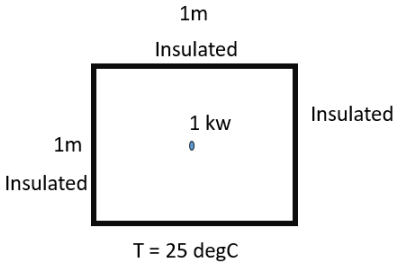

Fig 1

Case 1: One Side at Constant Temperature

- **Left side**: Insulated
- **Right side**: Insulated
- **Top side**: Insulated
- **Bottom side**: Maintained at constant temperature → T=25∘C
- **Point heat source at middle =** 1K W

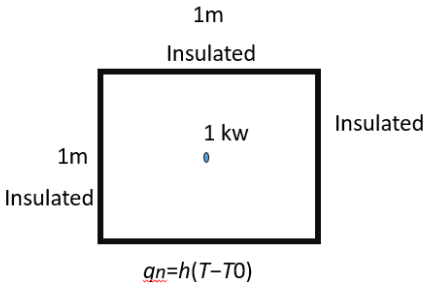

Fig 2

Case 2: One Side with Convective Boundary

- **Left side**: Insulated
- **Right side**: Insulated
- **Top side**: Insulated
- **Point heat source at middle =** 1K W
- **Boundary 4 (Bottom):** Subjected to convective heat transfer: qn=h(T−T0) where h=10 W/(m2.K) and T0=25∘C

Chapter -3
Methodology and Algorithm

Problem Setup

1. Plate dimensions: 1 m × 1 m × 0.01 m
2. Heat source: 1 kW = 1000 W in the center
3. Assumptions:
4. 2D steady-state heat conduction
5. Thickness is small → Out-of-plane conduction is negligible
6. Internal heat generation is **lumped at the center node**
7. Use linear triangular elements (CST)

The governing differential equation for 2D heat conduction problem with internal heat generation is

$$\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)-\dot{q}=0$$

Applying Galerkin's Weighted Residual method,

$$\int_{\Omega} N_i\left[\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial x}\right)+\frac{\partial}{\partial y}\left(k\frac{\partial T}{\partial y}\right)+\dot{q}\right]d\Omega+\int_{\Gamma_q} N_i'\left[k\frac{\partial T}{\partial n}+\bar{q}\right]d\Gamma\ =0$$

Simplifying by using the Green lemma equation and discretizing the domain, we get,

$$-\int_{\Omega_e}\left[\frac{\partial N_i}{\partial x}k\frac{\partial T}{\partial x}\ +\frac{\partial N_i}{\partial y}k\frac{\partial T}{\partial y}\right]\ d\Omega\ \ +\int_{\Omega_e} N_i\dot{q}\,d\Omega-\int_{\Gamma_{qe}} N_i\bar{q}\,d\Gamma\ =0$$

After substituting,

$$T^e=\ T_j^e N_j^e(x,y)$$

$$-\int_{\Omega_e}\left[\frac{\partial N_i^e}{\partial x}k\frac{\partial N_j^e}{\partial x}\ +\frac{\partial N_i^e}{\partial y}k\frac{\partial N_j^e}{\partial y}\right]T_j^e\ d\Omega\ \ +\int_{\Omega_e} N_i^e\dot{q}\,d\Omega-\int_{\Gamma_{qe}} N_i^e\bar{q}\,d\Gamma\ =0$$

This particular equation can be broken into

$$K_{ij}^e\,T_j^e=\ f_i^e$$

Where

$$K_{ij}^e=\ \int_{\Omega_e}\left[\frac{\partial N_i^e}{\partial x}k\frac{\partial N_j^e}{\partial x}\ +\frac{\partial N_i^e}{\partial y}k\frac{\partial N_j^e}{\partial y}\right]\ dxdy$$

$$f_i^e=\ \int_{\Gamma_{qe}} N_i^e\bar{q}\,d\Gamma\ -\int_{\Omega_e} N_i^e\dot{q}\,d\mathrm{xdy}$$

Now, In the problem set up it is told that we need to use linear triangular element,

For linear triangular element:

1. Shape functions N_i are linear.
2. Gradient of shape functions are constant
3. The strain-displacement matrix B is constant.

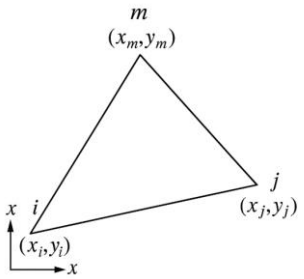

Fig 3

For linear Triangular element

$$T^e = N_1^e T_1^e + N_2^e T_2^e + N_3^e T_3^e$$

Shape Functions and Their Derivatives:

The shape function for node i in terms of area coordinates is given by:
Ni(x,y) = ai + bi * x + ci * y

where the partial derivatives w.r.t x and y are:
bi = ∂Ni/∂x
ci = ∂Ni/∂y

These bi and ci terms are computed using the geometry of the triangle with nodes 1, 2, and 3:
bi = (1 / (2A)) * (yj − yk)
ci = (1 / (2A)) * (xk − xj)

Here, (i,j,k) represents a cyclic permutation of (1,2,3), and A is the area of the triangular element.

The vectors b and c used in the code are scaled versions of the shape function gradients:
b = [b1 b2 b3]^T · 2A
c = [c1 c2 c3]^T · 2A

Stiffness Matrix Derivation:

The element stiffness matrix ke is derived using the following integral:
ke = ∫A k (∇ N)^T (∇ N) dA

where k is a material property (assumed constant here), and ∇ N is the gradient matrix of the shape functions.

The gradient matrix ∇ N is given by:
∇ N = (1 / (2A)) * [
  [b1 c1]
  [b2 c2]
  [b3 c3]
]

Now, (∇ N)^T (∇ N):
(∇ N)^T = (1 / (2A)) * [
  [b1 b2 b3]
  [c1 c2 c3]
]

(∇ N)^T (∇ N) = (1 / (2A))^2 * [
  [b1^2 + b2^2 + b3^2, b1c1 + b2c2 + b3c3]
  [b1c1 + b2c2 + b3c3, c1^2 + c2^2 + c3^2]
]

This matrix multiplication can be written as:
(∇ N)^T (∇ N) = (1 / (4A^2)) * (b b^T + c c^T)

Substituting into the stiffness matrix integral:
ke = k * ∫A (1 / (4A^2)) * (b b^T + c c^T) dA

Since k, b, and c are constants over the element,
ke = k * (1 / (4A^2)) * (b b^T + c c^T) * ∫A dA

As ∫A dA = A,
ke = k * (1 / (4A^2)) * (b b^T + c c^T) * A

Therefore,
ke = (k / (4A)) * (b b^T + c c^T)

Calculation of Force vector:

Case 1: One Side at Constant Temperature

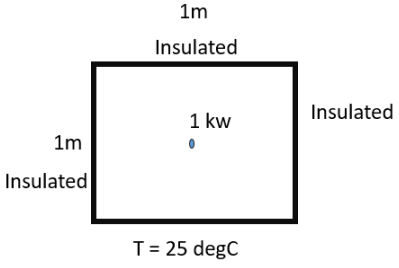

Fig 4

For Dirichlet conditions, no elemental force vector is computed. Except for 2 element problem, as no node will be there. In that case, we will formulate the source vector in the following format:-
We consider a body force (source term) applied to an element in the domain:

$$f^e = \int_{Ae} N(x, y) \cdot q(x, y)\, dA$$

Where:
- N(x, y) = [N1 N2 N3]^T are the shape functions of a linear triangle.
- q(x, y) is the source term applied to the element.

Assuming a point source represented by the Dirac delta function:

$$q(x, y) = Q\, \delta(x - xc, y - yc)$$

Substituting into the integral:

$$f^e = \int_{Ae} N(x, y) \cdot Q\, \delta(x - xc, y - yc)\, dA = Q \cdot N(xc, yc)$$

Input Parameters

1. Q = 1000
2. (xc, yc) is a point located inside Element 2.
3. N(x, y) = [N1(x, y) N2(x, y) N3(x, y)]^T

Element Force Vector for Element 2

The force vector for Element 2 due to the point source becomes:

$$f^e = 1000 \cdot [N1(xc, yc)\ N2(xc, yc)\ N3(xc, yc)]^T$$

This vector is then assembled into the global force vector by mapping the local node indices of Element 2 to the corresponding global indices.

Point Heat Source
The 1 kW point heat source is applied to the node closest to the center:

$$F(\text{center node}) = F(\text{center node}) + Q$$

Summary

• No elemental force vector is calculated for Dirichlet BC except 2 element for which $q = 1000\ \delta(x - xc, y - yc)$ W/m2 where δ is the Dirac-delta function
• The system is modified directly to apply T = 25°C on the bottom edge
• The 1 kW source is added at centre node

Case 2 : One Side with Convective Boundary

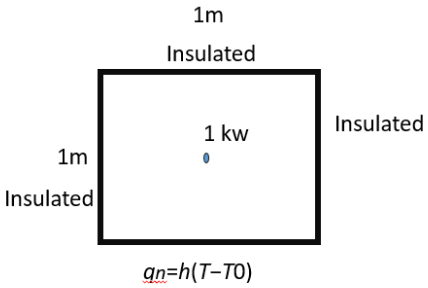

Fig 5

For each boundary edge with convection (two nodes per edge), the force vector is computed as:

f^(e) = (h * T_inf * L_e / 2) * [1; 1]

Where:
• h: Convection heat transfer coefficient
• T_inf: Ambient temperature
• L_e: Length of the boundary edge
This comes from integrating the shape functions over the 1D boundary element:

$f_i = \int h\, T_{inf}\, N_i\, dx = h\, T_{inf} * (L_e / 2)$, for each node on the edge

Point Heat Source

The internal point heat source (1 kW) is applied directly to the nearest node to the center of the domain. This is handled globally and not through elemental force vectors.

F (center node) = F (center node) + Q

Summary

- Convective BC contributes to the force vector only on boundary edges
- Point source is directly added to the global force vector at the center node
- No body force integration is needed since there is no volumetric source

Pseudocode for Matlab Program:

Case 1: One Side at Constant Temperature

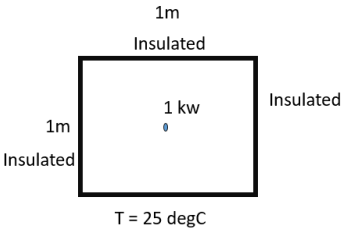

Fig 6

1. Define parameters:
   - k = 50 (thermal conductivity)
   - Q = 1000 (heat source)
   - T_fixed = 25 (boundary temperature)
   - Choose Nx and Ny as per your problem (elements in x and y directions)

2. Generate structured grid:
   - Create nodes coordinates (x_grid, y_grid)
   - n_nodes = total number of nodes
   - Create triangular elements by connecting nodes

3. Plot the mesh:
   - Visualize with node markers
   - Display mesh statistics (n_nodes, n_elements)

4. Initialize global matrices:
   - K = zeros(n_nodes, n_nodes) (stiffness matrix)
   - F = zeros(n_nodes, 1) (force vector)
   - T = zeros (n_nodes, 1) (temperature vector)

5. Assemble global stiffness matrix:
   for e = 1:n_elements do
      1. Get element node indices and coordinates
      2. Calculate element area A
      3. Compute shape function coefficients b and c
      4. Calculate element stiffness matrix ke
      5. Add ke to global K at appropriate indices
   end

6. Apply heat source:
   - Find center node closest to (0.5, 0.5)
   - Set F (center_node) = Q
   - For 2 element solve using dirac delta as given above

7. Apply boundary conditions:
   - Find fixed_nodes where y = 0
   - Set T (fixed_nodes) = T_fixed
   - Identify free_nodes (all others)

8. Modify system for boundary conditions:
   - F_mod = F (free_nodes) - K (free_nodes, fixed_nodes)*T (fixed_nodes)
   - K_mod = K(free_nodes, free_nodes)

9. Solve the system:
   - T (free_nodes) = K_mod \ F_mod

10. Display and visualize results:
    - Print nodal temperatures
    - Plot temperature contour
    - Plot temperature along centerline (y = 0.5)

Case 2 : One Side with Convective Boundary

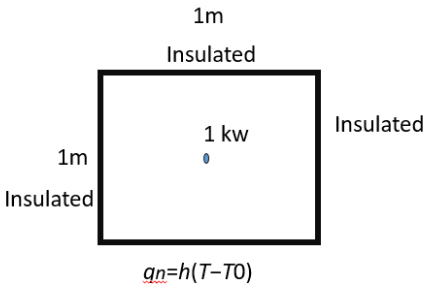

Fig 7

1. Define parameters:
   - k = 50 (thermal conductivity)
   - Q = 1000 (heat source)
   - T_inf = 25 (ambient temperature)
   - h = 10 (convection coefficient)
   - Choose Nx and Ny as per your problem (elements in x and y directions)

2. Generate structured grid:
   - Create nodes coordinates (x_grid, y_grid)
   - n_nodes = total number of nodes
   - Create triangular elements by connecting nodes (two triangles per square)

3. Plot the mesh:
- Visualize with node markers
- Display mesh statistics (n_nodes, n_elements)

4. Initialize global matrices:
- K = zeros (n_nodes, n_nodes) (stiffness matrix)
- F = zeros (n_nodes, 1) (force vector)

5. Assemble global stiffness matrix (conduction terms):
For e = 1:n_elements do
1. Get element node indices and coordinates
2. Calculate element area A
3. Compute shape function coefficients b and c
4. Calculate element stiffness matrix ke
5. Add ke to global K at appropriate indices
End

6. Apply convection boundary conditions:
- Identify bottom nodes where y = 0
- Find bottom edges (connected node pairs along boundary)
For each boundary edge do:
1. Calculate edge length L_edge
2. Compute convection stiffness matrix ke_conv = h*L_edge/6*[2 1; 1 2]
3. Compute convection force vector fe_conv = h*T_inf*L_edge/2*[1; 1]
4. Add ke_conv to global K
5. Add fe_conv to global F
End

7. Apply point heat source:
- Find center node closest to (0.5, 0.5)
- Add Q to F (center_node)

8. Solve the system:
- T = K \ F (solve for all nodes)

9. Display and visualize results:
- Print nodal temperatures
- Plot temperature contour with title "Case 2: Temp Distribution (~100 Elements + Convection)"
- Plot temperature along centerline (y = 0.5)

Chapter-3

Results and Discussion

3.1 Results

Case 1: One Side at Constant Temperature

When three sides of the plate are insulated and the 4th side is maintained at 25°C. Solve

(a) using 2 elements and $q = 1000\ \delta(x - x_c, y - y_c\ )$ W/m², where x_c and y_c are the coordinates of the center of the plate, and δ is the Dirac-delta function;

(b) using 4 elements intersecting at the center.

(c) using ~10 elements with heat source at the intersection of elements.

(d) using ~100 elements with heat source at the intersection of elements.

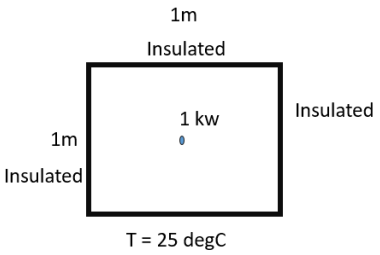

Fig 8

Matlab is used for finding simulation result of the problem as directed. Code is written based on pseudocode provided in chapter 2.

(a) For 2 elements

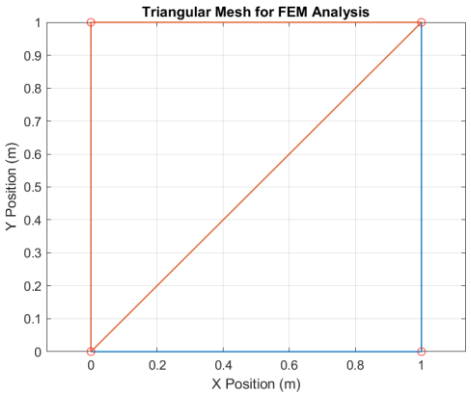

Fig 9

Mesh details:
Number of nodes: 4
Number of elements: 2
Nodal Temperatures:
Node 1 (0.0, 0.0): T = 25.00°C
Node 2 (1.0, 0.0): T = 25.00°C
Node 3 (1.0, 1.0): T = 38.33°C

Node 4 (0.0, 1.0): T = 31.67°C
Contour plot

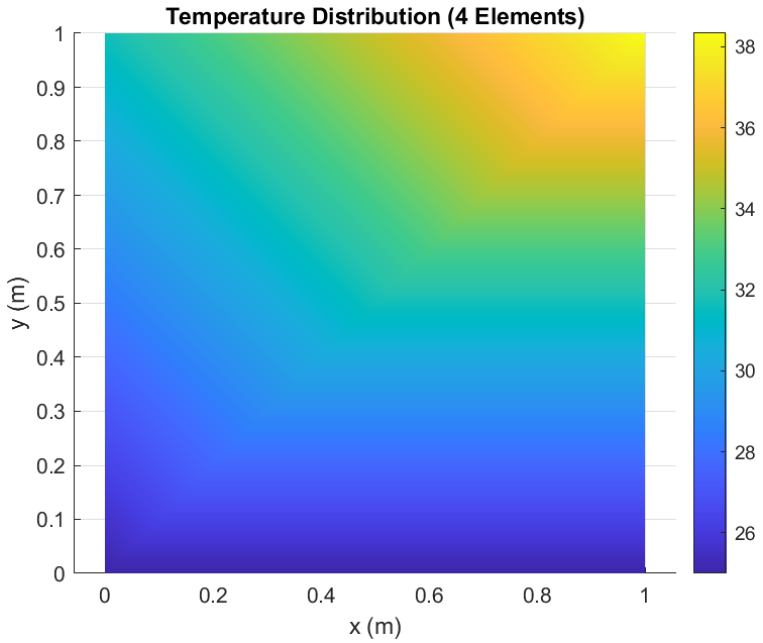

Fig 10

Temperature variation along a horizontal line passing through the center.

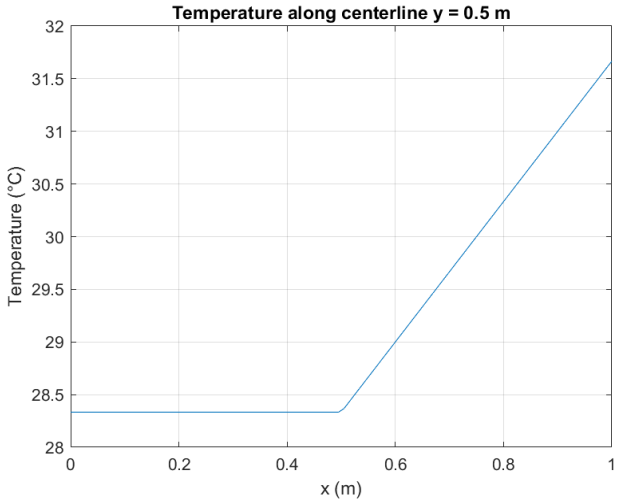

Fig 11

(b) 4 elements

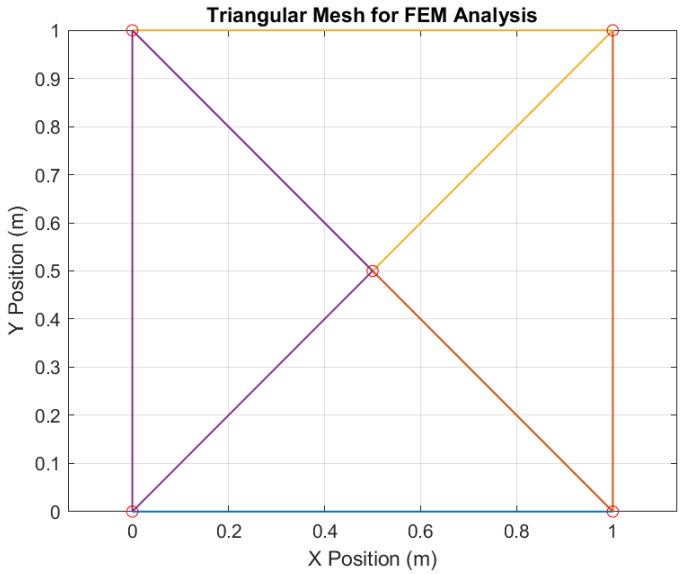

Fig 12

Mesh details:
Number of nodes: 5
Number of elements: 4
Nodal Temperatures:
Node 1 (0.0, 0.0): T = 25.00°C
Node 2 (1.0, 0.0): T = 25.00°C
Node 3 (1.0, 1.0): T = 35.00°C
Node 4 (0.0, 1.0): T = 35.00°C
Node 5 (0.5, 0.5): T = 35.00°C

Contour plot:

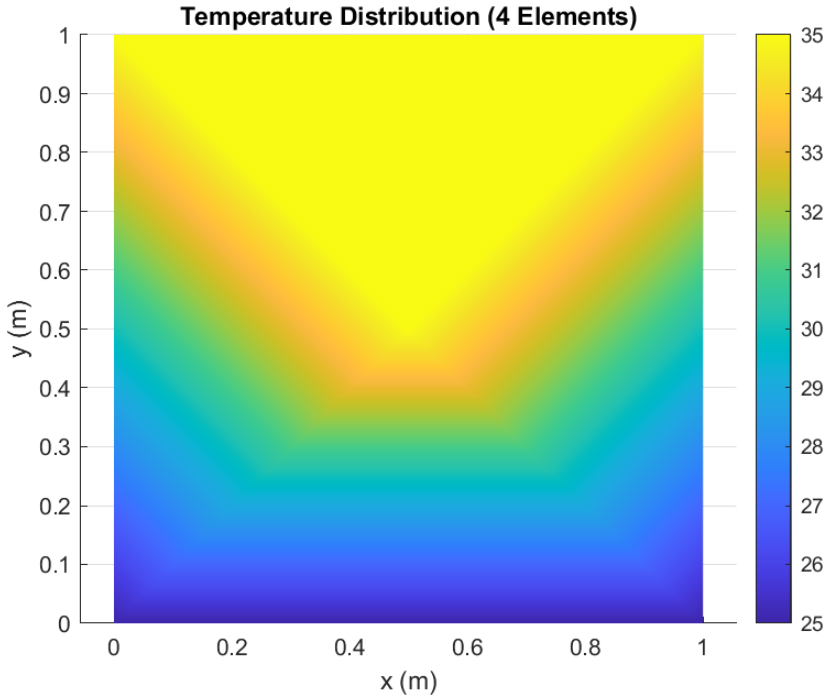

Fig 13

Temperature variation along a horizontal line passing through the center:

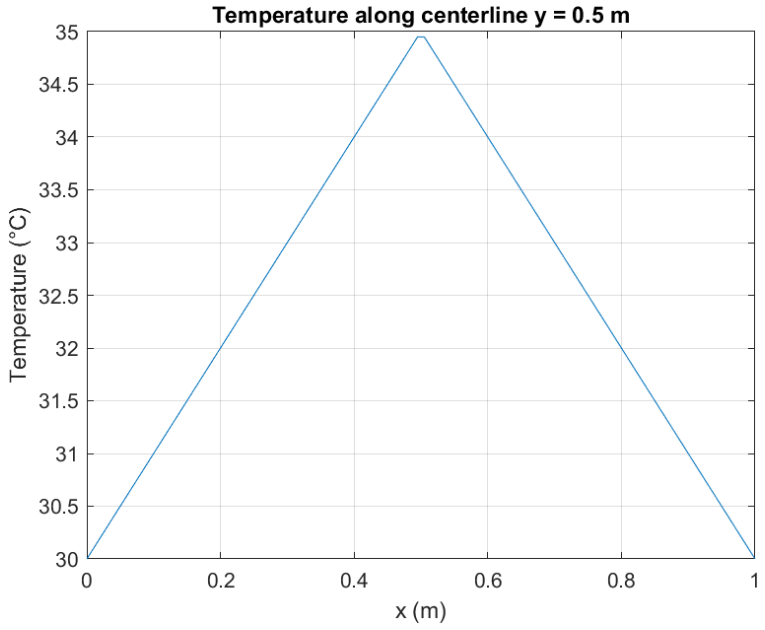

Fig 14

(c) 8 elements (near around 10 elements)

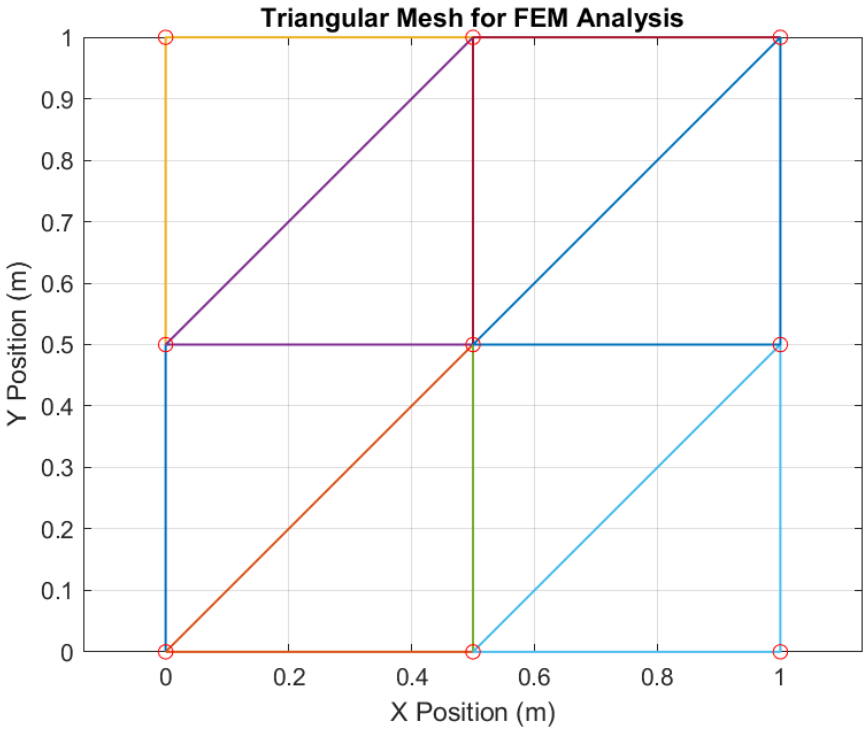

Fig 15

Mesh details:
Number of nodes: 9
Number of elements: 8
Nodal Temperatures:
Node 1 (0.0, 0.0): T = 25.00°C
Node 2 (0.0, 0.5): T = 33.24°C
Node 3 (0.0, 1.0): T = 34.41°C
Node 4 (0.5, 0.0): T = 25.00°C
Node 5 (0.5, 0.5): T = 36.76°C
Node 6 (0.5, 1.0): T = 35.59°C
Node 7 (1.0, 0.0): T = 25.00°C
Node 8 (1.0, 0.5): T = 33.24°C
Node 9 (1.0, 1.0): T = 34.41°C

Contour plot:

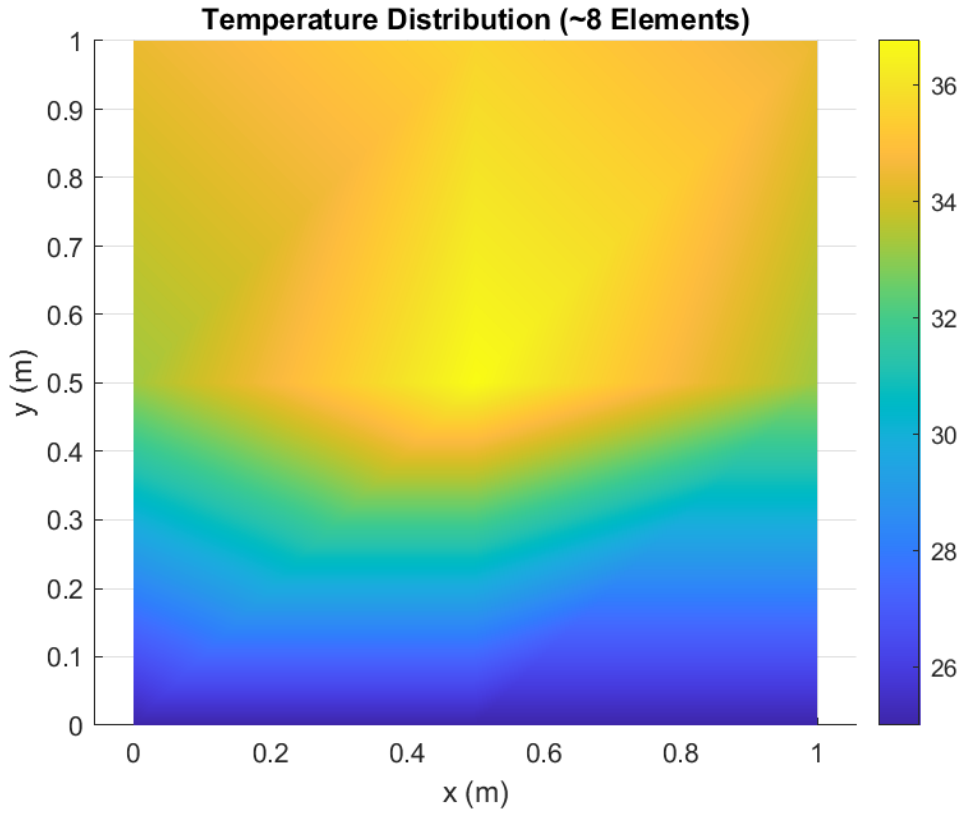

Fig 16

Temperature variation along a horizontal line passing through the center:

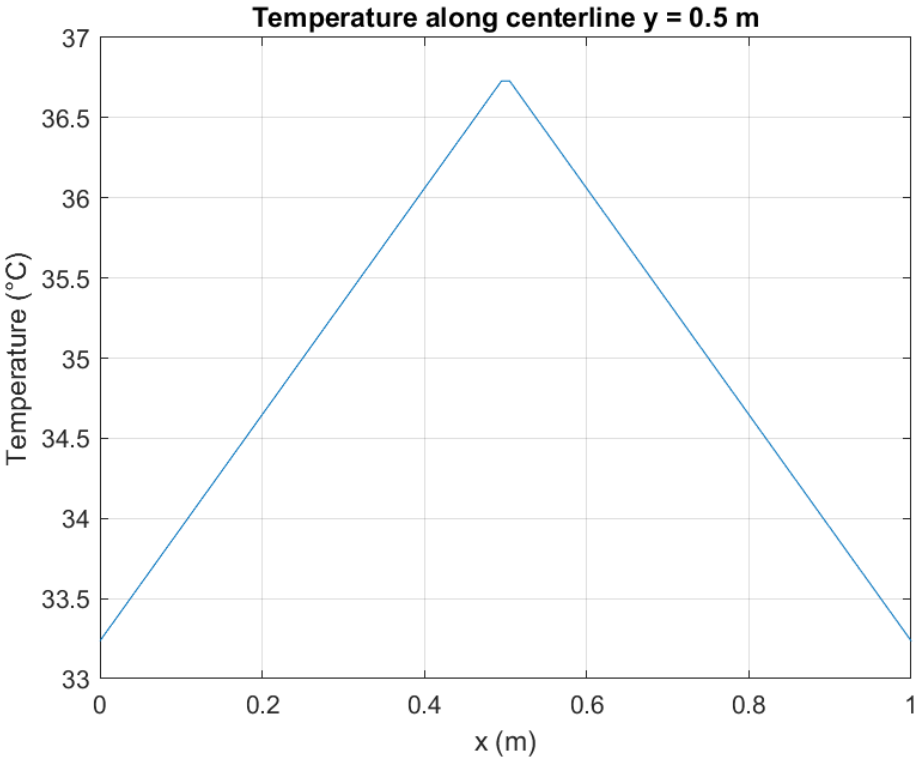

Fig 17

(d) 200 elements (to make mesh smoother for better results)

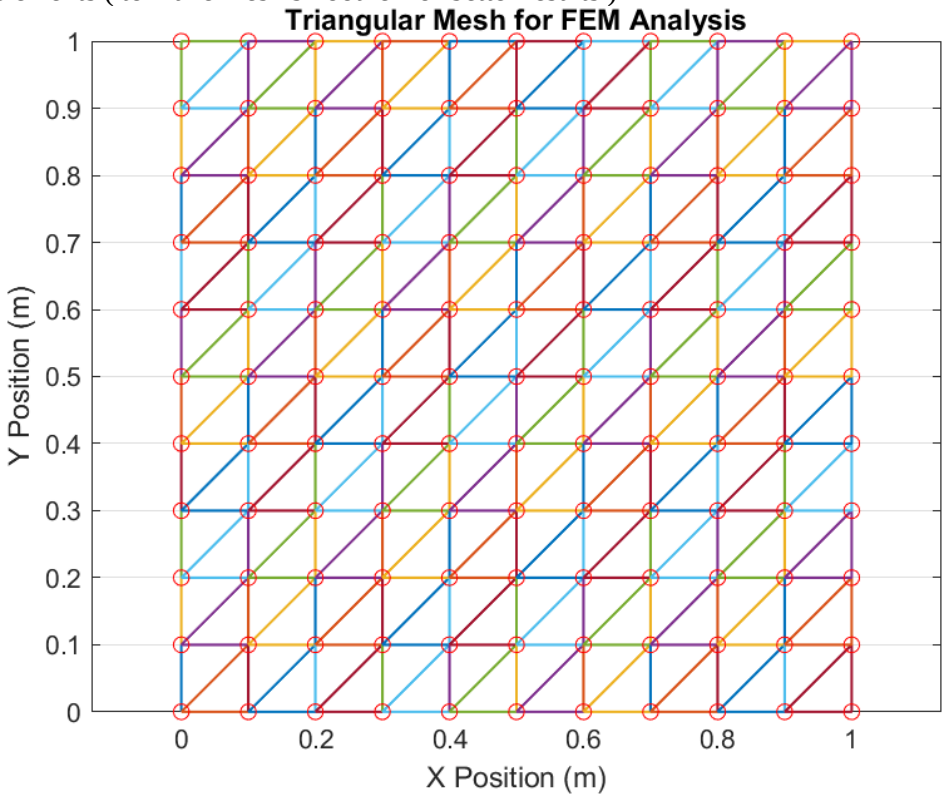

Fig 18

Mesh details:
Number of nodes: 121
Number of elements: 200
There are too many nodes, so the MATLAB code shows nodal temperatures.

Contour Plot:

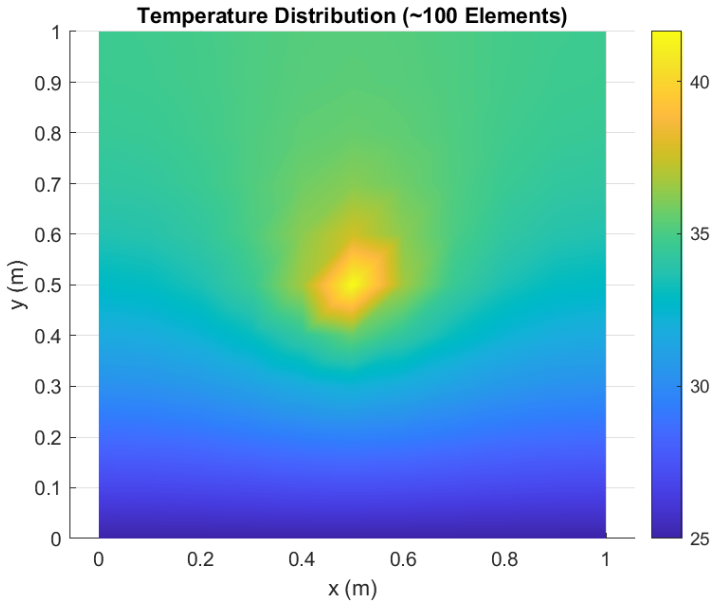

Fig 19

Temperature variation along a horizontal line passing through the center:

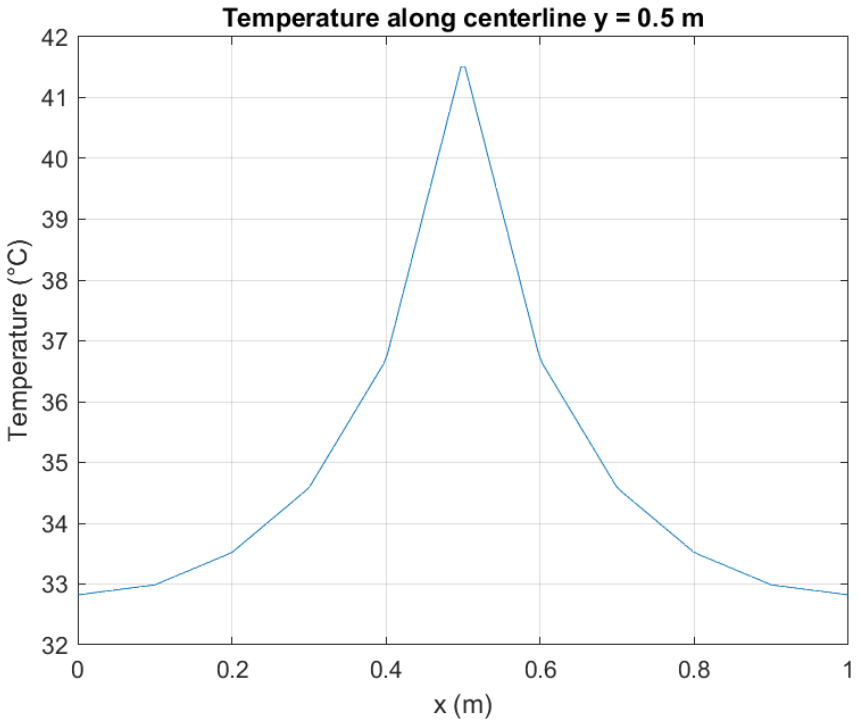

Fig 20

Case 2 : One Side with Convective Boundary:

When three sides of the plate are insulated and the 4th side is subjected to convective heat transfer such that $q_n = h(T - T_0)$ where T_0 = 25°C and h = 10 W/(m²-K). Solve using ~100 elements with heat source at the intersection of elements. Compare the solution to case 1(d).

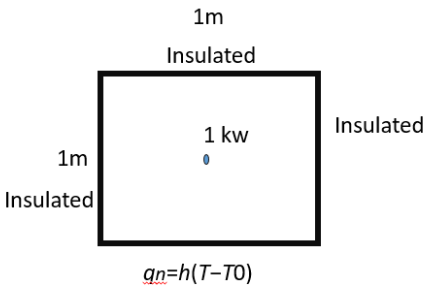

Fig 21

As this problem is needed to be compared with (d) part of case 1. So we need to take same no of elements as above .

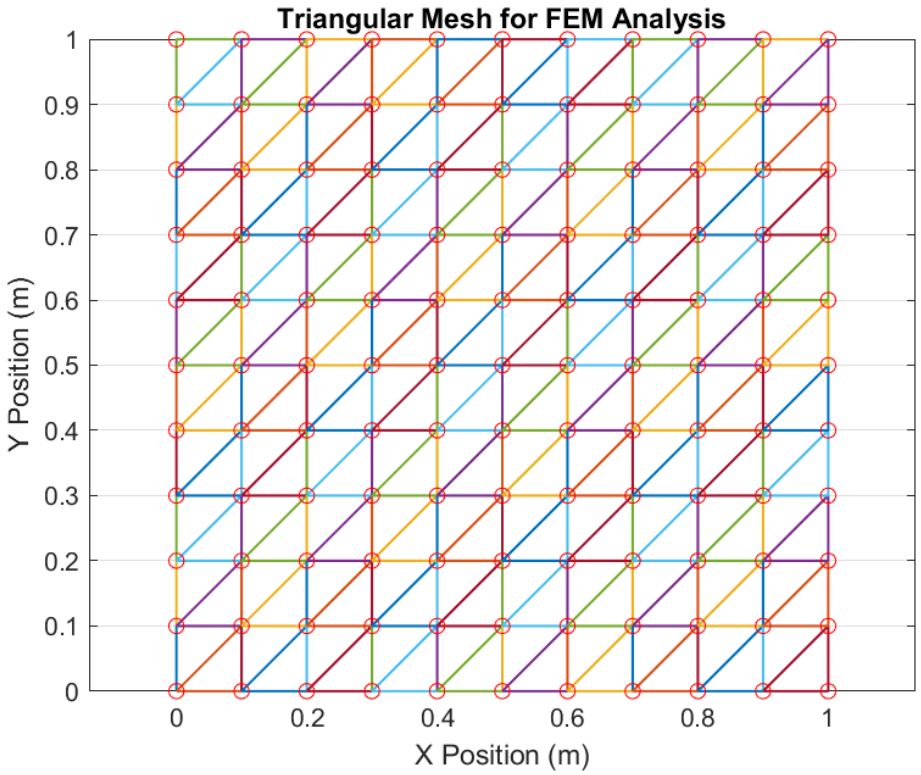

Fig 22

Mesh details:
Number of nodes: 121
Number of elements: 200

Contour Plot:

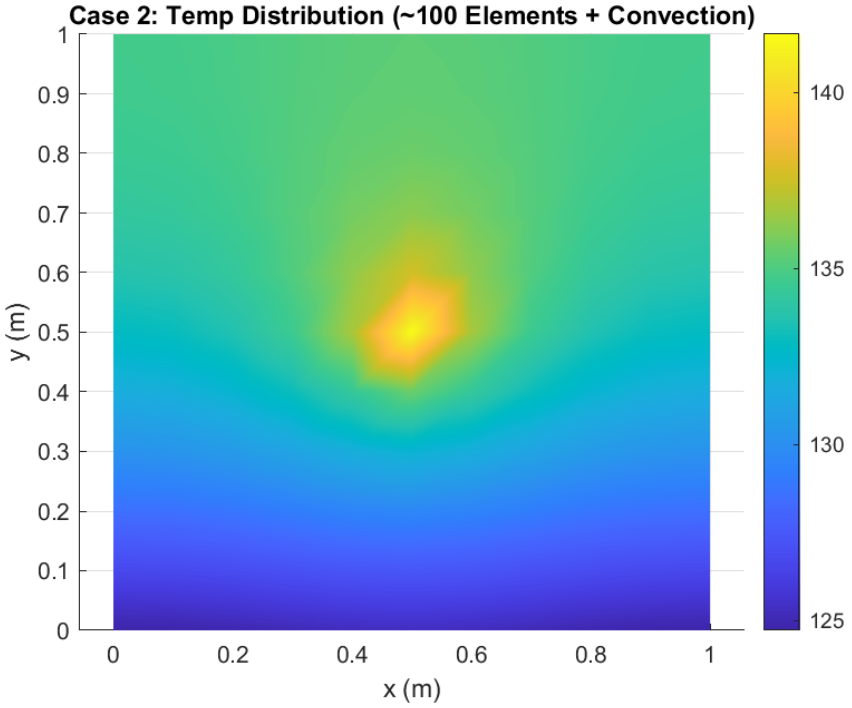

Fig 23

Temperature Distribution Along Horizonyal line :

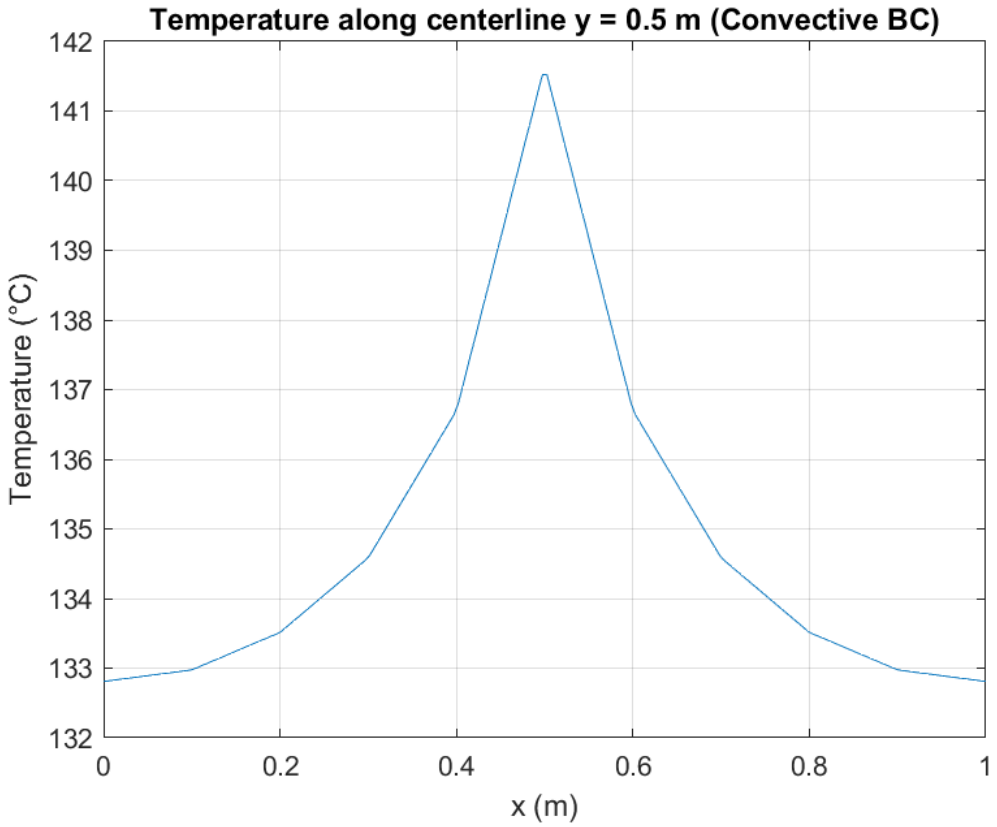

Fig 24

Discussion of results:-

1) As we can see in Case 1 , an increase no of elements is improving temperature contour distribution, and around 200 elements, we can see the maximum temperature at the point centre where the heat source is present(x=0.5, y = 0.5).
2) Now comparing case1 and case 2
    - **Temperature Levels:** The most significant difference is the drastically higher temperature levels in Case 2 compared to Case 1.This because of boundary condition.
    - The fixed temperature boundary in Case 1 effectively removes heat, preventing a large temperature buildup. The convective boundary in Case 2 allows for heat removal, but if the rate of heat input (1 kW source) exceeds the rate of heat removal by convection, and the other boundaries are insulated, the overall temperature will rise to a higher steady-state.
    - **Temmperature Distribution Shape:** While the overall temperature levels differ significantly, the general shape of the temperature distribution (a peak at the heat source location and decreasing temperatures away from it) is similar in both cases.

    - This indicates that way heat diffuses from the source within the insulated boundaries is comparable, but the absolute temperature scale is shifted due to the different bottom boundary conditions.

To compare we can calculate the normalized temperature distribution centre line
Normalized Temperature= $(T-25)/(TMAX-25)$

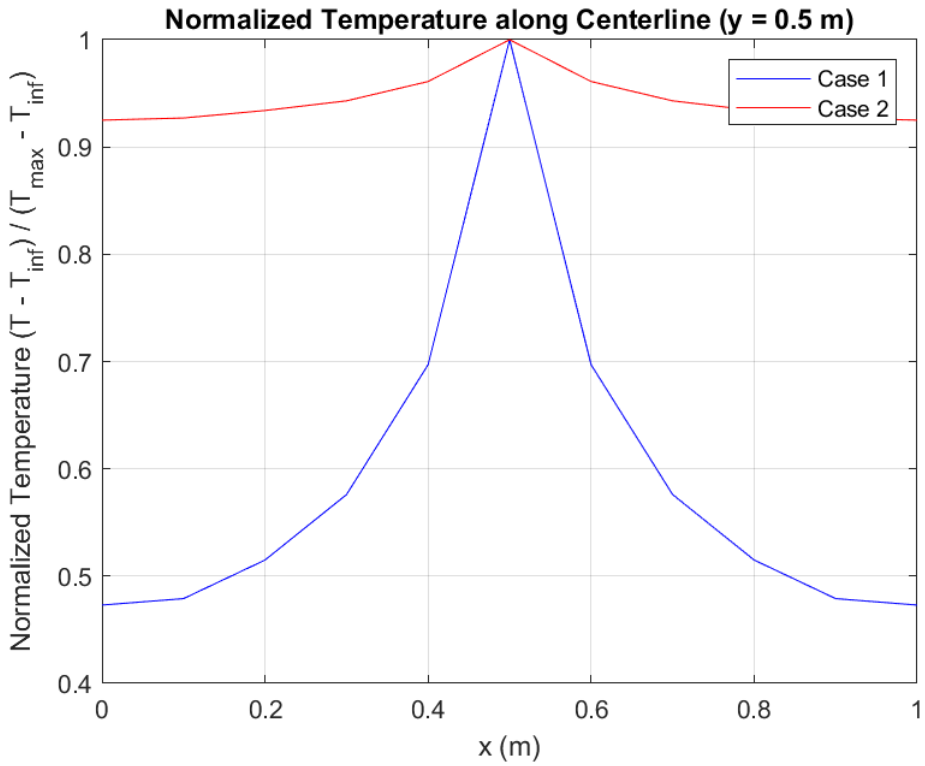

Fig 25

Observation from plot:-

1) **Peak at the Center:** Both curves show a peak normalized temperature of 1 at x = 0.5 m, which corresponds to the location of the heat source. This is expected since Tmax for normalization was taken as the peak temperature along this centerline for each case.
2) **Different Overall Levels:** The normalized temperature curve for Case 2 (red line) is significantly higher than the curve for Case 1 (blue line across most of the x-axis, except at the peak where they converge to 1.
3) **Similar Shape:** Both curves exhibit a similar bell-shaped profile, indicating that the relative distribution of temperature away from the heat source along the centreline follows a similar pattern in both cases.

Appendix

Providing Matlab code for all cases

Case1:

For 2 elements

```
% Parameters
k = 50;              % Thermal conductivity (W/m.K)
Q = 1000;            % Heat source strength (W/m^2)
L = 1;               % Plate length (m)
T_fixed = 25;        % Boundary temperature (°C)

% Node coordinates: [x, y]
nodes = [0 0;        % Node 1
         1 0;        % Node 2
         1 1;        % Node 3
         0 1;]       % Node 4];

% Elements (connect corners to center node)
elements = [1 2 3;
            1 3 4];

n_nodes = size(nodes, 1);
n_elements = size(elements, 1);


%% Plot the mesh
figure;
trimesh(elements, nodes(:,1), nodes(:,2), 'LineWidth', 1);
title('Triangular Mesh for FEM Analysis');
xlabel('X Position (m)');
ylabel('Y Position (m)');
axis equal;
grid on;

%% Add annotations (optional)
hold on;
plot(nodes(:,1), nodes(:,2), 'ro', 'MarkerSize', 6); % Show nodes as red circles

% Print mesh statistics
fprintf('Mesh details:\n');
fprintf('Number of nodes: %d\n', n_nodes);
fprintf('Number of elements: %d\n', size(elements,1));

K = zeros(n_nodes);    % Global stiffness matrix
F = zeros(n_nodes, 1); % Global load vector

% FEM Assembly
for e = 1:n_elements
   idx = elements(e, :);
   coords = nodes(idx, :);
```

```
    x = coords(:,1); y = coords(:,2);

    % Area of triangle
    A = polyarea(x, y);

    % Compute B matrix coefficients
    b = [y(2)-y(3); y(3)-y(1); y(1)-y(2)];
    c = [x(3)-x(2); x(1)-x(3); x(2)-x(1)];

    % Element stiffness matrix
    ke = (k / (4*A)) * (b*b' + c*c');

    % Assemble global stiffness matrix
    K(idx, idx) = K(idx, idx) + ke;
end

% --- SOURCE TERM: Dirac delta at center (node 3, shared node) ---

F(3) = F(3) + Q / 2;

% --- APPLY BOUNDARY CONDITIONS ---
fixed_nodes = [1, 2];
free_nodes = setdiff(1:n_nodes, fixed_nodes);

% Modify system for Dirichlet BC
T = zeros(n_nodes, 1);
T(fixed_nodes) = T_fixed;

F_mod = F(free_nodes) - K(free_nodes, fixed_nodes) * T(fixed_nodes);
K_mod = K(free_nodes, free_nodes);

% Solve the system
T(free_nodes) = K_mod \ F_mod;

%% Display results
disp('Nodal Temperatures:');
for i = 1:n_nodes
    fprintf('Node %d (%0.1f, %0.1f): T = %.2f°C\n', i, nodes(i,1), nodes(i,2), T(i));
end

% Plotting temperature contour
figure;
trisurf(elements, nodes(:,1), nodes(:,2), T, 'FaceColor','interp');
colorbar;
title('Temperature Distribution (4 Elements)');
xlabel('x (m)'); ylabel('y (m)'); zlabel('Temperature (°C)');
view(2); axis equal; shading interp;

% Plot temperature along horizontal centerline y = 0.5
x_line = linspace(0, 1, 100);
yline = 0.5 * ones(size(x_line));
T_line = griddata(nodes(:,1), nodes(:,2), T, x_line, yline);

figure;
```

```
plot(x_line, T_line, '-');
title('Temperature along centerline y = 0.5 m');
xlabel('x (m)'); ylabel('Temperature (°C)');
grid on;
```

for 4 elements :

```
% Parameters
k = 50;              % Thermal conductivity (W/m.K)
Q = 1000;            % Heat source strength (W/m^2)
L = 1;               % Plate length (m)
T_fixed = 25;        % Boundary temperature (°C)

% Node coordinates: [x, y]
nodes = [0 0;        % Node 1
         1 0;        % Node 2
         1 1;        % Node 3
         0 1;        % Node 4
         0.5 0.5];   % Node 5 (center)

% Elements (connect corners to center node)
elements = [1 2 5;
            2 3 5;
            3 4 5;
            4 1 5];

n_nodes = size(nodes, 1);
n_elements = size(elements, 1);


%% Plot the mesh
figure;
trimesh(elements, nodes(:,1), nodes(:,2), 'LineWidth', 1);
title('Triangular Mesh for FEM Analysis');
xlabel('X Position (m)');
ylabel('Y Position (m)');
axis equal;
grid on;

%% Add annotations (optional)
hold on;
plot(nodes(:,1), nodes(:,2), 'ro', 'MarkerSize', 6); % Show nodes as red circles

% Print mesh statistics
fprintf('Mesh details:\n');
fprintf('Number of nodes: %d\n', n_nodes);
fprintf('Number of elements: %d\n', size(elements,1));

K = zeros(n_nodes);    % Global stiffness matrix
F = zeros(n_nodes, 1); % Global load vector

% FEM Assembly
for e = 1:n_elements
    idx = elements(e, :);
```

```
    coords = nodes(idx, :);
    x = coords(:,1); y = coords(:,2);

    % Area of triangle
    A = polyarea(x, y);

    % Compute B matrix coefficients
    b = [y(2)-y(3); y(3)-y(1); y(1)-y(2)];
    c = [x(3)-x(2); x(1)-x(3); x(2)-x(1)];

    % Element stiffness matrix
    ke = (k / (4*A)) * (b*b' + c*c');

    % Assemble global stiffness matrix
    K(idx, idx) = K(idx, idx) + ke;
end

% Apply point heat source at center node (node 5)
F(5) = F(5) + Q;

% Apply Dirichlet BC on bottom side (y=0): nodes 1 and 2
fixed_nodes = [1, 2];
free_nodes = setdiff(1:n_nodes, fixed_nodes);

% Modify system for Dirichlet BC
T = zeros(n_nodes, 1);
T(fixed_nodes) = T_fixed;

F_mod = F(free_nodes) - K(free_nodes, fixed_nodes) * T(fixed_nodes);
K_mod = K(free_nodes, free_nodes);

% Solve the system
T(free_nodes) = K_mod \ F_mod;

%% Display results
disp('Nodal Temperatures:');
for i = 1:n_nodes
    fprintf('Node %d (%0.1f, %0.1f): T = %.2f°C\n', i, nodes(i,1), nodes(i,2), T(i));
end

% Plotting temperature contour
figure;
trisurf(elements, nodes(:,1), nodes(:,2), T, 'FaceColor','interp');
colorbar;
title('Temperature Distribution (4 Elements)');
xlabel('x (m)'); ylabel('y (m)'); zlabel('Temperature (°C)');
view(2); axis equal; shading interp;

% Plot temperature along horizontal centerline y = 0.5
x_line = linspace(0, 1, 100);
yline = 0.5 * ones(size(x_line));
T_line = griddata(nodes(:,1), nodes(:,2), T, x_line, yline);

figure;
```

```
plot(x_line, T_line, '-');
title('Temperature along centerline y = 0.5 m');
xlabel('x (m)'); ylabel('Temperature (°C)');
grid on;
```

for 8 elements :

```
% Parameters
k = 50;             % Thermal conductivity (W/m.K)
Q = 1000;           % Heat source (W/m^2)
T_fixed = 25;       % Boundary temperature (°C)
Nx = 2;             % Number of elements along x (2squares → ~8 triangles)
Ny = 2;             % Number of elements along y

% Generate grid
L = 1;
dx = L / Nx;
dy = L / Ny;
[x_grid, y_grid] = meshgrid(0:dx:L, 0:dy:L);
nodes = [x_grid(:), y_grid(:)];
n_nodes = size(nodes,1);

% Create elements (split each square into 2 triangles)
elements = [];
for j = 1:Ny
   for i = 1:Nx
      % Node indices
      n1 = i + (j-1)*(Nx+1);
      n2 = n1 + 1;
      n3 = n1 + (Nx+1);
      n4 = n3 + 1;

      % Two triangles per square
      elements = [elements;
                  n1 n2 n4;
                  n1 n4 n3];
   end
end

%% Plot the mesh
figure;
trimesh(elements, nodes(:,1), nodes(:,2), 'LineWidth', 1);
title('Triangular Mesh for FEM Analysis');
xlabel('X Position (m)');
ylabel('Y Position (m)');
axis equal;
grid on;

%% Add annotations (optional)
hold on;
plot(nodes(:,1), nodes(:,2), 'ro', 'MarkerSize', 6); % Show nodes as red circles

% Print mesh statistics
fprintf('Mesh details:\n');
```

```
fprintf('Number of nodes: %d\n', n_nodes);
fprintf('Number of elements: %d\n', size(elements,1));



n_elements = size(elements,1);

% Initialize stiffness matrix and load vector
K = zeros(n_nodes);
F = zeros(n_nodes,1);

% Assemble global stiffness matrix
for e = 1:n_elements
    idx = elements(e,:);
    coords = nodes(idx,:);
    x = coords(:,1); y = coords(:,2);

    A = polyarea(x, y);
    b = [y(2)-y(3); y(3)-y(1); y(1)-y(2)];
    c = [x(3)-x(2); x(1)-x(3); x(2)-x(1)];

    ke = (k / (4*A)) * (b*b' + c*c');

    K(idx, idx) = K(idx, idx) + ke;
end

% Apply point heat source at center node
center_coord = [0.5, 0.5];
[~, center_node] = min(vecnorm(nodes - center_coord, 2, 2));
F(center_node) = Q;

% Apply boundary conditions
% Bottom edge: y = 0 → Dirichlet
tol = 1e-6;
fixed_nodes = find(abs(nodes(:,2)) < tol);
T = zeros(n_nodes, 1);
T(fixed_nodes) = T_fixed;
free_nodes = setdiff(1:n_nodes, fixed_nodes);

% Modify K and F for Dirichlet BC
F_mod = F(free_nodes) - K(free_nodes, fixed_nodes) * T(fixed_nodes);
K_mod = K(free_nodes, free_nodes);

% Solve
T(free_nodes) = K_mod \ F_mod;

%% Display results
disp('Nodal Temperatures:');
for i = 1:n_nodes
    fprintf('Node %d (%0.1f, %0.1f): T = %.2f°C\n', i, nodes(i,1), nodes(i,2), T(i));
end

% Plotting
figure;
```

```
trisurf(elements, nodes(:,1), nodes(:,2), T, 'FaceColor','interp');
colorbar;
title('Temperature Distribution (~18 Elements)');
xlabel('x (m)'); ylabel('y (m)'); zlabel('Temperature (°C)');
view(2); axis equal; shading interp;

% Plot centerline (y = 0.5)
x_line = linspace(0, 1, 100);
T_line = griddata(nodes(:,1), nodes(:,2), T, x_line, 0.5*ones(size(x_line)));

figure;
plot(x_line, T_line, '-');
title('Temperature along centerline y = 0.5 m');
xlabel('x (m)'); ylabel('Temperature (°C)');
grid on;

for 200 elements:
% Parameters
k = 50;             % Thermal conductivity (W/m.K)
Q = 1000;           % Heat source (W/m^2)
T_fixed = 25;       % Boundary temperature (°C)
Nx = 10;            % Elements along x
Ny = 10;            % Elements along y

% Generate structured grid
L = 1;
dx = L / Nx;
dy = L / Ny;
[x_grid, y_grid] = meshgrid(0:dx:L, 0:dy:L);
nodes = [x_grid(:), y_grid(:)];
n_nodes = size(nodes,1);

% Create triangular elements
elements = [];
for j = 1:Ny
   for i = 1:Nx
      n1 = i + (j-1)*(Nx+1);
      n2 = n1 + 1;
      n3 = n1 + (Nx+1);
      n4 = n3 + 1;

      % Triangle 1: lower-left to upper-right
      elements = [elements;
                  n1 n2 n4;
                  n1 n4 n3];
   end
end


%% Plot the mesh
figure;
trimesh(elements, nodes(:,1), nodes(:,2), 'LineWidth', 1);
title('Triangular Mesh for FEM Analysis');
xlabel('X Position (m)');
```

```
ylabel('Y Position (m)');
axis equal;
grid on;

%% Add annotations (optional)
hold on;
plot(nodes(:,1), nodes(:,2), 'ro', 'MarkerSize', 6); % Show nodes as red circles

% Print mesh statistics
fprintf('Mesh details:\n');
fprintf('Number of nodes: %d\n', n_nodes);
fprintf('Number of elements: %d\n', size(elements,1));

n_elements = size(elements,1);

% Initialize global matrices
K = zeros(n_nodes);
F = zeros(n_nodes,1);

% Assemble global stiffness matrix
for e = 1:n_elements
    idx = elements(e,:);
    coords = nodes(idx,:);
    x = coords(:,1); y = coords(:,2);

    A = polyarea(x, y);
    b = [y(2)-y(3); y(3)-y(1); y(1)-y(2)];
    c = [x(3)-x(2); x(1)-x(3); x(2)-x(1)];

    ke = (k / (4*A)) * (b*b' + c*c');
    K(idx, idx) = K(idx, idx) + ke;
end

% Apply heat source at center node
center_coord = [0.5, 0.5];
[~, center_node] = min(vecnorm(nodes - center_coord, 2, 2));
F(center_node) = Q;

% Apply Dirichlet BC: bottom edge (y=0)
tol = 1e-6;
fixed_nodes = find(abs(nodes(:,2)) < tol);
T = zeros(n_nodes, 1);
T(fixed_nodes) = T_fixed;
free_nodes = setdiff(1:n_nodes, fixed_nodes);

% Modify system
F_mod = F(free_nodes) - K(free_nodes, fixed_nodes) * T(fixed_nodes);
K_mod = K(free_nodes, free_nodes);

% Solve
T(free_nodes) = K_mod \ F_mod;

%% Display results
disp('Nodal Temperatures:');
```

```
for i = 1:n_nodes
   fprintf('Node %d (%0.1f, %0.1f): T = %.2f°C\n', i, nodes(i,1), nodes(i,2), T(i));
end

% Plot: temperature contour
figure;
trisurf(elements, nodes(:,1), nodes(:,2), T, 'FaceColor','interp');
colorbar;
title('Temperature Distribution (~100 Elements)');
xlabel('x (m)'); ylabel('y (m)'); zlabel('Temperature (°C)');
view(2); axis equal; shading interp;

% Plot temperature along centerline (y = 0.5)
x_line = linspace(0, 1, 200);
T_line = griddata(nodes(:,1), nodes(:,2), T, x_line, 0.5 * ones(size(x_line)));

figure;
plot(x_line, T_line, '-');
title('Temperature along centerline y = 0.5 m');
xlabel('x (m)'); ylabel('Temperature (°C)');
grid on;
```

Case 2 :
For 200 elements

```
% Parameters
k = 50;              % Thermal conductivity (W/m.K)
Q = 1000;              % Heat input at center node
T_inf = 25;            % Ambient temp for convection
h = 10;              % Convection coefficient (W/m^2-K)
Nx = 10; Ny = 10;      % Grid size

% Generate structured mesh
L = 1;
dx = L / Nx;
dy = L / Ny;
[x_grid, y_grid] = meshgrid(0:dx:L, 0:dy:L);
nodes = [x_grid(:), y_grid(:)];
n_nodes = size(nodes,1);

% Element connectivity (triangulated squares)
elements = [];
for j = 1:Ny
   for i = 1:Nx
      n1 = i + (j-1)*(Nx+1);
      n2 = n1 + 1;
      n3 = n1 + (Nx+1);
      n4 = n3 + 1;
      elements = [elements;
                  n1 n2 n4;
                  n1 n4 n3];
   end
end
```

```
%% Plot the mesh
figure;
trimesh(elements, nodes(:,1), nodes(:,2), 'LineWidth', 1);
title('Triangular Mesh for FEM Analysis');
xlabel('X Position (m)');
ylabel('Y Position (m)');
axis equal;
grid on;

%% Add annotations (optional)
hold on;
plot(nodes(:,1), nodes(:,2), 'ro', 'MarkerSize', 6); % Show nodes as red circles

% Print mesh statistics
fprintf('Mesh details:\n');
fprintf('Number of nodes: %d\n', n_nodes);
fprintf('Number of elements: %d\n', size(elements,1));

n_elements = size(elements,1);

% Global matrices
K = zeros(n_nodes);
F = zeros(n_nodes,1);

% Assembly
for e = 1:n_elements
   idx = elements(e,:);
   coords = nodes(idx,:);
   x = coords(:,1); y = coords(:,2);

   A = polyarea(x, y);
   b = [y(2)-y(3); y(3)-y(1); y(1)-y(2)];
   c = [x(3)-x(2); x(1)-x(3); x(2)-x(1)];
   ke = (k / (4*A)) * (b*b' + c*c');
   K(idx, idx) = K(idx, idx) + ke;
end

% Convection BC on bottom edge (y = 0)
tol = 1e-6;
bottom_nodes = find(abs(nodes(:,2)) < tol);
bottom_edges = [];

% Find boundary edges (node pairs)
for i = 1:length(bottom_nodes)-1
   n1 = bottom_nodes(i);
   n2 = bottom_nodes(i+1);
   if abs(nodes(n1,1) - nodes(n2,1)) <= dx + 1e-6
      bottom_edges = [bottom_edges; n1 n2];
   end
end

% Add convection contribution
for e = 1:size(bottom_edges,1)
```

```
    n1 = bottom_edges(e,1);
    n2 = bottom_edges(e,2);
    x1 = nodes(n1,1); y1 = nodes(n1,2);
    x2 = nodes(n2,1); y2 = nodes(n2,2);
    L_edge = sqrt((x2 - x1)^2 + (y2 - y1)^2);

    ke_conv = h * L_edge / 6 * [2 1; 1 2];
    fe_conv = h * T_inf * L_edge / 2 * [1; 1];

    K([n1 n2],[n1 n2]) = K([n1 n2],[n1 n2]) + ke_conv;
    F([n1 n2]) = F([n1 n2]) + fe_conv;
end

% Apply point heat source at center
center_coord = [0.5, 0.5];
[~, center_node] = min(vecnorm(nodes - center_coord, 2, 2));
F(center_node) = F(center_node) +Q;

% Solve (all nodes are free now)
T = K \ F;


%% Display results
disp('Nodal Temperatures:');
for i = 1:n_nodes
    fprintf('Node %d (%0.1f, %0.1f): T = %.2f°C\n', i, nodes(i,1), nodes(i,2), T(i));
end

% Plot temperature distribution
figure;
trisurf(elements, nodes(:,1), nodes(:,2), T, 'FaceColor','interp');
colorbar;
title('Case 2: Temp Distribution (~100 Elements + Convection)');
xlabel('x (m)'); ylabel('y (m)'); zlabel('Temperature (°C)');
view(2); axis equal; shading interp;

% Plot centerline
x_line = linspace(0, 1, 200);
T_line = griddata(nodes(:,1), nodes(:,2), T, x_line, 0.5 * ones(size(x_line)));

figure;
plot(x_line, T_line, '-');
title('Temperature along centerline y = 0.5 m (Convective BC)');
xlabel('x (m)'); ylabel('Temperature (°C)');
grid on;
```