Finite Element Method

A Report on 2D Heat Flow Solution Using FEM

Department of Mechanical and Aerospace Engineering
Pulchowk Campus, Institute of Engineering
Nepal

Date: 07/03/2023

Submitted By:
Bibek Yonzan, 077BAS005
Simran Paudel, 077BAS043
Manish Tajpuriya, 077BAS023
Simonkrith Lamichhane, 077BAS042

Submitted To:
Asst. Prof. Kamal Darlami
Deputy Head of Department

1 Introduction

The global system of finite element equations for a 2D heat flow along an element can be expressed as:

$$\sum(K_T + H_T)T = R_\infty + R_Q - R_q$$

Each term specifically defines the elemental and boundary contribution in a 2D heat flow as 'Conductive Heat Transfer Coefficient', 'Convective Heat Transfer Coefficient', 'Temperature', 'Convective Heat Transfer', 'Heat Source', and 'Imposed Heat Normal Flux' respectively.

1.1 Boundary Conditions (BCs)

One of the following boundary conditions (BC) must be specified at the boundary of the domain:

1. **Prescribed Boundary Temperature (Dirichlet BC):**

$$T = T(x, y) \quad for \quad x, y \in \Gamma_T$$

where $T(x, y)$ is given.

2. **Prescribed Normal Boundary Flux (Neumann BC):**

$$q_n = q_n(x, y) \quad for \quad x, y \in \Gamma_q$$

where $q_n(x, y)$ is given. The flux normal to the boundary is defined as:

$$q_n = q_x \cos\alpha + q_y \sin\alpha = -k\frac{\partial T}{\partial x}\cos\alpha - k\frac{\partial T}{\partial y}\sin\alpha = -k\frac{\partial T}{\partial n}$$

3. **Convective Boundary Conditions (Cauchy BC):**

$$q_n = h(T - T_\infty) \quad for \quad x, y \in \Gamma_c$$

where h is the coefficient of convective heat transfer and T_∞ is the temperature of the convective medium.

2 Objective

The task is to solve dealing problems with 2D heat flow. The overall objective is to modify an existing MATLAB code which was able to solve a 2D heat flow problem with constant heat source per element, with prescribed normal heat flux as zero, wherever assigned.

Some useful data is provided to solve the problem and determine the temperature at nodes 1 and 2 :

- Normal heat flux $q = 0$
- Thermal conductivity $= 50\, W/m^\circ C$
- Prescribed temperature $= 0^\circ C$
- Ambient temperature $T_\infty = 30^\circ C$
- Convection coefficient $h = 30\, W/m^{2\circ} C$
- Uniform volumetric heat source strength $= 100\, W/m^3$

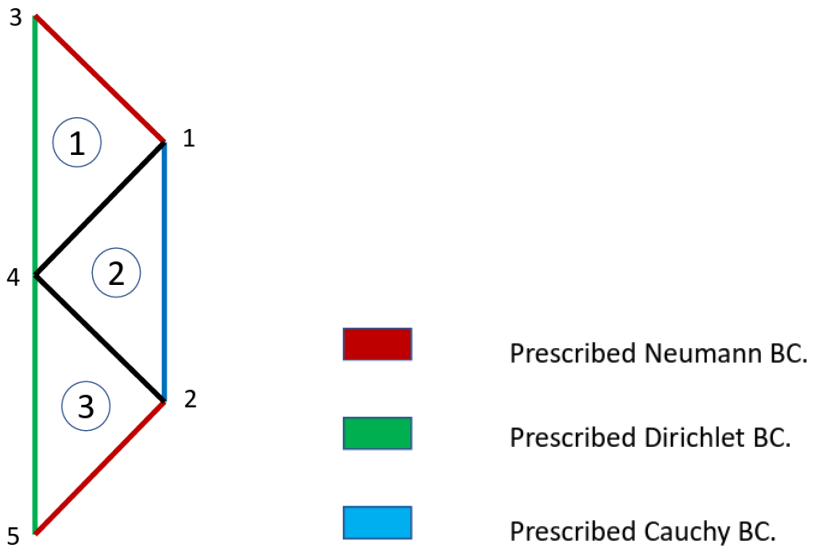


Figure 1: Given domain and its associated boundary conditions

3 Methodology

We used MATLAB® to modify the code. Three different methods were adopted by the team to generate the .dat file for the solver. The methodology for each one is described in the subsequent sections.

3.1 Manual Meshing

For this method, we identified the individual equations for the boundaries of the heat flow domain. Once the equations were identified, they were plotted using an graphing tool like Desmos®. In the plots, equations with increments in the y-intercept were added and the points of intersection between the incremental equations and the domain boundary equations were noted. The points of intersection then served as the nodes for mesh of the domain. The obtained points were then assembled into a .dat with the necessary boundary conditions and general requirements.

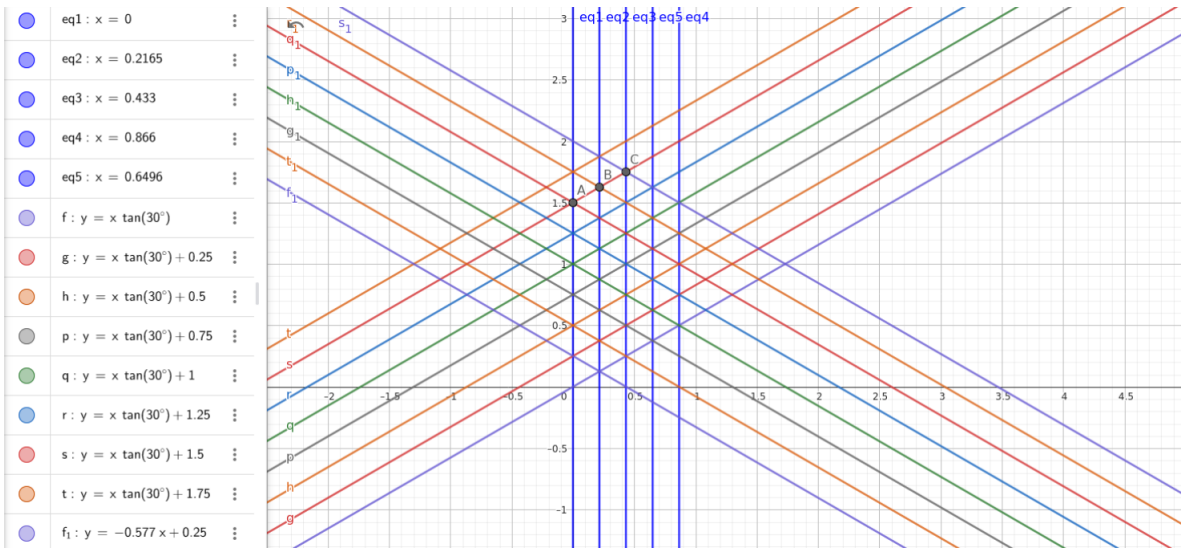


Figure 2: Plotting the mesh using a series of equations.

3.2 Using PDE Toolbox

For this, we write the following code to create and write ".dat" file.

```
%%geom definitions

model = createpde("thermal","steadystate");

T1 = [2,4,0,0.866,0.866,0,0,0.5,1.5,2]';
g = decsg(T1);

geometryFromEdges(model,g);
mesh = generateMesh(model,"GeometricOrder","linear","Hmax",0.1);
% pdegplot(model,'Edgelabels','on')
pdeplot(model,"NodeLabels","on")
```

```
%% BC setting

thermalBC(model,"Edge",2,"ConvectionCoefficient",30,"AmbientTemperature",30);
thermalBC(model,"Edge",4,"Temperature",0);
thermalBC(model,"Edge",[1,3],"HeatFlux",0);
internalHeatSource(model,100)
thermalProperties(model,"ThermalConductivity",50)

temperatures = solve(model).Temperature;
conductivity = 50;
heatSource = 100;
convectionCoefficient = 30;
ambientTemperature = 30;
```

```
%% node list and element list generation

NL = mesh.Nodes';
EL = mesh.Elements';
EL_size = length(EL);
NL_size = length(NL);
EL_list = zeros(EL_size,1);
NL_list = zeros(NL_size,1);
for i = 1:EL_size
    EL_list(i) = i;
    EL_to_print(i,1:6) = [EL_list(i) EL(i,1:3) conductivity heatSource];
end
for i = 1:NL_size
    NL_list(i) = i;
end
```

```
%% writing element data to the dat file
fileID = fopen('data.dat','a+');
fprintf(fileID,'TITLE = %d-elem-problem\n',EL_size);
fprintf(fileID,'\nELEMENTS = %d\n',EL_size);
writematrix(EL_to_print,'data.dat','Delimiter',' ',WriteMode='append');
```

```
%% writing node data to the dat file
fprintf(fileID,'\nNODE_COORDINATES = %d', NL_size);

writematrix([NL_list NL],'data.dat','Delimiter',' ',WriteMode='append');
```

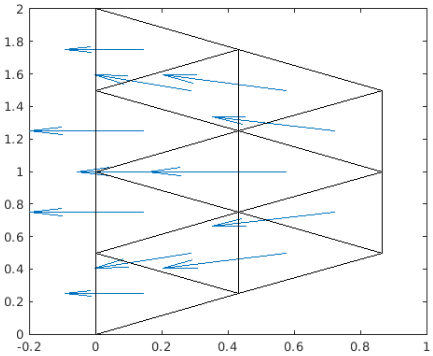
```
%% calculating known edge temperatures

NL_x = NL(:,1);
c = 1;
for i = 1:NL_size
    if NL_x(i) == 0
        known_temp_node(c,1) = i;
        known_temp_node(c,2) = temperatures(i);
        c = c + 1;
    end
end

fprintf(fileID,'\nNODES_WITH_PRESCRIBED_TEMPERATURE = %d',c-1);
writematrix(known_temp_node,'data.dat','Delimiter',' ',WriteMode='append');
```

```
%% calculating convection edges

c_2 = 1;
for i = 1:NL_size
    if NL_x(i) == 0.866
        known_convec_node(c_2,1) = i;
        known_convec_node(c_2,2:3) = NL(i,:);
        c_2 = c_2 + 1;
    end
end
sorted_convec_nodes = sortrows(known_convec_node,3);

for i = 1:length(sorted_convec_nodes)
    if i < length(sorted_convec_nodes)
        edges_to_print(i,1:4) = [sorted_convec_nodes(i,1) sorted_convec_nodes(i+1,1) convectionCoefficient ambientTemperature];
    end
end

fprintf(fileID,'\nEDGES_WITH_PRESCRIBED_CONVECTION = %d',c_2-2);
writematrix(edges_to_print,'data.dat','Delimiter',' ',WriteMode='append');
```

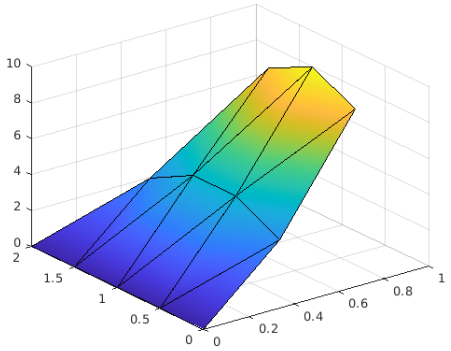
```
%% closing the file
fclose(fileID)
```

Figure 4: source Code

Using this code, the following number of meshes were generated and was solved.

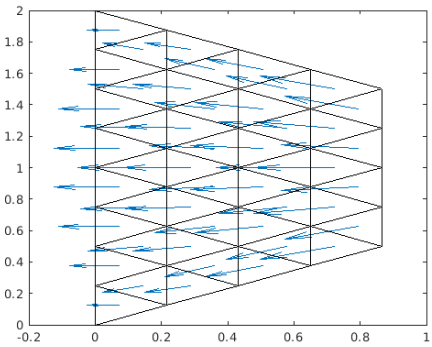


(a) Mesh Generation

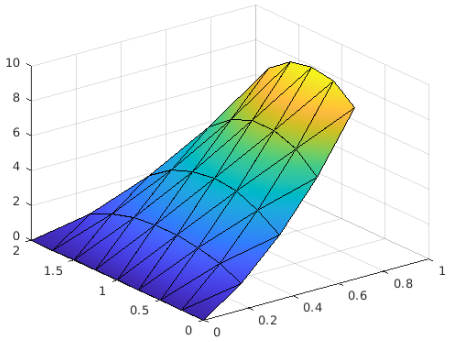


(b) Heat Convection

Figure 5: 12-elements

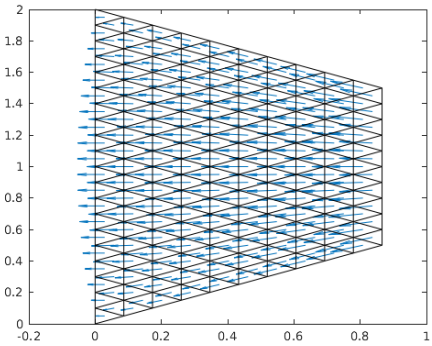


(a) Mesh Generation

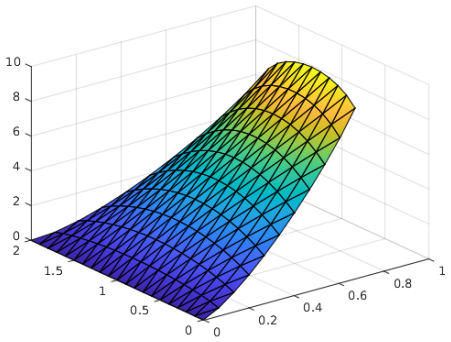


(b) Heat Convection

Figure 6: 48-elements

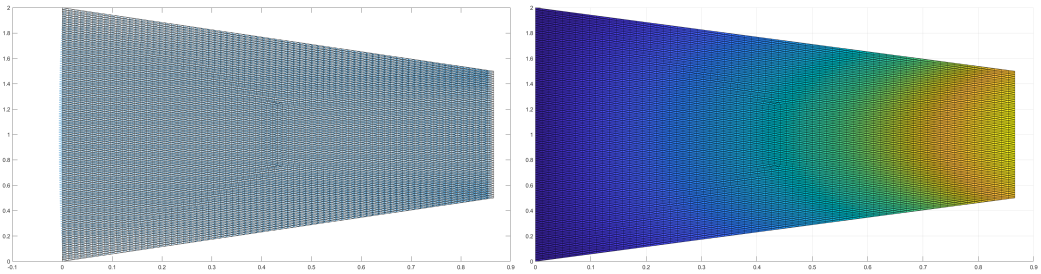


(a) Mesh Generation



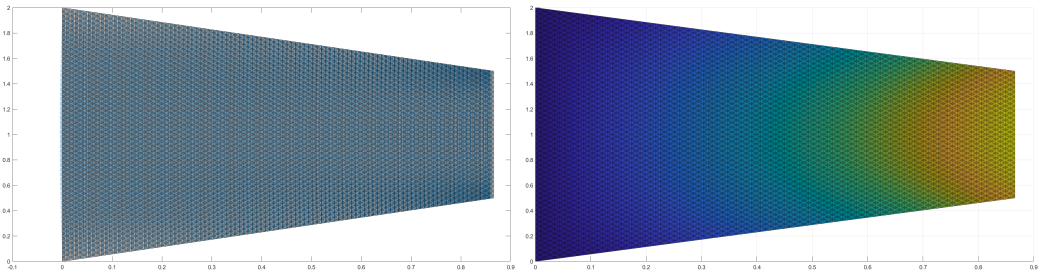
(b) Heat Convection

Figure 7: 300-elements



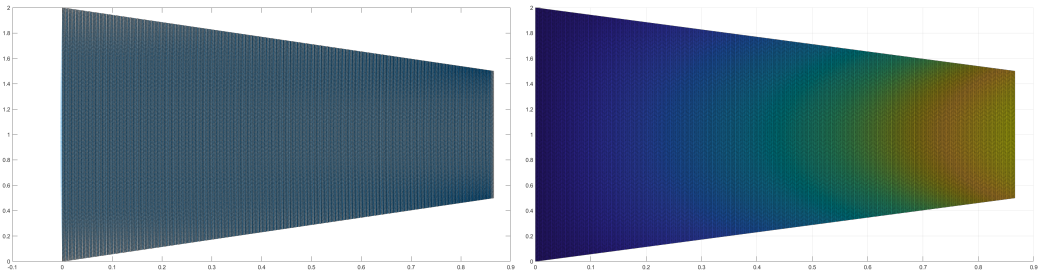
(a) Mesh Generation (b) Heat Convection

Figure 8: 15000-elements



(a) Mesh Generation (b) Heat Convection

Figure 9: 30000-elements



(a) Mesh Generation (b) Heat Convection

Figure 10: 46000-elements

3.3 Achievement

We've modified the code and achieved the following:

- Simulated linear internal heat source, defined node by node.
- Calculated, plotted and displayed both the temperature and reactive heat flux at each node with the prescribed temperature.
- Solved the problem with finer meshing to develop a better understanding of FEM.

4 Results and Findings

The code was run for different number of elements. The comparison table is as follow:

Number of elements	Temperature at node 1	Temperature at node 2
3	8.2135	8.2135
12	8.0451	8.0451
48	8.0048	8.0048
75	8.0008	8.0008
185	7.9961	7.9961
300	7.9957	7.9957
1200	7.9948	7.9948
15266	7.99474	7.99474
30000	7.99474	7.99474
46875	7.99475	7.99475
61189	7.99475	7.99475

Table 1: Comparison Table

The graphical representation of the comparison table is shown below, which shows us the converging solution at 185 number of elements.

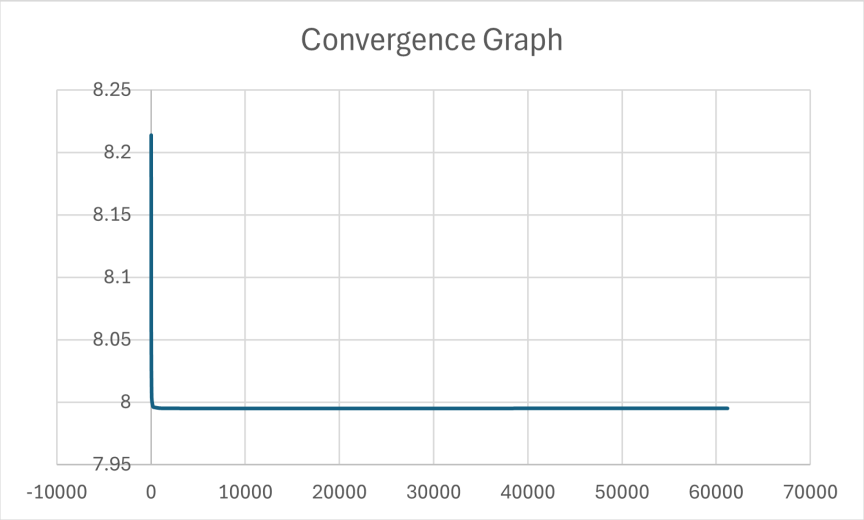


Figure 11: Graph of Temperature Vs Number of Elements

5 Conclusion

Hence, from the table it is clear that the converging solution for temperature at node 1 and 2 are 7.99475 degrees respectively. For 3 elements, the temperatures at nodes 1 and 2 is 8.2135 each. As the number of elements increases the temperatures for the nodes also decreases. However, once the mesh starts getting finer, the temperature values converge. The team calculated the nodal temperatures for a domain with up-to 61,189 elements. But the temperatures at the nodes 1 and 2 remained 7.99 each after 185 elements. Thus, after a certain point, it is not efficient computationally and resource-wise to increase the number of elements.

6 Appendix

- DAT File Link