ME 544 Lab 4 Report
Christopher Ng - Russell Occhipinti
4/30/24

1. Using the MatLab PDE Toolbox, a FEM temperature distribution using a line of thermal symmetry down half of the x-length can be seen in Figure 1. Compared to the previous temperature distributions seen in Lab 2 and 3, this plot matches overall very closely, but using a higher node count and with a much higher density of nodes near the top left corner, the resolution near the corner is significantly better than those seen in the FDM temperature distribution. As seen in Figure 2, when comparing the FEM, FDM, and Analytical solutions for the temperature at x and y positions of 0.4m, a noticeable difference in the number of nodes can be seen for the two numerical solutions to reach the analytical solution. The FEM solution only requires 169 nodes (n = 2) while the FDM solution requires 625 nodes (Nx = 25). A few explanations could be given for this difference. The node distribution for the FEM could help improve the convergence of certain regions of the model compared to the square FDM nodes that do poorly in quickly changing temperature gradient areas. The automatic relaxation applied to the PDE Toolbox could have helped the convergence to reach the correct temperature compared to the optimal relaxed FDM model.

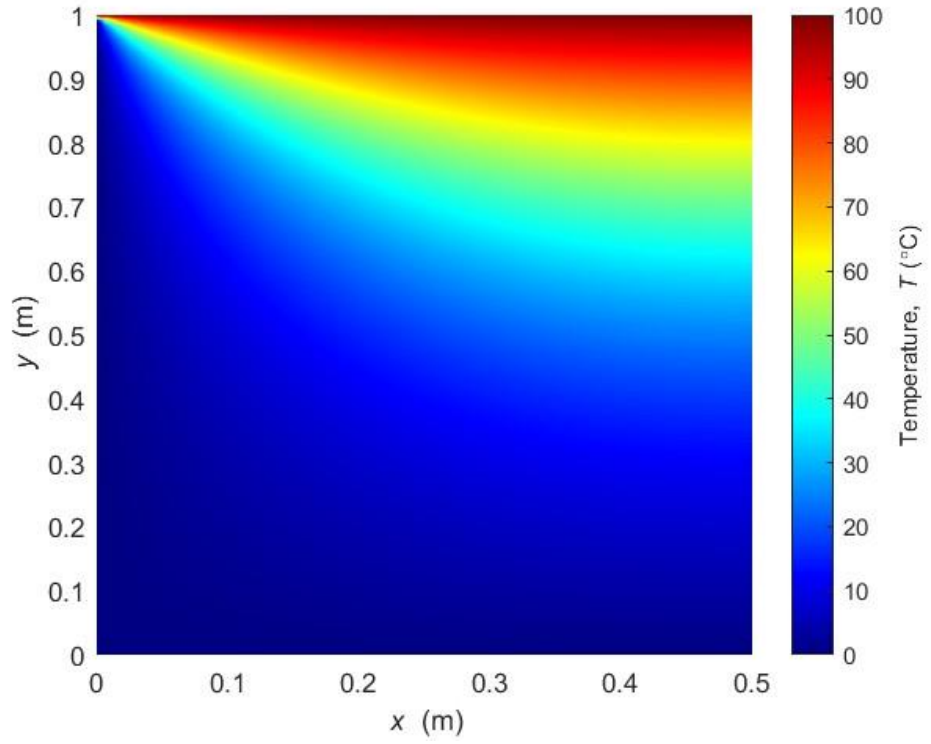

Figure 1. Steady Thermal Symmetry FEM Temperature Distribution (Hmax = 0.02)

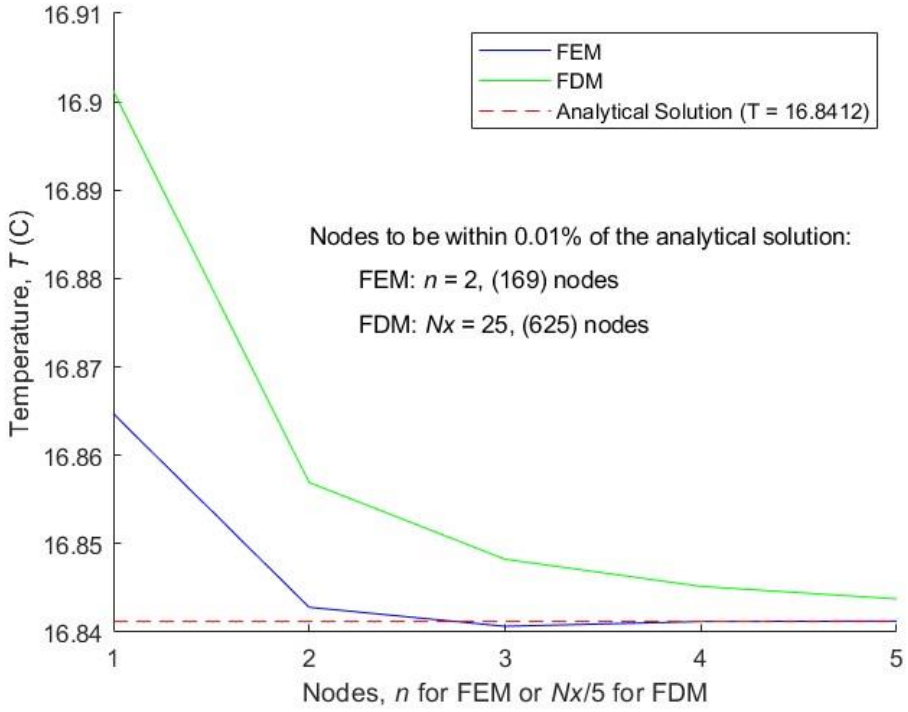

Figure 2. FEM and FDM Temperature Plotted Over Number of Nodes (for x = y = 0.4m)

2.

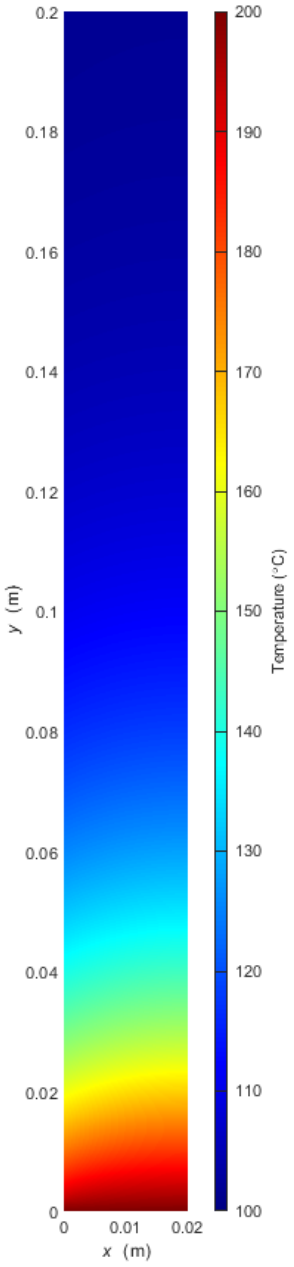

Figure 3. Steady Thermal Symmetry FEM Temperature Distribution for Fin (Hmax = 0.004)

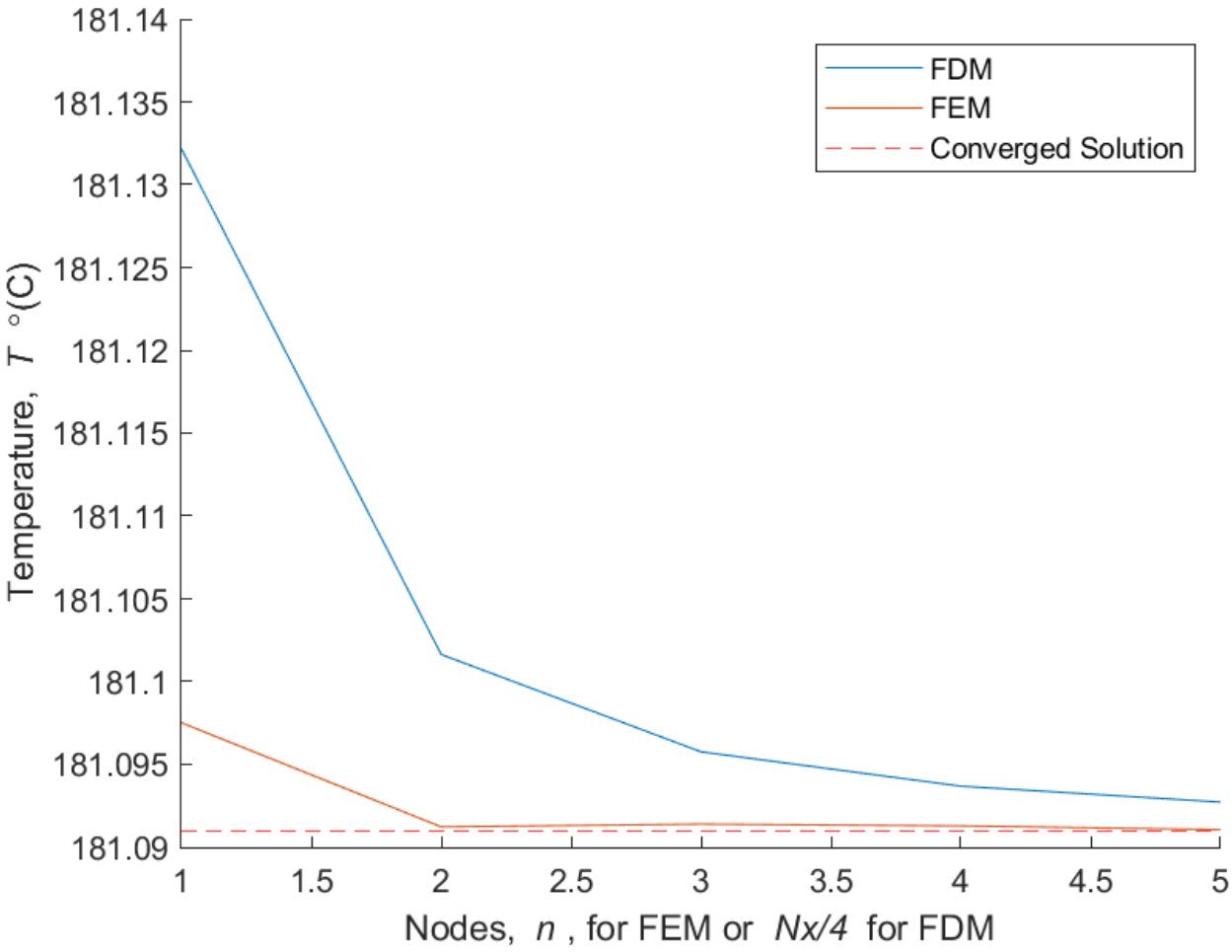

Figure 4. FEM and FDM Temperature for Fin Plotted Over Number of Nodes (for x = y = 0.1m)

Using MATLAB's PDE toolbox, the temperature distribution in half of a fin can be found and is displayed in Figure 3. Previously, the same problem has been investigated in labs 2 and 3. The solution found here correlates exactly with solutions found previously. To compare the results of this analysis with previous methods, the temperature at location x = 0.01 m and y = 0.01 m was calculated with each method for a number of settings and displayed in Figure 4. For FEM, n = 1 and for FDM, Nx = 8 was sufficient to be within 0.01% of the converged value. This corresponded with a value of 82 elements for FEM and 720 nodes for FDM.

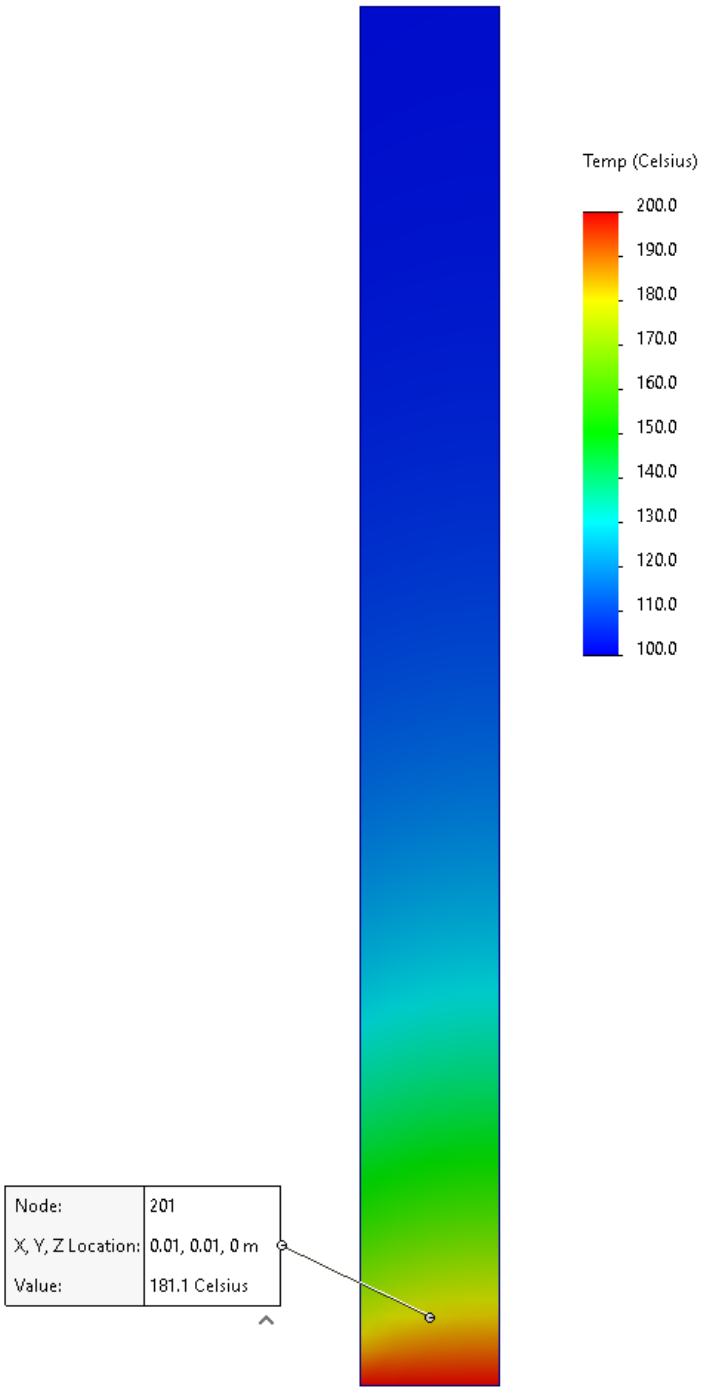

Figure 5. Solidworks 2D Solution for Fin. Temperature at point x = 0.1m, y = 0.1m Labeled

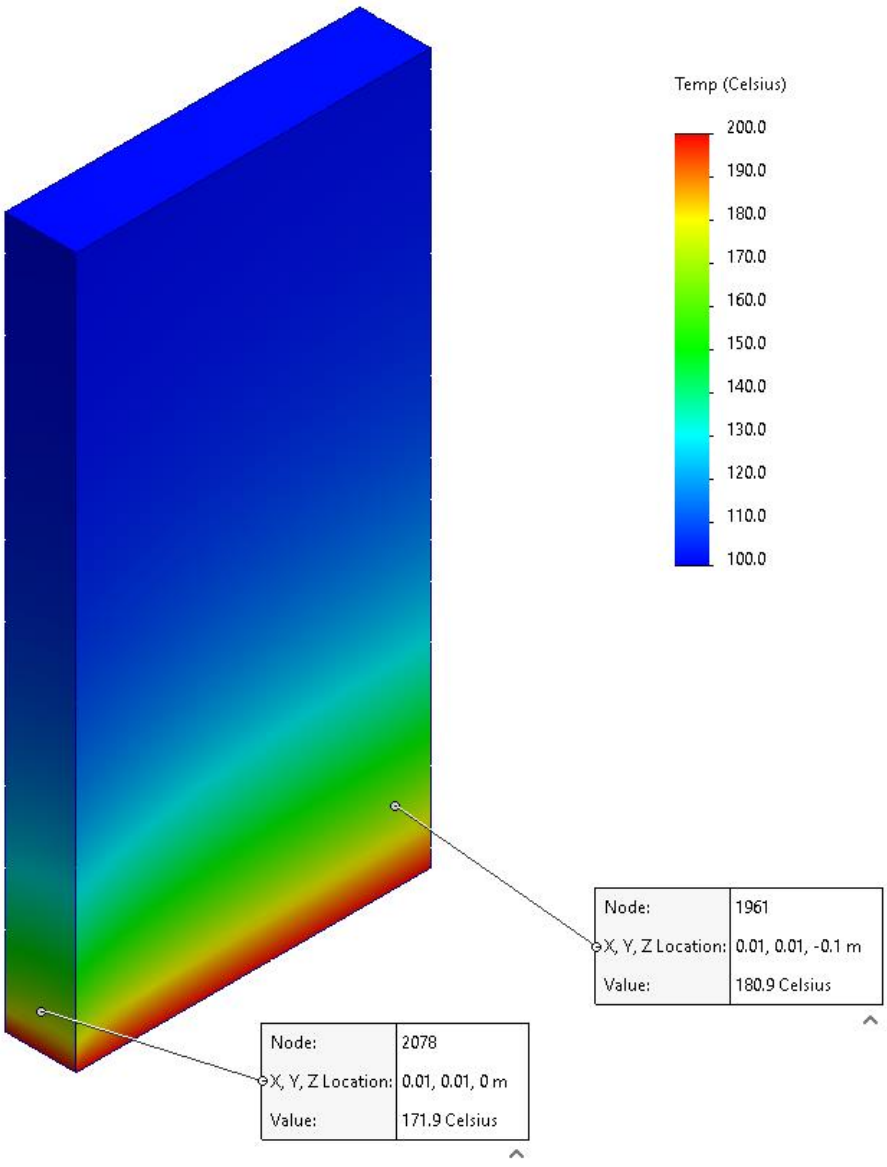

Figure 6. Solidworks 3D Simulation Solution for Fin. Temperature at point x = 0.1m, y = 0.1m, z = -0.1m and z = 0m Labeled

Figures 5 and 6 display the results of a SolidWorks Simulation of the previously solved fin. The results of the 2D simulation match our previous 2D solutions. However, the temperature distribution changes when the model is improved to 3D likely due to the 3-dimensional geometry distributing heat across more volume and thus we see smaller temperature numbers. A SolidWorks Simulation Report is appended at the end of the document.

3. Utilizing the PDE Toolbox again, a temperature distribution can be seen in Figure 7 for the steady state case of a non-linear thermal conductivity, *k,* that is dependent on temperature. The results of this FEM solution are once again very similar to the FDM solution with the main difference being the higher density meshing done at the corners. The FEM solution, however, was observed to have issues converging in the top right corner, which was more evident at lower node amounts. This could have occurred due to the sharp change in temperature at the corners compared to taking the average of the two corner surfaces for the FDM solution. The number of nodes can also be compared to the linear model, where more nodes were required for convergence for the non-linear model. This is reasonable since the temperature dependence for thermal conductivity makes the residual less accurate to the prior solution with varying values of thermal conductivity.

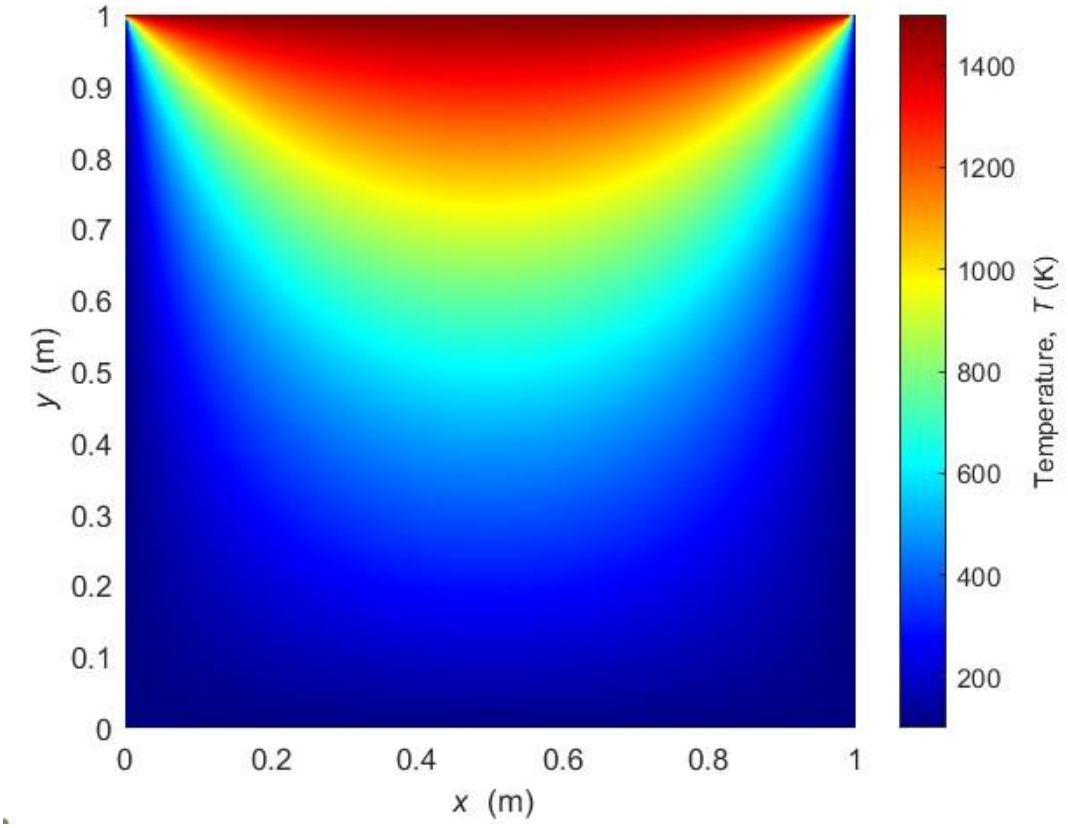

Figure 7. Steady Non-Linear Conductivity FEM Temperature Distribution (Hmax = 0.02)

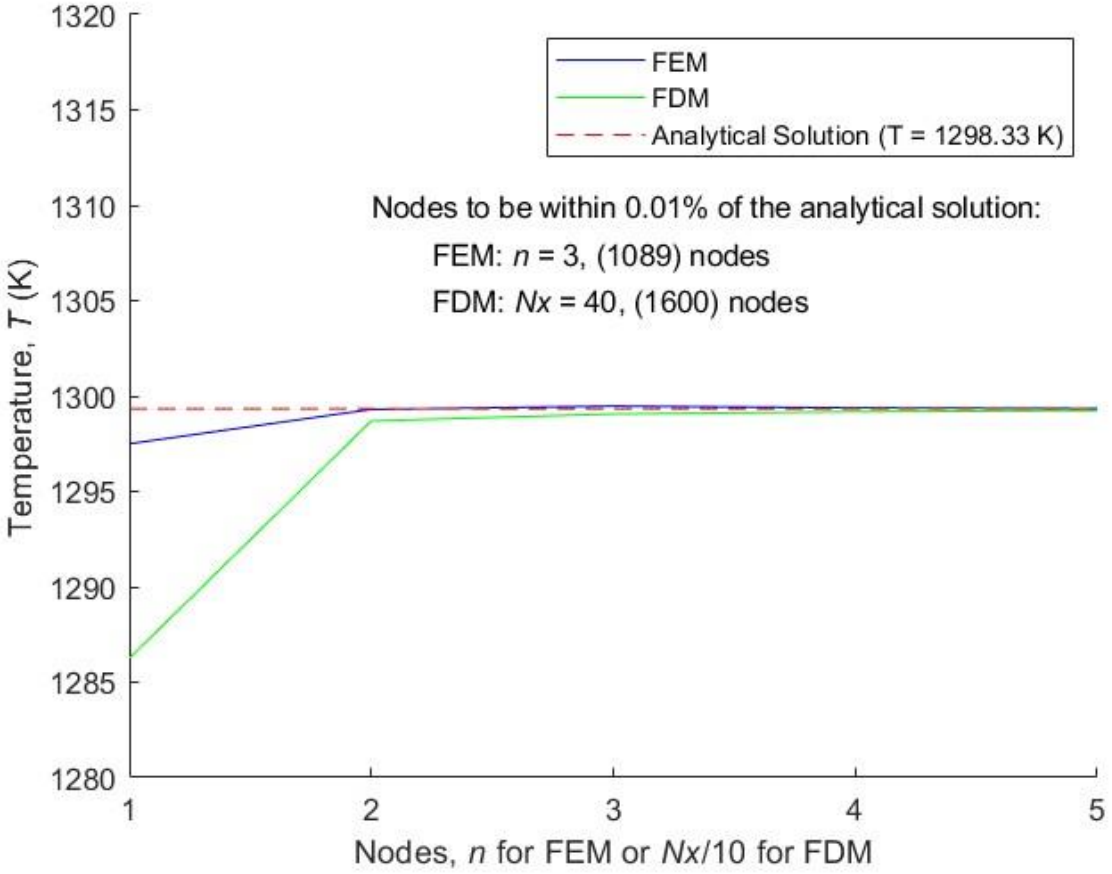

Figure 8. FEM and FDM Temperature Plotted Over Number of Nodes (for x = 0.5m, y = 0.9m)

4.

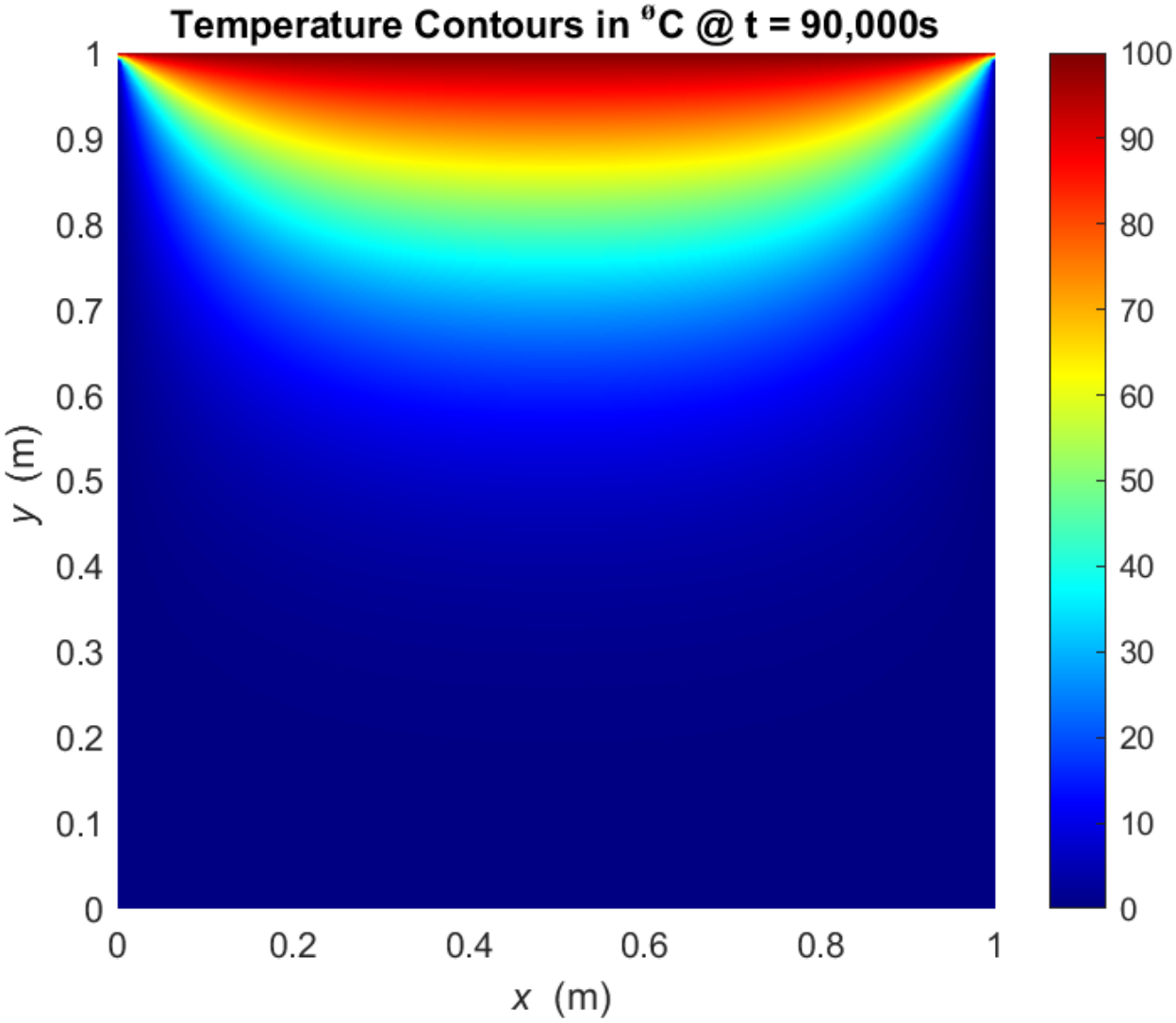

Figure 9. Transient FEM Temperature Distribution (Hmax = 0.02)

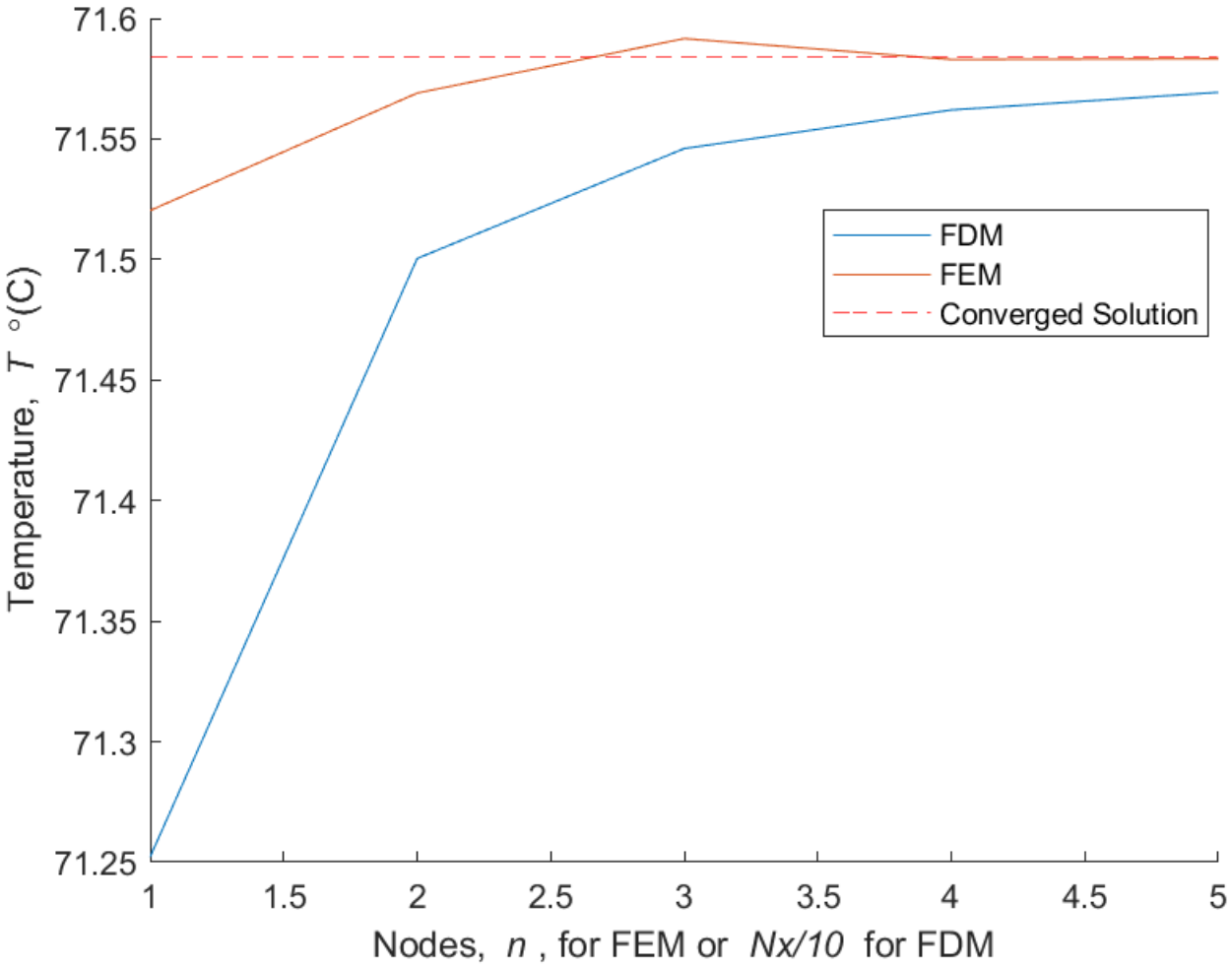

Figure 10. Transient FEM and FDM Solution vs Nodes (x = 0.5m, y = 0.9m, t = 90000s)

Using MATLAB's PDE toolbox, the transient temperature distribution of a brick can be found and is displayed in Figure 9. Previously, the same problem has been investigated in lab 3 pt 5. The solution found here correlates with solutions found previously. To compare the results of this analysis with previous methods, the temperature at location x = 0.5 m and y = 0.9 m was calculated with each method for a number of settings and displayed in Figure 10. For FEM, n = 3 was sufficient to be within 0.01% of the converged value. This corresponded with a value of 1432 elements. However, the FDM method was not able to be within 0.01% of the converged value up to Nx = 50 and thus 2500 elements was not enough to reach the converged value.

Raw Code for Part 2:

```
% ME554_Lab04_Part02.m
% Author: Christopher Ng, Russell Occhipinti
% Created: 4/23/24
% Last Modified: 4/29/24
clear, clc, close all
n = 5;
T_SS = ones(n,1)*16.8412;
%% FEM Model
T_FEM = zeros(n,1);
n_elem = zeros(n,1);
for i = 1:n
% Create PDE thermal model container
thermalmodel = createpde('thermal','steadystate');
% Create geometry for a rectangle, R1, where:
% The origin of the shape is located at x = 3, y = 4
% next 4 rows are x-coord. relative to the origin for points of edges
% final 4 rows are respective y-coord. relative to the origin for points of edges
Lx = 1/2; % m
Ly = 1; % m
R1 = [3;4; 0;Lx;Lx;0; 0;0;Ly;Ly];
gd = [R1]; % geometry description matrix
sf = 'R1'; % set formula
ns = char('R1')'; % name space matrix
dl = decsg(gd,sf,ns); % decomposed geometry matrix
% Add geometry to model
pg = geometryFromEdges(thermalmodel,dl);
% Plot geometry and display edge and face labels
w = 500; % pixels, width of figure windows
h = 400; % pixels, height of figure windows
off = 100; % pixels, offset between figure windows
% figure('Name','Geometry With Edge and Subdomain Labels',...
% 'Position',[off,off,w,h]);
% pdegplot(thermalmodel,'EdgeLabels','on','FaceLabels','on')
% axis equal
% Generate and plot mesh
Hmax = Lx/(2*i);
msh = generateMesh(thermalmodel,'Hmax',Hmax,'Hgrad',1.5);
Ne = length(msh.Elements);
% figure('Name',['Mesh with ',num2str(Ne),' Elements'],...
% 'Position',[2*off,2*off,w,h]);
% pdemesh(thermalmodel);
% axis equal tight
% Set boundary conditions
T1 = 0; % deg. C
T2 = 100; % deg. C
Q1 = 0; % W
thermalBC1 = thermalBC(thermalmodel,'Edge',[1 4],'Temperature',T1);
thermalBC2 = thermalBC(thermalmodel,'Edge', 3, 'Temperature',T2);
thermalBC3 = thermalBC(thermalmodel,'Edge', 2, 'HeatFlux',Q1);
% Alternatively, for specified heat flux use 'HeatFlux', for specified convection
use
% 'ConvectionCoeffient' and 'AmbientTemperature', and for specified radiation use
```

```
% 'Emissivity' and 'AmbientTemperature' along with their corresponding values.
% Specify thermal properties of the material.
k = 1.0;
thermalProperties(thermalmodel,'ThermalConductivity',k);
% Find the solution
thermalmodel.SolverOptions.ReportStatistics ='on';
thermalmodel.SolverOptions.RelativeTolerance = 1e-4;
results = solve(thermalmodel);
% Plot the solution
T = results.Temperature;
figure('Name',['Temperature Contours in',char(176),'C'],...
'Position',[3*off,3*off,w,h]);
pdeplot(thermalmodel,'XYData',T,'ColorMap','jet');
caxis([T1 T2]);
c = colorbar;
c.Label.String = 'Temperature, {\it T} ({\circ}C)';
axis([0 Lx 0 Ly]);
xlabel('\it x \rm (m)');
ylabel('\it y \rm (m)');
% Interpolate for temperature at each (x,y) coordinate
%N = 10; % number of locations
%x = Lx/2*ones(N+1,1); % x-coordinates
%y = [0:Ly/N:Ly]'; % y-coordinates
x = 0.4;
y = 0.4;
Tint = interpolateTemperature(results,x,y);
fprintf('T(x =%4.1f m, y =%4.1f m, Hmax =%4.1f) = %5.4f deg. C \n', ...
[x y Hmax Tint]');
T_FEM(i) = Tint;
n_elem(i) = length(msh.Nodes);
end
T_PE_FEM = abs(T_FEM-T_SS)./T_SS;
qv = 0.01/100;
T_extrap_FEM1 = interp1(T_PE_FEM,1:n,qv,'nearest','extrap');
T_extrap_FEM2 = interp1(T_PE_FEM,n_elem,qv,'nearest','extrap');
%% FDM Model
n_FDM = n;
T_FDM = zeros(n_FDM,1);
for idx = 1:n_FDM
% System Parameters
T1 = 0; % deg. Celsius
T2 = 100; % deg. Celsius
k = 1; % W/m-K, thermal conductivity
w = 1; % m, width
Lx = 0.5; % m
Ly = 1; % m
Nx = idx*5; % -, number of x nodes
dx = Lx/Nx; % m, increment in x-direction
dy = dx; % m, increment in y-direction
Ny = Ly/dy; % -, number of y nodes
Ny = floor(Ny);
x = 0:dx:(Lx*2);
y = 0:dy:Ly;
pmax = 1e9; % -, max number of iterations
tol = 1e-9; % deg. Celsius, tolerance
```

```
% Analytical Solution
T = ones(Nx+1,Ny+1);
T = T*(T1+T2)/2;
T(:,Ny+1) = T2;
T(:,1) = T1;
T(1,:) = T1;
T(1,Ny+1) = (T1+T2)/2;
res = zeros(Nx+1,Ny+1);
omega_opt = 1;
p = 0;
Res = tol+1;
while p <= pmax && Res >= tol
for j = 2:Ny
for i = 2:(Nx+1)
if i == (Nx+1)
T(i,j) = (1/4)*omega_opt*(2*T(i-1,j)+T(i,j-1)+T(i,j+1))+(1-omega_opt)*T(i,j);
else
T(i,j) = (1/4)*omega_opt*(T(i-1,j)+T(i+1,j)+T(i,j-1)+T(i,j+1))+(1-
omega_opt)*T(i,j);
end
end
end
for j = 2:Ny
for i = 2:(Nx+1)
if i == (Nx+1)
res(i,j) = abs((1/4)*(2*T(i-1,j)+T(i,j-1)+ ...
T(i,j+1))-T(i,j));
else
res(i,j) = abs((1/4)*(T(i-1,j)+T(i+1,j)+T(i,j-1)+ ...
T(i,j+1))-T(i,j));
end
end
end
Res = sum(res,"all")/sum(T,"all");
fprintf('Iter = %8.0d \n', p);
fprintf('Res = %4.2e \n', Res);
p = p+1;
if p == pmax
print('Code did not converge');
end
end
T_FDM(idx) = T((0.4+dx)/dx,(0.4+dy)/dy);
end
T_PE_FDM = abs(T_FDM-T_SS)./T_SS;
qv = 0.01/100;
T_extrap_FDM = interp1(T_PE_FDM,1:n,qv,'nearest','extrap');
%% Plotting
figure(6);
hold on;
plot(1:n,T_FEM,'b');
plot(1:n,T_FDM,'g');
plot(1:n,T_SS,'--r');
txt = ['Nodes to be within 0.01% of the analytical solution:'];
text(2,16.885,txt);
```

```
txt = ['FEM:{\it n} = ',num2str(T_extrap_FEM1), ', (',num2str(T_extrap_FEM2),')
nodes'];
text(2.25,16.88,txt);
txt = ['FDM:{\it Nx} = ',num2str(5*T_extrap_FDM), ',
(',num2str((5*T_extrap_FDM)^2),') nodes'];
text(2.25,16.875,txt);
legend('FEM', 'FDM', 'Analytical Solution (T = 16.8412)','Location','northeast');
xlabel('Nodes,{\it n} for FEM or{\it Nx}/5 for FDM')
ylabel('Temperature,{\it T} (C)')
xticks(1:5);
```

Raw Code for Part 3:

```
% ME554_Lab04_Part03.m
% Author: Christopher Ng, Russell Occhipinti
% Created: 4/27/24
% Last Modified: 4/29/24

clc;
clear;
close all;

thermalmodel = createpde('thermal','steadystate');

% Define geometry
Lx = 0.02;
Ly = 0.2;
R1 = [3;4;0;Lx;Lx;0;0;0;Ly;Ly];

gd = [R1];
sf = 'R1';
ns = char('R1')';
dl = decsg(gd,sf,ns);

% Add geometry to model
pg = geometryFromEdges(thermalmodel,dl);

% Plot geometry and display edge and face labels
w = 200;
h = 1000;
off = 100;
figure('Name','Geometry With Edge and Subdomain Labels',...
     'Position',[off,off,w,h]);
pdegplot(thermalmodel,'EdgeLabels','on','FaceLabels','on');
axis equal

% Generate and plot mesh
n_h = 1;
h_max = Lx/(2*n_h);
msh = generateMesh(thermalmodel,'Hmax',h_max,'Hgrad',1.5);
Ne = length(msh.Elements);
figure('Name',['Mesh with' num2str(Ne),'Elements'],'Position',[2*off,2*off,w,h]);
pdemesh(thermalmodel);
axis equal tight
```

```
% Set boundary conditions
T1 = 100;
T2 = 200;
q = 0;
h_conv = 500;    % W/m^2K

thermalBC1 = thermalBC(thermalmodel,'Edge',1,'Temperature',T2);
thermalBC2 =
thermalBC(thermalmodel,'Edge',[3,4],'ConvectionCoefficient',h_conv,'AmbientTemperat
ure',T1);
thermalBC3 = thermalBC(thermalmodel,'Edge',2,'HeatFlux',q);

% Specific thermal properties
k = 50; %W/m*K
thermalProperties(thermalmodel,'ThermalConductivity',k);

% Find solution
thermalmodel.SolverOptions.ReportStatistics ='on';
thermalmodel.SolverOptions.RelativeTolerance = 1e-4;
results = solve(thermalmodel);

% Plot
% Plot the solution
T = results.Temperature;
figure('Name',['Temperature Contours in',char(176),'C'],...
 'Position',[3*off,3*off,w,h]);
pdeplot(thermalmodel,'XYData',T,'ColorMap','jet');
caxis([T1 T2]);
axis([0 Lx 0 Ly]);
xlabel('\it x \rm (m)');
ylabel('\it y \rm (m)');
c = colorbar;
c.Label.String = 'Temperature ({\circ}C)';

% Interpolate for temperature at each (x,y) coordinate
N = 20; % number of locations
x = Lx/2*ones(N+1,1); % x-coordinates
y = [0:Ly/N:Ly]'; % y-coordinates
Tint = interpolateTemperature(results,x,y);
fprintf('T(x =%4.1f m, y =%4.1f m) = %5.1f deg. C \n', ...
 [x y Tint]');

%% Plot temp at x = y = 0.01 vs total # of nodes
T_FEM = [   181.097518639079;
            181.091227226923;
            181.091403153014;
            181.091293279039;
            181.091065818797];

T_FDM = [   181.132167404199;
            181.101631425093;
            181.095760749674;
            181.093688303270;
            181.092723713212];
```

```
figure;
hold on;
plot([1 2 3 4 5],T_FDM)
plot([1 2 3 4 5],T_FEM)
yline(181.091,"--r");
legend("FDM","FEM","Converged Solution")

ylim([181.09 181.14])
xlabel('Nodes, \it n \rm, for FEM or \it Nx/4 \rm for FDM');
ylabel('Temperature, \it T \rm {\circ}(C)');
```

Raw Code for Part 4:

```
% ME554_Lab04_Part04.m
% Author: Christopher Ng, Russell Occhipinti
% Created: 4/23/24
% Last Modified: 4/29/24
clear, clc, close all
n = 5;
T_SS = ones(n,1)*1299.33;
%% FEM Model
% Plot geometry and display edge and face labels
w = 500; % pixels, width of figure windows
h = 400; % pixels, height of figure windows
off = 100; % pixels, offset between figure windows
T_FEM = zeros(n,1);
n_elem = zeros(n,1);
for i = 1:n
% Create PDE thermal model container
thermalmodel = createpde('thermal','steadystate');
% Create geometry for a rectangle, R1, where:
% The origin of the shape is located at x = 3, y = 4
% next 4 rows are x-coord. relative to the origin for points of edges
% final 4 rows are respective y-coord. relative to the origin for points of edges
Lx = 1; % m
Ly = 1; % m
R1 = [3;4; 0;Lx;Lx;0; 0;0;Ly;Ly];
gd = [R1]; % geometry description matrix
sf = 'R1'; % set formula
ns = char('R1')'; % name space matrix
dl = decsg(gd,sf,ns); % decomposed geometry matrix
% Add geometry to model
pg = geometryFromEdges(thermalmodel,dl);
% Generate and plot mesh
Hmax = Lx/(5*i);
msh = generateMesh(thermalmodel,'Hmax',Hmax,'Hgrad',1.5);
Ne = length(msh.Elements);
% Set boundary conditions
```

```
T1 = 100; % deg. K
T2 = 1500; % deg. K
thermalBC1 = thermalBC(thermalmodel,'Edge',[1 2 4],'Temperature',T1);
thermalBC2 = thermalBC(thermalmodel,'Edge', 3, 'Temperature',T2);
% Specify thermal properties of the material.
k = @k_fun;
thermalProperties(thermalmodel,'ThermalConductivity',k);
% Find the solution
thermalmodel.SolverOptions.RelativeTolerance = 1e-9;
thermalmodel.SolverOptions.ReportStatistics ='on';
results = solve(thermalmodel);
% Plot the solution
T = results.Temperature;
% figure('Name',['Temperature Contours in',char(176),'K'],...
% 'Position',[3*off,3*off,w,h]);
% pdeplot(thermalmodel,'XYData',T,'ColorMap','jet');
% caxis([T1 T2]);
% c = colorbar;
% c.Label.String = 'Temperature, {\it T} (K)';
% axis([0 Lx 0 Ly]);
% xlabel('\it x \rm (m)');
% ylabel('\it y \rm (m)');
x = 0.5;
y = 0.9;
Tint = interpolateTemperature(results,x,y);
fprintf('T(x =%4.1f m, y =%4.1f m, Hmax =%4.4f) = %5.4f K \n', ...
[x y Hmax Tint]');
T_FEM(i) = Tint;
n_elem(i) = length(msh.Nodes);
end
T_PE_FEM = abs(T_FEM-T_SS)./T_SS;
qv = 0.01/100;
T_extrap_FEM1 = interp1(T_PE_FEM,1:n,qv,'nearest','extrap');
T_extrap_FEM2 = interp1(T_PE_FEM,n_elem,qv,'nearest','extrap');
%% FDM Model
% System Parameters
T1 = 100; % deg. Kelvin
T2 = 1500; % deg. Kelvin
w = 1; % m, width
Lx = 1; % m
Ly = 1; % m
pmax = 1e9; % -, max number of iterations
tol = 1e-9; % deg. Celsius, tolerance
T_FDM = zeros(n,1);
for idx = 1:n
Nx = idx*10; % -, number of x nodes
dx = Lx/Nx; % m, increment in x-direction
dy = dx; % m, increment in y-direction
Ny = Ly/dy; % -, number of y nodes
Ny = floor(Ny);
x = 0:dx:Lx;
y = 0:dy:Ly;
T_k = ones(Nx+1,Ny+1);
T_k = T_k*(T1+T2)/2;
T_k(:,Ny+1) = T2;
```

```
T_k(:,1) = T1;
T_k(1,:) = T1;
T_k(Nx+1,:) = T1;
T_k(1,Ny+1) = (T1+T2)/2;
T_k(Nx+1,Ny+1) = (T1+T2)/2;
k = k_SS(T_k);
omega = 0.25;
res = zeros(Nx+1,Ny+1);
p = 0;
Res = tol+1;
while p <= pmax && Res >= tol
for j = 2:Ny
for i = 2:Nx
k(i,j) = k_SS(T_k(i,j));
k_r = (1/2)*(k(i,j)+(1/4)*(k(i-1,j)+k(i+1,j)+k(i,j-1)+k(i,j+1)));
T_k(i,j) = (1/4)*omega*((k(i,j)+k(i-1,j))/(2*k_r)* ...
T_k(i-1,j)+(k(i,j)+k(i+1,j))/(2*k_r)*T_k(i+1,j)+(k(i,j) ...
+k(i,j-1))/(2*k_r)*T_k(i,j-1)+(k(i,j)+k(i,j+1))/ ...
(2*k_r)*T_k(i,j+1))+(1-omega)*T_k(i,j);
end
end
for j = 2:Ny
for i = 2:Nx
k(i,j) = k_SS(T_k(i,j));
k_r = (1/2)*(k(i,j)+(1/4)*(k(i-1,j)+k(i+1,j)+k(i,j-1)+k(i,j+1)));
res(i,j) = abs((1/4)*((k(i,j)+k(i-1,j))/(2*k_r)* ...
T_k(i-1,j)+(k(i,j)+k(i+1,j))/(2*k_r)*T_k(i+1,j)+(k(i,j) ...
+k(i,j-1))/(2*k_r)*T_k(i,j-1)+(k(i,j)+k(i,j+1))/ ...
(2*k_r)*T_k(i,j+1))-T_k(i,j));
end
end
Res = sum(res,"all")/sum(T_k,"all");
%fprintf('Iter = %8.0d \n', p);
%fprintf('Res = %4.2e \n', Res);
p = p+1;
if p == pmax
print('Code did not converge');
end
end
% Nc = 20;
% dT = (T2-T1)/Nc;
% v = T1:dT:T2;
% figure(1);
% colormap(jet)
% contourf(x,y,T_k',v,'LineStyle','none');
% c = colorbar;
% c.Label.String = 'Temperature, {\it T} (K)';
% Tmax = max(T1,T2);
% Tmin = min(T1,T2);
% clim([Tmin, Tmax]);
% axis equal tight;
% %title('Table 1. Temperature data for two-dimensional steady state conduction
with fixed temperature boundary conditions.')
% xlabel('X Position, {\it x} (m)')
% ylabel('Y Position, {\it y} (m)')
```

```
T_FDM(idx) = T_k(floor((0.5+dx)/dx),floor((0.9+dy)/dy));
end
T_PE_FDM = abs(T_FDM-T_SS)./T_SS;
qv = 0.01/100;
T_extrap_FDM1 = interp1(T_PE_FDM,(1:n)*10,qv,'nearest','extrap');
T_extrap_FDM2 = interp1(T_PE_FDM,((1:n)*10).^2,qv,'nearest','extrap');
%% Plotting
figure(6);
hold on;
plot(1:n,T_FEM,'b');
plot(1:n,T_FDM,'g');
plot(1:n,T_SS,'--r');
txt = ['Nodes to be within 0.01% of the analytical solution:'];
text(2,1310,txt);
txt = ['FEM:{\it n} = ',num2str(T_extrap_FEM1), ', (',num2str(T_extrap_FEM2),')
nodes'];
text(2.25,1307.5,txt);
txt = ['FDM:{\it Nx} = ',num2str(T_extrap_FDM1), ', (',num2str(T_extrap_FDM2),')
nodes'];
text(2.25,1305,txt);
legend('FEM', 'FDM', 'Analytical Solution (T = 1298.33 K)','Location','northeast');
xlabel('Nodes,{\it n} for FEM or{\it Nx}/10 for FDM')
ylabel('Temperature,{\it T} (K)')
xticks(1:n);
axis([1,n,1280,1320]);
```

Raw Code for Part 5:

```
% ME554_Lab04Part05.m
% Author: Christopher Ng, Russell Occhipinti
% Created: 4/29/24
% Last Modified: 4/29/24

clc;
clear;
close all;

thermalmodel = createpde('thermal','transient');

% Define geometry
Lx = 1;
Ly = 1;
R1 = [3;4;0;Lx;Lx;0;0;0;Ly;Ly];

gd = [R1];
sf = 'R1';
ns = char('R1')';
dl = decsg(gd,sf,ns);

% Add geometry to model
pg = geometryFromEdges(thermalmodel,dl);

% Plot geometry and display edge and face labels
w = 500;
```

```
h = 400;
off = 100;
figure('Name','Geometry With Edge and Subdomain Labels',...
     'Position',[off,off,w,h]);
pdegplot(thermalmodel,'EdgeLabels','on','FaceLabels','on');
axis equal

% Generate and plot mesh
n_h = 5;
h_max = Lx/(5*n_h);
msh = generateMesh(thermalmodel,'Hmax',h_max,'Hgrad',1.5);
Ne = length(msh.Elements);
figure('Name',['Mesh with' num2str(Ne),'Elements'],'Position',[2*off,2*off,w,h]);
pdemesh(thermalmodel);
axis equal tight

% Set boundary conditions
T1 = 0;
T2 = 100;
thermalBC1 = thermalBC(thermalmodel,'Edge',[1 2 4],'Temperature',T1);
thermalBC2 = thermalBC(thermalmodel,'Edge',3,'Temperature',T2);

% Specify initial temperature
Tinit = 0; % deg. C
thermalIC(thermalmodel, Tinit);

% Specific thermal properties
k = 0.72; % W/m-K
rho = 1920; % kg/m^3
c = 835; % J/kg-K
thermalProperties(thermalmodel,'ThermalConductivity',k,...
 'MassDensity',rho,...
'SpecificHeat',c);

% Find solution at Nt points in time between 0 and tmax
Nt = 10;
tmax = 90000; % s
tlist = 0:tmax/Nt:tmax;
thermalmodel.SolverOptions.ReportStatistics ='on';
results = solve(thermalmodel, tlist);

% Plot
% Plot the solution
figure;
T = results.Temperature;
for j = 1:Nt
 pdeplot(thermalmodel,'XYData',T(:,j),'ColorMap','jet');
 caxis([T1 T2]);
 axis([0 Lx 0 Ly]);
 xlabel('\it x \rm (m)');
 ylabel('\it y \rm (m)');
 title('Temperature Contours in ^oC @ t = 90,000s')
 Mv(j) = getframe;
end
```

```
%% Interpolate for temperature at (x,y) coordinate and time step iT
xtest = 0.5; % m, x-coordinates
ytest = 0.9; % m, y-coordinates
iT = 11; % index for time step
Tint = interpolateTemperature(results,xtest,ytest,iT);
fprintf('T(x =%4.1f m, y =%4.1f m, t =%4.2f s) = %5.4f deg. C \n', ...
 xtest, ytest, (iT-1)*tmax/Nt, Tint);

%% Plot temps at x = 0.5

T_PDE = [   71.5203;
            71.5690;
            71.5916;
            71.5829;
            71.5832];

T_FDM = [   71.2522;
            71.5004;
            71.5460;
            71.5620;
            71.5693];

figure;
hold on;
plot([1 2 3 4 5],T_FDM)
plot([1 2 3 4 5],T_PDE)
yline(71.5840,"--r");
legend("FDM","FEM","Converged Solution")
ylim([71.2500 71.6000])
xlabel('Nodes, \it n \rm, for FEM or \it Nx/10 \rm for FDM');
ylabel('Temperature, \it T \rm {\circ}(C)');
```