CHEN 3600 Computer Aided Chemical Engineering

Department of Chemical Engineering

Auburn University, AL 36849

**MEMORANDUM**

**Date:** October 8, 2019

**To:**

**From:** Hazl Torres, Chemical Engineer

**Subject:** Calculating and creating temperature profiles in cartesian and cylindrical coordinates.

The main purpose of this project is to calculate the rate of change of temperature and create a temperature profile at steady state using Euler’s method. This project involves heat transfer in both cartesian and cylindrical coordinates, creating 3D temperature profiles, and determining numerical instability.

**Attachments:**

Attachment 1 – Problem 1

Attachment 2 – Problem 2

Attachment 3 – Problem 3

**Attachment 1 – Problem 1**

**Problem:** A 0.50m by 0.75m copper plate at 25°C with the four sides initially held at temperatures:

At x = 0.00m:

At y = 0.00m:

At x = 0.50m: 6

At y = 0.75m:

The equation for two dimensional heat conduction: , where

= 8960 kg/m3 , Cp = 385 J/kg K, k =84.8 J/m K s

Using Euler’s Method, calculate how long it takes to reach steady state and create a 3D plot of the plate’s steady state temperature profile.

**My Solution:**

General Algorithm:

1. Initialize problem variables
   1. Define parameters given in problem statement
   2. Define parameters that are choices
   3. Initialize temperatures
2. Calculate steady state temperatures and time
   1. Apply boundary conditions for temperature
   2. Calculate temperature and time
      1. Define step size
      2. Define steady state tolerance
      3. Initialize temperature change matrix and time
      4. Solve for change in temperature using Euler’s Method
         1. Multiply change in time by (k/*p* Cp ) times the sum of second derivatives with respect to point x and point y (written as finite difference between points in a grid)
      5. Stop if at steady state
      6. If not at steady state, T = T + dT and t = t +dt. Repeat b.
3. Report Results
   1. Print the total time to reach steady state
   2. Plot

MATLAB Code:

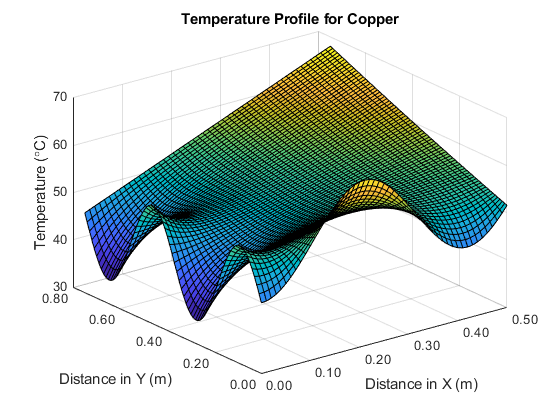
clc, clear all  
k = 84.8;  
p = 8960;  
cp = 385;  
a = k/(cp\*p);  
  
Lx = 0.5; % m  
Ly = 0.75; % m  
T0 = 25; % C  
t0 = 0;  
  
y = linspace(0,Ly,76);  
x = linspace(0,Lx,51);  
dy = y(2)-y(1);  
dx = x(2)-x(1);  
  
T = T0\*ones(76,51); % rows in y cols in x  
  
% apply boundary conditions  
for m = 1:76  
 T(m,1) = 45+10\*sin(17\*y(m)); % at x=0  
 T(m,51) = 2.5\*y(m)^2 + 20\*y(m) + 51.5336; % at x = 0.5  
end  
for n = 1:51  
 T(1,n) = 55-10\*cos(15\*x(n)); % at y = 0  
 T(76,n) = 42.2277\*x(n) + 46.8260; % at y = 0.75  
end  
[Temp, time] = steady(T, dx,dy);  
fprintf('Reaching steady state took %.2f seconds\n', time)  
surf(x,y,Temp)  
title('Temperature Profile for Copper');  
xlabel('Distance in X (m)');  
ylabel('Distance in Y (m)');  
zlabel('Temperature (\circC)');  
xtickformat('%0.2f');  
ytickformat('%0.2f');  
ax = gca;  
ax.TickLength = [0,0];

function [T,t] = steady(T,dx,dy)  
k = 84.8;  
p = 8960;  
cp = 385;  
a = k/(cp\*p);  
  
  
%%%%%%  
dt = 0.1; % s  
ss = 0.1/60;  
ss = dt\*ss; % tolerance  
t = 0;  
dT = zeros(76,51);  
steadystate=false;  
while ~steadystate  
 steadystate = true;  
 for i = 2:75  
 for j = 2:50  
 dT(i,j) = a\*dt\*(((T(i,j+1) + T(i,j-1) - 2\*T(i,j))/dx^2) + ...  
 ((T(i+1,j) + T(i-1,j) - 2\*T(i,j))/dy^2));  
 if abs(dT(i,j))> ss  
 steadystate = false;  
 end  
 if steadystate  
 continue  
 end  
 end  
 end  
T = T + dT;  
t = t + dt;  
end

[*Published with MATLAB® R2018b*](https://www.mathworks.com/products/matlab)

**Answer:**

Reaching steady state took 2614.50 seconds



[*Published with MATLAB® R2018b*](https://www.mathworks.com/products/matlab)

**Conclusions:**

This answer is reasonable because all of the temperatures are in the range of the minimum initial temperature of 25°C and the maximum temperature, which was set by the initial conditions. The shape of the plot results from temperatures initially being a function of sin or cos, and then varying linearly as it approaches the ends of the plate. Furthermore, the color change represents the temperature gradient. As the distance in x and y increase, the temperature increases, which can be seen as the color comes closer to orange/yellow. Furthermore, the time it took to reach steady state was 2614.50 seconds, or about 43.5 minutes, which is a reasonable amount of time.

In this problem, heat transfer is related to the change in temperature over the change in time, which is dependent on the second derivatives of temperature with respect to x and y. These second derivatives were found by representing the temperatures as a grid and finding the finite difference between points. Then, Euler’s method was applied to find the temperatures throughout the plate by calculating the changes in temperature and adding it to the previous temperature. Furthermore, steady state is a condition in which there is no change in temperature with respect to time. Because computers cannot calculate this, a tolerance was set to determine when the changes in temperature were small enough to be considered steady state.

**Attachment 2 – Problem 2**

**Problem:** Using the same system as problem 1 and a step size of 1cm between points in the x and y directions

What is the smallest step size that causes numerical instability?

**My Solution:**

General Algorithm:

1. Redo Problem 1 using step sizes of 1cm for x and y to get the steady state temperatures
2. Redo Problem 1 using a different change in time
   1. Initialize dt so that the steady state temperature and the temperature for unsteady state are equal
   2. Calculate the new temperatures with this dt
      1. If the differences between steady state temperatures and these temperatures is below tolerance, increase dt
      2. Repeat b
   3. Stop when the differences between steady state temperatures and these temperatures are above tolerance
3. Report results
   1. Print the time step that caused numerical instability
   2. Plot

MATLAB Code:

## Problem 2 - Redo Problem 1

clc, clear all  
k = 84.8;  
p = 8960;  
cp = 385;  
a = k/(cp\*p);  
  
Lx = 0.5; % m  
Ly = 0.75; % m  
T0 = 25; % C  
t0 = 0;  
  
y = linspace(0,Ly,76);  
x = linspace(0,Lx,51);  
dy = y(2)-y(1);  
dx = x(2)-x(1);  
  
T = T0\*ones(76,51); % rows in y cols in x  
% apply boundary conditions  
for m = 1:76  
 T(m,1) = 45+10\*sin(17\*y(m)); % at x=0  
 T(m,51) = 2.5\*y(m)^2 + 20\*y(m) + 51.5336; % at x = 0.5  
end  
for n = 1:51  
 T(1,n) = 55-10\*cos(15\*x(n)); % at y = 0  
 T(76,n) = 42.2277\*x(n) + 46.8260; % at y = 0.75  
end

## Problem 2

function [T,dt] = stability(T,dx,dy,dt)  
k = 84.8;  
p = 8960;  
cp = 385;  
a = k/(cp\*p);  
  
  
%%%%%%  
ss = 0.1/60;  
ss = dt\*ss; % tolerance  
t = 0;  
dT = zeros(76,51);  
steadystate=false;  
while ~steadystate  
 steadystate = true;  
 for i = 2:75  
 for j = 2:50  
 dT(i,j) = a\*dt\*(((T(i,j+1) + T(i,j-1) - 2\*T(i,j))/dx^2) + ...  
 ((T(i+1,j) + T(i-1,j) - 2\*T(i,j))/dy^2));  
 if abs(dT(i,j))> ss  
 steadystate = false;  
 end  
 if steadystate  
 continue  
 end  
 end  
 end  
T = T + dT;  
end

[*Published with MATLAB® R2018b*](https://www.mathworks.com/products/matlab)

[T\_stable] = steady(T,dx,dy);  
dt = 0.1; % initial  
T\_unstable = T\_stable; % initial  
% change dt until unstable  
while abs(T\_unstable-T\_stable) < 1  
% changed function to make dt an input and output  
[T\_unstable, dt] = stability(T,dx,dy,dt);  
dt = dt+0.01;  
T\_unstable = T\_unstable;  
end  
dt = dt - 0.01;  
fprintf('The smallest step size causing instability is %0.2f seconds\n', dt)  
  
  
surf(T\_unstable)  
surf(x,y,T\_unstable)  
title('Unstable Temperature Profile for Copper');  
xlabel('Distance in X (m)');  
ylabel('Distance in Y (m)');  
zlabel('Temperature (\circC)');  
xtickformat('%0.2f');  
ytickformat('%0.2f');  
ax = gca;  
ax.TickLength = [0,0];

**Answer:**

The smallest step size causing instability is 1.02 seconds

A close up of a map

Description automatically generated

[*Published with MATLAB® R2018b*](https://www.mathworks.com/products/matlab)

**Conclusions.**

In this problem, the smallest time step that caused numerical instability was 1.02 seconds. This answer is reasonable because the plot for using this time step is very different from the steady state plot, but the outline of this plot is the same. This is due to the boundary conditions being specified as the same thing for both time steps. However, because of the numerical instability, MATLAB® could not calculate the temperature for the interior points, so the plot is blank inside of the boundary. Although the step sizes for time and space are independent of one another, they have an effect on each other that can cause numerical instability when the time step is too large for the spatial steps.

**Attachment 3 – Problem 3**

**Problem:** A circular iron disk with a radius of 30cm at 25°C. The ends of the disk are initially at °C

The equation for two dimensional heat conduction: , where

= 7874 kg/m3 , Cp = 450 J/kg K, k =83.5 J/m K s

Using Euler’s Method, calculate how long it takes to reach steady state and plot the temperature profile.

**My Solution:**

General Algorithm:

1. Initialize problem variables
   1. Define parameters given in problem statement
   2. Define parameters that are choices
   3. Convert r and theta to cartesian coordinates
   4. Initialize temperatures
      1. Initialize temperature at the ends of the disk (given)
      2. Initialize temperature when radius is 0cm
2. Calculate steady state temperatures and time
   1. Apply boundary conditions for temperature
   2. Calculate temperature and time
      1. Define step size
      2. Define steady state tolerance
      3. Initialize temperature change matrix
      4. Initialize time
      5. Solve for change in temperature using Euler’s Method
         1. Multiply change in time by (k/*p* Cp ) times the right hand derivatives of the given equation (written as the finite difference between points in a grid)
            1. If theta is zero (at point 1), the grid point corresponding to the point before it is the same as the grid point to the right of it
            2. If theta is (point at the end), the grid point corresponding to the point after it is the same as the as the point after where theta is 0
      6. Stop if at steady state
      7. If not at steady state, T = T + dT and t = t +dt. Repeat b.
3. Report Results
   1. Print the total time to reach steady state
   2. Plot

MATLAB Code:

% conductive heat transfer for iron, cylindrical coordinates  
clc, clear all  
  
Lr = 0.30; % m  
T0 = 25; % C  
p = 7874;  
cp = 450;  
k = 83.5;  
a = k/(p\*cp);  
  
r = linspace(0,Lr,31);  
theta = linspace(0,2\*pi,100);  
dr = r(2) - r(1);  
d\_th = theta(2) - theta(1);  
for e = 1:length(r)  
 X(e,:) = e.\*cos(theta);  
 Y(e,:) = e.\*sin(theta);  
end  
T = T0\*ones(length(r), length(theta));  
% boundary condition  
T(end,:) = 52+10\*cos(3\*theta); %C  
T(1,:) = 52; % when theta is 0  
% find T profile and time  
dt = 0.01; % s  
ss = 0.01;  
ss = dt\*ss; % tolerance  
t = 0;  
dT = zeros(length(r), length(theta));  
steadystate=false;  
while ~steadystate  
 steadystate = true;  
 for i = 30:-1:2  
 for j = 1:length(theta)  
 if j == 1  
 dT(i,j) = a\*dt\*((((r(i))^2))\*((T(i,j+1) + T(i,2) - 2\*T(i,j))/d\_th^2) + ...  
 ((T(i+1,j) + T(i-1,j) - 2\*T(i,j))/(dr^2))+...  
 (dr/(2\*(dr^2)\*r(i)))\*((T(i+1,j)-T(i-1,j))));  
 elseif j == length(theta)  
 dT(i,j) = a\*dt\*((((r(i))^2))\*((T(i,2) + T(i,j-1) - 2\*T(i,j))/d\_th^2) + ...  
 ((T(i+1,j) + T(i-1,j) - 2\*T(i,j))/(dr^2))+...  
 (dr/(2\*(dr^2)\*r(i)))\*((T(i+1,j)-T(i-1,j))));  
 else dT(i,j) = a\*dt\*((((r(i))^2))\*((T(i,j+1) + T(i,j-1) - 2\*T(i,j))/d\_th^2) + ...  
 ((T(i+1,j) + T(i-1,j) - 2\*T(i,j))/(dr^2))+...  
 (dr/(2\*(dr^2)\*r(i)))\*((T(i+1,j)-T(i-1,j))));  
 end  
 if abs(dT(i,j))> ss  
 steadystate = false;  
 end  
 if steadystate  
 continue  
 end  
  
 end  
 end  
T = T + dT;  
t = t + dt;  
end  
fprintf('Reaching steady state took %.2f seconds\n', t)  
surf(X,Y,T)  
title('Temperature Profile of a Disk');  
xlabel('Distance from Center (cm)');  
ylabel('Distance from Center (cm)');  
zlabel('Temperature (\circC)');  
xtickformat('%0.2f');  
ytickformat('%0.2f');  
ax = gca;  
ax.TickLength = [0,0];

**Answer:**

Reaching steady state took 1153.03 seconds

A close up of a logo

Description automatically generated

[*Published with MATLAB® R2018b*](https://www.mathworks.com/products/matlab)

**Conclusions:**

The time it took to reach steady state was 1153.03 seconds, or about 19.2 minutes, which is a reasonable amount of time. This answer is also reasonable because all of the temperatures are in the range of the minimum initial temperature of 25°C and the maximum temperature defined by the boundary condition. The shape and color of the plot results from the initial temperature being a function of cosine. The color changes represent the temperature gradient.

Because solving for derivatives involves using points on a grid and the right hand side of the given equation contains and , it becomes difficult to solve for the temperature change when the radius is 0. Therefore, the problem was simplified by using the initial boundary condition to state that the temperature was 52 °C at r = 0 cm. The temperatures could be calculated using the finite difference when theta was at the first or last point. This is because the variable theta is an angle; therefore, the cosine of a point to the left of theta is equal to that of a point an equal distance away to the right.

Overall, the same method was used as in problem 1 to calculate the change in temperature, temperature, and time. The difference between cylindrical coordinates and cartesian coordinates was in calculating the derivatives on the right hand side of the equation in terms of points on a grid. After this, solving the problem was the same as problem 1, and, Euler’s method was applied to find the temperatures throughout the plate by calculating the changes in temperature and adding it to the previous value. A tolerance was again used to determine if the system was at steady state.