Programming Project1

EAS230 Engineering Computations
Spring 2018

Joshua Emerling   50104912   Lab: H2
Drew Wentka   50110060   Lab: H2

Table of Contents

List of Tables and Figures

1. Summary/Abstract

The objective of this project was to calculate the temperature at different node locations on a fin as well as Rate of Heat Transfer and Fin Efficiency. The purpose of the second problem was to be able to calculate thermal conductivity given an alloy name as well as temperature(s). The function is able to handle an input of a scalar, vector or matrix. The third problem used both the first and second problems to keep calculating thermal conductivity and fin temperatures until the values of temperature were within a tolerance of 1E-8. Our approach consisted of: For problem 1 calculating the coefficients for the exterior nodes first, since they used different equations. Then using a loop to calculate the ones for the interior nodes. This made scaling our solution easy as we only need to change one variable node_limit in order to scale the problem. For problem 2 we used a switch case for each alloy name then check if the given temperature value(s) were valid. For Problem 3 we started with an initial value for the vector of T values. We then inside a loop passed this to the ThCond function to find thermal conductivity. Next we repeated the steps in Problem 1 to solve the system of equations. We found the error by taking the difference between the new and old T values. If it was bigger than our threshold we repeated by passing the new T values to ThCond to find a new k. Finally, we computed a k value for each temperature value. We concluded that Copper had the highest Rate of Heat Transfer and the best Fin Efficiency. Both the numerical and analytical solutions produced very similar results.

2. Introduction

The problem to be solved was to find properties of Fin given the type of metal it was and its size. We used 2 approaches to solve this problem the first was to create a system of equations using the different equations of each T value and augmenting this matrix with the left side of each equation. After we used the rref function to solve for the T values. These Tvalues were used to compute Rate of Heat Transfer and Fin Efficiency. This was the numerical solution. The other solution for this was to use the modified Bessel function and initial values for T to calculate the Temperature at each node location, Rate of Heat Transfer and Fin Efficiency. This was the analytical solution. The purpose and application of this would be to show how changing different constants of the fin such as length, width, thickness and metal type would affect both the Rate of Heat Transfer and Fin Efficiency.

3. Deliverables

3.1 **Part 1**

3.1.1 **Methods:**

Script Name: PP1P1.m

Description:

Values in () represent corresponding variable name in the script.

This script calculates the temperature of each node on a fin given the number of nodes (node_limit), an initial temperature (T0), T infinity (Tinf) ,length (L), coefficient of heat transfer (h), Thermal Conductivity (k) and the thickness of the base (base_thickness), by using both the numerical and analytical solutions. It then calculates the rate of heat transfer (Qfin) and fin efficiency(nfin) for both solutions.

Pseudocode:

Define all given constants

Calculate theata and delta_x by delta_x = L/(node_limit-1) and theta=atan((base_thickness/2)/L);

Initialize A to a node_limit X node_limit of all zeros

Initialize b to a column vector of length node_limit

Set values of row 1 of A and b

A(1,1) = 1;

b(1,1) = T0;

Set values of row node_limit of A and b

A(node_limit,node_limit-1) = 1;

delta_x_ex = delta_x;

A(node_limit,node_limit) = -1*(1+(h*delta_x/(k*sin(theta))));

b(node_limit,1) = -1*((h*delta_x_ex/(k*sin(theta)))) * Tinf;

for m =2 to m = node_limit -1

calculate the previous, current and next coefficients of T with the following formulas and assign the appropriate values of row m of A and b.

coff_T_prev = 1-(m-(1/2))*delta_x/L;

coff_T = -1*((2-2*m*delta_x/L) + h*delta_x^2/(k*L*sin(theta)));

coff_T_next = 1-((m+(1/2))*delta_x/L);

b(m,1) = -1*(h*delta_x^2/(k*L*sin(theta)))*Tinf;

Solve T values using rref([A b])

Calculate numerical Qfin and nfin

Create a vector of node locations using delta_x

%Analytical solution

Use given formulas to calculate T values, Qfin_Ana and nfin_Ana by the analytical solutions

Print out Qfin, nfin, Qfin_Ana and nfin_Ana

Plot the numerical T values and analytical values vs each node location.

3.1.2 **Results:**

1)

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0.7 & -1.2 & 0.5 & 0 & 0 & 0 \\ 0 & 0.5 & -0.8 & 0.3 & 0 & 0 \\ 0 & 0 & 0.3 & -0.4 & 0.1 & 0 \\ 0 & 0 & 0 & 0.1 & -0.002 & -0.1 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix} \begin{bmatrix} T_1 \\ T_2 \\ T_3 \\ T_4 \\ T_5 \\ T_6 \end{bmatrix} = \begin{bmatrix} 200 \\ -0.0419 \\ -0.0419 \\ -0.0419 \\ -0.0419 \\ 0.2094 \end{bmatrix}$$

2) Solved using rref([A b])

T_values =

200.0000

198.5602

197.1259

195.6964

194.2670

192.8612

3)

Qfin =

258.406437

nfin =

0.979520

5)

e)

```
>> PP1P1
node_limit =
    11

T_values =
  200.0000
  199.2813
  198.5641
  197.8483
  197.1340
  196.4211
  195.7097
  194.9996
  194.2907
  193.5819
  192.8789
```

```
>> PP1P1

node_limit =

   21

T_values =

  200.0000

  199.6409

  199.2821

  198.9237

  198.5657

  198.2081

  197.8508

  197.4939

  197.1374

  196.7812

  196.4254

  196.0700

  195.7149

  195.3602

  195.0059

  194.6519

  194.2983

  193.9450

  193.5921

  193.2391

  192.8876


>> PP1P1
```

```
node_limit =

   101

T_values =


  200.0000

  199.9282

  199.8564

  199.7847

  199.7129

  199.6412

  199.5695

  199.4977

  199.4261

  199.3544

  199.2827

  199.2111

  199.1394

  199.0678

  198.9962

  198.9246

  198.8531

  198.7815

  198.7100

  198.6384

  198.5669

  198.4954

  198.4240
```

198.3525

198.2810

198.2096

198.1382

198.0668

197.9954

197.9240

197.8526

197.7813

197.7100

197.6386

197.5673

197.4960

197.4248

197.3535

197.2823

197.2110

197.1398

197.0686

196.9974

196.9263

196.8551

196.7840

196.7129

196.6417

196.5706

196.4996

196.4285

196.3575

196.2864

196.2154

196.1444

196.0734

196.0024

195.9315

195.8605

195.7896

195.7187

195.6478

195.5769

195.5060

195.4351

195.3643

195.2935

195.2226

195.1518

195.0811

195.0103

194.9395

194.8688

194.7981

194.7274

194.6567

194.5860

194.5153

194.4447

194.3740

194.3034

194.2328

194.1622

194.0916

194.0211

193.9505

193.8800

193.8095

193.7390

193.6685

193.5980

193.5275

193.4571

193.3867

193.3162

193.2458

193.1754

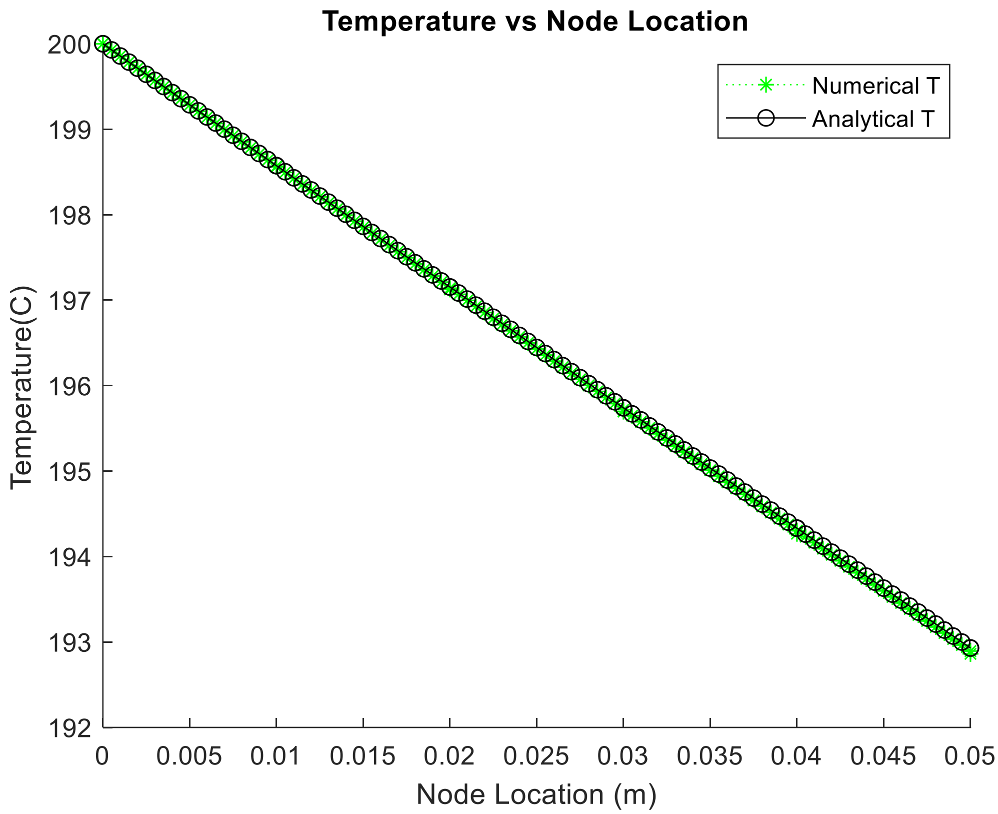
193.1051

193.0347

192.9643

192.8940

f)



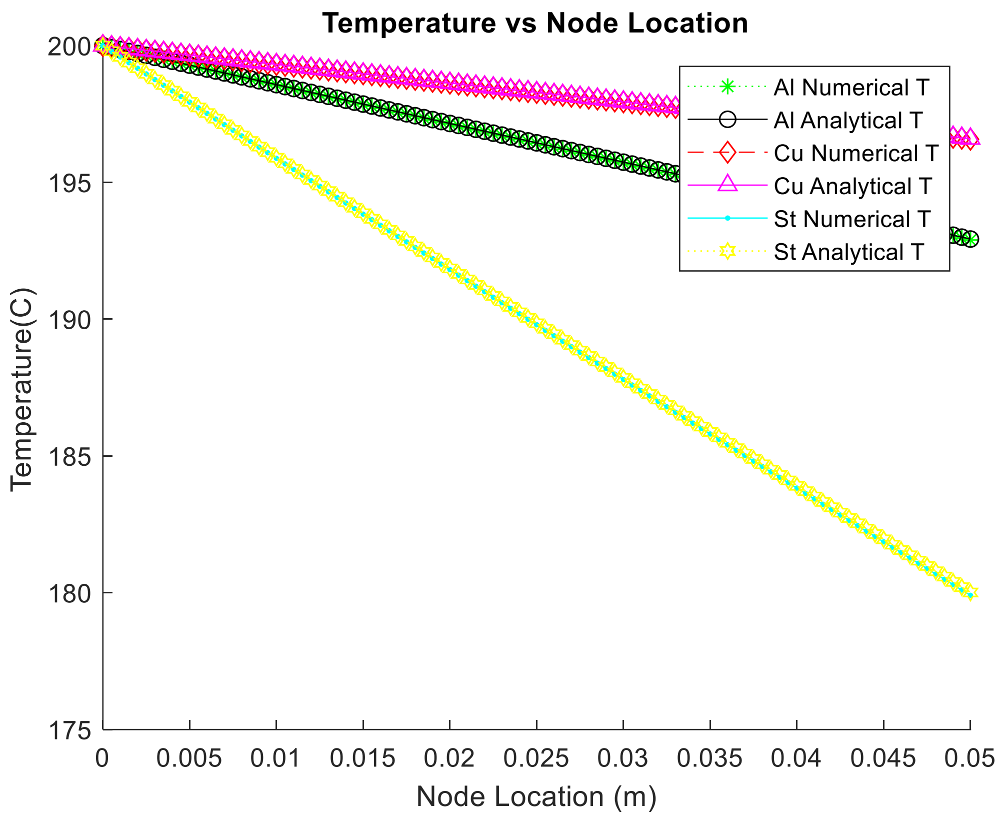
g)

>> PP1P1

Numerical : Rate of Heat Transfer 258.434578, Fin Efficiency 0.979627

Analytical : Rate of Heat Transfer 257.178930, Fin Efficiency 0.974867

h)




```
>> hold on
>> %Aluminum Alloy
>> PP1P1
Numerical : Rate of Heat Transfer 258.434578, Fin Efficiency 0.979627
Analytical : Rate of Heat Transfer 257.178930, Fin Efficiency 0.974867
>> %Copper
>> PP1P1
Numerical : Rate of Heat Transfer 261.226747, Fin Efficiency 0.990211
Analytical : Rate of Heat Transfer 259.943168, Fin Efficiency 0.985345
>> %Steel
>> PP1P1
Numerical : Rate of Heat Transfer 248.507640, Fin Efficiency 0.941997
```

Analytical : Rate of Heat Transfer 247.352785, Fin Efficiency 0.937620

>> legend('Al Numerical T','Al Analytical T','Cu Numerical T','Cu Analytical T','St Numerical T','St Analytical T')

>> hold off

3.1.3 **Conclusions:**

Our numerical and analytical solutions were almost the same with Rate of heat transfer differing by less than 1 and Fin Efficiency differing by less than 0.05. This may be due to rounding error.

Out of the 3 metals Copper has the Highest Rate of Heat Transfer and the best Fin Efficiency, followed by Aluminum Alloy. Steel has the lowest Rate of Heat Transfer and the worst Fin Efficiency.

3.2 **Part 2**

3.2.1 **Methods:**

Script Names: ThCond.m PP1P2.m

Description:

Created a function that will determine the thermal conductivity given a temperature value and alloy name. Temperature may be a scalar, a vector or a matrix. If the alloy name is invalid or the temperature is outside of the range of the given alloy name the function will error. PP1P2 will call ThCond for each metal type and the temperature range they are defined on. Then plot the return k for each metal vs their respective temperatures.

Pseudocode:

ThCond:

In a case statement have a case for each alloy name then check if the given temperature is within that alloys range, if its then calculate thermal conductivity (k) with the respective formula for the alloy. Else error.

In the otherwise clause of the case statement error.

PP1P2: For each alloy call ThCond for the correct temperature range then plot result vs temperature range.

3.2.2 **Results:**

I

>> ThCond(400,'Pl1')

Error using ThCond (line 173)

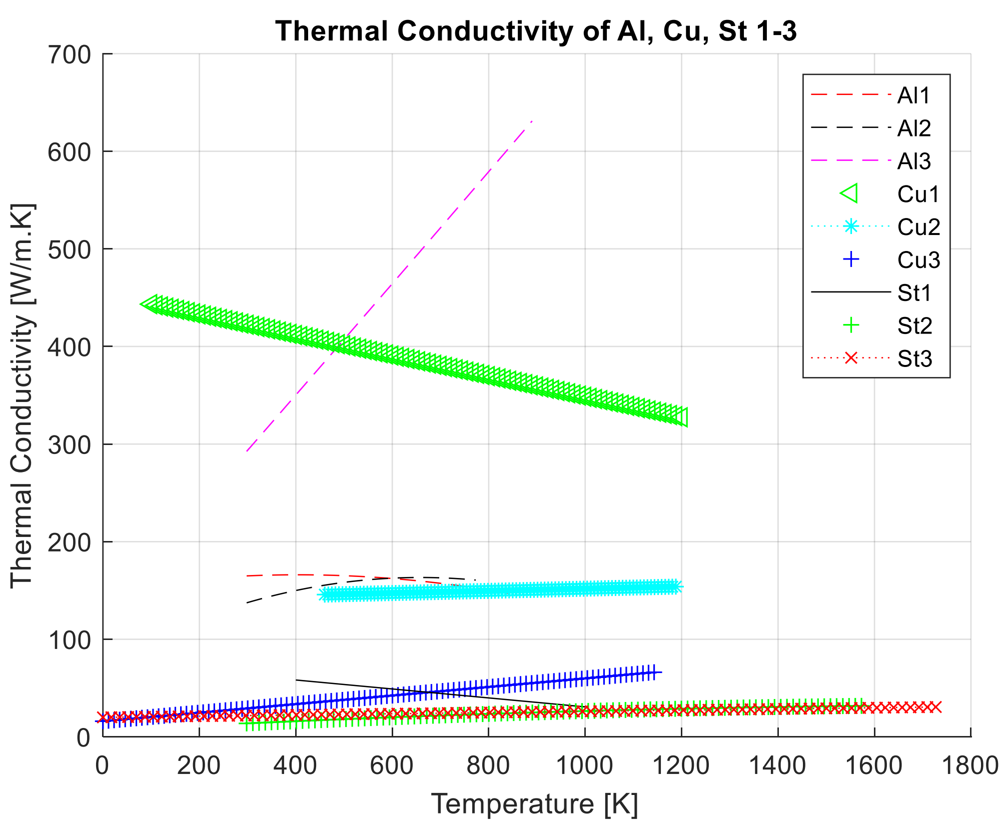
Not a valid Alloy

>> ThCond(300,'St1')

Error using ThCond (line 128)

Temperature out of Range for Alloy

II



3.3 **Part 3**

3.3.1 **Methods:**

Script Names: PP1P3.m

Description:

This scripts is similar to the numerical solution of Part 1 except that it is using the ThCond function to recalculate thermal conductivity(k) and with that k value recalculate the value of T using the same solution as in Part 1. It continues to do this until either the difference between the previous T values and the new ones is less than or equal to 1E-8 or 100 iterations have passed. Then it calculates rate of heat transfer and fin efficiency. Lastly it saves the determined values of T and node location to the PP1P3.dat file. Finally using the command line these values are loaded and plotted for each alloy.

Pseudocode:

Ask user for an alloy name

Initilize all constant values from part 1 except for k.

Initialize Told to a vector of T0's.

Set a counter variable equal to 0.

Set error to 1 to ensure the loop can be entered the first time

While error>1E-8 and counter less than or equal to 100

Calculate k using ThCond with the alloy name and Told

Solve using the numerical solution of part one and the new k value for Tnew

Find the difference between Told and Tnew and assign that to error

Calculate node locations

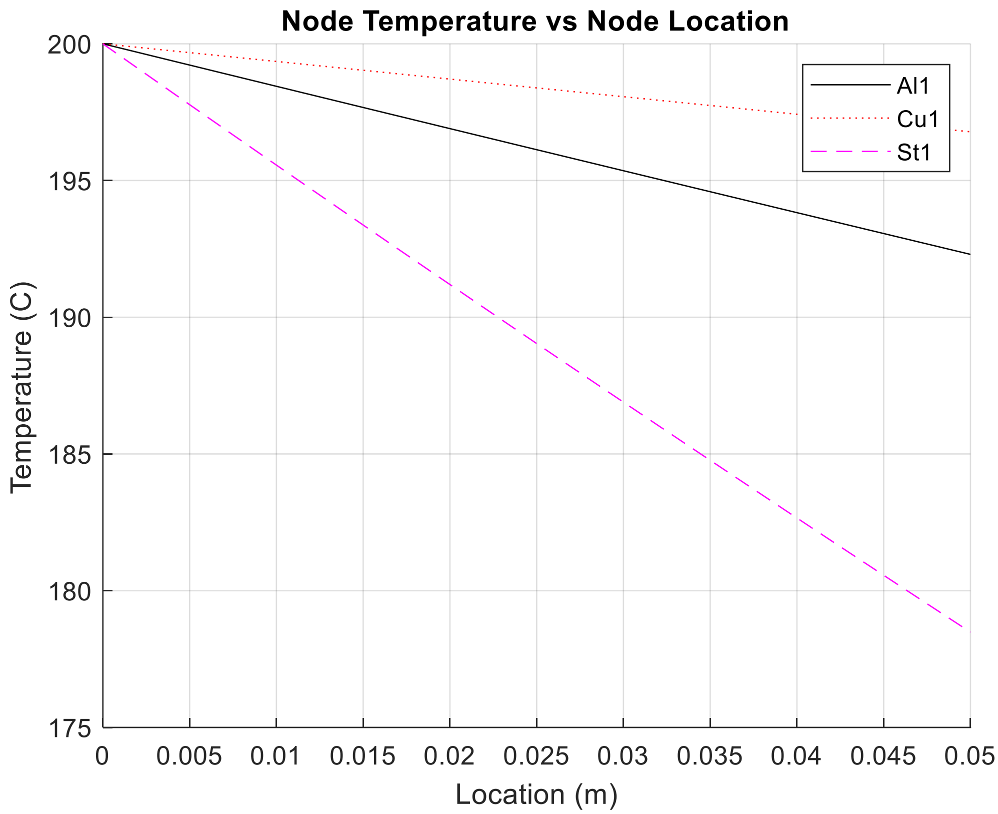
Save node locations and Tnew to PP1P3.dat

Calculate Qfin and nfin using same method as in Part 1.

3.3.2 **Results:**

e)

```
>> data = load('PP1P3.dat','-ascii');
>> Al_x = data(1:101);
>> Al_T = data(102:202);
>> Cu_x = data(203:303);
>> Cu_T = data(304:404);
>> St_x = data(405:505);
>> St_T = data(506:606);
>> hold on
>> plot(Al_x,Al_T,'k');
>> plot(Cu_x,Cu_T,'r:');
>> plot(St_x,St_T,'m--');
>> title('Node Temperature vs Node Location');
>> ylabel('Temperature (C)');
>> xlabel('Location (m)');
>> legend('Al1','Cu1','St1');
>> grid on
>> hold off
```



f)

```
>> PP1P3
Enter an Alloy Name 'Al1'
Rate of Heat Transfer 257.982891, Fin Efficiency 0.977915
>> PP1P3
Enter an Alloy Name 'Cu1'
Rate of Heat Transfer 261.379897, Fin Efficiency 0.990791
>> PP1P3
Enter an Alloy Name 'St1'
Rate of Heat Transfer 247.387577, Fin Efficiency 0.937752
```

3.3.3 **Conclusions:**

<u>Al1</u>

Rate of Heat Transfer 257.982891, Fin Efficiency 0.977915

<u>Cu1</u>

Rate of Heat Transfer 261.379897, Fin Efficiency 0.990791

<u>St1</u>

Rate of Heat Transfer 247.387577, Fin Efficiency 0.937752

4. Conclusion

Writing this solution in Matlab allowed us to perform calculations a lot faster and on a much larger number of nodes. Some good programming practices we used in our solution are meaningful variable names, comments, white space, and proper indentation. Meaningful variable names helped improve readability. Because we gave our variables explicit names future readers will more easily understand their functionality. Use of proper indentation and white spaces made our program flow clearer and easier on the eyes of potential code readers. Proper comments also made our code more readable. If a reader gets confused about one of our code blocks they can refer to the comments to help clear things up. Finally, we needed to use the thermal conductivity calculation often, so without a function we would have to copy a lot of code.