

# ASEN 3802 Lab 2 Part 3

Andrew Patella, Lauren Lajoie, Skylar Harris, Yaseen Mustapha

March 31, 2025

# 1 Task 1: Variance in Thermal Diffusivity

## 1.1 Plot and Table

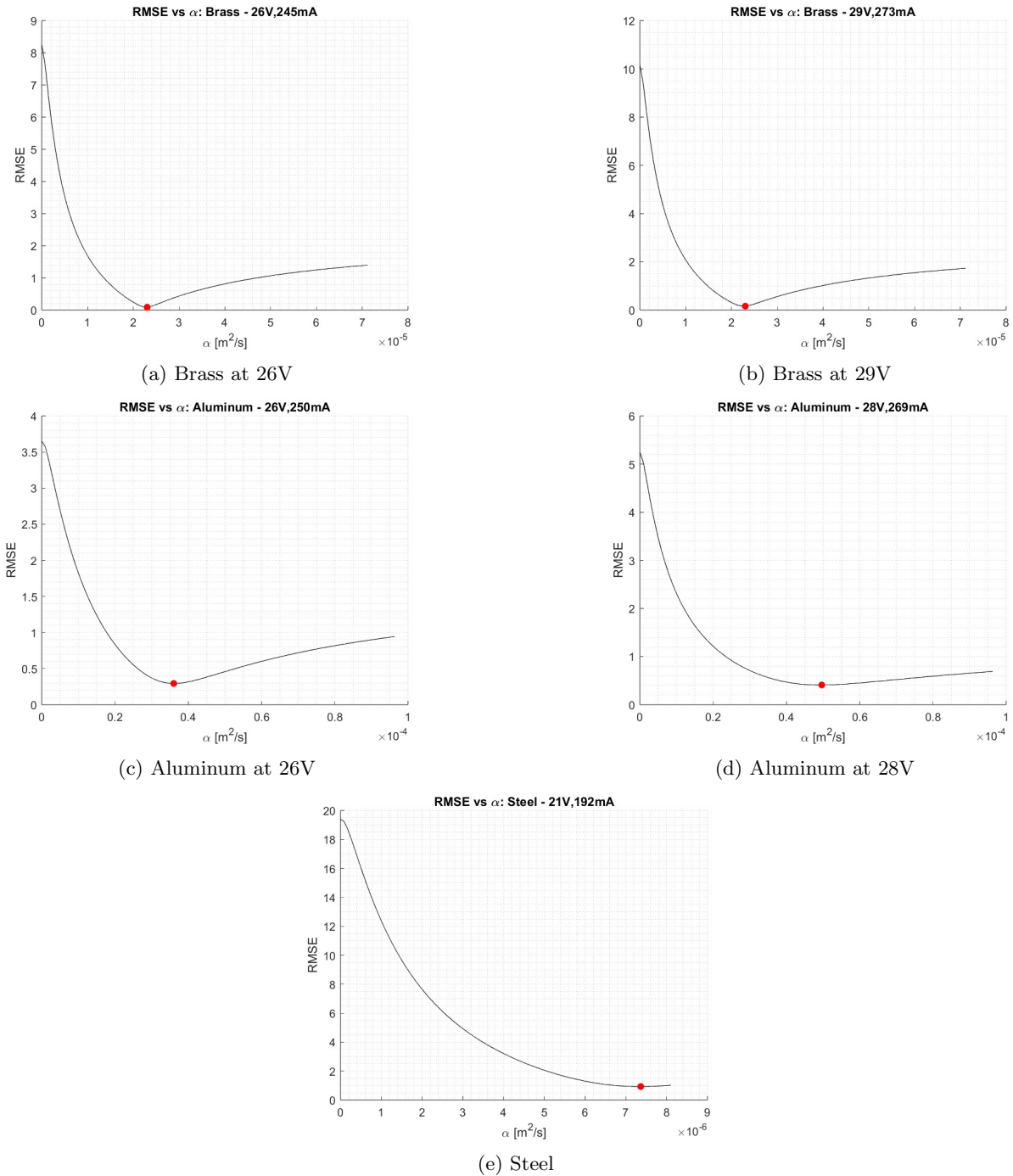
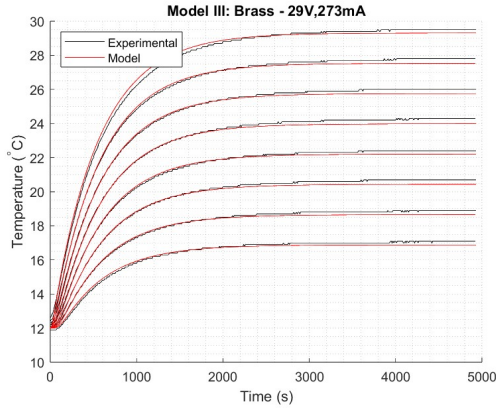
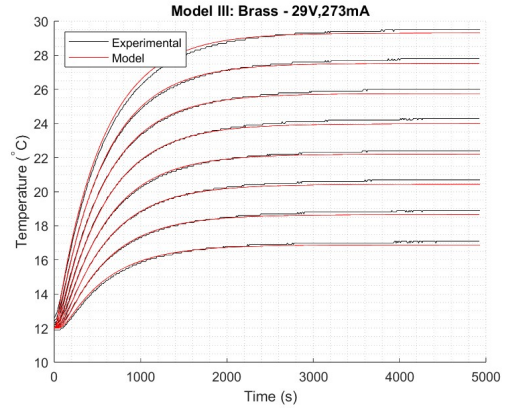


Figure 1: Alpha Vs. Root Mean Square Error for the 5 different cases

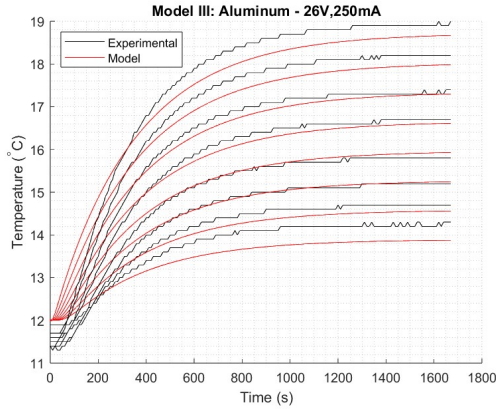
Material	$\alpha \left[ \frac{m^2}{s} \right]$	$\alpha_{adj} \left[ \frac{m^2}{s} \right]$
Steel 21V,192mA	$4.05 \times 10^{-6}$	$7.36 \times 10^{-6}$
Brass 26V,245mA	$3.56 \times 10^{-5}$	$2.30 \times 10^{-5}$
Brass 29V,273mA	$3.56 \times 10^{-5}$	$2.30 \times 10^{-5}$
Aluminum 26V,250mA	$4.82 \times 10^{-5}$	$3.60 \times 10^{-5}$
Aluminum 28V,269mA	$4.82 \times 10^{-5}$	$4.97 \times 10^{-5}$



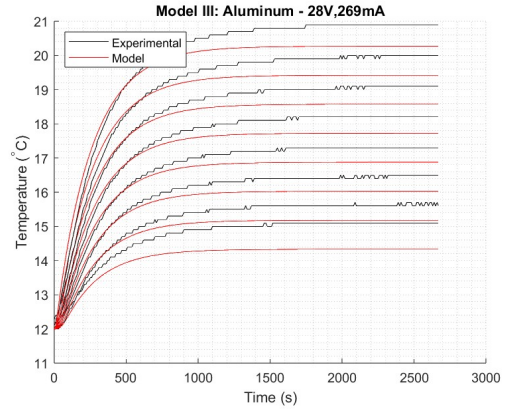
(a) Brass at 26V



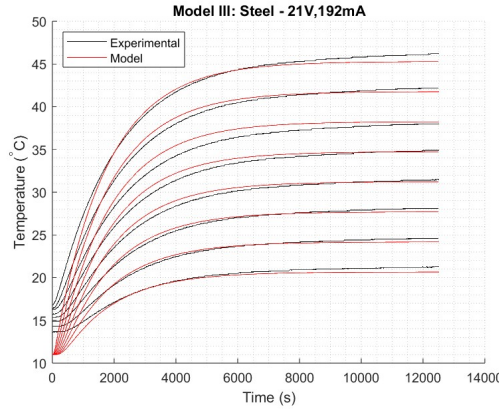
(b) Brass at 29V



(c) Aluminum at 26V



(d) Aluminum at 28V



(e) Steel

Figure 2: Plot of Model 3 against the experimental data

## 1.2 Discussion

To find our adjusted value for the thermal diffusivity, we utilized the root mean square error. This approach tries multiple values for  $\alpha$  and determines which one decreases the error overall for all the channels at each tie. For each material, we found the root mean square error of the Temperature vs. Time curve for various values of thermal diffusivity. We found the error of 100 different thermal diffusivity coefficients from 0 to  $2\alpha$ . We chose the thermal diffusivity of each case that resulted in the smallest root mean square error because this value would result in the closest model of reality possible. Varying the thermal diffusivity

changes the rate at which the model reaches steady state. A lower value of  $\alpha$  causes a curve to reach steady state slower than a higher value of  $\alpha$ . With our method, we were able to find  $\alpha_{adj}$  that best matches the experimental data; as shown in Figure 2. Figure 1 illustrates our method for finding the minimum root mean square error in varying the thermal diffusivity. Figure 2 shows the application of the said thermal diffusivity compared to the experimental data. We used a root mean square for our model to take into account all thermocouples at all times. We wanted to minimize the error everywhere instead of just the error at a particular place on the rod or at a particular moment in time. This offers a more accurate overall model instead of focusing on just one thermocouple. The adjusted thermal diffusivity coefficients found differed for each case. Steel's thermal diffusivity was found to be nearly twice the given value. Brass's was identical for both cases and less than the given value. In the case of aluminum, we found that the thermal diffusivity in the 26V case was less than the given value, while the 28V case was nearly identical to the given value. Overall for each adjusted value we found for thermal diffusivity the value was in the same order of magnitude and seemed reasonable when compared to the values given in the table. We suspect that the deviation between the given and analytical thermal diffusivity coefficients is due to some of the simplifications we made to derive this model, such as the assumption of constant properties.

## 2 Task 2: Time to steady state

Material	$t_{ss}$ [s]	$F_o$ No adjustment	$F_o$ With Adjustment
Steel 21V, 192mA	7,300	1.3281	2.4136
Brass 26V, 245mA	1,790	2.8629	1.8495
Brass 29V, 273mA	2,000	3.1988	2.0664
Aluminum 26V, 250mA	880	1.9051	1.4231
Aluminum 28V, 269mA	910	1.9700	2.0317

To find the approximate time to steady state, we used MATLAB to find where the temperature of the last thermocouple is within 2.5% of the final temperature in the experimental data. To verify our results for the steady state time we also used the MATLAB function `grad` to find where the change in temperature is zero as well as compared visually with our plots. We then used this steady-state time to find the Fourier numbers using  $\frac{\alpha t_{ss}}{L^2}$ . We found our Fourier number with the original thermal diffusivity and then the new Fourier number with adjusted thermal diffusivity. We then put them into the same table for easier comparison.

### 2.1 Discussion

The Fourier number is important because when it is greater than 0.2, a one-term approximation is appropriate. The Fourier number is physically significant because it represents the heat conduction versus the heat storage of a material. This means that the higher the Fourier number, the slower the heat transfer. This makes sense because a material with higher heat conduction will have more rapid heat transfer, while a material with higher heat storage will have a slower rate of heat transfer.

We saw the Fourier number change with the adjusted thermal diffusivity. When our adjusted thermal diffusivity increased, our Fourier number increased. This makes sense because the Fourier number is in direct relationship to the thermal diffusivity. For each case, we saw a different change in the Fourier number. We would expect aluminum to have the lowest Fourier number because it has the highest thermal conductivity and therefore time to steady state would be lower. We found that steel had the highest Fourier number. We think this propagates from our code to find the adjusted thermal diffusivity, where our main goal was just to match the experimental data. This process may have led to Fourier numbers that are not the most accurate representation of the physical materials. For our adjusted thermal diffusivity for steel we saw an increase in the Fourier number, which would physically represent more rapid heat transfer. For brass, we calculated the same adjusted thermal diffusivity for each case. The adjusted thermal diffusivity smaller than the original value and thus the Fourier number decreases for both cases. This physically represents a more slow heat transfer due to a decrease in thermal diffusivity. Finally, for aluminum, we found two different thermal diffusivity coefficients for the two cases. For the 26 volt case we found that the adjusted thermal diffusivity decreased from the original value. Therefore, the Fourier number also decreased. This physically represented that with the adjusted thermal diffusivity, the heat transfer would take longer. For the 28 volt case, we found that the adjusted thermal diffusivity increased, therefore the Fourier number also increased. Physically for the 28 volt case with the new thermal diffusivity the heat transfer would happen more rapidly.

Time to steady state is affected by thermophysical properties of the material as well as the geometry of the object. A longer length will cause a longer time for the furthest thermocouple to reach steady state because the heat transfer has to travel further. Time to steady state is also affected by the thermal diffusivity of the material. A higher diffusivity means a shorter time to steady state. Physically, this makes sense because the more the heat can spread (diffuse) the quicker the material can reach steady state. Finally, within the thermal diffusivity equation, specific heat, density and thermal conductivity also affect the time to steady state. To decrease time to steady state, from  $\alpha = \frac{k}{c_p \rho}$  we can see how each property effects the thermal diffusivity. We found that increasing thermal conductivity, decreasing density and decreasing specific heat capacity all increase thermal diffusivity which will then decrease the time to steady state.

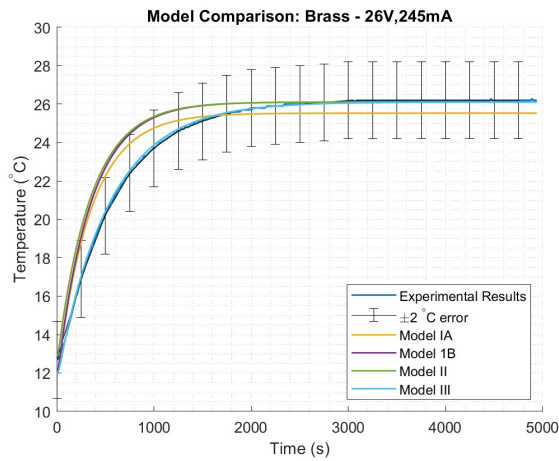
$$F_o = \frac{\alpha t_{ss}}{L^2} \Rightarrow t_{ss} = \frac{F_o L^2}{\alpha}$$

$$t_{ss} \propto \frac{L^2}{\alpha}$$

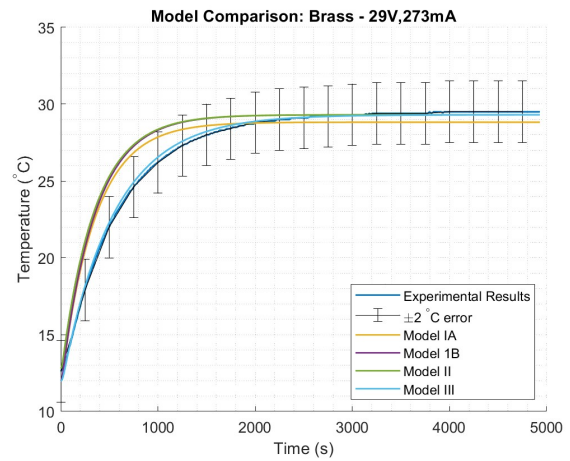
With the Fourier Number being held constant, increasing  $\alpha$  will decrease the time to steady state at a linear rate. This is because the thermal diffusivity is a measure of how quickly heat can spread through a material, so if heat can spread quicker, then the time to steady state will decrease. Since they are linearly related, doubling  $\alpha$  will result in half the time to steady state. The length of the bar has a proportional quadratic impact on the time to steady state. This makes sense as well. If the bar is longer, the heat will take longer to distribute along the bar. Due to the quadratic relationship, doubling the length of the bar will quadruple the time to steady state.

### 3 Task 3: Model Validation and Application

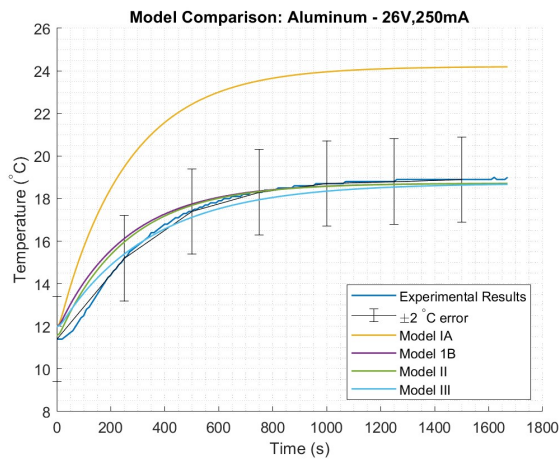
#### 3.1 Plot



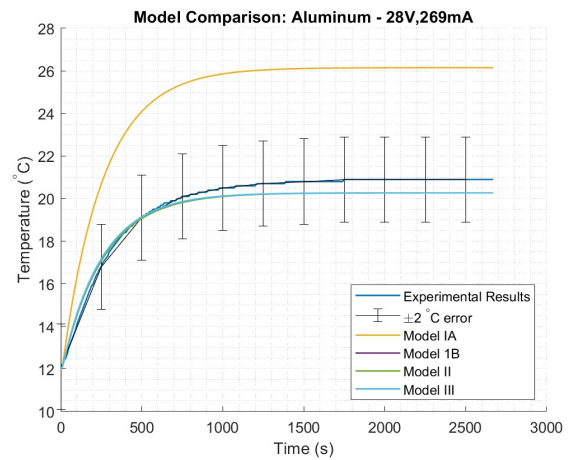
(a) Brass at 26V



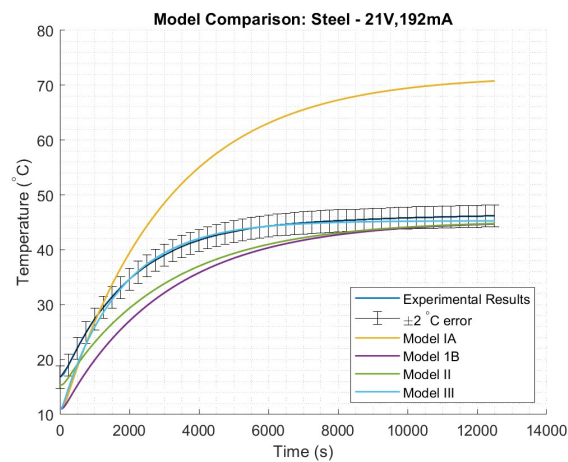
(b) Brass at 29V



(c) Aluminum at 26V



(d) Aluminum at 28V



(e) Steel

Figure 3: Alpha Vs. Root Mean Square Error for the 5 different cases

## 3.2 Discussion

There are three main phases of the experiment that need to be modeled: The beginning heat distribution and the beginning of the warming, the main heating phase where the most change in temperature occurred and the steady state at the end. Each of the models does a better job of predicting the different phases. The beginning steady state is best modeled by Model II, which takes into account the initial heat distribution. The model that does the best in the heating phase is the Model III, which takes into account the actual thermal properties for the metal and improves the estimate of  $\alpha$  to match reality closer. All of the models perform about the same for the end steady state since the exponential dominates and the rate of change decreases significantly so there are fewer sources of error from other constants. The results from the plots match this hypothesis. When looking at the plots from varying our models we were able to see distinct improvements in the regions we were trying to correct, like model II correcting for the initial state condition. The models all gave us the expected improvement in our plots.

The best model to use for modeling the time to steady state would be Model III. This model more accurately matches the experimental data with  $\alpha$  because of how the model is derived. Since  $\alpha$  matches more accurately, the Fourier number will match better. Since the time to steady state is most dependent on the Fourier number, this model will most accurately describe the time to steady state. It has some more error on the beginning by that is small error proportional to the change in time to steady state. The worst model to use is model IA, which neglects any improvements in slope, and initial condition. We can see this result in our plots of each model with the experimental data. We can see that our initial model, Model 1A was outside of the error bars of 2 degrees. But once we reached Model 3 every point was within the error bars.



## 4 Appendix

### 4.1 Contribution Report

**Lauren Lajoie:**

Part 1 - Tabled our results and wrote discussion.

Part 2 - Helped with code for tasks 1 and 2 and wrote discussion.

Part 3 - Found  $T_{ss}$  and  $F_0$  for task 2 and wrote discussion.

**Skylar Harris:**

Part 1 - Helped with code, calculated the analytical slopes, and wrote discussion.

Part 2 - Wrote code for Tasks 1, 2, 3, and 4. Wrote discussion in the report.

Part 3 - Wrote code for Task 1 and wrote discussion in the report.

**Yaseen Mustapha:**

Part 1 - Worked on analysis and the report.

Part 2 - Did the derivation for task 1. Worked on analysis and the report.

Part 3 - Worked on analysis and the report.

**Andrew Patella:**

Part 1 - Calculated the steady-state slope from temperature data. Helped with analysis and data entry

Part 2 - Calculated the Fourier coefficients for the new model, did LaTeX typesetting, and wrote analysis. Helped with analysis and data entry.

Part 3 - Did error analysis graphs and LaTeX typesetting. Helped with analysis and data entry.

### 4.2 Code for Task 1

```
1      %% ASEN 3802 Lab 2 Part 3 Task 1
2
3      clc;clear;close all;
4
5      al26 = readmatrix("Aluminum_26V_250mA");
6      al28 = readmatrix("Aluminum_28V_269mA");
7      br26 = readmatrix("Brass_26V_245mA");
8      br29 = readmatrix("Brass_29V_273mA");
9      st = readmatrix("Steel_21V_192mA");
10
11     dx = 0.0127; %m
12     x0 = 0.034925;
13     xpos = x0:dx:x0 + 0.1016;
14     % breaking data into channels
15     % cell row is which thermocouple
16     al26_c = cell(8,1);
17     al28_c = cell(8,1);
18     br26_c = cell(8,1);
19     br29_c = cell(8,1);
20     st_c = cell(8,1);
21     al26_t = al26(:,1);
22     al28_t = al28(:,1);
23     br26_t = br26(:,1);
24     br29_t = br29(:,1);
25     st_t = st(:,1);
26     for i = 1:8
27         al26_c{i} = al26(:,i+1);
28         al28_c{i} = al28(:,i+1);
29         br26_c{i} = br26(:,i+1);
```

```

30 br29_c{i} = br29(:,i+1);
31 st_c{i} = st(:,i+1);
32 end
33
34 al26_notime = al26(:,2:end);
35 al28_notime = al28(:,2:end);
36 br26_notime = br26(:,2:end);
37 br29_notime = br29(:,2:end);
38 st_notime = st(:,2:end);
39
40 x1 = 0.0349; % [m]
41 x2 = 0.0476; % [m]
42 x3 = 0.0603; % [m]
43 x4 = 0.073; % [m]
44 x5 = 0.0857; % [m]
45 x6 = 0.0984; % [m]
46 x7 = 0.1111; % [m]
47 x8 = 0.1238; % [m]
48 x = [x1,x2,x3,x4,x5,x6,x7,x8];
49
50 const_st.H_exp = 277; % [C/m]
51 const_st.T0 = 11; % [C]
52 const_st.k = 16.2; % [W/mK]
53 const_st.cp = 500; % [J/kgK]
54 const_st.rho = 8000; % [kg/m^3]
55 const_st.alpha = const_st.k/(const_st.rho*const_st.cp); % [m^2/s]
56 const_st.L = x8+0.0508; % [m]
57
58 const_br26.H_exp = 114; % [C/m]
59 const_br26.T0 = 12; % [C]
60 const_br26.k = 115; % [W/mK]
61 const_br26.cp = 380; % [J/kgK]
62 const_br26.rho = 8500; % [kg/m^3]
63 const_br26.alpha = const_br26.k/(const_br26.rho*const_br26.cp); % [m^2/s]
64 const_br26.L = x8+0.0508; % [m]
65
66 const_br29.H_exp = 139.8; % [C/m]
67 const_br29.T0 = 12; % [C]
68 const_br29.k = 115; % [W/mK]
69 const_br29.cp = 380; % [J/kgK]
70 const_br29.rho = 8500; % [kg/m^3]
71 const_br29.alpha = const_br29.k/(const_br29.rho*const_br29.cp); % [m^2/s]
72 const_br29.L = x8+0.0508; % [m]
73
74 const_al26.H_exp = 54.3; % [C/m]
75 const_al26.T0 = 12; % [C]
76 const_al26.k = 130; % [W/mK]
77 const_al26.cp = 960; % [J/kgK]
78 const_al26.rho = 2810; % [kg/m^3]
79 const_al26.alpha = const_al26.k/(const_al26.rho*const_al26.cp); % [m^2/s]
80 const_al26.L = x8+0.0508; % [m]
81
82 const_al28.H_exp = 66.8; % [C/m]
83 const_al28.T0 = 12; % [C]
84 const_al28.k = 130; % [W/mK]
85 const_al28.cp = 960; % [J/kgK]
86 const_al28.rho = 2810; % [kg/m^3]
87 const_al28.alpha = const_al28.k/(const_al28.rho*const_al28.cp); % [m^2/s]

```

```

88 const_al28.L = x8+0.0508; % [m]
89
90 alpha_br26 = linspace(0,2*const_br26.alpha,100);
91 alpha_br29 = linspace(0,2*const_br29.alpha,100);
92 alpha_al26 = linspace(0,2*const_al26.alpha,100);
93 alpha_al28 = linspace(0,2*const_al28.alpha,100);
94 alpha_st = linspace(0,2*const_st.alpha,100);
95
96 t_st = 0:10:12510; % [s]
97 t_al26 = 0:10:1670; % [s]
98 t_al28 = 0:10:2670; % [s]
99 t_br26 = 0:10:4940; % [s]
100 t_br29 = 0:10:4930; % [s]
101
102
103 for j = 1:length(alpha_st)
104     const_st.alpha = alpha_st(j);
105     for i = 1:length(t_st)
106         u_st(i,:) = heatdistribution(t_st(i),x,const_st);
107     end
108     error_st(j) = mean(rmse(u_st, st_notime));
109 end
110 [~,idx] = min(error_st);
111 const_st.alpha = alpha_st(idx);
112
113 for j = 1:length(alpha_al26)
114     const_al26.alpha = alpha_al26(j);
115     for i = 1:length(t_al26)
116         u_al26(i,:) = heatdistribution(t_al26(i),x,const_al26);
117     end
118     error_al26(j) = mean(rmse(u_al26, al26_notime));
119 end
120 [~,idx] = min(error_al26);
121 const_al26.alpha = alpha_al26(idx);
122
123 for j = 1:length(alpha_al28)
124     const_al28.alpha = alpha_al28(j);
125     for i = 1:length(t_al28)
126         u_al28(i,:) = heatdistribution(t_al28(i),x,const_al28);
127     end
128     error_al28(j) = mean(rmse(u_al28, al28_notime));
129 end
130 [~,idx] = min(error_al28);
131 const_al28.alpha = alpha_al28(idx);
132
133 for j = 1:length(alpha_br26)
134     const_br26.alpha = alpha_br26(j);
135     for i = 1:length(t_br26)
136         u_br26(i,:) = heatdistribution(t_br26(i),x,const_br26);
137     end
138     error_br26(j) = mean(rmse(u_br26, br26_notime));
139 end
140 [~,idx] = min(error_br26);
141 const_br26.alpha = alpha_br26(idx);
142
143 for j = 1:length(alpha_br29)
144     const_br29.alpha = alpha_br29(j);
145     for i = 1:length(t_br29)

```

```

146     u_br29(i,:) = heatdistribution(t_br29(i),x,const_br29);
147     end
148     error_br29(j) = mean(rmse(u_br29, br29_notime));
149 end
150 [~,idx] = min(error_br29);
151 const_br29.alpha = alpha_br29(idx);
152
153 figure()
154 hold on
155 grid minor
156 plot(alpha_st,error_st,'k')
157 scatter(const_st.alpha,min(error_st),'filled','r')
158 xlabel('\alpha[m^2/s]')
159 ylabel("RMSE")
160 title('RMSE vs \alpha: Steel-21V,192mA')
161
162 figure()
163 hold on
164 grid minor
165 plot(alpha_al26,error_al26,'k')
166 scatter(const_al26.alpha,min(error_al26),'filled','r')
167 xlabel('\alpha[m^2/s]')
168 ylabel("RMSE")
169 title('RMSE vs \alpha: Aluminum-26V,250mA')
170
171 figure()
172 hold on
173 grid minor
174 plot(alpha_al28,error_al28,'k')
175 scatter(const_al28.alpha,min(error_al28),'filled','r')
176 xlabel('\alpha[m^2/s]')
177 ylabel("RMSE")
178 title('RMSE vs \alpha: Aluminum-28V,269mA')
179
180 figure()
181 hold on
182 grid minor
183 plot(alpha_br26,error_br26,'k')
184 scatter(const_br26.alpha,min(error_br26),'filled','r')
185 xlabel('\alpha[m^2/s]')
186 ylabel("RMSE")
187 title('RMSE vs \alpha: Brass-26V,245mA')
188
189 figure()
190 hold on
191 grid minor
192 plot(alpha_br29,error_br29,'k')
193 scatter(const_br29.alpha,min(error_br29),'filled','r')
194 xlabel('\alpha[m^2/s]')
195 ylabel("RMSE")
196 title('RMSE vs \alpha: Brass-29V,273mA')
197
198 for i = 1:length(t_st)
199     u_st(i,:) = heatdistribution(t_st(i),x,const_st);
200 end
201 for i = 1:length(t_al26)
202     u_al26(i,:) = heatdistribution(t_al26(i),x,const_al26);
203 end

```

```

204 for i = 1:length(t_al28)
205     u_al28(i,:) = heatdistribution(t_al28(i),x,const_al28);
206 end
207 for i = 1:length(t_br26)
208     u_br26(i,:) = heatdistribution(t_br26(i),x,const_br26);
209 end
210 for i = 1:length(t_br29)
211     u_br29(i,:) = heatdistribution(t_br29(i),x,const_br29);
212 end
213
214 figure()
215 hold on
216 for i = 1:8
217     plot(al26_t,al26_c{i},'k')
218     plot(t_al26,u_al26(:,i),'r')
219 end
220 grid minor
221 xlabel("Time (s)")
222 ylabel('Temperature_\(\circ C\)')
223 legend('Experimental','Model','','','','','','','','','','','','','Location','northwest')
224 title("Model III: Aluminum - 26V,250mA")
225
226 figure()
227 hold on
228 for i = 1:8
229     plot(al28_t,al28_c{i},'k')
230     plot(t_al28,u_al28(:,i),'r')
231 end
232 grid minor
233 xlabel("Time (s)")
234 ylabel('Temperature_\(\circ C\)')
235 legend('Experimental','Model','','','','','','','','','','','','','Location','northwest')
236 title("Model III: Aluminum - 28V,269mA")
237
238 figure()
239 hold on
240 for i = 1:8
241     plot(br26_t,br26_c{i},'k')
242     plot(t_br26,u_br26(:,i),'r')
243 end
244 grid minor
245 xlabel("Time (s)")
246 ylabel('Temperature_\(\circ C\)')
247 legend('Experimental','Model','','','','','','','','','','','','','Location','northwest')
248 title("Model III: Brass - 26V,245mA")
249
250 figure()
251 hold on
252 for i = 1:8
253     plot(br29_t,br29_c{i},'k')
254     plot(t_br29,u_br29(:,i),'r')
255 end
256 grid minor
257 xlabel("Time (s)")
258 ylabel('Temperature_\(\circ C\)')
259 legend('Experimental','Model','','','','','','','','','','','','','Location','northwest')
260 title("Model III: Brass - 29V,273mA")
261

```

```

262 figure()
263 hold on
264 for i = 1:8
265 plot(st_t,st_c{i},'k')
266 plot(t_st,u_st(:,i),'r')
267 end
268 grid minor
269 xlabel("Time (s)")
270 ylabel('Temperature_\(^{\circ}\)C')
271 legend('Experimental','Model','','','','','','','','','','','','','Location','northwest')
272 title("Model III: Steel - 21V,192mA")
273
274
275 function u = heatdistribution(t,x,const)
276     sum = [0,0,0,0,0,0,0,0];
277     for n = 1:10
278         b_n = (8*const.H_exp*const.L*(-1)^n)/((2*n-1)^2*pi^2);
279         lambda_n = pi*(2*n-1)/(2*const.L);
280         sum = sum + b_n.*sin(lambda_n.*x).*exp(-1*lambda_n^2*const.alpha*t);
281     end
282     u = const.T0+const.H_exp.*x + sum;
283 end

```

### 4.3 Code for Task 2

```

1 close all;
2 clear;
3 clc;
4
5
6
7 %% Data
8 al26 = readmatrix('Aluminum_26V_250mA');
9 al28 = readmatrix('Aluminum_28V_269mA');
10 br26 = readmatrix('Brass_26V_245mA');
11 br29 = readmatrix('Brass_29V_273mA');
12 st = readmatrix('Steel_21V_192mA');
13
14 %% Constant
15 L = 0.1492;
16 const_st.k = 16.2; % [W/mK]
17 const_st.cp = 500; %J/kgK
18 const_st.rho = 8000; % [kg/m^3]
19 const_st.alpha = const_st.k/(const_st.rho*const_st.cp); % [m^2/s]
20 const_st.alphaadj = 7.36*10^-6;
21
22 const_br26.k = 115; % [W/mK]
23 const_br26.cp = 380; %J/kgK
24 const_br26.rho = 8500; % [kg/m^3]
25 const_br26.alpha = const_br26.k/(const_br26.rho*const_br26.cp); % [m^2/s]
26 const_br26.alphaadj = 2.30*10^-5;
27
28 const_br29.k = 115; % [W/mK]
29 const_br29.cp = 380; % [J/kgK]
30 const_br29.rho = 8500; % [kg/m^3]
31 const_br29.alpha = const_br29.k/(const_br29.rho*const_br29.cp); % [m^2/s]
32 const_br29.alphaadj = 2.3*10^-5;

```

```

33
34 const_al26.k = 130; % [W/mK]
35 const_al26.cp = 960; % [J/kgK]
36 const_al26.rho = 2810; % [kg/m^3]
37 const_al26.alpha = const_al26.k/(const_al26.rho*const_al26.cp); % [m^2/s]
38 const_al26.alphaadj = 3.6*10^-5;
39
40 const_al28.k = 130; % [W/mK]
41 const_al28.cp = 960; % [J/kgK]
42 const_al28.rho = 2810; % [kg/m^3]
43 const_al28.alpha = const_al28.k/(const_al28.rho*const_al28.cp); % [m^2/s]
44 const_al28.alphaadj = 4.97*10^-5;
45
46 %% Original Alpha
47 final_st = 0.975*st(1252,9);
48 grad_st = gradient(st(:,9));
49 tss_st = st(find(st(:,9)>final_st,1));
50
51
52 final_al26 = 0.975*al26(168,9);
53 grad_al26 = gradient(al26(:,9));
54 tss_al26 = al26(find(al26(:,9)>final_al26,1));
55
56
57 final_al28 = 0.975*al28(268,9);
58 grad_al28 = gradient(al28(:,9));
59 tss_al28 = st(find(al28(:,9)>final_al28,1));
60
61
62 final_br26 = 0.975*br26(495,9);
63 grad_br26 = gradient(br26(:,9));
64 tss_br26 = br26(find(br26(:,9)>final_br26,1));
65
66
67 final_br29 = 0.975*br29(494,9);
68 grad_br29 = gradient(br29(:,9));
69 tss_br29 = br29(find(br29(:,9)>final_br29,1));
70
71
72 Fo_st = (const_st.alpha*tss_st)/L^2;
73 Fo_br26 = (const_br26.alpha*tss_br26)/L^2;
74 Fo_br29 = (const_br29.alpha*tss_br29)/L^2;
75 Fo_al26 = (const_al26.alpha*tss_al26)/L^2;
76 Fo_al28 = (const_al28.alpha*tss_al28)/L^2;
77
78 %% New Alpha
79
80 newFo_st = (const_st.alphaadj*tss_st)/L^2;
81 newFo_br26 = (const_br26.alphaadj*tss_br26)/L^2;
82 newFo_br29 = (const_br29.alphaadj*tss_br29)/L^2;
83 newFo_al26 = (const_al26.alphaadj*tss_al26)/L^2;
84 newFo_al28 = (const_al28.alphaadj*tss_al28)/L^2;

```

#### 4.4 Code for Task 3

```

1 clc;clear;close all;
2

```

```

3 al26 = readmatrix("Aluminum_26V_250mA");
4 al28 = readmatrix("Aluminum_28V_269mA");
5 br26 = readmatrix("Brass_26V_245mA");
6 br29 = readmatrix("Brass_29V_273mA");
7 st = readmatrix("Steel_21V_192mA");
8
9 %% Material and Geometric Properties
10 dx = 0.0127; %m
11 x0 = 0.034925;
12 xpos = x0:dx:x0 + 0.1016;
13 % breaking data into channels
14 % cell row is which thermocouple
15 al26_c = cell(8,1);
16 al28_c = cell(8,1);
17 br26_c = cell(8,1);
18 br29_c = cell(8,1);
19 st_c = cell(8,1);
20 al26_t = al26(:,1);
21 al28_t = al28(:,1);
22 br26_t = br26(:,1);
23 br29_t = br29(:,1);
24 st_t = st(:,1);
25 for i = 1:8
26 al26_c{i} = al26(:,i+1);
27 al28_c{i} = al28(:,i+1);
28 br26_c{i} = br26(:,i+1);
29 br29_c{i} = br29(:,i+1);
30 st_c{i} = st(:,i+1);
31 end
32
33 al26_notime = al26(:,2:end);
34 al28_notime = al28(:,2:end);
35 br26_notime = br26(:,2:end);
36 br29_notime = br29(:,2:end);
37 st_notime = st(:,2:end);
38
39 x1 = 0.0349; % [m]
40 x2 = 0.0476; % [m]
41 x3 = 0.0603; % [m]
42 x4 = 0.073; % [m]
43 x5 = 0.0857; % [m]
44 x6 = 0.0984; % [m]
45 x7 = 0.1111; % [m]
46 x8 = 0.1238; % [m]
47 x = [x1,x2,x3,x4,x5,x6,x7,x8];
48
49 const_st.H_exp = 277; % [C/m]
50 const_st.H_an = 491.2; % [C/m]
51 const_st.M_exp = 35.2; % [C/m]
52 const_st.T0 = 11; % [C]
53 const_st.k = 16.2; % [W/mK]
54 const_st.cp = 500; % [J/kgK]
55 const_st.rho = 8000; % [kg/m^3]
56 const_st.alpha = const_st.k/(const_st.rho*const_st.cp); % [m^2/s]
57 const_st.L = x8+0.0508; % [m]
58 const_st.alpha2 = 7.36e-6;
59
60 const_br26.H_exp = 114; % [C/m]

```



```

61 const_br26.H_an = 109.3; % [C/m]
62 const_br26.M_exp = 7; % [C/m]
63 const_br26.T0 = 12; % [C]
64 const_br26.k = 115; % [W/mK]
65 const_br26.cp = 380; % [J/kgK]
66 const_br26.rho = 8500; % [kg/m^3]
67 const_br26.alpha = const_br26.k/(const_br26.rho*const_br26.cp); % [m^2/s]
68 const_br26.L = x8+0.0508; % [m]
69 const_br26.alpha2 = 2.3e-5;
70
71
72 const_br29.H_exp = 139.8; % [C/m]
73 const_br29.H_an = 135.9; % [C/m]
74 const_br29.M_exp = 6.6; % [C/m]
75 const_br29.T0 = 12; % [C]
76 const_br29.k = 115; % [W/mK]
77 const_br29.cp = 380; % [J/kgK]
78 const_br29.rho = 8500; % [kg/m^3]
79 const_br29.alpha = const_br29.k/(const_br29.rho*const_br29.cp); % [m^2/s]
80 const_br29.L = x8+0.0508; % [m]
81 const_br29.alpha2 = 2.3e-5;
82
83 const_al26.H_exp = 54.3; % [C/m]
84 const_al26.H_an = 98.6; % [C/m]
85 const_al26.M_exp = -3.4; % [C/m]
86 const_al26.T0 = 12; % [C]
87 const_al26.k = 130; % [W/mK]
88 const_al26.cp = 960; % [J/kgK]
89 const_al26.rho = 2810; % [kg/m^3]
90 const_al26.alpha = const_al26.k/(const_al26.rho*const_al26.cp); % [m^2/s]
91 const_al26.L = x8+0.0508; % [m]
92 const_al26.alpha2 = 3.6e-5;
93
94 const_al28.H_exp = 66.8; % [C/m]
95 const_al28.H_an = 114.3; % [C/m]
96 const_al28.M_exp = 0.3; % [C/m]
97 const_al28.T0 = 12; % [C]
98 const_al28.k = 130; % [W/mK]
99 const_al28.cp = 960; % [J/kgK]
100 const_al28.rho = 2810; % [kg/m^3]
101 const_al28.alpha = const_al28.k/(const_al28.rho*const_al28.cp); % [m^2/s]
102 const_al28.L = x8+0.0508; % [m]
103 const_al28.alpha2 = 4.97e-5;
104
105 alpha_br26 = linspace(0,2*const_br26.alpha,100);
106 alpha_br29 = linspace(0,2*const_br29.alpha,100);
107 alpha_al26 = linspace(0,2*const_al26.alpha,100);
108 alpha_al28 = linspace(0,2*const_al28.alpha,100);
109 alpha_st = linspace(0,2*const_st.alpha,100);
110
111 t_st = 0:10:12510; % [s]
112 t_al26 = 0:10:1670; % [s]
113 t_al28 = 0:10:2670; % [s]
114 t_br26 = 0:10:4940; % [s]
115 t_br29 = 0:10:4930; % [s]
116
117 %% Finding Analytical Solutions
118

```

```

119 % Analytical solutions for model IA
120 for i = 1:length(t_st)
121     u_stIA(i,:) = heatdistributionIA(t_st(i),x8,const_st);
122 end
123 for i = 1:length(t_al26)
124     u_al26IA(i,:) = heatdistributionIA(t_al26(i),x8,const_al26);
125 end
126 for i = 1:length(t_al28)
127     u_al28IA(i,:) = heatdistributionIA(t_al28(i),x8,const_al28);
128 end
129 for i = 1:length(t_br26)
130     u_br26IA(i,:) = heatdistributionIA(t_br26(i),x8,const_br26);
131 end
132 for i = 1:length(t_br29)
133     u_br29IA(i,:) = heatdistributionIA(t_br29(i),x8,const_br29);
134 end
135
136 % Analytical solutions for model IB
137 for i = 1:length(t_st)
138     u_stIB(i,:) = heatdistributionIB(t_st(i),x8,const_st);
139 end
140 for i = 1:length(t_al26)
141     u_al26IB(i,:) = heatdistributionIB(t_al26(i),x8,const_al26);
142 end
143 for i = 1:length(t_al28)
144     u_al28IB(i,:) = heatdistributionIB(t_al28(i),x8,const_al28);
145 end
146 for i = 1:length(t_br26)
147     u_br26IB(i,:) = heatdistributionIB(t_br26(i),x8,const_br26);
148 end
149 for i = 1:length(t_br29)
150     u_br29IB(i,:) = heatdistributionIB(t_br29(i),x8,const_br29);
151 end
152
153 % Analytical Solutions for model II
154 for i = 1:length(t_st)
155     u_stII(i,:) = heatdistributionII(t_st(i),x8,const_st);
156 end
157 for i = 1:length(t_al26)
158     u_al26II(i,:) = heatdistributionII(t_al26(i),x8,const_al26);
159 end
160 for i = 1:length(t_al28)
161     u_al28II(i,:) = heatdistributionII(t_al28(i),x8,const_al28);
162 end
163 for i = 1:length(t_br26)
164     u_br26II(i,:) = heatdistributionII(t_br26(i),x8,const_br26);
165 end
166 for i = 1:length(t_br29)
167     u_br29II(i,:) = heatdistributionII(t_br29(i),x8,const_br29);
168 end
169
170 % Analytical Solution for model III
171 for i = 1:length(t_st)
172     u_stIII(i,:) = heatdistributionIII(t_st(i),x8,const_st);
173 end
174 for i = 1:length(t_al26)
175     u_al26III(i,:) = heatdistributionIII(t_al26(i),x8,const_al26);
176 end

```

```

177 for i = 1:length(t_al28)
178     u_al28III(i,:) = heatdistributionIII(t_al28(i),x8,const_al28);
179 end
180 for i = 1:length(t_br26)
181     u_br26III(i,:) = heatdistributionIII(t_br26(i),x8,const_br26);
182 end
183 for i = 1:length(t_br29)
184     u_br29III(i,:) = heatdistributionIII(t_br29(i),x8,const_br29);
185 end
186
187
188 %% Data analysis
189 dx = 0.0127; %m
190 x0 = 0.034925;
191 xpos = x0:dx:x0 + 0.1016;
192 % breaking data into channels
193 % cell row is which thermocouple
194 al26_c = cell(8,1);
195 al28_c = cell(8,1);
196 br26_c = cell(8,1);
197 br29_c = cell(8,1);
198 st_c = cell(8,1);
199 al26_t = al26(:,1);
200 al28_t = al28(:,1);
201 br26_t = br26(:,1);
202 br29_t = br29(:,1);
203 st_t = st(:,1);
204 for i = 1:8
205     al26_c{i} = al26(:,i+1);
206     al28_c{i} = al28(:,i+1);
207     br26_c{i} = br26(:,i+1);
208     br29_c{i} = br29(:,i+1);
209     st_c{i} = st(:,i+1);
210 end
211
212 %% Plotting Results\
213
214 % AL 26
215 plotvec = al26_t(1:25:end);
216 err = 2*ones(length(plotvec),1);
217
218 figure()
219 hold on
220 p1=plot(al26_t,al26_c{8},'LineWidth',1);
221 p2=errorbar(al26_t(1:25:end),al26_c{8}(1:25:end),err,'k');
222 p3=plot(t_al26,u_al26IA(:,8),'LineWidth',1);
223 p4=plot(t_al26,u_al26IB(:,8),'LineWidth',1);
224 p5=plot(t_al26,u_al26II(:,8),'LineWidth',1);
225 p6=plot(t_al26,u_al26III(:,8),'LineWidth',1);
226 grid minor
227 xlabel("Time (s)")
228 ylabel('Temperature_\(^{\circ}\text{C}\)')
229 legend([p1,p2,p3,p4,p5,p6],{'Experimental_Results','\pm2_\(^{\circ}\text{C}\)error','Model_IA','Model_1B','Model_II','Model_III'},'Location','southeast')
230 legend
231 title("Model Comparison: Aluminum - 26V,250mA")
232
233

```

```

234 % Steel
235 plotvec = st_t(1:25:end);
236 err = 2*ones(length(plotvec),1);
237
238 figure()
239 hold on
240 p1=plot(st_t,st_c{8},'LineWidth',1);
241 p2=errorbar(st_t(1:25:end),st_c{8}(1:25:end),err,'k');
242 p3=plot(t_st,u_stIA(:,8),'LineWidth',1);
243 p4=plot(t_st,u_stIB(:,8),'LineWidth',1);
244 p5=plot(t_st,u_stII(:,8),'LineWidth',1);
245 p6=plot(t_st,u_stIII(:,8),'LineWidth',1);
246 grid minor
247 xlabel("Time (s)")
248 ylabel('Temperature_\circC')
249 legend([p1,p2,p3,p4,p5,p6],{'Experimental_Results','\pm_\circC_error','Model_IA','Model_1B','Model_II','Model_III'},'Location','southeast')
250 legend
251 title("Model Comparison: Steel - 21V,192mA")
252
253
254 % AL 28
255 plotvec = al28_t(1:25:end);
256 err = 2*ones(length(plotvec),1);
257
258 figure()
259 hold on
260 p1=plot(al28_t,al28_c{8},'LineWidth',1);
261 p2=errorbar(al28_t(1:25:end),al28_c{8}(1:25:end),err,'k');
262 p3=plot(t_al28,u_al28IA(:,8),'LineWidth',1);
263 p4=plot(t_al28,u_al28IB(:,8),'LineWidth',1);
264 p5=plot(t_al28,u_al28II(:,8),'LineWidth',1);
265 p6=plot(t_al28,u_al28III(:,8),'LineWidth',1);
266 grid minor
267 xlabel("Time (s)")
268 ylabel('Temperature_\circC')
269 legend([p1,p2,p3,p4,p5,p6],{'Experimental_Results','\pm_\circC_error','Model_IA','Model_1B','Model_II','Model_III'},'Location','southeast')
270 legend
271 title("Model Comparison: Aluminum - 28V,269mA")
272
273
274 % Br 26
275 plotvec = br26_t(1:25:end);
276 err = 2*ones(length(plotvec),1);
277
278 figure()
279 hold on
280 p1=plot(br26_t,br26_c{8},'LineWidth',1);
281 p2=errorbar(br26_t(1:25:end),br26_c{8}(1:25:end),err,'k');
282 p3=plot(t_br26,u_br26IA(:,8),'LineWidth',1);
283 p4=plot(t_br26,u_br26IB(:,8),'LineWidth',1);
284 p5=plot(t_br26,u_br26II(:,8),'LineWidth',1);
285 p6=plot(t_br26,u_br26III(:,8),'LineWidth',1);
286 grid minor
287 xlabel("Time (s)")
288 ylabel('Temperature_\circC')
289 legend([p1,p2,p3,p4,p5,p6],{'Experimental_Results','\pm_\circC_error','Model_IA','Model_1B','Model_II','Model_III'},'Location','southeast')

```

```

289 legend
290 title("Model Comparison: Brass - 26V,245mA")
291
292 % Br 28
293 plotvec = br29_t(1:25:end);
294 err = 2*ones(length(plotvec),1);
295
296 figure()
297 hold on
298 p1=plot(br29_t,br29_c{8}, 'LineWidth',1);
299 p2=errorbar(br29_t(1:25:end),br29_c{8}(1:25:end),err,'k');
300 p3=plot(t_br29,u_br29IA(:,8), 'LineWidth',1);
301 p4=plot(t_br29,u_br29IB(:,8), 'LineWidth',1);
302 p5=plot(t_br29,u_br29II(:,8), 'LineWidth',1);
303 p6=plot(t_br29,u_br29III(:,8), 'LineWidth',1);
304 grid minor
305 xlabel("Time (s)")
306 ylabel('Temperature_\u00b0C')
307 legend([p1,p2,p3,p4,p5,p6],{'Experimental_Results','\u03c0m2_\u00b0C_error','Model_IA','Model_1B','Model_II','Model_III'},'Location','southeast')
308 legend
309 title("Model Comparison: Brass - 29V,273mA")
310
311 %% Functions
312 function u = heatdistributionIA(t,x,const)
313     sum = [0,0,0,0,0,0,0,0];
314     for n = 1:10
315         b_n = (8*const.H_an*const.L*(-1)^n)/((2*n-1)^2*pi^2);
316         lambda_n = pi*(2*n-1)/(2*const.L);
317         sum = sum + b_n.*sin(lambda_n.*x).*exp(-1*lambda_n^2*const.alpha*t);
318     end
319     u = const.T0+const.H_an.*x + sum;
320 end
321
322 function u = heatdistributionIB(t,x,const)
323     sum = [0,0,0,0,0,0,0,0];
324     for n = 1:10
325         b_n = (8*const.H_exp*const.L*(-1)^n)/((2*n-1)^2*pi^2);
326         lambda_n = pi*(2*n-1)/(2*const.L);
327         sum = sum + b_n.*sin(lambda_n.*x).*exp(-1*lambda_n^2*const.alpha*t);
328     end
329     u = const.T0+const.H_exp.*x + sum;
330 end
331
332 function u = heatdistributionII(t,x,const)
333     sum = [0,0,0,0,0,0,0,0];
334     for n = 1:10
335         b_n = (8*(const.M_exp-const.H_exp)*const.L*(-1)^(n+1))/((2*n-1)^2*pi^2);
336         lambda_n = pi*(2*n-1)/(2*const.L);
337         sum = sum + b_n.*sin(lambda_n.*x).*exp(-1*lambda_n^2*const.alpha*t);
338     end
339     u = const.T0+const.H_exp.*x + sum;
340 end
341
342 function u = heatdistributionIII(t,x,const)
343     sum = [0,0,0,0,0,0,0,0];
344     for n = 1:10
345         b_n = (8*const.H_exp*const.L*(-1)^n)/((2*n-1)^2*pi^2);

```

```
346     lambda_n = pi*(2*n-1)/(2*const.L);
347     sum = sum + b_n.*sin(lambda_n.*x).*exp(-1*lambda_n^2*const.alpha2*t);
348 end
349 u = const.T0+const.H_exp.*x + sum;
350 end
```