Heat Transport Mechanisms for an End-Heated Aluminum Rod

Akshiv Bansal - 35072131, Group 11

July 6, 2015

ENPH 257 Dr. Waltham

Contents

List of Figures

List of Tables

1 Introduction

The laws of thermodynamics govern the transfer of energy between objects within our system. Here we will investigate the properties that govern heat transfer between an aluminum rod and the environment. The major mechanisms by which an object loses heat are radiation, convection, and conduction. All of these processes are driven by a temperature gradient between our system and the environment, because all entities are constantly trying to reach thermal equilibrium. Each method of heat transfer, transports energy in a different way and is effected by its own set of parameters. For simplicity we reduce each transport mechanism to a constant and a polynomial dependence between the temperature of the system and the surroundings. This way we can eliminate a great deal of complexity that comes from trying to accurately model thermodynamic systems, as we are really trying to describe the outcome of atom-atom interactions. This is particularly useful in the case of convection, as it shows a strong dependence on factors like geometry, orientation, and the normal considerations of material properties.

2 Purpose

We would like to investigate the thermodynamic properties of an aluminum rod that affect the transfer of heat by radiation, convection, and conduction, as well as a consideration of power loss in the special case of evaporation of water from the surface of the rod. We model these three transfers of heat as:

$$Power_{Radiation} = \epsilon\sigma A(T^4 - T_{amb}^4)\ |A = Area \tag{1}$$

$$Power_{Convection} = K_C A \Delta T \tag{2}$$

$$Power_{Conduction} = KA\frac{\delta T}{\delta x} \tag{3}$$

Given this we would like to find,

Constant	Description
K_{Alu}	Thermal Conductivity
K_{C_H}	Horizontal Convection Coefficient
K_{C_V}	Vertical Convection Coefficient
ϵ_s	Emissivity for Sand Blasted Rod
ϵ_b	Emissivity for the Black Painted Rod
P_{Evap}	Power Lost from a Wet Surface due to Evaporation
C_p	Heat Capacity of Aluminum Rod
P_{Loss_R}	Percent Power lost by the Resistor

Table 1: List of Needed Parameters

3 Methodology

3.1 Experiment to Determine Heat Capacity

In this experiment we wanted to use calorimetry methods to find the heat capacity of the aluminum rod. Heat capacity and thermal energy supplied are related by the governing equation.

$$Q = m_{mass} \times C_{Heat\ Capacity} \times \Delta T_{temperature} \tag{4}$$

We want to heat the rod up to a known temperature, then put the rod in a cold bath of water and measure the change in the temperature of the water. This way we know that the energy the rod has goes into heating the water. From the change in water temperature and the given value of the heat capacity of water we can calculate the heat capacity of aluminum. We use the lab balance to find the mass of the rod. Next we create a calorimeter from insolation, which holds our cold store of water. We want to maximize the change in temperature, to minimize our uncertainty, so we use the smallest mass of water. We place the rod in a pan of water and heat it up using a hot plate, while measuring the temperature. We heat the rod until it is in steady state and thus at its maximum temperature. We take a measurement for the temperature of the water and drop the heated rod into the cold bath, taking care to avoid losing water and cooling the rod in the environment. The components then come to thermal equilibrium and we measure the final temperature.
Note, in all experiments except the calorimetry, we treat the power resister as a permanent portion of the rod.

3.2 Experiment to Determine Percent Power Lost in Resistor

Using the vacuum tube and an insolation cap we seal the rod completely, leaving only the face with the power resistor exposed. Inside the tube there is the aluminum rod, water, and a thermocouple to measure the temperature of the water. We now turn the power resistor on and record the power being output by the bench-top power supply, while simultaneously logging the data from the thermocouple. We make sure the power being supplied is small so that the water and the rod heat up at the same rate. Once we observe a temperature change of 20K we stop the experiment.

3.3 Experiments of a Rod Cooling in Air

These are a series of experiments with different orientations and rod conditions, that follow very similar procedures. We place our rod on our lab stand and raise it. We attach 6 temperature sensors at regular intervals down the length of the rod. These sensors are read by an Ardiuno into CSV files at 1Hz. We log data from the temperature sensors to establish a baseline for the room temperature. Now we turn on the power resistor and set the power output by the power supply to 10W. We record the voltage and current being drawn by the circuit. We try to limit the changes to the environment. Periodically, we plot the data to see if we are in steady state. Once in steady state, we remain there for 10 minutes to obtain a long sample time for the steady state calculations. Now we turn the power supply off and take cooling data [1] as the rod is allowed to cool back to the ambient temperature. For the insulated rod experiments, the rod will most likely not be able to reach steady state on a reasonable timescale. As such take data for at least two hours and adjust your time varying simulations.

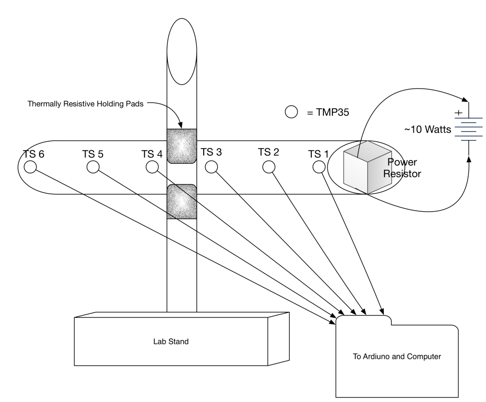

Figure 1: Major experiment setup

[1] Not actually used in data analysis
[2] These experiments had very noisy data and were eventually discarded

	Steady State	Time Varying
Horizontal Rod	✓	✓[2]
Horizontal Black Painted Rod	✓	✓
Vertical Rod	✓	
Vertical Black Painted Rod	✓	✓
Horizontal Insulated Rod		✓
Vertical Insulated Rod		✓[2]

Table 2: Experiments Conducted

3.4 Experiment to Determine the Power Lost to Evaporation

To determine the power lost from a wet surface at steady state we modify the previous experiment. First, we setup the same experiment as a horizontal rod cooling in air, but we add a piece of cloth that runs down the length of the rod. Now we use the steady state simulation to establish a baseline for values of K, K_C, and ϵ. Then we run the same experiment again with the bottom of the cloth hanging in a trough of water, allowing the cloth to wick up water through it and keep the surface of the aluminum wet. We now run the same simulation as before with the same K, K_C, and ϵ. The graph of this will not fit until we add back the power lost due to evaporation, we adjust this value until the model matches the steady state data we obtained.

4 Analysis

4.1 Determining Heat Capacity

The analysis for this experiment is relatively straight forward as the heat capacity is found directly from analysis of the calorimetry equation.

$$C_{Alu} = \frac{m_{Water} C_{Water} (T_f - T_{i_{Water}})}{m_{Alu}(T_f - T_{i_{Alu}})} \tag{5}$$

4.2 Determining Percent Power Lost in Resistor

In order to find the percent power lost in the resistor, we use a combination of calorimetry and simple conservation of energy. The power going into heating the rod must be equal to the power being supplied less the power being lost due to the environment. We isolate the system well, to ensure that the only point of power loss is at the power resistor. This portion of the rod should be set up identical to the rod cooling in air experiments.

$$P_{heating} = (m_{alu} C_{alu} + m_{water} C_{water}) \frac{\delta T}{\delta t} \tag{6}$$

$$P_{loss} = 1 - \frac{P_{heating}}{P_{supply}} \tag{7}$$

4.3 Parameters Obtained from a Rod Cooling in Air

This series of experiments requires more complex analysis as well as modelling of the system. We obtain three useful series of data[3] from each trial, the data as the rod heats up, and the data from

[3] This would have been five but we abandoned the Vertical Insulated Rod and the Horizontal Rod

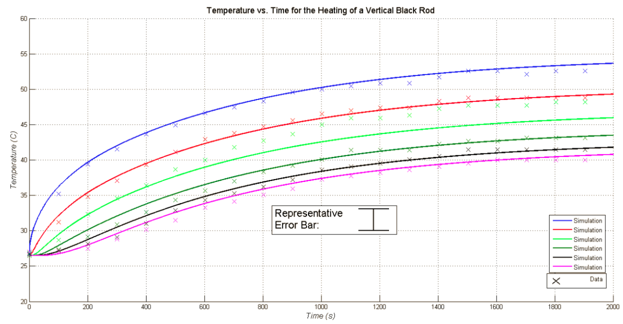

Figure 2: Heating of a Vertical Black Rod

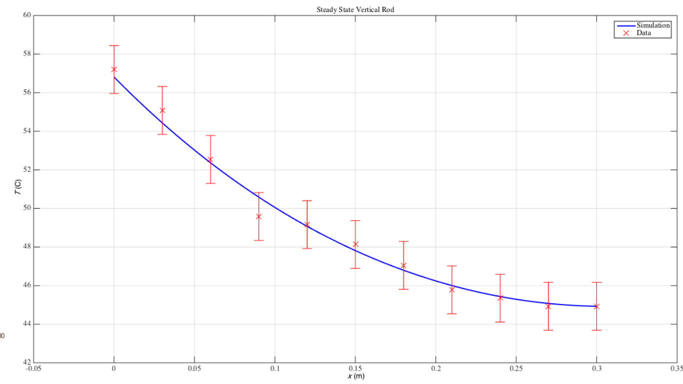

Figure 3: Heating of an Insulated Rod

the rod in steady state. We can solve the heat equation using a finite differences calculation, and use this to generate a temperature series for various points along the rod over time. We use the heat equation to model the transfer of heat over time in stead of the steady state conduction equation 3.

$$\frac{\partial T}{\partial t} = \frac{K}{C_p}\frac{\partial^2 T}{\partial x^2} \tag{8}$$

This simulation takes several input parameters to try and fit to our data. The simulation takes, K, K_C, ϵ, P_{in}, P_{Loss_R}, and C_p all as inputs. We take our data and estimates for these values from other experiments and vary the parameters until we arrive at a visually satisfactory fit. The second series of data is averaged over at least ten minute in steady state. This gives us a snapshot of the temperature down the rod. The crucial difference between this simulation and the previous one is the ability to ignore the power going into heating the rod. At steady state the power we supply is exactly equal to the power the rod loses, and thus the rod is at a constant temperature. We can vary three parameters to get our model to fit the data and they are K, K_C, and ϵ. Using this we simulate the power being input into the rod and try to see if that number is consistent with other results, as a secondary check for the thermal contact resistance. The code and the remainder of the figures are in appendices B and C.

Different rod setups provide us with different parameters based on assumptions and configurations of the rods. The following table lists each experiment and the variables that were obtained from the combination of experiment and simulation. Some of these experiments were not used to obtain the values but rather to create confidence bounds on the values, see discussion.

Original Experiments	Params Varied	Params Held Constant
Horizontal Rod Steady State	ϵ_s, K_{C_H}	K, $[K_{C_H}, \epsilon_s]$ Varied Together
Horizontal Black Painted Rod Steady State	ϵ_b, K_{C_H}	K, $[K_{C_H}, \epsilon_b]$ Varied Together
Vertical Rod Steady State	ϵ_s, K_{C_V}	K, $[K_{C_V}, \epsilon_s]$ Varied Together
Vertical Black Painted Rod Steady State	ϵ_b, K_{C_V}	K, $[K_{C_V}, \epsilon_b]$ Varied Together
Horizontal Insulated Rod Time Varying[4]	K	$K_C = 0$,$\epsilon = 0$, C_p, P_{loss_R}

Table 3: Parameters from each experiments

[4]This simulation has a power loss term which is varied to get to the correct curvature, and we hold the two standard power loss parameters to zero. This is to compensate for the fact that we don't understand how heat leaves the rod in the insulated rod case.

Consistency Checks	Parameters Verified
Horizontal Black Paint Rod TV	ϵ_b, K_{C_H}, K, C_p, P_{loss_R}
Vertical Black Painted Rod TV	ϵ_b, K_{C_V}, K, C_p, P_{loss_R}

Table 4: Consistency Checks

4.3.1 Determining the Power Lost to Evaporation

The data analysis for power lost to evaporation is two successive steady state simulations. The first one gives us an estimate for the new K_C and $\epsilon_{Effective}$ of the rod. We assume K is the same as in other experiments because the cloth cannot effect the internal properties of the rod. This then allows us to fit our model to the data obtained from the first dry cloth experiment.

We then take our model and parameters and plug them in to the wet cloth simulation, we expect that we will undershoot the actual data. We add in additional power that was lost to evaporation until the simulation matches up with the data. This power term is the power lost due to evaporation at steady state. We can then obtain the rate of evaporation from the rod. As a consistency check we measured the mass of the water at the beginning of the experiment and the end to provide and upper bound on the power lost based on the latent heat of evaporation. We expect the power term to be well inside this value and it is. It is not possible to do an effective time varying simulation of this power loss because of its strong correlation with temperature and ambient wind flow. The environment in which the experiments were conducted are not controlled enough to provide meaningful data in this situation. We can find the evaporation rate from:

$$Rate_{Evap} = \frac{P_{Loss_R} \times P_{delivered}}{LatentHeat_{fusion}} \tag{9}$$

5 Discussion

5.1 Calibration and Offsets

To deal with non-ideal sensors and inconsistent offsets between temperature sensors, we did a calibration between the temperature sensors and a MCP9808 high accuracy temperature sensor. We attached them together and added thermal paste. Then we placed them in a beaker of water that we heated while stirring constantly. This data series provided us with a mapping between the Ardiuno analog voltage reading and the actual temperature. To deal with the asymmetry in resolution we averaged the temperature per Ardiuno voltage reading. This gives us a reliable way to deal with non linear behaviour of our sensors as well as offset them in the correct direction. The code for this is in appendix D.

5.2 Uncertainty Propagation

The uncertainties for each experiment involve similar propagation and can be found in appendix A. There are sections for the uncertainty in Calorimetry, Percent Power Lost in Resistor, Time Varying Temperature data, and Steady State Temperature Data. The uncertainties are taken from combining our offsets and the various reading uncertainties in all of the apparatus.

5.3 χ^2 Analysis

We compute χ^2 by adding the squares of the difference between the model and the data and dividing by the uncertainty found. This method was used for to compute χ^2 for the steady state simulations.

$$\chi^2 = \frac{1}{\sigma^2} \sum_{i=1}^{n} (Model_i - Data_i)^2 \tag{10}$$

We use the "chi-by-eye" method to find satisfactory fits for the time varying simulations.

5.4 Confidence Intervals

We know the bounds on our emissivity are 0 for a minimum and 1 for a maximum. We use our simulation comparing two rods with the same orientation, but different surface emissivity conditions to find bounds for emissivity and K_C. For the same value of K_C, we find that the emissivity of the black painted rod has to be higher than the emissivity for sand-blasted rod, to obtain a similar fit. Using this fact we know that the lower bound on ϵ_b is the upper bound on ϵ_s . We now set the emissivities to their extreme values and find the bounds for the K_C. This provides us with new information again because the K_C must be constant between aluminum surface states and the K_C for horizontal rod is greater than the K_C for the vertical rod. This method gives us rather large bounds, but this is because the different modes of heat transfer are difficult to separate. We compared to the literature with our values, and decided ones that provided the best fit and were reasonable given what we know about the expected values of emissivity from the material and the expected values for K_C from the geometry of the rod.
For the other bounds we set all but one variable constant, we vary this variable until χ^2 changes by 1. This gives us a quantitive way of saying when a fit is no longer viable.

6 Summary of Results

These are the final parameters we found along with their respective uncertainties.

Constant	Value	Confidence Interval
K_{Alu}	190 $Wm^{-1}K^{-1}$	[175, 210]
K_{C_H}	12.5 $Wm^{-2}K^{-1}$	[10.5, 14.5]
K_{C_V}	11 $Wm^{-2}K^{-1}$	[9.5, 13.5]
ϵ_s	0.2	[0, 0.35]
ϵ_b	0.9	[0.35, 1]
P_{Evap}	5.8 W	[4.9, 6.7]

Table 5: Results - Rod in Air Experiments

Constant	Value
C_p	860 ± 130 $Jkg^{-1}K^{-1}$
P_{Loss_R}	0.70 ± 0.1
$Rate_{Evap}$	9.1 $g\ hour^{-1}$

Table 6: Results - Other Experiments

7 References and Acknowledgements

I would like to acknowledge that this lab was largely done by Team 11, including the setup, experimentation, and data analysis. I would also like to acknowledge Alex Goddijn with whom I developed the bulk of the time varying simulations.

A Uncertainty Analysis

A.1 Calorimetry

Uncertainties in directly measured quantities:

$$\delta_{V_{water}} = 0.01L \; ; \; \delta_{M_{water}} = \delta_{V_{water}} * 1kg \; ; \; \delta_{M_{al}} = 0.005kg;$$

Uncertainties in quantities taken from temperature plots. These were estimated by looking at variation in plots

$$\delta_{Water_{T_i}} = 0.3K \; ; \; \delta_{T_f} = 0.3K \; ; \; \delta_{Rod_{T_i}} = 1K;$$

$$\delta_{T_{Alu}} = \sqrt{\delta_{T_f}^2 + \delta_{T_{Rod_i}}^2} \qquad \delta_{T_{Water}} = \sqrt{\delta_{T_f}^2 + \delta_{T_{Water_i}}^2}$$

$$\delta_Q = \sqrt{(C_{water}\delta_{T_{water}} m_{water})^2 + (\delta_{T_{water}} C_{water} \delta_{m_{water}})^2}$$

A.2 Percent Power Lost in Resistor

We find the uncertainity in power going into the resistor by progating the uncertainity from the voltage and current readings from the power supply.

$$dP_{out} = \sqrt{((m_{al}C_{al} + m_{water}C_{water})\delta Slope)^2 + (Slope m_{al}\delta C_{al})^2 + (Slope C_{al}\delta m_{al})^2 + (Slope C_{water}\delta m_{water})^2}$$

$$\delta P_{frac} = \sqrt{(\frac{\delta P_{in}}{P_{in}})^2 + (\frac{P_{in}}{P_{out}^2}\delta P_{out})^2}$$

A.3 Rod Cooling in Air

Uncertainities in rod dimensions

$$\delta_r = 0.05\text{E-}3 \; m$$

$$\delta_l = 0.003m$$

$$\delta_{T_{amb}} = 2K$$

$$\delta_{V_{Ardiuno}} = 0.5(integer)$$

$$\delta_{V_{TMP}} = 1(integer)$$

Maximum and minimum temperature readings

$$maxReading = 170 \; minReading = 50$$

$$slopeConversion = 0.4523 \pm 0.01$$

$$\delta_{T_{max}} = sqrt(\delta_{Reading} SlopeConversion)^2 + (maxReading \delta_{Conversion})^2$$

$$\delta_{T_{min}} = sqrt(\delta_{Reading} slopeConversion)^2 + (minReading \delta_{Conversion})^2$$

Uncertainty in the placement of our sensors along the length of the rod.

$$\delta_{Position} = 0.005m$$

Final uncertainty in temperature measurement

$$\delta_T = \frac{\delta_{T_{min}} + \delta_{T_{max}}}{2} = 1.2K$$

B Final Figures

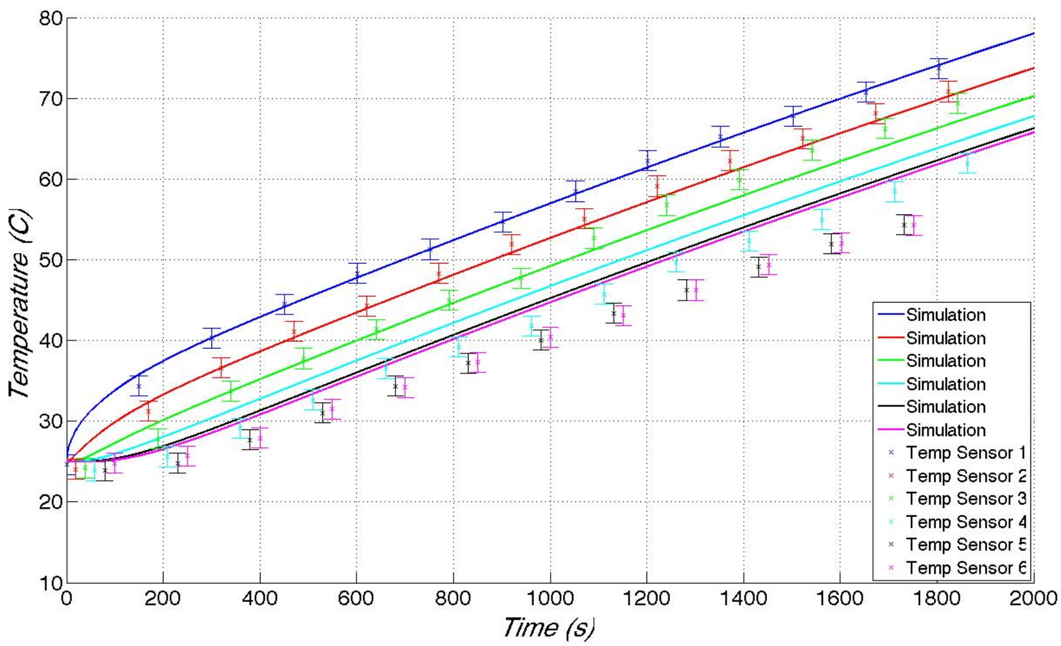

Figure 4: Horizontal Insulated Rod - Time Varying Simulation

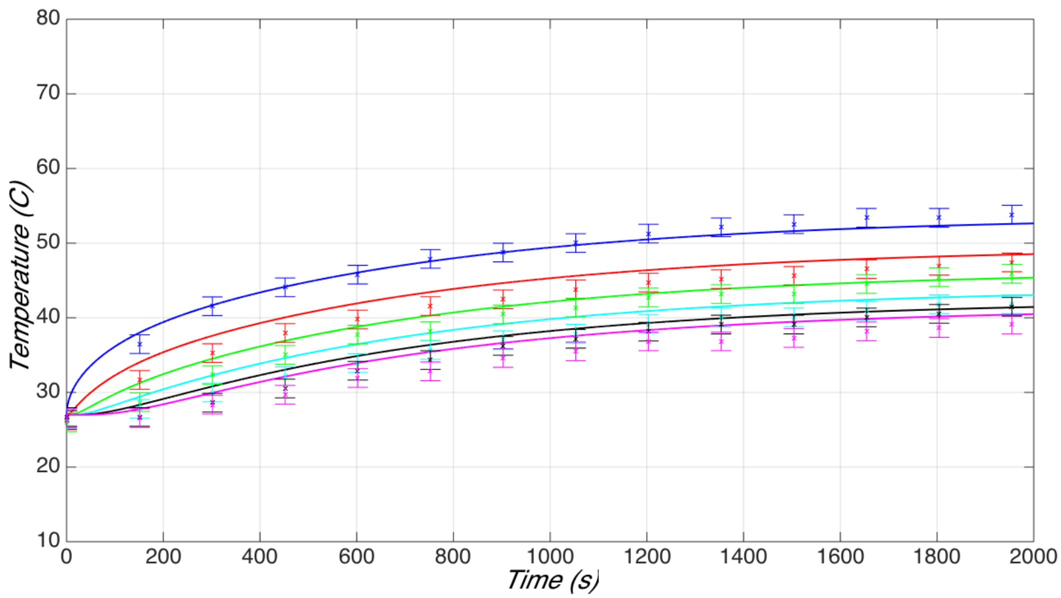

Figure 5: Horizontal Black Painted Rod - Time Varying Simulation

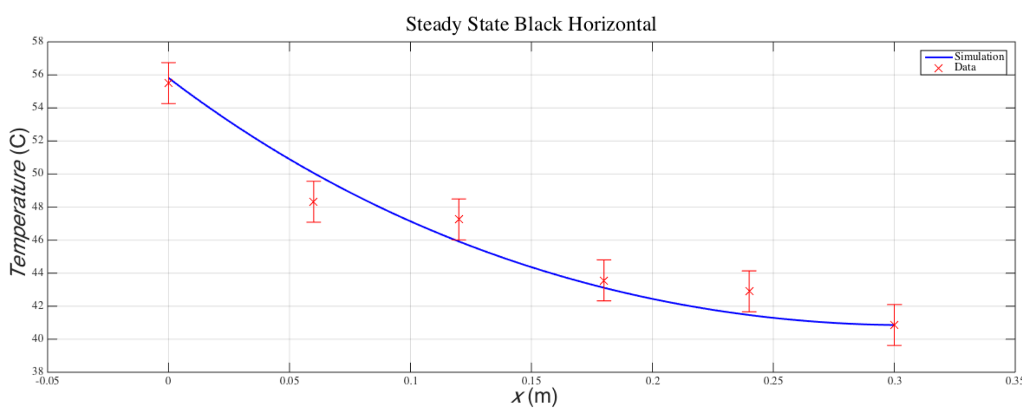

Figure 6: Horizontal Black Painted Rod - Steady State Simulation

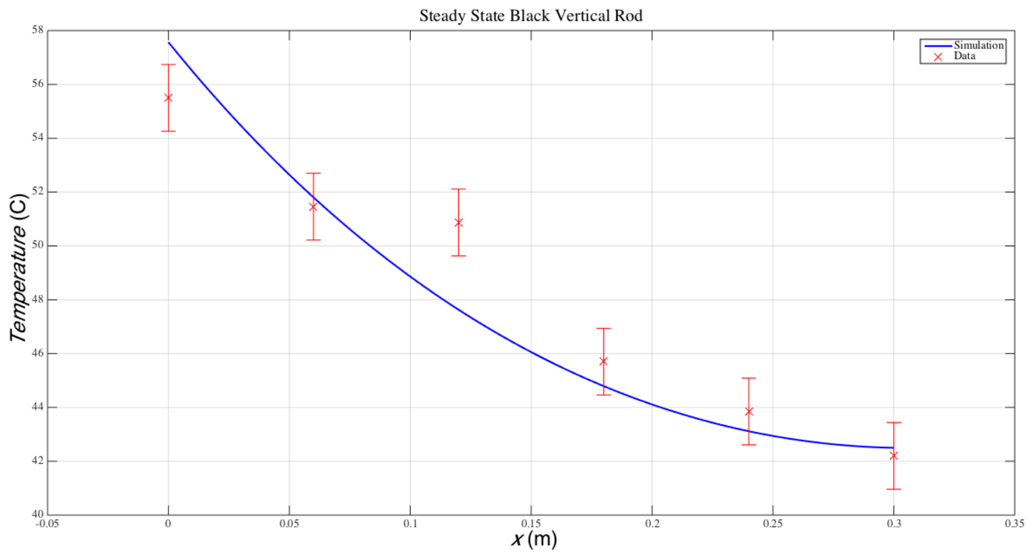

Figure 7: Vertical Black Painted Rod - Steady State Simulation

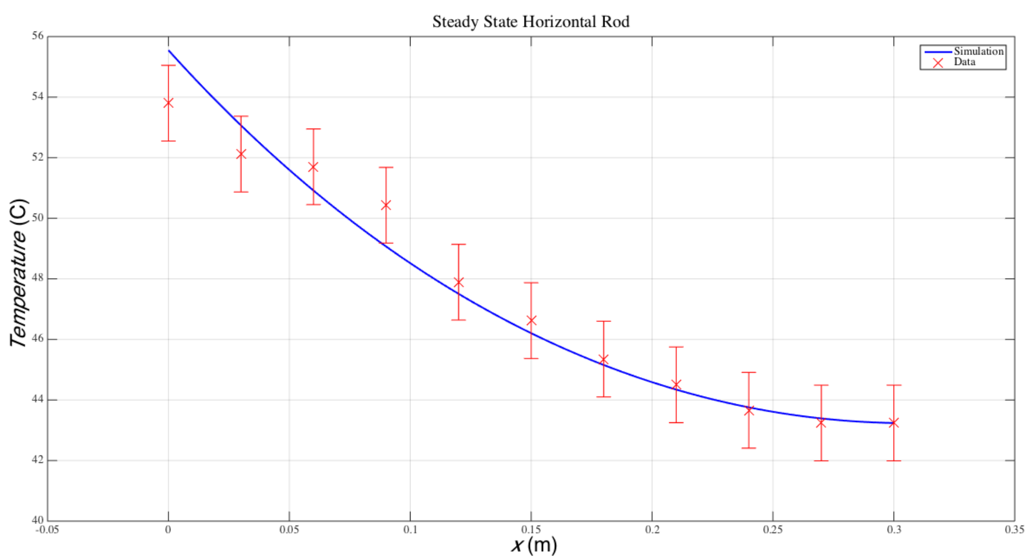

Figure 8: Horizontal Rod - Steady State Simulation

C MATLAB Simulation Code

In this section there will be a representative simulation from each kind of simulation we wrote. The specific simulations given here are Vertical Rod Steady State, Vertical Black Painted Rod Time Varying, Calorimetry, and Percent Power Lost in Resistor.

C.1 Temperature of a Rod at Steady State

Heating Along Vertical Sand Blasted Rod at Steady State

Contents

- Rod Geometry
- Thermodynamic Properties
- Calculations
- Plotting the Simulation
- Plotting the Collected Data
- Residuals

Rod Geometry

```
r = 0.02245/2;  (m) Radius of rod - Assumed (Given in class)
D = 2*r;  (m) Diameter
l = 0.3;  (m) Length of rod
ndiv = 300;  Number of divisions for approximation
dx = l/ndiv;  (m)
A = pi*r^2;  (m^2) Area of rod
S = 2*pi*r*dx;  (m^2) Surface Area of one segment
```

Thermodynamic Properties

Measured Ambient Temp in Lab

```
T_amb = 273 + 25;  (K)
 Calculated from time varying experiment
k_rod = 190;  (W/m/K)
Varied to make curve fit data
kc = 13.5;  (W*m^2*K)

Calculated from comparison of painted to unpainted rod
e = 0.0;
Stefan-Boltzmann Constant
o = 5.67E-8;  (W/(m^2 K^4)
```

Calculations

```
x = linspace(l, 0, ndiv+1);
T = zeros(1, ndiv+1);
P_flow = zeros(1, ndiv+1);
```

```
Steady State temp at unheated end of rod, measured
T0 = 273+44.93;  (K)
T(1) = T0;
dT = T(1) - T_amb;

Initial Effective Convection
keff = kc * dT + e*o*(T(1)^4-T_amb^4);
```

$$K_{eff} = K_c \delta T + \epsilon\sigma(T_1^4 - T_{amb}^4)$$

```
 Find power out of the end of the rod
P_out  = keff * (S+A); (W)
```

$$P_{out} = K_{eff} * (Area_{surface} + Area_{end})$$

```
P_flow(1) = P_out; (W)
 Power flow in and out are same
P_flow(2) = P_flow(1); (W)
 Find Tin from our equation
T(2) = P_flow(2)*dx/(k_rod*A) + T(1);  (K)
```

$$T_2 = \frac{P_{flow} * \delta x}{K_{rod} A + T_1}$$

```
Ptot = P_out; (W)

 Stepping through the length of the rod

for i = 2:ndiv

   P_out = P_flow(i); (W)

   dT = T(i)-T_amb; (K)

   keff = kc * dT+ e*o*(T(i)^4-T_amb^4);

   P_loss = keff * S; (W)
    Summing total power loss from rod to check for consistency with thermal contact resistance
   Ptot =Ptot+P_loss; (W)
    Find the new P_in
```

```
   P_flow(i+1) = P_out + P_loss; (W)
    Find Tin from our equation
   T(i+1) = P_flow(i+1)*dx/(k_rod*A) + T(i); (W)
```

$$T_{i+1} = \frac{P_{flow_{i+1}} * \delta x}{K_{rod}A + T_i}$$

```
end
```

Plotting the Simulation

```
figure(1)
plot(x, T-273, 'Linewidth', 2,'color','b');
hold
xlabel('{\it Position} (m)','FontSize',30)
ylabel('{\it Temperature} (C)','FontSize',30)
set(gca, 'FontSize', 20)
set(gca, 'FontName', 'TimesRoman')
title('Steady State Vertical Rod','FontSize',13);
```

```
Current plot held
```

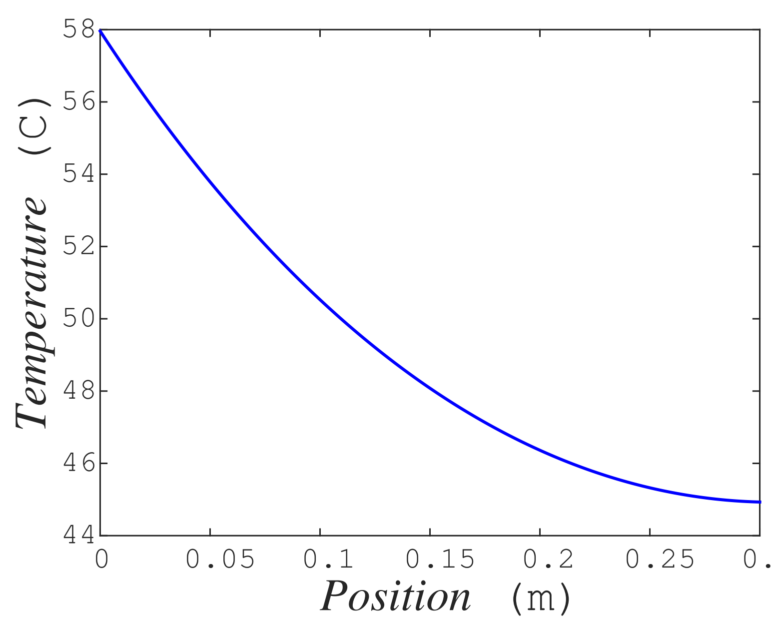

Figure 9: Vertical Steady State Simulation

Plotting the Collected Data

```
Xdat = linspace(0,0.30,11);
Ydat = [57.2 55.08 52.54 49.58 49.16 48.13 47.05 45.78 45.35 44.93 44.93];
Ytemp = Ydat;

error = 1.24;
figure(1)
hold on;
grid on;
errorbar(Xdat,Ytemp,error*ones(length(Xdat),1),'x','MarkerSize',12,'color','r');
legend('Simulation','Data');
```

```
 Observations from graph
 plot fits for lower data very well, but at higher temperatures it is
 slightly off. This may be due to not allowing the sensor to reach steady
 state when measuring the hot end.
```

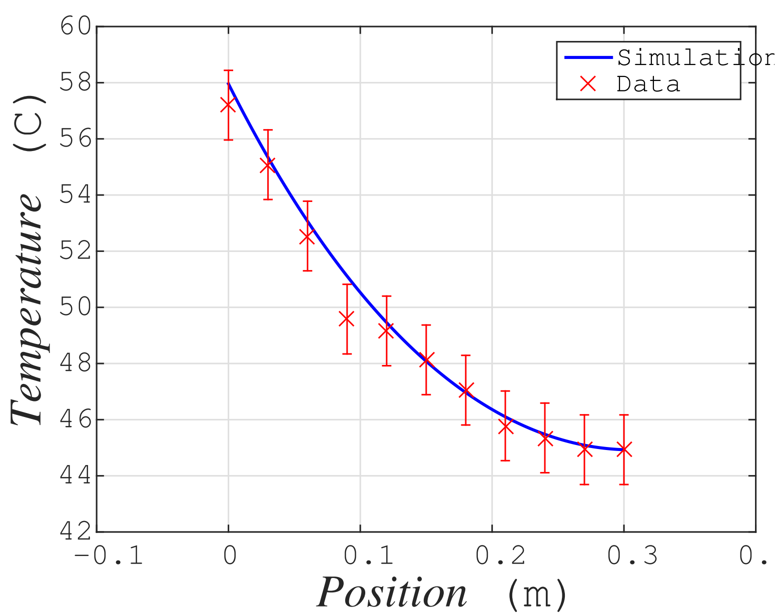

Figure 10: Vertical Steady State Simulation and Data

Residuals

```
sigma = 1.24;
[chiSquare, residuals] = squareSumAnalysis(Ydat, T, sigma);
```

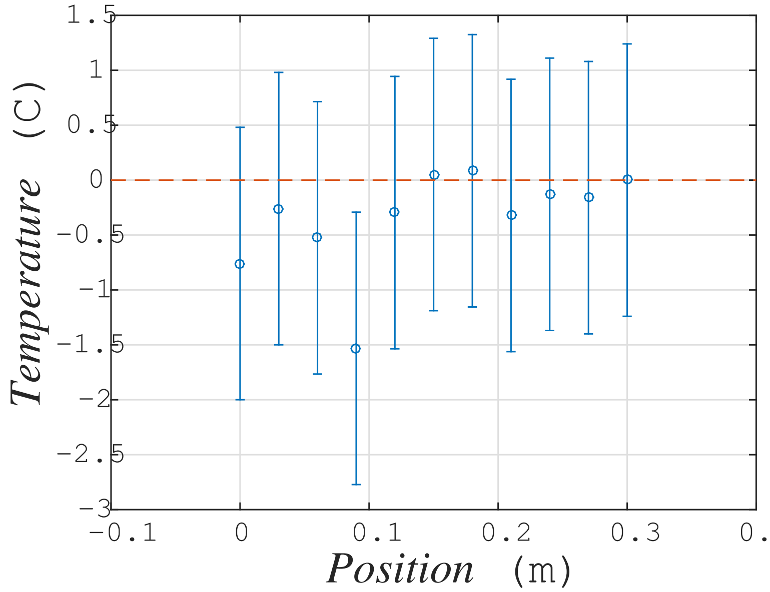

Figure 11: Vertical Steady State Simulation Residuals

C.2 Time Varying Simulation for a Rod

Time Varying Simulation for a Vertical Black Painted Rod

Plotting the temperature of six points along the rod over time

Contents

Defining Parameters for our Finite Differences Calculation

```
Nx = 120; Number of steps in length
Nt = 300000; Number of steps in time
t0 = 0; Start time (s)
tf = 10000; End time (s)
x0 = 0; Start length (m)
xf = 0.3; End length (m)
spacing = 0.06; (m) - real world physical sensor placement
dx = (xf - x0)/(Nx); Position slice size
dt = (tf - t0)/(Nt); Time slice size
```

Known parameters

Found from other experiments, scientifically accepted constants, or measured in lab.

```
a = 0.011225; Radius of the rod (m)
SBc = 5.67e-8; Stefan-Boltzmann constant (W/(m^2K^4))
Tamb = 273+26.5; Ambient temperature (K)

rho = 2.7e3; Density of Al (kg/m^3)
midpoint_P_loss = 0; No longer a factor in the simulation
midpoint_size = 5;
```

Simulation Parameters

```
PercentPowerTransfered = 0.72;
kc = 12.5; Convection coefficient of horizontal Al rod (W/(m^2K))
k = 195; Constant of conductivity of Al (W/(mK))
kc_end = 12.5;
e = 0.9; Emmisivity of Black Al rod
Cp = 857.05; Specific heat capacity of Al (J/(K kg))
```

Defining Temperature Equations

$$\Delta Temperature_{conv} = \frac{K_c \times 2(T_x - T_{amb}) \times \delta Time}{C_p \rho a}$$

```
Temperature change due to convection (K)
dT_convec       = @(Tx) (kc*2*(Tx-Tamb)*dt)/(Cp*rho*a);
dT_convec_end   = @(Tx) (kc_end*(Tx-Tamb)*dt)/(Cp*rho*dx);
```

$$\Delta Temperature_{rad} = \frac{\epsilon \times \sigma \times 2(T_x^4 - T_{amb}^4) \times \delta Time}{C_p \rho a}$$

Temperature change due to radiation (K)

```
dT_rad          = @(Tx) (e*SBc*2*(Tx.^4-Tamb^4)*dt)/(Cp*rho*a);
dT_rad_end      = @(Tx) (e*SBc*(Tx.^4-Tamb^4)*dt)/(Cp*rho*dx);

Temperature change due to midpoint loss (K)
dT_midpoint     = @(Tx) (midpoint_P_loss*(Tx-(273+35))*dt)/(Cp*rho*dx);
```

$$\Delta Temperature = \frac{Power \times \delta Time}{C_p \rho a^2 \times \delta X}$$

```
Temperature change in slice heating (K)

dT              = @(P) (P * dt)/(Cp * pi*a^2*dx*rho);
```

Simulation Code

```
Constants
Power into the left side of the rod (W)
Pinl = 10.625 * PercentPowerTransfered;

Storage
Array o temp over time, indices are (x,t)
T = zeros(Nx, Nt);

Initial Setup
Set all temperatures to Tamb
T(:,1) = ones(Nx,1)*(Tamb);

for t = 1:Nt-1
    power in to rod
    T(1,t+1)=T(1,t)+dT(Pinl)+((k/(Cp*rho))*(T(2,t)-T(1,t))./dx^2)*dt;
```

```
    temperature changes due to conduction along rod
    T(2:Nx-1,t+1)=T(2:Nx-1,t)+(k/(Cp*rho))*((T(3:Nx,t)-2*T(2:Nx-1,t)+T(1:Nx-2,t))./dx^2)*dt;
    T(Nx,t+1) = T(Nx,t)-((k/(Cp*rho))*(T(Nx,t)-T(Nx-1,t))./dx^2)*dt;

    temperature changes due to loss in convection and radiation along rod
    T(1:Nx/2-(midpoint_size+1),t+1)=T(1:Nx/2-(midpoint_size+1),t+1) - dT_convec(T(1:Nx/2-(midpo
    T(Nx/2-midpoint_size:1:Nx/2+midpoint_size,t+1)=T(Nx/2-midpoint_size:1:Nx/2+midpoint_size,t+
    T(Nx/2+(midpoint_size+1):Nx,t+1)=T(Nx/2+(midpoint_size+1):Nx,t+1) - dT_convec(T(Nx/2+(midpo

    At cold and hot end of rod
    T(Nx,t+1)=T(Nx,t+1) - dT_convec_end(T(Nx,t+1)) - dT_rad_end(T(Nx,t+1));
end
```

Plotting Simulation

```
Plotting the time varying simulation at the x positions corresponding to
the sensor placements in our actual data collection

timeSim = linspace(0,tf,Nt);

figure(1);
grid on;
hold on;
sensorPositions = 1:20:Nx;
plot(timeSim, T(sensorPositions(1),:)-273,'linewidth',2,'color','b');
plot(timeSim, T(sensorPositions(2),:)-273,'linewidth',2,'color','r');
plot(timeSim, T(sensorPositions(3),:)-273,'linewidth',2,'color','g');
plot(timeSim, T(sensorPositions(4),:)-273,'linewidth',2,'color','c');
plot(timeSim, T(sensorPositions(5),:)-273,'linewidth',2,'color','k');
plot(timeSim, T(sensorPositions(6),:)-273,'linewidth',2,'color','m');


xlabel('\it{Time (S)}','FontSize',13);
ylabel('\it{Temp (C)}','FontSize',13);

startTime = 310;
endTime   = length(time);
time1 = time(startTime:endTime)-(startTime-1);
T1 = temp1(startTime:endTime);
T2 = temp2(startTime:endTime);
T3 = temp3(startTime:endTime);
T4 = temp4(startTime:endTime);
T5 = temp5(startTime:endTime);
T6 = temp6(startTime:endTime);

plotInterval = 150;
error = 1.25;
```

```
Estimate of error in temperature made by propagating voltage
uncertainty of the temperature sensors, and the arduino readings.

errorbar(time1(1:plotInterval:end),T1(1:plotInterval:end),...
    error*ones(length(time1(1:plotInterval:end)),1),'x');
errorbar(time1(20:plotInterval:end),T2(20:plotInterval:end),...
    error*ones(length(time1(20:plotInterval:end)),1),'x','color','r');
errorbar(time1(40:plotInterval:end),T3(40:plotInterval:end),...
    error*ones(length(time1(40:plotInterval:end)),1),'x','color','g');
errorbar(time1(60:plotInterval:end),T4(60:plotInterval:end),...
    error*ones(length(time1(60:plotInterval:end)),1),'x','color','c');
errorbar(time1(80:plotInterval:end),T5(80:plotInterval:end),...
    error*ones(length(time1(80:plotInterval:end)),1),'x','color','k');
errorbar(time1(100:plotInterval:end),T6(100:plotInterval:end),...
    error*ones(length(time1(100:plotInterval:end)),1),'x','color','m');


plot (time1, T1 );
plot (time1, T2 );
plot (time1, T3 );
plot (time1, T4 );
plot (time1, T5 );
plot (time1, T6 );

 Specifying the specifics of the graph
axis([0,2000,20,60]);
set(gca,'FontSize',18);
title('Temperature vs. Time for the Heating of a Vertical Black Rod','FontSize',13);
xlabel('\it{Time (s)}','FontSize',30);
ylabel('\it{Temperature} (C)','FontSize',30);
h = legend( 'Simulation','Simulation','Simulation','Simulation',...
'Simulation','Simulation','Temp Sensor 1','Temp Sensor 2',...
'Temp Sensor 3','Temp Sensor 4',...
'Temp Sensor 5','Temp Sensor 6');
set(h,'Location','SouthEast');
legend('Location','SouthEast');
```

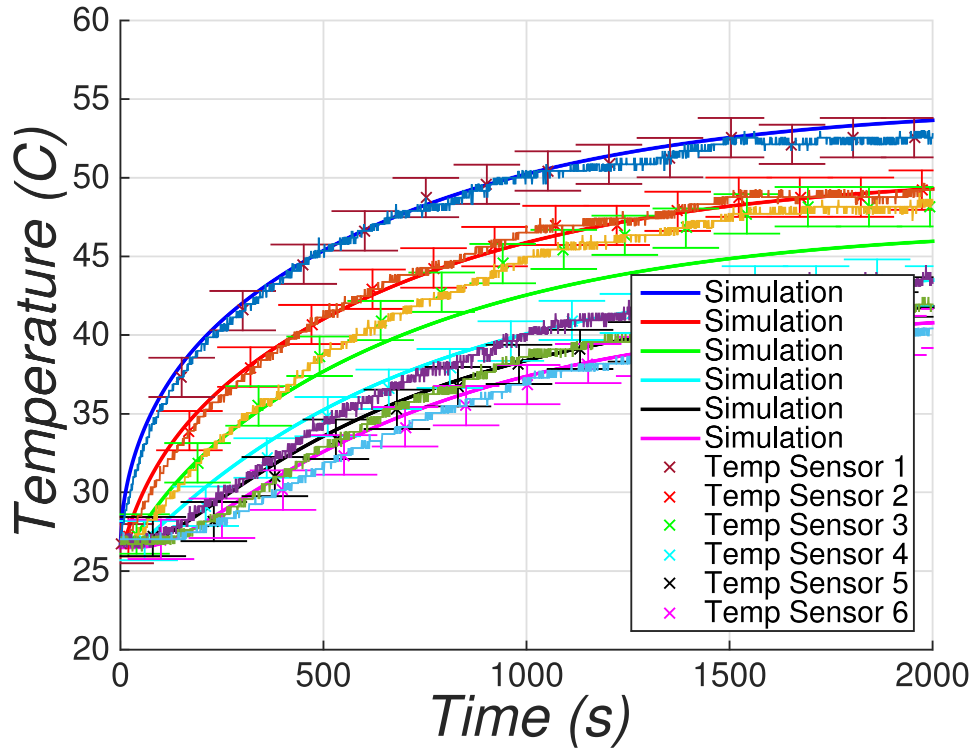

Figure 12: Vertical Black Time Varying Simulation

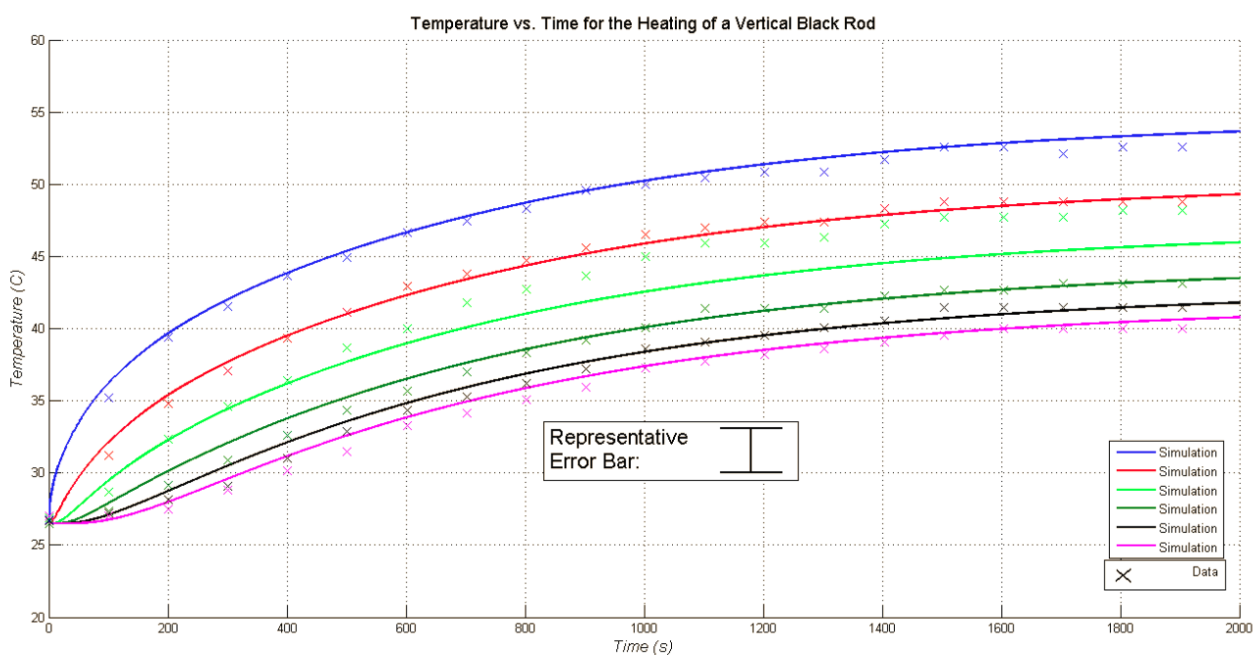

Figure 13: Vertical Black Time Varying Simulation - With Adjusted Error Bars

C.3 Data Analysis for Calorimetry

Data Analysis for Calorimetry

Contents

- Reading Temperatures from the Thermocouple
- Use the initial stable temperature to find our initial water temp
- Add Constants for our calculations

```
 Import data from our experiment to give us our data for the heating and
 initial temperature of the rod as time and temp

 Import data from our experiment to give us our data for the steady state
 then heat transfer to the water as tBox and Vbox
```

Reading Temperatures from the Thermocouple

From curve fitting our calibration data for our thermocouples, we see that the voltage can be mapped to temperature using the equation $T = 0.02502 * V^{1.252} + 11.35$

```
Tbox = 0.02502*Vbox.^1.252+11.35;
```

Use the initial stable temperature to find our initial water temp

```
stableT = Tbox(1:140);
WaterT0 = mean(stableT);

 We noted that our cooling was almost stable at around 300 seconds so take
 this range to see final temp

endT = Tbox(300:540);

 We calculate the mean tempurature of two different sections of time that
 showed our stable final temperature. These two regions were separated by
 a discontinuity in the measurement caused by issues with the
 thermocouple. We take the average of these two sections of time.
endTemp1 = mean(endT(1:75));
endTemp2 = mean(endT(190:240));
Tf = (endTemp1 + endTemp2)/2;

 Find our final temperature for the rod
rodTemp = temp(950:1189);
RodT0 = mean(rodTemp);

 However we found that because of the hot glue and the positioning of the
 temperature sensor. We were not reading exactly the temperature of the
 rod. After taking our data, we once again heated the rod until we saw the
 same reading from the end temperature sensor, then we took a reading
```

```
 from the middle of the rod with another temperature sensor. We found that
 the rod was actually reading a temperature 5 degrees lower without the
 water in the connection interface and the hot glue surrounding it.
 Because of this we add an adjustment factor of -5 degrees to our rod
 initial temperature.
RodT0Adj = RodT0 - 5;
```

Add Constants for our calculations

```
VWater = 0.5;  (L) volume of water used
mWater = VWater * 1;  (kg)) mass of water used
cWater = 4186;  (J/kg K) heat capacity of water
mAl = .32;  (kg) mass of rod

dTAl = (Tf - RodT0Adj);
dTWater = (Tf - WaterT0);

QWater = dTWater * cWater * mWater;
cAl = -QWater/mAl/dTAl;

fprintf( 'Heat Capacity of aluminum is: f.\n\n', cAl);
```

```
Heat Capacity of aluminum is: 857.045569.
```

C.4 Percent Power Lost Data Analysis

Percent Power Lost Data Analysis

Contents

Power Resistor Information

```
i_in = 0.85;  (A) current into resistor
v_in = 12.5;  (V) voltage into resistor
p_in = i_in * v_in;  Power into resistor
```

Thermodynamic Information

```
VWater = 0.37;  (L) volume of water used
mWater = VWater * 1;  (kg)) mass of water used
cWater = 4186;  (J/kg K) heat capacity of water
mAl = 0.326;  Mass of Aluminum
cAl = 857.046;  Aluminum Heat Capacity
```

Plotting and Fitting the Data

First trim of all trailing zeros in our dataset

```
time = Time(2600:4402);
voltage = Voltage(2600:4402);

temperature = 19.64*voltage.^1.252+11.35;

plot( time, temperature);

 Now fit the data with a line
[xData, yData] = prepareCurveData( time, temperature );

 Set up fittype and options.
ft = fittype( 'poly1' );

 Fit model to data.
[fitresult{1}, gof(1)] = fit( xData, yData, ft );

coefficients = coeffvalues(fitresult{1});

 From curve fitting, obtain slope of this linear portion
slope =  coefficients(1);  (K/s) rate of temperature change

p_out = (mAl*cAl + mWater * cWater)*slope;

p_ratio = p_out/p_in;

fprintf( 'The power delivery ratio is: f.\n\n', p_ratio);
fprintf( 'The percent power lost is: f.\n\n', 1-p_ratio);
```

```
The power delivery ratio is: 0.705006.

The percent power lost is: 0.294994.
```

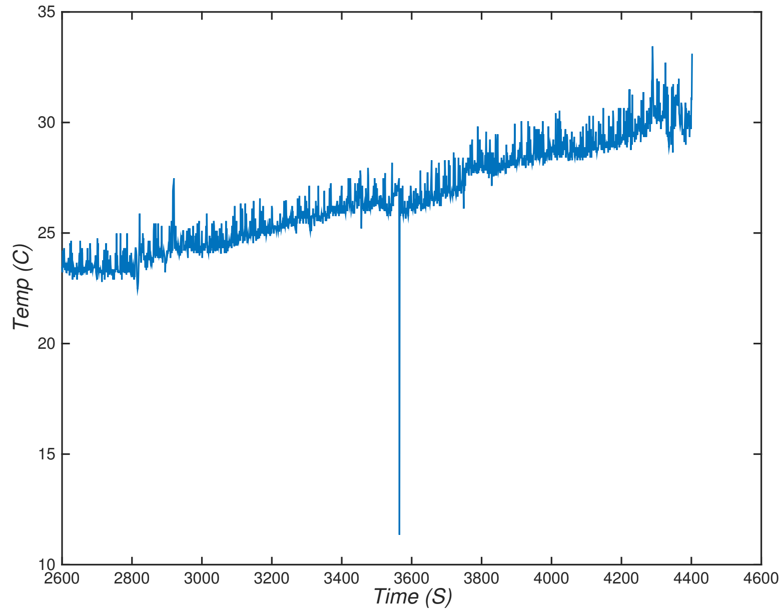

Figure 14: Water Temperature during Percent Power Lost Experiment

D TMP35 Sensor Calibration Matlab Code

```
% -----------------------------------------------------------------------
%TO USE THIS CODE, MUST FIRST IMPORT THE CSV DATA FROM THE 6 TEMP SENSORS.
% -----------------------------------------------------------------------

%Takes the arduino output from the 6 temperature sensors, converts them
%to degrees celcious, and shifts the data to eliminate offsets in the data.
%Temperature sensor 1 is taken to be the zero point that all the other
%data is shifted to at room temp.

%Vectors containing the slope and intercept corrections for each of the sensors relative
%to sensor 3. These values were found by plotting the readings of each
%sensor vs the reading of sensor 3 in the curve fitting toolbox (as the sensors were being collectively heated up to 80 degrees)
%, and finding the slopes and intercepts of the data.
slopeDifs = [0.935420000000000;0.994920000000000;1;0.967420000000000;1.052100000000000;0
intercepts = [4.456700000000000;0.901140000000000;0;3.963300000000000;-1.420900000000000;

%These two values correspond to the conversion factor necessary to shift
%temperature sensor 3's reading to the proper temperature, given by the
%MCP9808 temperature sensor. They were found the same way that slopeDifs
%and intercepts were found.
slopeConversion = 0.4523;
interceptConversion = -0.2385;

%below are the functions that convert from the arduino number corresponding
%to the voltage of each sensor, to the temperature in degrees celcius.
```

```
V1 = temperature1*slopeDifs(1)+intercepts(1);
V2 = temperature2*slopeDifs(2)+intercepts(2);
V3 = temperature3*slopeDifs(3)+intercepts(3);
V4 = temperature4*slopeDifs(4)+intercepts(4);
V5 = temperature5*slopeDifs(5)+intercepts(5);
V6 = temperature6*slopeDifs(6)+intercepts(6);

temp1 = slopeConversion*V1 + interceptConversion;
temp2 = slopeConversion*V2 + interceptConversion;
temp3 = slopeConversion*V3 + interceptConversion;
temp4 = slopeConversion*V4 + interceptConversion;
temp5 = slopeConversion*V5 + interceptConversion;
temp6 = slopeConversion*V6 + interceptConversion;
```

E Uncertainty Calculation Matlab Code

E.1 Uncertainty in Temperature Value Given by TMP35 Sensors

```
%% Uncertainty Analysis for our Rod Steady State Temperature Fits:

%% Uncertainties in rod dimensions:
%These are insignificant to our simulations as they are relatively small uncertainties
dr = 0.05E-3; % (m)
dl = 0.003; % (m)

%% The uncertainty in the ambient temperature will also be present in our
%simulation. We can however, assume that this uncertainty is
dT_amb = 2; % (K)

%% Uncertainty in voltage readings of arduino sensors
dArduino = 0.5; % (arduino number)
%Our TMP uncertainty was determined by observing fluctuations in the
%baseline for a steady state set of voltage data.
dTMP = 1; % (arduino number)
dReading = sqrt(dArduino^2+dTMP^2);

%% Maximum and minimum temperature readings
%Values in arduino numbers
maxReading = 170;
minReading = 50;

%% Conversion factors taken from our TMP35 calibration results
slopeConversion = 0.4523;
%This value is an estimate in the uncertainty of the slope conversion factor
%from arduino numbers to real temperatures
dConversion = 0.01;
dTmax = sqrt((dReading * slopeConversion)^2+(maxReading*dConversion)^2);
dTmin = sqrt((dReading * slopeConversion)^2+(minReading*dConversion)^2);

%% Uncertainty in the placement of our sensors along the length of the rod.
dPosition = 0.005; % (m)
%% Final uncertainty in temperature measurement
%This value is the average of the maximum
%uncertainty, and the minimum uncertainty.
dT = (dTmin + dTmax)/2;

```

```
fprintf('Our uncertainty in each temperature measurement is: %.1f
   kelvin.',dT);
```

E.2 Uncertainty in Specific Heat Capacity Value

```
% Thermal Contact Uncertainty Analysis

% Uncertainties in directly measured quantities
deltaVwater = 0.01;
deltaMwater = deltaVwater * 1;
deltaMal = 0.005;

%Uncertainties in quantities taken from temperature plots. These
    were
%estimated by looking at variation in plots
deltaWaterT0 = 0.3;
deltaTf = 0.3;
deltaRodT0 = 1;

% Propagated uncertainty for calculated values:
deltadTAl = sqrt((deltaTf)^2+(deltaRodT0)^2);
deltadTWater = sqrt((deltaTf)^2+(deltaWaterT0)^2);

deltaQwater = sqrt((cWater*deltadTWater*mWater)^2+(dTWater * cWater
    * deltaMwater)^2);
deltaCal = sqrt((deltaQwater/mAl/dTAl)^2+(QWater/mAl^2/dTAl*deltaMal
    )^2+(QWater*deltadTAl/mAl/dTAl^2)^2)
```

E.3 Uncertainty in Power Delivery Fraction

```
% Thermal Contact Uncertainty Analysis

dVin = 0.2;
dIin = 0.02;
dMal = 0.005;
dVwater = 0.01;
dMwater = 1 * dVwater;
dCal = 130;

dPin = sqrt((dVin * i_in)^2 + (dIin * v_in)^2);

dVoltage = 0.015;
%Temperature = 19.64*Voltage.^1.252 So:
% dTemperature = 19.64*1.252*Voltage^0.252*dVoltage, but since voltage
% varies we will find upper and lower bounds:

Vupper = 1.1;
Vlower = 0.65;
dTupper = 19.64*1.252*Vupper^0.252*dVoltage;
dTlower = 19.64*1.252*Vlower^0.252*dVoltage;

%Average the two:
dT = (dTupper + dTlower)/2;

%We extract an upper bound of the uncertainty on the slope of our line that
%we calculated based on having a maximum error on the slope from taking our
%vertical portion of the slope calculation off by twice dT, as we have
%error of dT for the first point, and error of dT for the last point. This
%is an overestimate, but for our purposes, it should work well as we are
%not super sure about the linearity of our data and that the slope will be
%consistent throughout the entire line.
dSlope2 = dT*2/(length(time));

dPout = sqrt((dSlope2*(mAl*cAl + mWater * cWater))^2+(slope*mAl*dCal)^2+(slope*cAl*dMal)^2 + (slope*cWater*dMwater)^2);

dPratio = sqrt((dPout/p_in)^2+(p_out/p_in^2*dPin)^2);
fprintf( 'The uncertainty in our power ratio is: %.6f.\n', dPratio);
```

E.4 χ^2 Calculation Code

```
%Returns a Chi Squared value and a vector containing all of the residual
%values

function [chiSquare, residuals] = squareSumAnalysis(data, T, sigma)
% Plotting residuals & sum of squares for the fit:
Num_points = length(data);
div_size = 300/(Num_points-1);

residuals = zeros(1,Num_points);

%Calculating the sum of the squares of all the residuals
squareSum = 0;
for i = 1:Num_points
    residuals(i) = data(i)- (T((301+div_size)-(div_size)*i)-273);
    squareSum = squareSum + (residuals(i))^2;
end

position = linspace(0, 0.3, Num_points);

chiSquare = squareSum/sigma^2;
figure;
errorbar(position, residuals, ones(1, Num_points)*sigma, 'o');
hold on;
plot(linspace(-0.1, 0.4, 10), zeros(1, 10), '--');
xlabel('{\it x} (m)','FontSize',13);
ylabel('{\it dT} (C)','FontSize',13);
set(gca, 'FontSize', 13)
set(gca, 'FontName', 'TimesRoman')
title('Residuals');
hold off;
```

F Illustrations of Experimental Setup

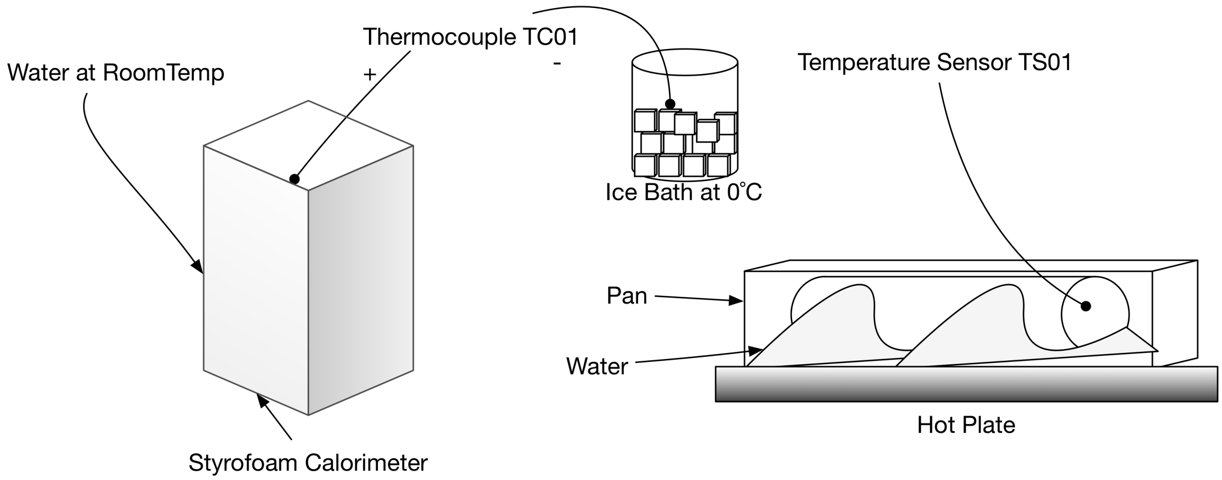

Figure 15: Setup for Calorimetry

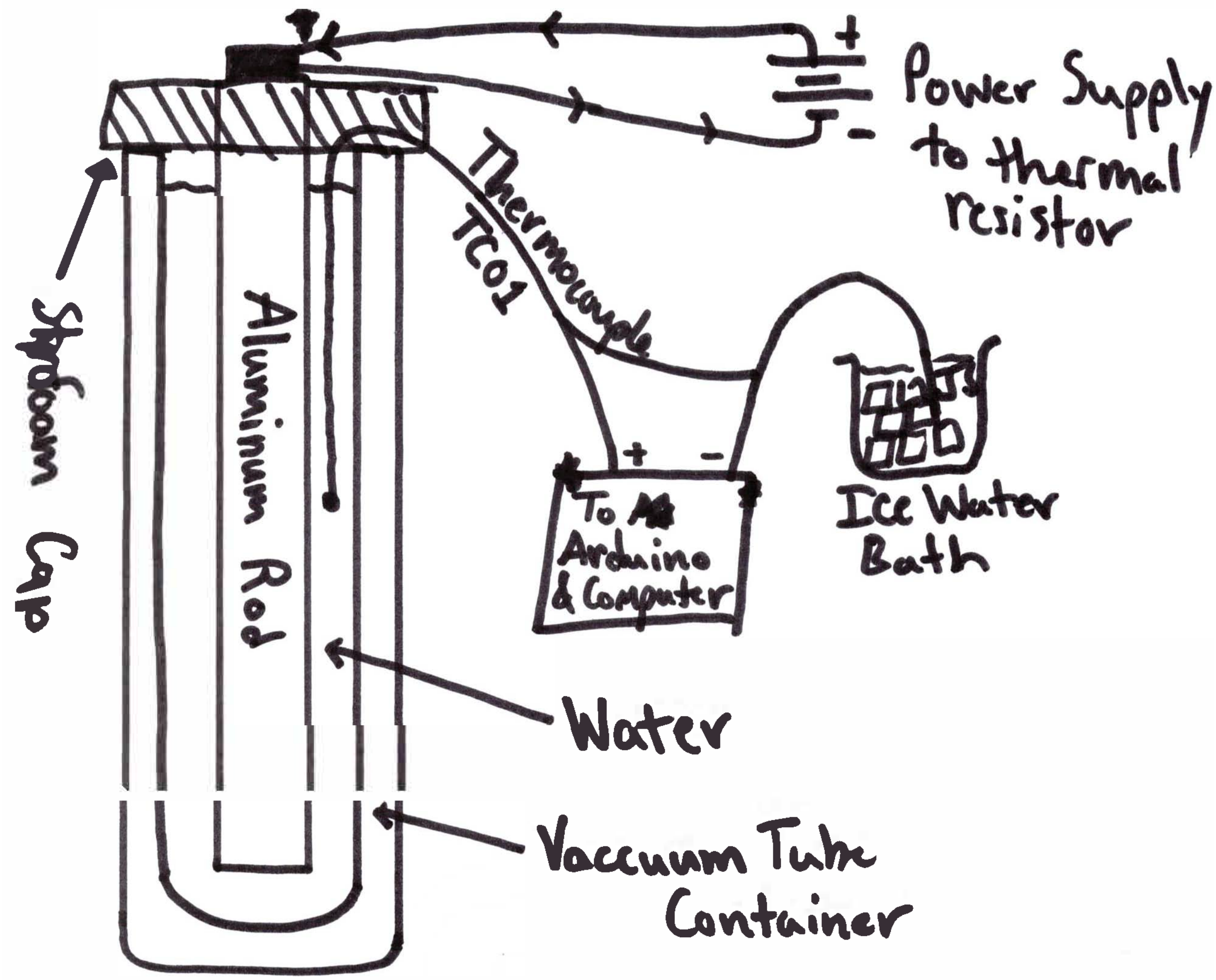

Figure 16: Setup for Percent Power Loss by the Resistor

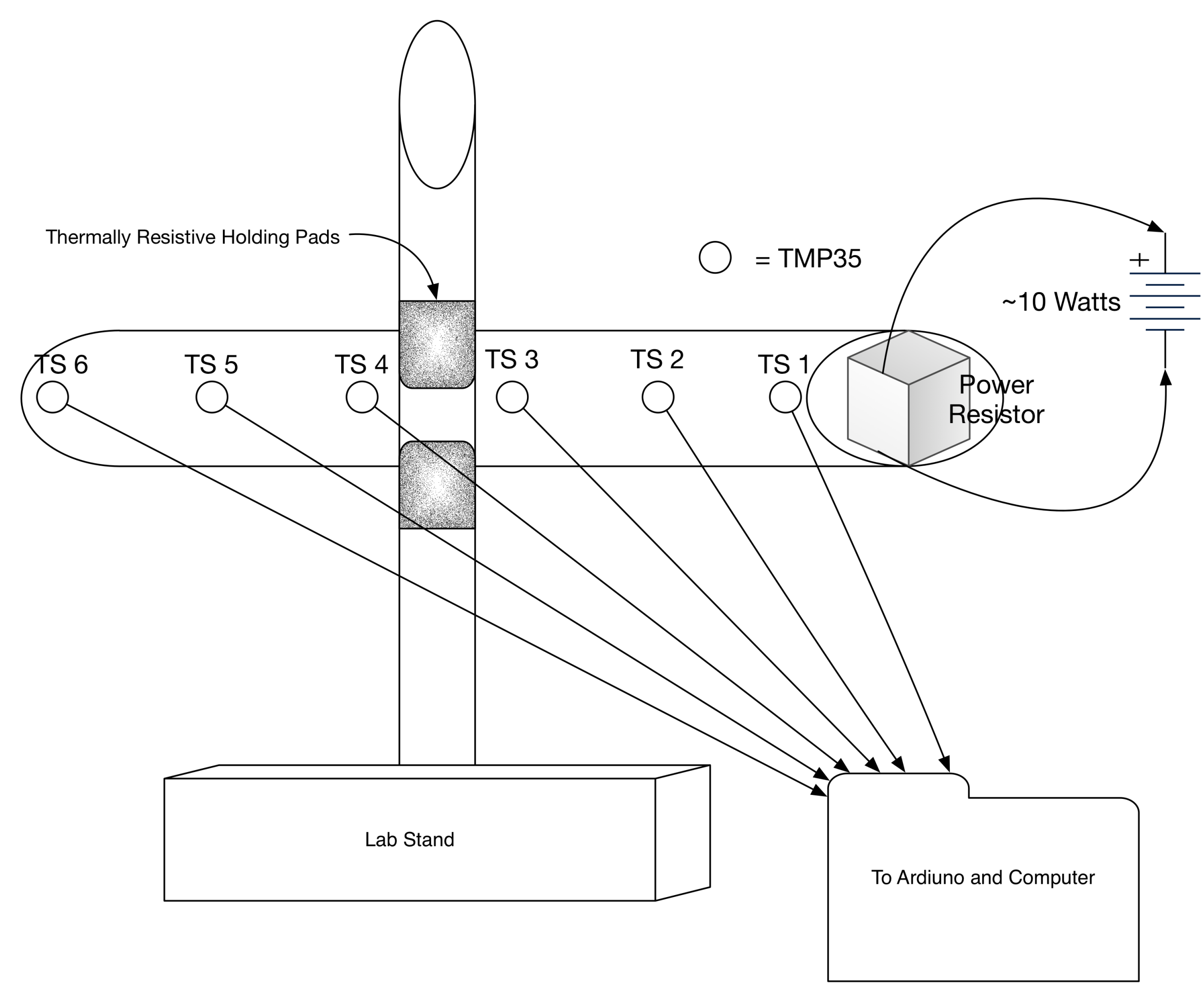

Figure 17: Setup for Major Experiment

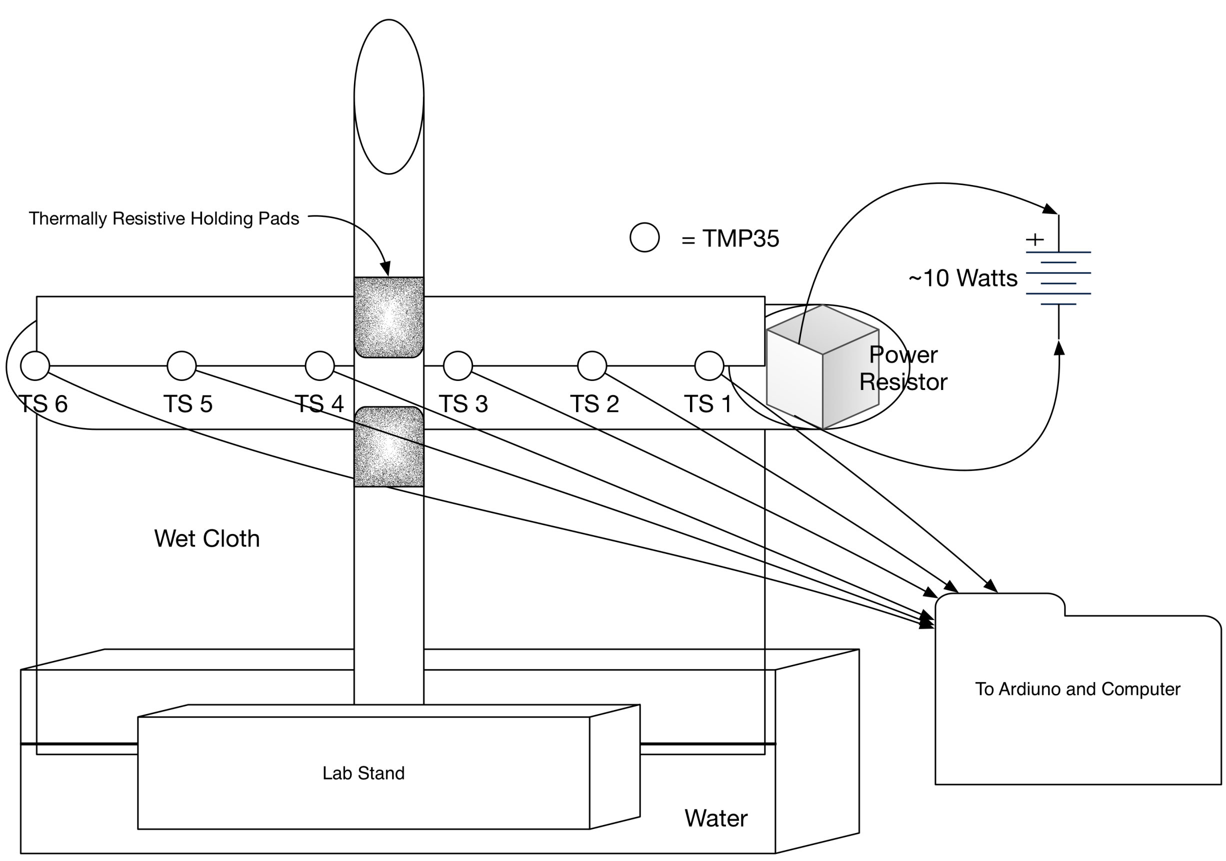

Figure 18: Setup for Power Lost to a Wet Surface

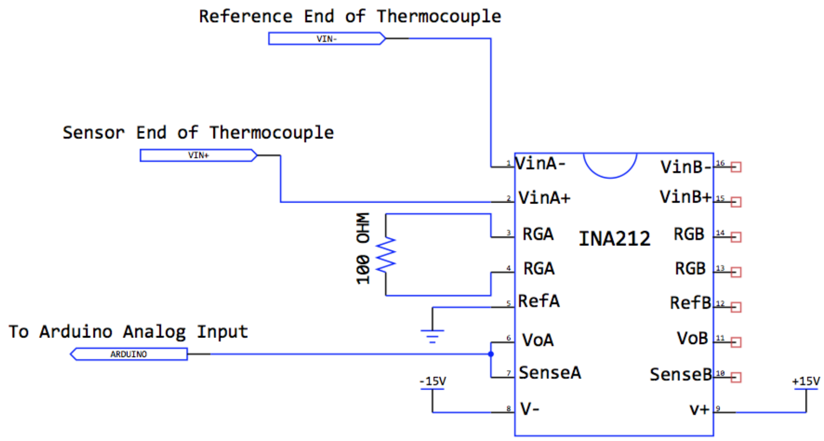

Figure 19: Circuit Diagram for Thermocouple Input

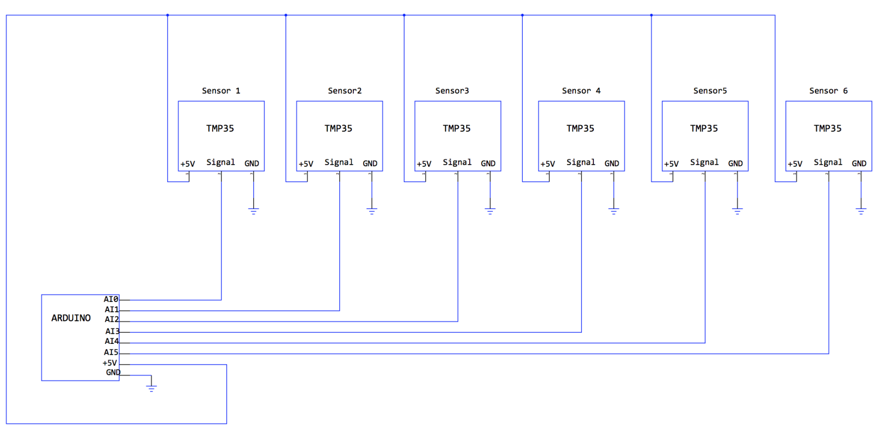

Figure 20: Circuit Diagram for Temperature Sensor Input