|  |  |
| --- | --- |
| **Team Member Names:** | Andrew Gan, Neel Lingam, John Papas Dennerline, Nick Sherman |
| **Purdue Logins:** | gan35, ilingam, jpapasde, nsherma |
| **Section Number:** | 012 |
| **Team Number:** | 26 |

**Your work will be graded using the following learning objectives:**

|  |
| --- |
| **Learning Objective (LO): 22.00 Reflect on feedback for the purpose of improvement**  ***Evidence of Proficiency Requires*:**   * Feedback summarization is clear and useful * Response plan is clear and practical |

|  |
| --- |
| **Learning Objective (LO): 12.00 Perform linear regression** |
| **Learning Objective (LO): 13.00 Perform function discovery and data transformations** |

|  |
| --- |
| **Learning Objective (LO): 07.00 Create and evaluate x-y plots suitable for technical presentation (this includes all appropriate sub-LOs)** |

|  |
| --- |
| **Learning Objective (LO): 21.02 Communicate ideas clearly and concisely**  ***Evidence of Proficiency Requires:***   * Purpose of communication is clear * Improvements are fully but concisely described   + All steps are included   + Appropriate technical language is used   + Clarifying images (e.g., sketches, graphs and/flow charts) are provided (as necessary) * External research is accompanied by an in-text citation and full reference |
| **Learning Objective (LO): 21.03 Evaluate model or algorithm development (e.g. ideas, work, functionality) using evidence-based rationales**  ***Evidence of Proficiency Requires:***   * Assumptions, claims, and critical decisions are clearly stated * An appropriate source of evidence is used to support assumptions, claims, and critical decisions * The evidence is clearly articulated * External research is accompanied by an in-text citation and full reference |

# M3 Feedback Review

1. In your own words, summarize the feedback you received on project milestone M3 that could lead to improvements in your work.

|  |
| --- |
| We received three main points of improvement for milestone 3. Improve the general parameter ID by minimizing the modified SSE values even further, increase comments for clarity, and improve the consistency of algorithm outputs (minimize standard deviation). These improvements target the issues regarding seemingly arbitrary threshold values and can lead to reevaluating how threshold values are implemented. |

1. Based on your feedback, what do you need to do to improve your parameter identification approaches? (Do not just reword your response to Part A. Do consider how you will incorporate your feedback into your work.)

|  |
| --- |
| In order to incorporate these changes, we plan to code a way to find the most accurate threshold for a specific data set, rather than have a single, arbitrary threshold that does not adapt to each data set. In order to improve clarity, we will code these changes into the algorithm and comment said changes. We feel that these changes will help minimize the standard deviation and SSE statistics we calculate from the algorithm, thus improving the parameters themselves. |

# Iteration

1. Improvements (remember to use metrics and rationales for each)

|  |
| --- |
| **Improvement 1** |
| Description  The method as outlined in Improvement 2 will improve the results for both statistical parameters and the temperature-based parameters. The method to accomplish this is identical to Improvement 2, which involves running the algorithm with a variable threshold component, modifying the step size and threshold repeat values.  By taking the optimal parameters and using those in the algorithm, we can find the best set of step size and threshold repeat value to use for the data. In addition to finding the optimal step size and threshold value, this iteration will return improved temperature parameters, since their accuracy directly depends on the accuracy of the statistical parameters, step size, and threshold repeat values. |
| Metrics To Determine Improvement  The effectiveness of this improvement will be evaluated based on how closely its returned temperature-based parameters match the ones that were given. While not the primary focus in this situation, the statistical parameters can also be considered, since they can indicate the precision of the algorithm as well. R\_Squared should be as close to 1 as possible (Coefficient of Determination, 2019), and SSEmod should be as close to zero as possible to achieve the best accuracy. |
| Rationale for Improvement and Metrics  It is appropriate to use a similar method to Improvement 2 because the improvement of the step size, threshold values, and statistical parameters will directly affect the outputs of the temperature-based parameters. A higher degree of accuracy (with respect to the given temperature parameters) is indicated by improved statistical parameters. Since Improvement 2 is designed to improve the statistical parameters, it will directly improve the temperature parameters as well. |

|  |
| --- |
| **Improvement 2** |
| Description  This improvement involves running the algorithm a predetermined number of times, with each run containing a modification in the step size and threshold repeat values (number of times in a row that conditions must be met).  Improvement 2 will run a known number of times instead of the variable threshold component of improvement 1. As with improvement 1, we will then find the optimized set of thresholds to use. This allows for a more accurate prediction on how long the algorithm will take to run and more accurate calculation of the parameters and statistics from the data.        For example, we could run the algorithm 15 times total, with each run containing a modification in the step size and threshold repeat values.  This would generate 60 parameters in total **for each set of data instead** of just 4. We would then take this data and run our SSEmod function for every set of the algorithm (resulting in 15 SSEmod values), since there are currently 1 SSEmod values for every 4 parameters generated. We would find the minimum SSEmod out of each set of 15, and find the corresponding data set and parameters associated with that SSEmod value. This would allow us to isolate the best, cleanest results from multiple thresholds of the algorithm to find the data that most closely represents the raw data. |
| Metrics To Determine Improvement  By finding the optimized set of threshold values, it will improve the calculation of the parameters by giving the best step size and threshold repeat value for the set of data given to us.  To measure the improvement of this improvement, we will analyze the statistical parameters. Most importantly, SSEmod and R\_Squared should be considered, as each refers to the (original) time vs. temperature and price vs. Tau data, respectively. SSEmod should be as close to zero as possible, and the R\_Squared should be as close to 1 as possible (Coefficient of Determination, 2019). The desired temperature-based parameters should also be more closely matched to the given ones, to obtain the greatest accuracy possible. |
| Rationale for Improvement and Metrics  Optimizing the SSEmod values, most likely by using a min() function and a for-loop to run the algorithm multiple times, will allow the cleanest given data for each dataset to be used. This in turn will yield the most precise and accurate temperature and time parameters. The threshold value and step size will be varied for each run as part of the effort to find the lowest SSEmod value, to accomodate for all raw data variance. Increasing the threshold value will verify that each moving average is increasing for a longer consecutive time. Increasing the step size of the moving average vector itself will allow for a more precise average calculation (movmedian, 2019). The precision of our modifications will be shown in the new SSEmod and R\_Squared values, which are useful as they assign a numeric value to indicate the precision of the data analysis.  Even though this improvement calculates new statistical and temperature parameters, the new results are still just as valid, as they are taken and analyzed from the same raw data. Also, since the statistical parameters should be improved, this iteration also functions as an improvement for temperature-based parameter identification. The only difference between this iteration and the previous algorithm is that the optimal step size and threshold repeat values are automatically (as opposed to manually) selected. |

# Parameter Identification Algorithm Improvements Pseudocode

1. **Algorithm Improvements Pseudocode**

|  |
| --- |
| **Algorithm Improvements Pseudocode** |
| Initialize paramArray as 4-by-5 matrix  Initialize counter as 0  Initialize sseMaxArray as 1-by-15 matrix  For consec\_thres from 4 to 8, increment by 1  For step\_size from 26 to 28, increment by 1  Get input data and validate  Define consec\_thresh to 6 as an algorithm constant  Define step\_size to 27 to determine the range of the moving average  Determine if heating or cooling  Assign initial average temperature to min or max temp based on heating or cooling  Create an initial array with step\_size elements with defined low and high indices  Initialize count at 0  While count is less than consec\_thresh  Calculate the average of the initial array and compare with initial array  if (heating and average increases OR cooling and average decreases)  Add one to count and increment indices by step\_size  else  set count back to 0  Set new vector with updated indices  Replace initial average temp with new average temp  end  Go back to the first time point at the last array of the loop and set equal to t\_s  Define y\_H or y\_L based on heating or cooling.  Create a vector of step\_size elements with defined low and high indices  Initialize count to 0  While count is less than consec\_thresh  Calculate the average of the first block of data  If (cooling and the average increases OR heating and the average decreases)  Add 1 to count  Else  Reset count at zero  Decrease endpoint indices by step\_size  Set average vector to correspond to new endpoint indices  Update initial temperature to equal the new calculated average temperature  If the temperature is heating  Go back to last time point of last array in loop. Set this equal to y\_H  (= index + step\_size \* consec\_thresh)  Else if temperature is cooling  Go back to last time point of last array in loop. Set this equal to y\_L  (= index + step\_size \* consec\_thresh)  Calculate temperature range using y\_H and y\_L  If the temperature is heating  Set “TauPoint”  = y\_L + 0.632(temperature range)  Else if temperature is cooling  Set “TauPoint”  = y\_H - 0.632(temperature range)  Initialize currentTemp as temperature at t\_s  Initialize count variable at t\_s location (an index).  While current temp is less than tauPoint and heating OR current temp is greater than tauPoint and cooling  Increment count variable by 1  Update current temp  Set tauTime equal to the time data at the count variable  Calculate Tau  Increment counter by 1  Store y\_L, y\_H, t\_s, and tau values in paramArray, column by column.  For x from 1 to the number of columns of paramArray  Call M2y\_of\_t\_012\_26, passing in timeData, tempData and column x of paramArray  Store returned value to predict\_tempData  Compute sum of squared difference between tempData and predict\_tempData divided by 10240  Store computed sum as sseMaxArray element index x  Return the column of paramArray with minimum sseMaxArray values as returnArray |

1. In MATLAB, translate your pseudocode into a user-defined function named **M4ParameterID\_*sss*\_**

Comment your code so it shows your two improvements.

# Parameter Identification Comparisons

**Table 1 Actual Parameter Values For Calibration Data**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | **HEATING** | | **COOLING** | |
| **Parameter** | **Unit** | **Clean** | **Noisy** | **Clean** | **Noisy** |
| **ts** | [sec] | 1.50 | 1.50 | 1.50 | 1.50 |
| 𝝉 | [sec] | 0.31 | 1.65 | 1.82 | 1.12 |
| **yL** | [°F] | 0.00 | -0.7 | 0.96 | -0.67 |
| **yh** | [°F] | 100.00 | 98.9 | 100.00 | 98.91 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **HEATING** | | **M3 UDF** | | **M4 UDF** |  |
| **Parameter** | **Unit** | **Clean** | **Noisy** | **Clean** | **Noisy** |
| **ts** | **[sec]** | **1.48** | **1.50** | **1.50** | **1.50** |
|  | **[sec]** | **0.33** | **1.50** | **0.31** | **1.56** |
| **yL** | **[°F]** | **0** | **-0.54** | **0** | **-0.54** |
| **yh** | **[°F]** | **100.00** | **96.45** | **100.00** | **97.93** |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **COOLING** | | **M3 UDF** | | **M4 UDF** |  |
| **Parameter** | **Unit** | **Clean** | **Noisy** | **Clean** | **Noisy** |
| **ts** | **[sec]** | **1.48** | **1.42** | **1.50** | **1.42** |
|  | **[sec]** | **1.81** | **1.12** | **1.79** | **1.15** |
| **yL** | **[°F]** | **0.95** | **0.57** | **0.95** | **-0.40** |
| **yh** | **[°F]** | **100.00** | **99.19** | **100.00** | **99.18** |

# SSEmod Comparisons

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **SSEmod** | **ACTUAL PARAMETERS** | | **M3 UDF**  **PARAMETERS** | | | **M4 UDF PARAMETERS** | |
| **Regime** | **Clean** | **Noisy** | **Noisy** | **Clean** | **Noisy** | | **Clean** |
| **Heating** | 0.00 degF2 | 0.85 degF2 | 3.19 | 0.377 | 1.30 | | 0.00262 |
| **Cooling** | 0.54 degF2 | 1.04 degF2 | 3.73 | 0.414 | 2.75 | | 0.261 |
| *Note:* Verify your SSEmod calculations. |  |  |  |  |  | |  |

# Statistics Comparisons

|  |  |  |  |
| --- | --- | --- | --- |
|  | **M3 Statistics** | |  |
| **Model Number** | **τ Characteristics** | | **Mean SSEmod [°F2]** |
| **Mean [sec]** | **Standard**  **Deviation [sec]** |
| **FOS-1** | 0.157 | 0.0361 | 0.5120 |
| **FOS-2** | 0.348 | 0.0321 | 0.4685 |
| **FOS-3** | 0.872 | 0.0596 | 0.6508 |
| **FOS-4** | 1.009 | 0.0831 | 0.8014 |
| **FOS-5** | 1.399 | 0.1543 | 1.4028 |

|  |  |  |  |
| --- | --- | --- | --- |
|  | **M4 Statistics** | |  |
| **Model Number** | **τ Characteristics** | | **Mean SSEmod [°F2]** |
| **Mean [sec]** | **Standard**  **Deviation [sec]** |
| **FOS-1** | 0.149 | 0.0312 | 0.3995 |
| **FOS-2** | 0.340 | 0.0367 | 0.4195 |
| **FOS-3** | 0.875 | 0.0523 | 0.4897 |
| **FOS-4** | 1.025 | 0.0784 | 0.5500 |
| **FOS-5** | 1.482 | 0.1103 | 0.6983 |

# Price vs. τ Regression Model Comparisons

1. **Regression Plots**

|  |
| --- |
| **M3 Regression Plot** |
| A screenshot of a cell phone  Description automatically generated |

|  |
| --- |
| **M4 Regression Plot** |
| A screenshot of a cell phone  Description automatically generated |

1. **Regression Results**

|  |  |  |
| --- | --- | --- |
|  | **M3 Results** | **M4 Results** |
| **Function Type** | Exponential | Exponential |
| **Linearized Data** | Log10(price) and tau (M = -1.028 , B = 1.337) | Log10(price) and tau (M = -0.982, B = 1.319) |
| **Linearized Form of The Function** | Log10(price) = -1.028(tau) + 1.337 | Log10(price) = -0.982(tau) + 1.319 |
| **SSE** | 2.30 | 91.05 |
| **SST** | 153.56 | 3071.13 |
| **r2** | 0.98 | 0.97 |
| **General Form of Best-Fit Equation** | Price = -1.028 \* (10 ^ (21.734 \* Tau)) | Price = -0.982 \* (10 ^ (20.836 \* Tau)) |

# Algorithm Efficiency

|  |  |
| --- | --- |
| **Algorithm** | **Execution Time (sec)** |
| **M3 UDF** | 0.000909 |
| **M4 UDF** | 0.0153 |

# Algorithm Insight

|  |
| --- |
| **Algorithm Insight Plot(s)** |
| A screenshot of a cell phone  Description automatically generated  Figure  A close up of a map  Description automatically generated  Figure |

|  |
| --- |
| **Algorithm Insight Explanation** |
| FIGURE 1:  The plot displays the tau vs. price coordinates of the M3 and M4 algorithms overlaid on top of each other. As seen in the plot, the M4 values have a tighter grouping than those of M3, indicating a more precise analysis of temperature parameters. Because M4 contains our modification of optimized statistical parameters, its final grouping of Tau vs. price points is tighter than that of M3. Additionally, this plot allowed us to see the appropriate pricing for each model of thermocouple, and where the actual price outlined by FOS matches it.  FIGURE 2:  This plot graphs the modified SSE for each thermocouple from M3 and M4.  By optimizing the 2 thresholds for step size and repeat values, we were able to minimize our SSE values for the data set in M4.  As shown in the graph below, the SSE values for all thermocouples were lower than the M3 counterparts. In addition, this shows that our revised algorithm in M4 was more accurate in determining the parameter values.  Knowing the prices of the thermocouples, this also shows us that the cheaper thermocouples have a greater SSE values, while the more expensive thermocouples have a lower SSE. This is expected because lower quality thermocouples should be less accurate than an expensive, high quality thermocouple. |

|  |
| --- |
| **References Used in Evidence-Based Rationales** |
| Coefficient of Determination. Mathworks, Inc. (2019). Retrieved April 10, 2019, from  https://www.mathworks.com/help/stats/coefficient-of-determination-r-squared.html  Movmedian. Mathworks, Inc. (2019). Retrieved April 10, 2019, from  https://www.mathworks.com/help/matlab/ref/movmedian.html |