***EEET– 425 Section 3***

**Digital Signals Processing**

**Rochester Institute of Technology**

**Professor** David Orlicki

**Lab #2**

**Student**: Aliana Tejeda

**September 10, 2018**

|  |  |
| --- | --- |
| LABORATORY GRADE | |
| Title, Name of Lab Partners, Two Column Format  Plots for grade B paper - complete set, correct labels, brief discussion  Plots for A paper - complete set, correct labels, brief discussion  Lab participation | \_\_\_\_\_ /20  \_\_\_\_\_ /40  \_\_\_\_\_ /20    \_\_\_\_\_ /20 |
| Final Grade | \_\_\_\_\_ /100 |

[[1]](#footnote-1)

D.S.P. Lab 1- Statistical Introduction and Hardware Setup

Aliana Tejeda, Member, CPET

*Abstract*—Microsoft excel is a robust application which can handle all of the statistical needs of this lab. The application also has an add-on tool called “Data-Analysis” which adds the feature of being able to chart a histogram of the data collected. This add-on allowed for a graphical view of the distribution of the signal data. The use of this application is very handy when it comes to analyzing the data collected from the LM61 temperature sensor from the Arduino application.

*Index Terms*— Microprocessors, Statistical Learning,

# INTRODUCTION

T

HE purpose of this lab was to introduce the statistical tools that Microsoft excel provides, and the hardware setup needed to the future labs. The hardware was that of connecting the LM61 temperature sensor to the colored wires and connect them in the correct order to the Red Yourduino board. After successful set up, testing of the hardware was conducted to make sure it is functioning as intended. The data collected from these testing was analyzed and graphically represented in Microsoft excel. This lab is an introduction to statistical learning, and how to work with microprocessors, and how to analyze the data from the microprocessor with Microsoft excel.

# LM61 Temperature Sensor Response

## Response to Finger Touch (500msec delay)

Fig. 1 shows the response of the LM61 temperature sensor to a finger touch over about 60 seconds. The sensor stayed at steady state for about 40 milli-seconds, and after 40 milliseconds, the temperature sensor was touched till it reached and peak of about 205. After it reached a peak of about 205, the finger was released from the sensor, and the voltage from the sensor slowly stepped down to a steady integer voltage around 188. The data collected from this sensor was sampled every 500 milliseconds with the use of the Arduino software, using the analogRead syntax.

## Interrupts & Control

Fig.2 displays the data collected from the temper sensor over a 60 second time interval, sampled at 100 milliseconds using an interrupt. The temperature sensor has a finger held onto the sensor for about 10 seconds then let to observe the sensor slowly ripple back down to stability.

## Sample Set Defined

Fig.3 displays the data collected from the temperature sensor when the Arduino application was given the parameter of only sampling 265 data samples and ceasing after such. The temperature sensor was still tested for the temperature sensing by having a finger held on the sensor for about 10 seconds, however the sensor was not given the time it needed to go fully back into a stable low temperature. Which is the reason for the high value of base line.

## Running Average & Standard Deviation

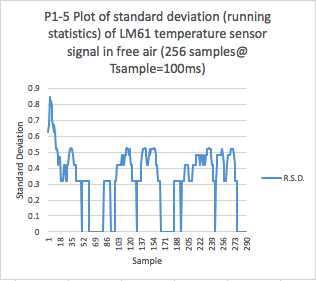
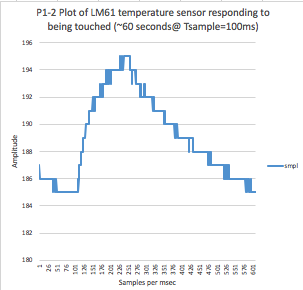
Fig.4 displays the Running Average, Fig.5 displays the standard deviation, and last Fig.6 is the source code of those calculations. The data collected from the temperature sensor had a running average of about 188 for the 20th sample was taken while the sensor was responding to a finger touch. The running standard deviation was very jumpy at the beginning but fell into a pattern after about 20 samples were taken. After the running standard deviation stabilized, it stayed between 0.5-0.6 standard deviation. The reason for this uniform instability is due to the fact that the data being collected was being compared to no previous data so the difference from 0 to a large number like 180 will cause instability in calculations of both the running mean and running standard deviation. The source code was based off of the interrupt code. To make analytic easier, a library called QuickStats was added. This library made calculating the standard deviation 4 steps instead of several.

Fig. 4. displays a line plot of the running mean. The running mean shoots up to about 183 in response to a finger touch and then gradually levels out to about 188 over time and stays there for the rest of the experiment.

Fig. 1. A line plot of the temperature sensor responding to finger touch. The data gradually climbs up to peak at 206 then gradually steps down back into its stable temperature of 185 once the finger is released. The resulting plot looks that of a ocean wave.

Fig. 2. A line plot of the temperature sensor responding to finger touch. The data gradually climbs with many ripples up to peak at 195 then gradually ripples down back into its stable temperature of 185 once the finger is released. The resulting plot looks that of a ocean wave.

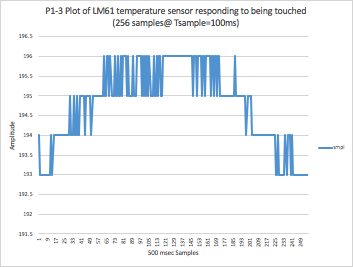


Fig. 3. A line plot of the temperature sensor responding to finger touch. sporadically ripples up to peak at 195 then gradually ripples down back into its stable temperature of 185 once the finger is released. The resulting plot looks that of a volcano.

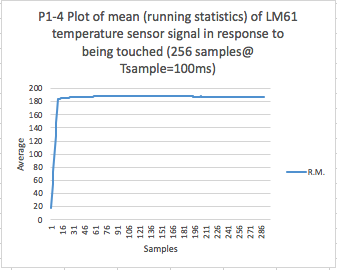


Fig. 5. This graph represents the running standard deviation of the data collected from the temperature sensor when it responds to a finger touch. The data starts out sporadically then settles down into a parttern at about a range of 0.3-0.53 stdev. Levels out at 0.3 stdev, and then dips all the way down to 0.00 stdev. Then it repeats this process for the rest of the time it was sampled.

#include <MsTimer2.h>

#include <QuickStats.h>

QuickStats stats; // initialize the class

const int TSAMP\_MSEC = 100;

const int numReadings = 10; // array size

const int NUM\_SAMPLES = 300;

volatile boolean sampleFlag = false;

int nSmpl = 1, sample;

float signalReadings[numReadings]; //making a array for the readings

int currentReading = 0; //the index of the reading we are currently at

int total = 0; //the running total

int averageReading = 0;

float stdev = 0;

int inputPin = A0;

void setup() {

// put your setup code here, to run once:

Serial.begin(9600);

Serial.println("Enter 'g' to go .....");

while (Serial.read() != 'g'); // spin until 'g' entry

// initialize all readings to zero

for(int initialReading = 0; initialReading < numReadings; initialReading++) {

signalReadings[initialReading] = 0;

}

MsTimer2::set(TSAMP\_MSEC, ISR\_Sample); // Set sample msec, ISR name

MsTimer2::start(); // start running the Timer

}

void loop() {

while (sampleFlag == false); // spin until ISR trigger

sampleFlag = false; // disarm flag: enable next dwell

// take away the last reading form the total

total = total - signalReadings[currentReading];

// read a new sample from the signal

signalReadings[currentReading] = analogRead(inputPin);

// add this reading to the total

total = total + signalReadings[currentReading];

// next array index

currentReading = currentReading + 1;\

//if i so happen to run out of space then i go back to the begining

if(currentReading >= numReadings) {

currentReading = 0;

}

// Calculate the averge here

averageReading = total / numReadings;

// Calculate the standard Deviation here

stdev = sqrt((1/numReadings)\*pow(total-averageReading,2));

// Display results to console

if (nSmpl == 1) Serial.print("\nR.M.\tR.S.D.\n");

Serial.print(averageReading); Serial.print('\t');

Serial.println(stats.stdev(signalReadings,numReadings)); Serial.print('\n');

if (++nSmpl == NUM\_SAMPLES) MsTimer2::stop();

delay(300);

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

void ISR\_Sample()

{

sampleFlag = true;

}

Fig. 6. This is the source code which calculated the running average and running standard deviation. The code was based off of the last interrupt code form the lab procedure.

# Q&A

Q2-2-1. Answer and address the following: Theoretically, what should the mean of the sum of the signals be equal to? Is this the case for the data given? (exactly or approximately?) (Why, or Why not?)

**A:** The mean of a sum of signals should be equal to those respective signal's means added together. So the mean of signal 1 plus the mean of signal 2 is equal to the mean of both those signals together. For the data given the mean for the data of "Signal 1 + Signal 2" is exactly what it should be, and that is the mean of signal 1 added to the mean of signal 2.

Q2-2-2. Answer and address the following: Theoretically, what should the variance of the sum of the signals be equal to? Is this the case for the data given? (exactly or approximately?) (Why, or Why not?)

**A:** Theoretically the variance of the sum of signals must be equal to the individual signals added together. This is the exact case for this data. The variance of (signal 1 + signal 2) is equal to the variance of signal 1 plus the variance of signal 2.

Q2-2-3 Is the standard deviation of the sum (sig1+sig2) equal to, less than or greater the sum of the standard deviations of the individual components sig1, sig2? (exactly or approximately?) (Why, or Why not?)

**A:** The standard deviation of the sum of signals 1&2 is greater than the standard deviation of both signal 1 and signal 2 individually. This is because when taking the standard deviation of the signal1+signal2 data set, we are looking at how far the power of that signal has deviated from the mean. The mean is different and greater for this data set when compared to signal 1 and signal 2 individually.

Q2-2-4. What distribution is the data in P2-2-6 best described by (uniform, triangular, normal, other)?

**A:** the distribution of the histogram is of a bimodal normal distribution characteristic.

Q2-3-1. Carefully compare the distributions (from the histogram) of Signal 1, Signal 1+2, and the sum of signal 1-12. How are each of these distributions best described (uniform, triangular, normal, other)? How does this compare to theory? What theory applies to the sum of the signals 12?

**A:** Signal 1 has steady distribution that does not fluctuate much; So the distribution is said to be uniform. The histogram for Signal 1+2 is of a normal distribution but is very spread out and the same characteristic for the sum of signals 1-12 which has a more normally distributed characteristic. The sum of signals 1-12 should all add up to be a merging of all the signals , so the amplitude should increase for every signal combined into the total.

Q2-4-1. How would you describe the distribution of the error of the estimate (normal, uniform, triangular, other)?

**A:** The distribution is at most normally distributed, though the distribution is negatively skewed to the right of the graph more so the left.

Q2-4-2. What is the standard deviation of the error?

**A:** the standard deviation of the error is 8.33

Q2-4-3. If the estimator predicts that the temperature today will be 55 degrees, what is the signal to noise ratio of this estimate, given that the standard deviation of the error was the value you just calculated?

**A:** the signal noise ratio would be 55/8.33 making the ratio be a 6.6 SNR value.

Q3-1-1. How would you describe the distribution of the error of the estimate (normal, uniform, triangular, other)?

**A:** I would describe the distribution pretty much uniform, the data does not fluctuate much. Though it could also be argued to be normal distributed due to the falls at both ends of the peak in the middle.

Q3-2-1. How would you describe the distribution of the error of the estimate (normal, uniform, triangular, other)?

**A:** The distribution of the error has a tri-modal normal distribution. There are three normally distributed peaks in the data distribution graph.

Q3-2-2. Are the distributions of the quantization error significantly different or largely the same between P3-1-1 and P3-2-1?

**A:** The distributions are significantly different, while P3-1-1 has a normal/uniform distribution, P3-1-2 has a tri-modal normal distribution, it has three peaks and those peaks fall at either end.

From the data, estimate what the true value of the analog signal is, using a minimum number of samples, so that your estimate has a typical error of less than 100 microvolts.

Q3-3-1. What is the estimated true value of the constant analog signal?

**A:** The estimated true value of the constant signal would be 3.6 V

Q3-3-2. How many samples must you use in your estimate?

**A:** 100 samples were used

# Discussion

The LM61 temperature sensor is not the best of the best, but it gets the job done. The sensor during experimentation took a long time to heat up but took a short time to cool down. The resulting graph of these characteristics would change depending on how the data was sampled. In terms of the 100ms interrupt. That interrupt fired off every time the count of the sample reached the intended number of samples for the data set. The sampleFlag was initialized to false, and for every time the calculations were done for the running mean, the sample Flag would be set back to false in the loop after it had entered the interrupt and was set to true. So, for 256 samples, the sampleFlag would be set false 256 times. This is done to make sure the code dose not sit in the interrupt and discontinue running calculations after it had been set true. It is only told to stop with the if statement that says, “if you reach the intended number of samples, stop interrupting and continue on with the rest of the code”.

The running standard deviation of the data collected shows just how far away from the truth we really are. If someone were to breath on the temperature sensor at rest. The sensor’s mean would be pretty much leveled out. This is because the sensor does not have enough time to cool down again till the next breath. However, when it comes to looking at the data collected and analyzed using the standard deviation of the data. We can see just how far we are from the truth. The data would be jumpy and sporadically jumping all over the graph. This lets us know that even though the data is not changing or fluctuating a whole lot, the actual temperature that would be shown is drastically different from what is being collected.

1. Manuscript received September 10, 2018.This work was required by class for grade. [↑](#footnote-ref-1)