Ryerson University

Department of Electrical & Computer Engineering

BME 632 - Signals & Systems II

Lab Report - EMG

Lab Number: 2

Instructor: April Khademi

Section: 2

Due Date: February 16th, 2020

Student Name (ID): [redacted]

Signature: [redacted]

Student Name (ID): **Pass Fail** **Kushal Shah** **500843903**

Signature: KS

By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work. Any suspicion of copying or plagiarism in this work will result in an investigation of Academic Misconduct and may result in a 0 on the work, an F in the course, or possibly more severe penalties, as well as a Disciplinary Notice on your academic record under the Student Code of Academic Conduct, which can be found online at: www.ryerson.ca/senate/current/pol60.pdf.

Before submitting your report, your TA asks questions about your report. If there is no consistency between your oral answer and your report, you will lose 50% of your total mark. For this part, Pass or fail will be circled next to your name accordingly.

Introduction:

The human body is one large energy source. It creates electrical signals in response to outside sources, and these electrical signals can be measured with sensors. An EMG signal is the summation of all action potentials in a muscle group, which in other terms means it is the summation of all electrical signals. Therefore, these are the body signals which can be measured. EMG signals, for the purpose of this lab, are created by contracting forearm muscles in an isometric fashion. Isometric muscle contractions are muscle contractions, but unlike a normal contraction the muscle will not change length. But this would still create the same response as if it was a normal contraction. Comparing the force output to the EMG signals will be the focus of this lab. Exploring the relationship between the exerted force and the electrical inputs to those muscles. To compare, there are many calculations/methodologies which can be used. The methodologies include variance, dynamic range, mean squared, etc. There are all statistical values which can be used to find abnormalities in signals. This lab will test and make us explore deeper into what it means to study biomedical signals. [1]

Pre-Lab Questions:

1. An EMG signal is the summation of all action potentials in a muscle at one time. For an action potential, and in turn a muscle contraction (for purposes of this experiment), to happen an electrical signal must begin in the brain, run down the spinal cord, and then it goes to the motor neuron in the muscle. Each muscle fiber is innervated by a single neuron. Chemically many things happen to create the action potential but that is beyond the scope of this lab. Recruitment is a term used to explain which and how many neurons are being used to innervate a muscle. Spatial recruitment is the activation of many motor neurons to create a larger force. Temporal recruitment is the ability of neurons to fire faster than normal, which has the possibility of exciting another motor unit. In terms of this course, the frequency of the firing response increases during temporal recruitment.
2. As we know the force output of a muscle can be related to the amplitude of the recorded EMG signal. For weaker contractions, fewer muscle fibers are used and this will result in a lower amplitude signal. When a larger force is applied spatial summation is used, which we know activates more motor neurons, and will create a larger amplitude signal. We know that the EMG signal, as stated in the previous answer, is the summation of all action potentials in a muscle. Therefore, when more motor neurons are firing, there are more action potentials, which results in a larger amplitude signal.
3. When we contract or extend a muscle we are both applying a force and changing the length of the muscle. During isometric contraction, the muscle is activated and a force is applied but the length of the muscle stays constant.
4. Short time analysis is when you take a portion/window of your data and perform functions on the windows instead of the full data set. This allows us to take a systematic approach to working with the data, instead of working with the entire data set at once. Especially if the data set is infinitely long, this will be efficient and allows us to locally analyze data. This window would move across the whole data set, creating segmented

analyzed data outputs, but also allowing for better use of functions such as mean functions.

5. Mean will find the average output signal value across the window and compare it to other windows. Variance will tell us how spread out the data points of the emg signal are, and compare it to other windows. Dynamic range will tell us the maximum and minimum values of the signal in the windows. Mean square is similar to mean where it gives us average power in the window. Root mean square
6. For higher volitional effort I would expect the metric values to lower. For example, mean force would probably lower because the individual will try to keep a more consistent force output. This would take away from any random peaks of energy which is from the individual lowering in energy and then suddenly raising in force to compensate. I would expect dynamic range to decrease because again the individual will be more consistent. Variance is very similar and so are mean squared and root mean square in the reasoning behind the values lowering.

Experiment 1:

Exercise 1.1:

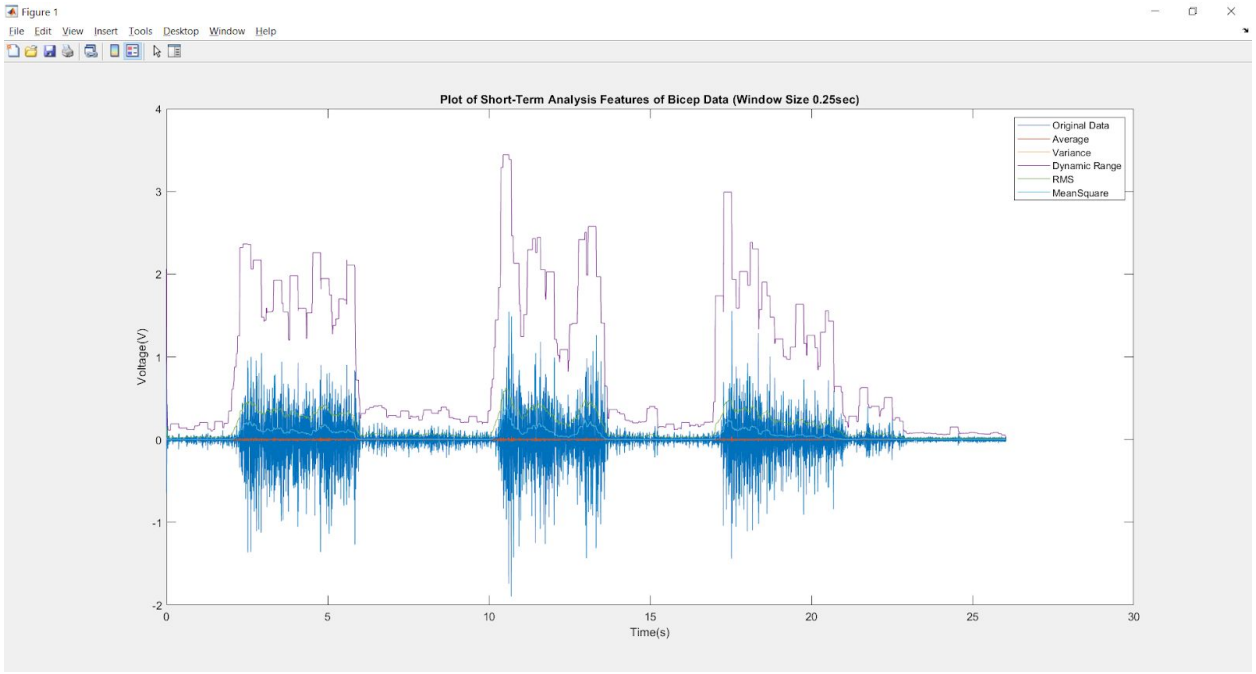

Figure 1. Short-time analysis of the isometric bicep contraction emg data with an analysis window size of 0.25 sec.

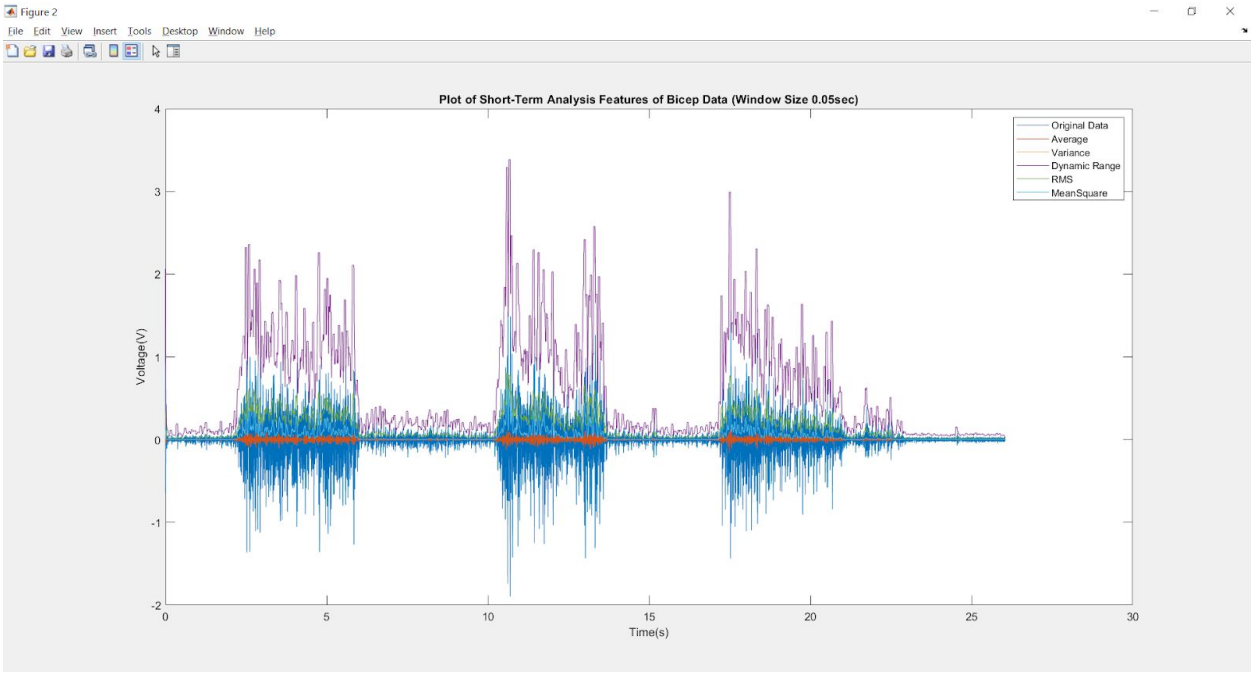

Figure 2. Short-time analysis of the isometric bicep contraction emg data with an analysis window size of 0.05 sec.

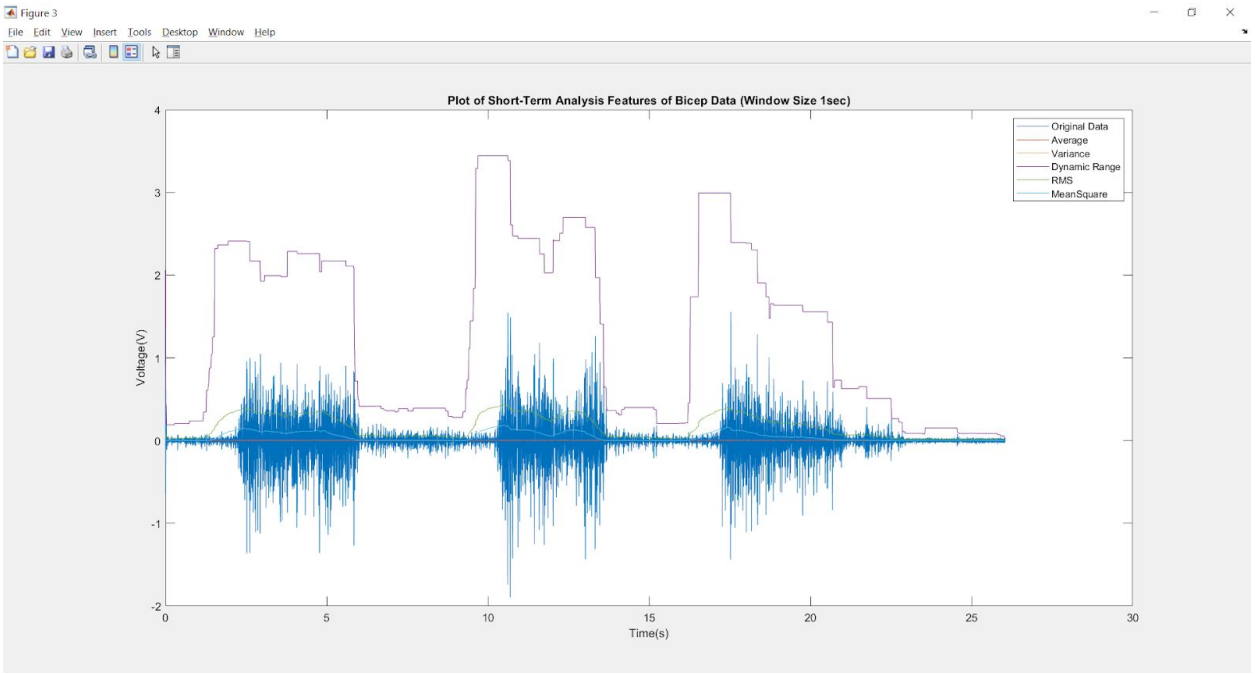

Figure 3. Short-time analysis of the isometric bicep contraction emg data with an analysis window size of 1 sec.

In this experiment we looked at isometric contractions of the biceps. To analyze this data we created a window function which would go across the entire data set and determine certain metrics. These metrics include mean value, variance, dynamic range, mean squared, and root mean squared. To do this, in the window function we created a padded vector so that while the window moves across the data it looks at one value and it puts zeros in the other spaces. This allows us to analyze the data piece by piece without affecting the data itself.

Experiment 2:

Exercise 2.1:

a)

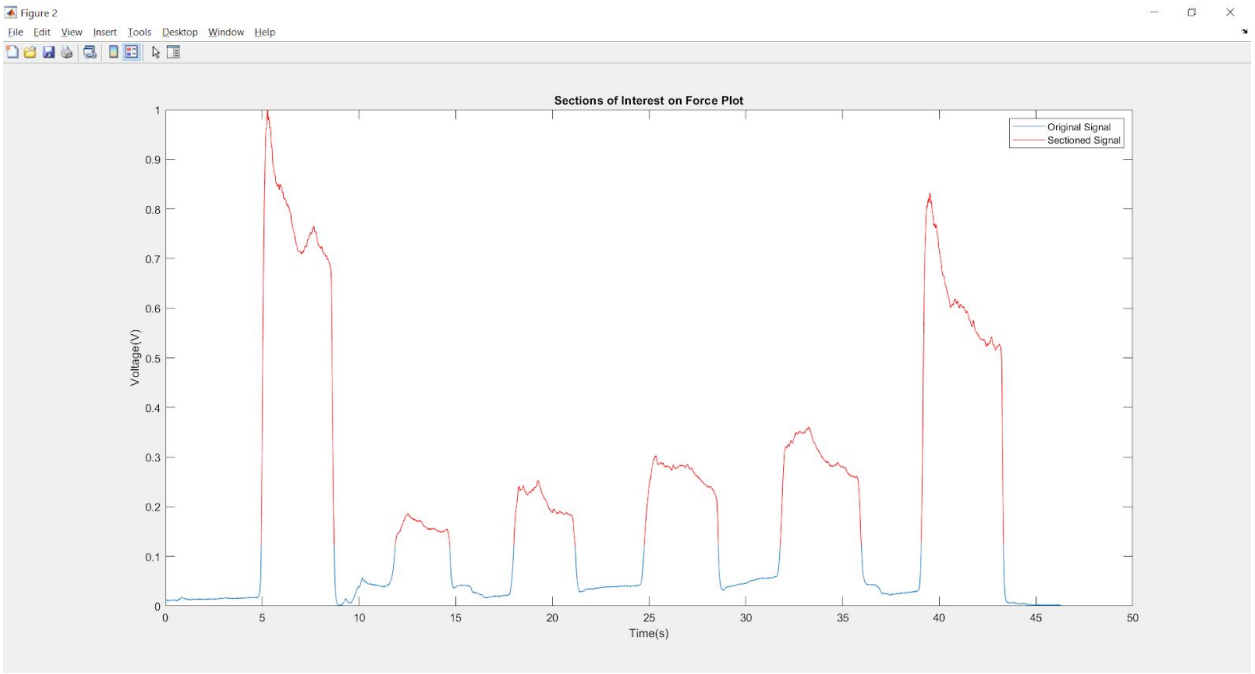

Figure 4. Plot of the force graph with the sectioned area of interest marked off in red.

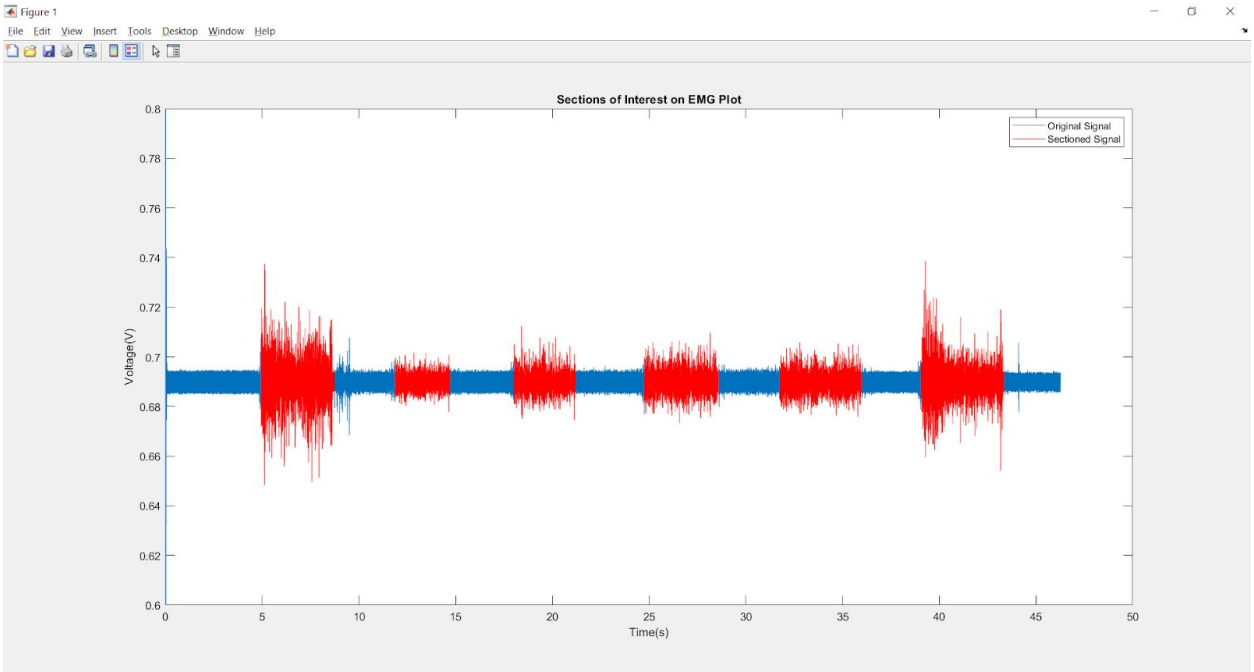

Figure 5. Plot of the EMG graph with the sectioned area of interest marked off in red.

To analyze the data we wanted to only look at the sections at which force was actually applied. To do so we thresholded the whole dataset and made the values that were under the threshold zero and then ‘nan’. The threshold was arbitrarily chosen, but it gave us a value which did exactly correspond to the time values where the force was applied.

b)

Table 1. Results from all of the statistical values of the EMG signal.

Segment	1	2	3	4	5	6
Force	0.7557	0.1605	0.2077	0.2621	0.2980	0.6007
Mean	0.6670	0.6593	0.6626	0.6673	0.6693	0.6696
Variance	0.0054	0.0071	0.0063	0.0052	0.0048	0.0048
DR	0.0986	0.0782	0.0792	0.0710	0.0641	0.0809
RMS	0.6747	0.6695	0.6718	0.6749	0.6762	0.6764
MS	0.4602	0.4548	0.4571	0.4604	0.4617	0.4619

The window function, from part 1 was used for this part. However, instead of the function going through the whole dataset, it would begin and stop at the intervals of each section which was found out by checking where the function was ‘nan’. The window function would also be used to create a vector of all the sections.

c)

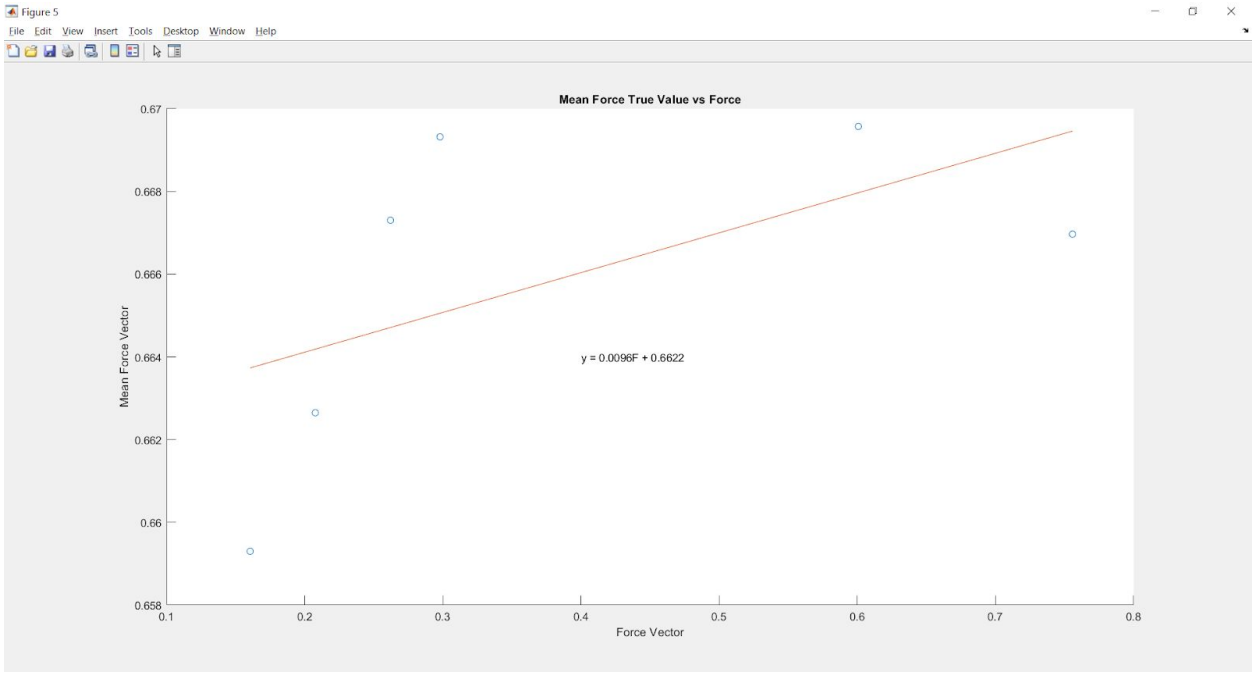

Figure 6. Linear regression plot of mean force true value vs force.

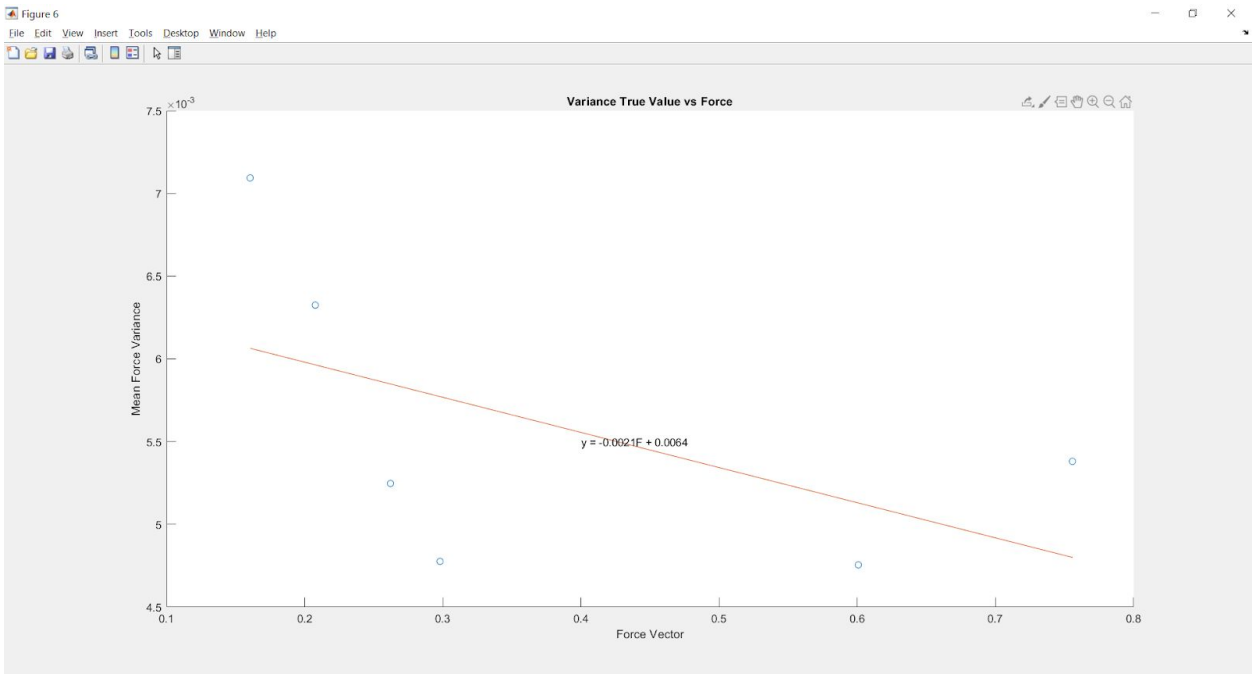

Figure 7. Linear regression plot of variance true value vs force.

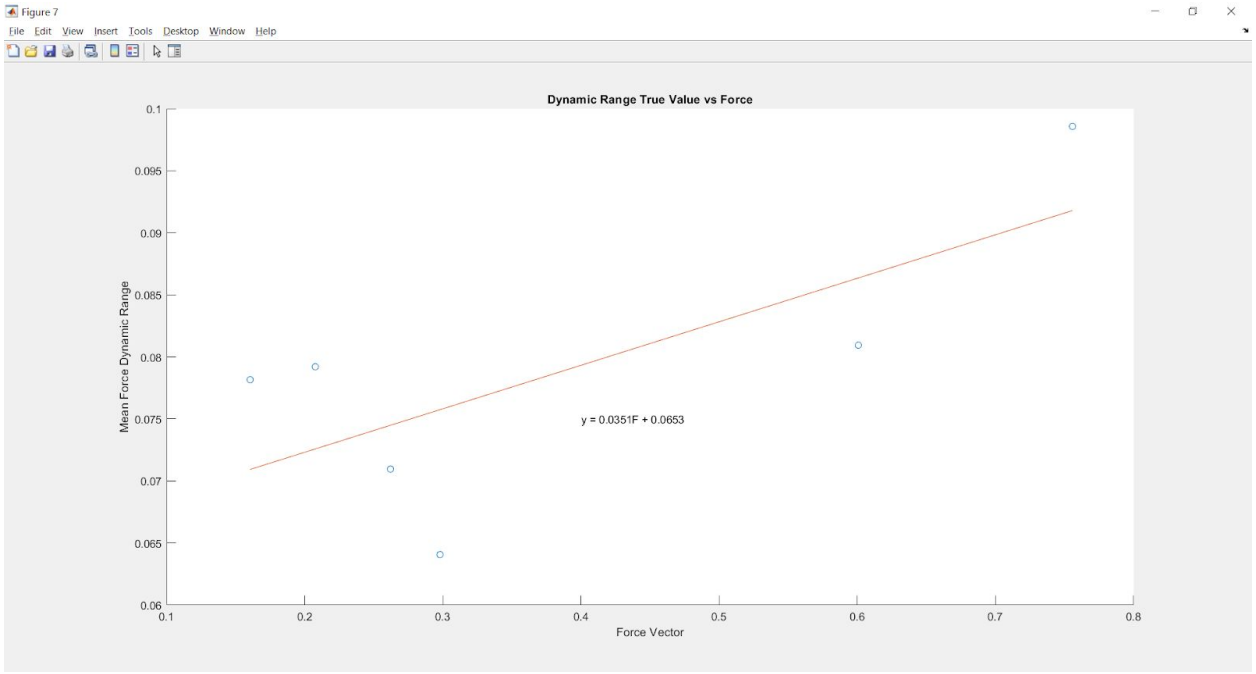

Figure 8. Linear regression plot of dynamic range true value vs force.

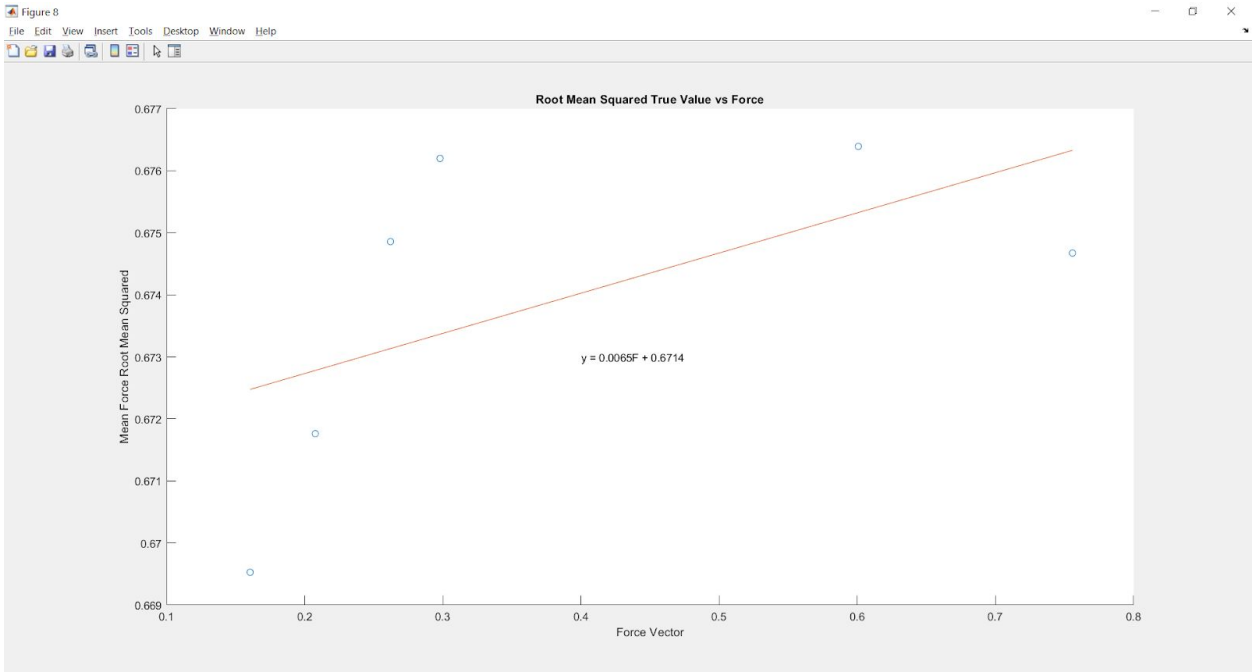

Figure 9. Linear regression plot of root mean squared true values vs force.

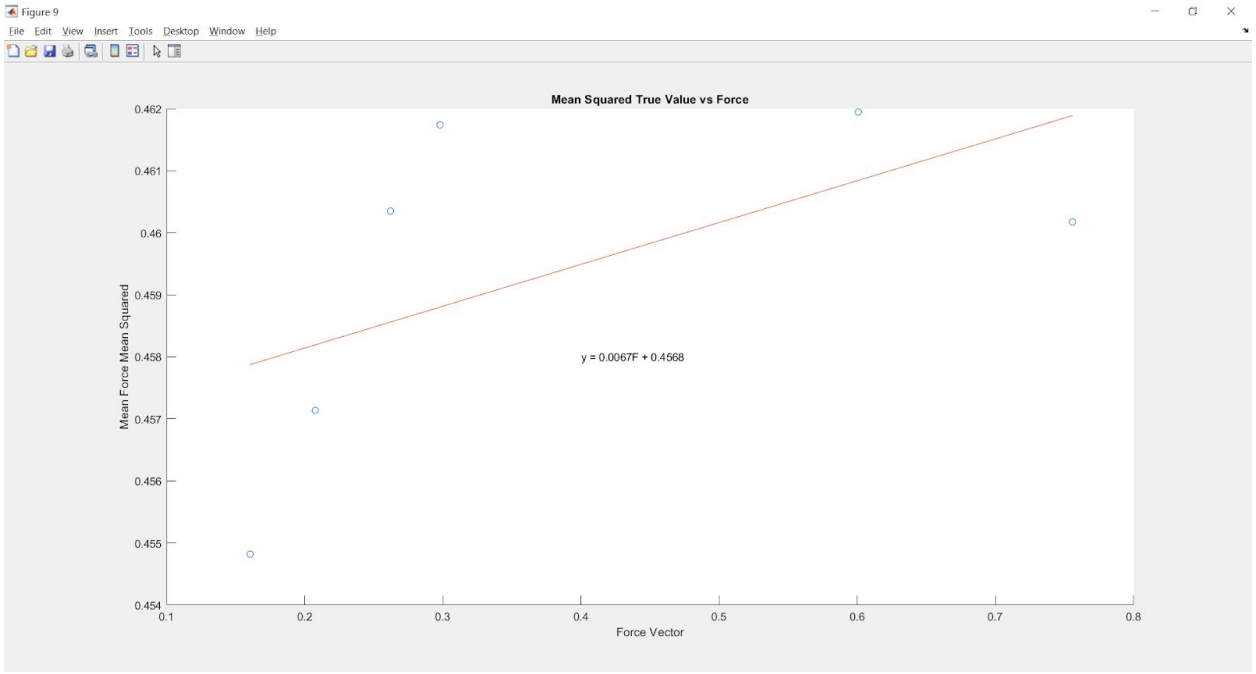

Figure 10. Linear regression plot of mean square true values vs force.

To create these plots we used the vectors created in the previous section and plot them against the average force vector. The scatter and plot matlab functions were used to create these plots. Interestingly we can see that the trend lines for each of these plots except for the variance plot. In this lab we only had 6 sections which in turn means we had a very low amount of points. Therefore, since we had a small dataset it could create data which we don't expect and a negative slope for the variance plot is something we did not expect. However, it is a good sign that the other plots had positive slopes.

d)

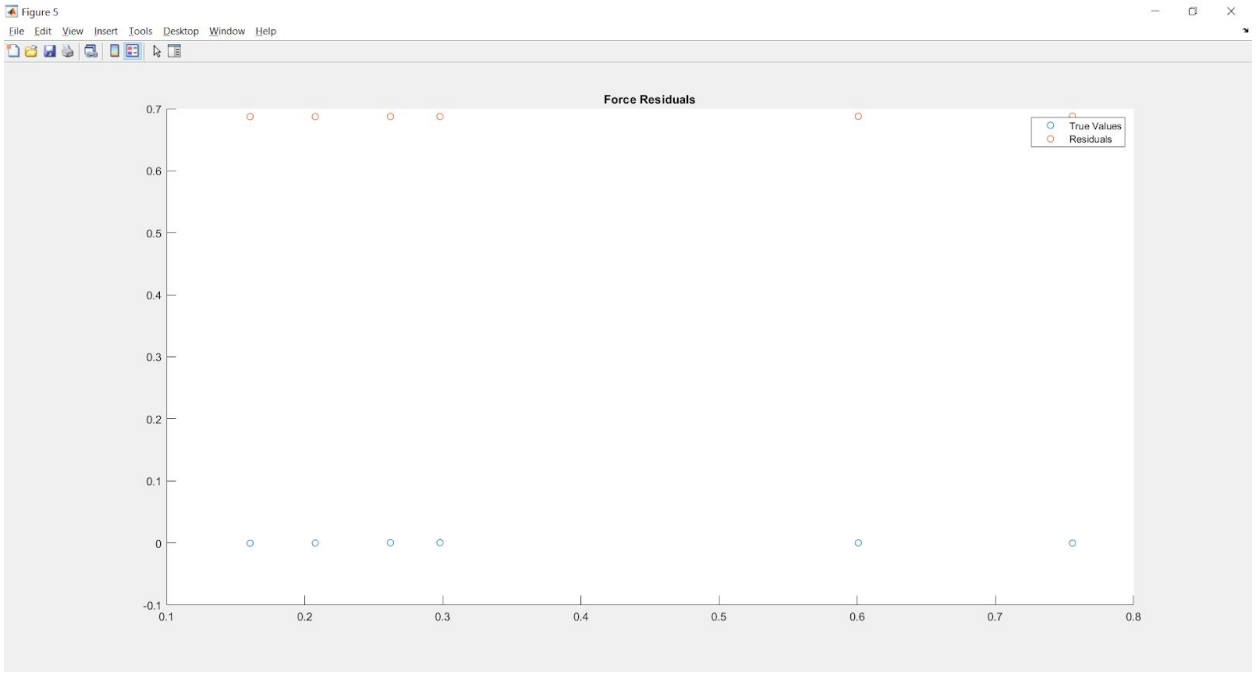

Figure 11. Plot of the true and residual values of the force applied.

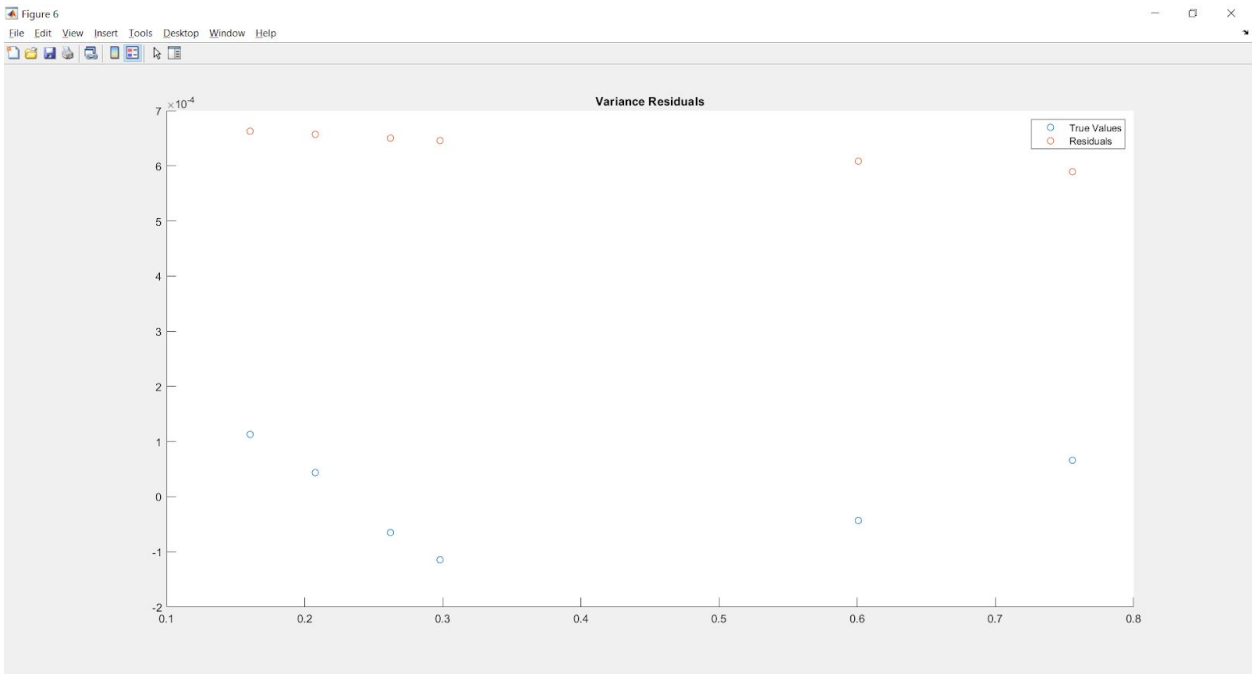

Figure 12. Plot of the true and residual values of the variance.

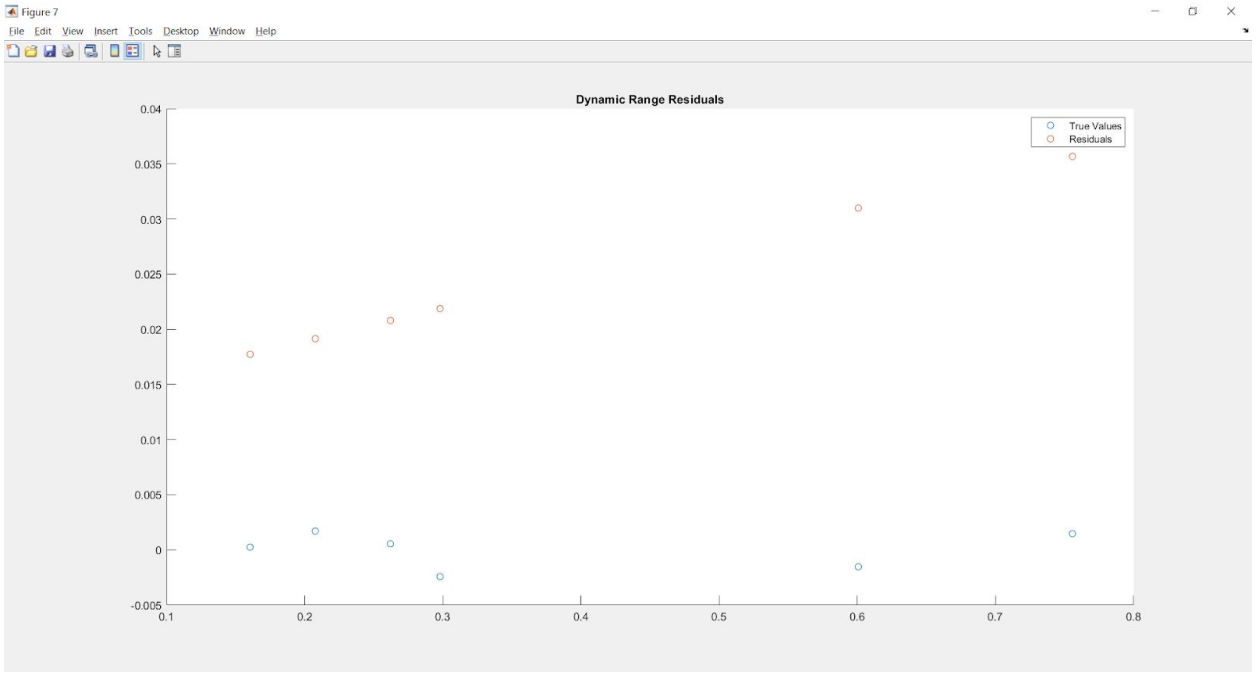

Figure 13. Plot of the true and residual values of the dynamic range.

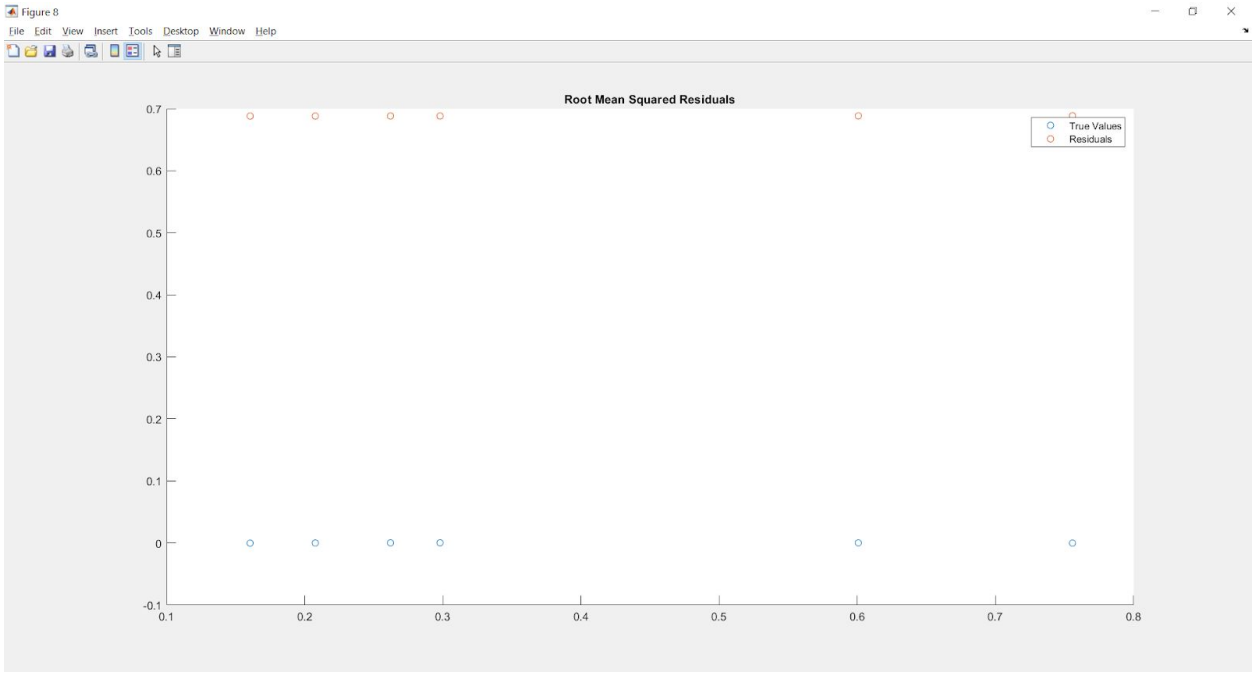

Figure 14. Plot of the true and residual values of the RMS.

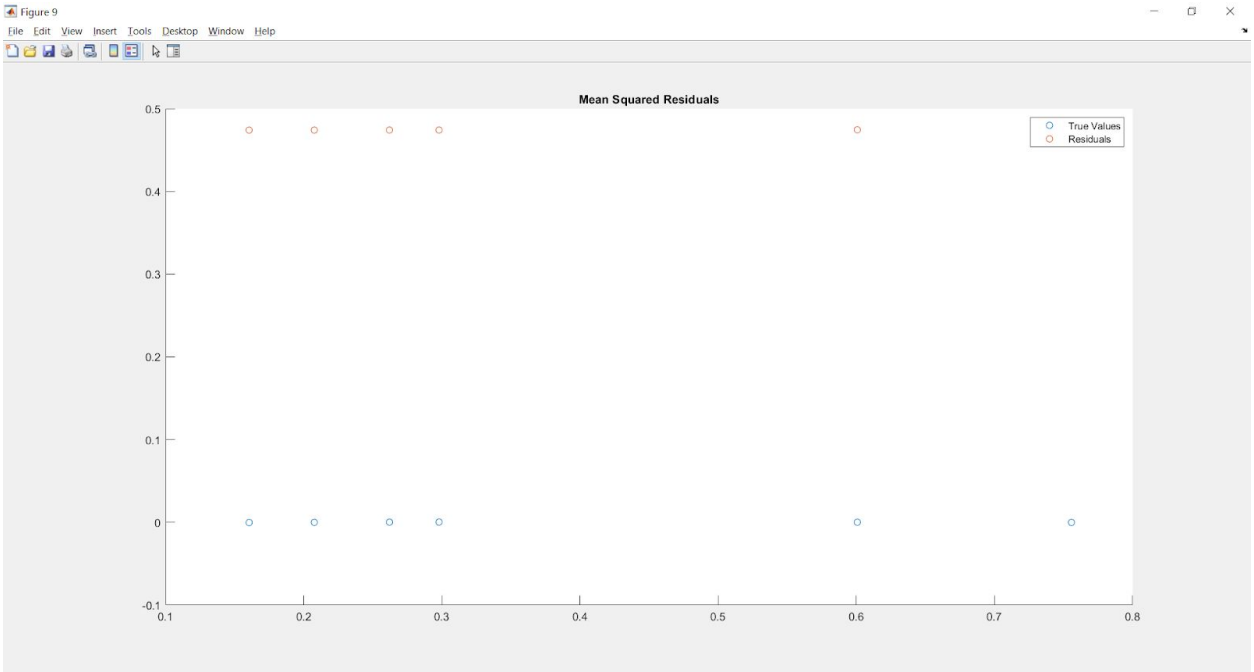

Figure 15. Plot of the true and residual values of the MS.

The residual value is the difference between the true values and the estimated values. Therefore, the lower the residuals, the closer the observed values are to the theoretical values. All of these plots above have very low residual values except for the variance plot. Therefore, we can determine that variance is not the best feature to use to analyze our data. This conforms to our previous assessment where the variance had a negative slope, and we determined that the data wasn't good.

e) **Table 2.** MSE and correlation coefficient for each of the statistical values.

	MSE	Correlation Coefficient
Force	0.1420	0.5691
Variance	0.1926	-0.5486
DR	0.1722	0.7246
RMS	0.1425	0.5740
MS	0.0567	0.5759

When we look at MSE and correlation coefficients we want to have a low MSE and a high correlation coefficient. Therefore, when we look at this table we can say that MS is the best statistical value to analyze the data. To also note, we can confirm again that variance is a poor choice because of the negative correlation coefficient.

Exercise 2.2:

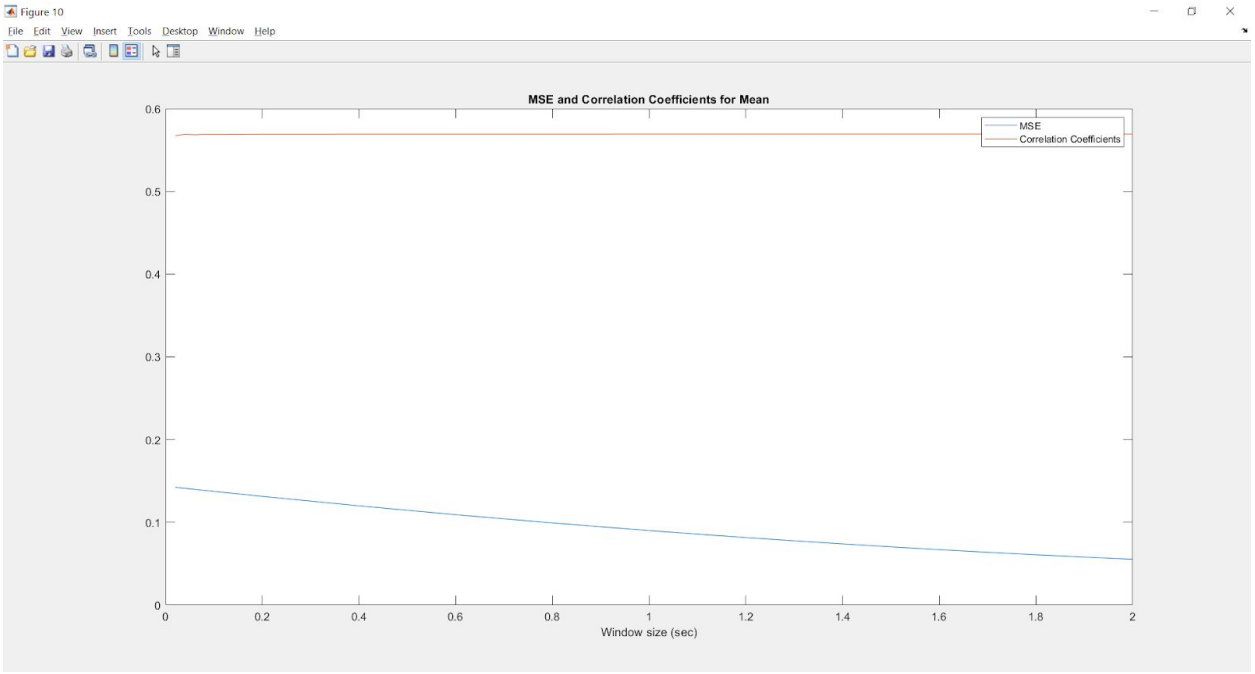

Figure 16. Plot of MSE and correlation coefficients for mean force, which contains the values from twenty different window sizes (0.2-2).

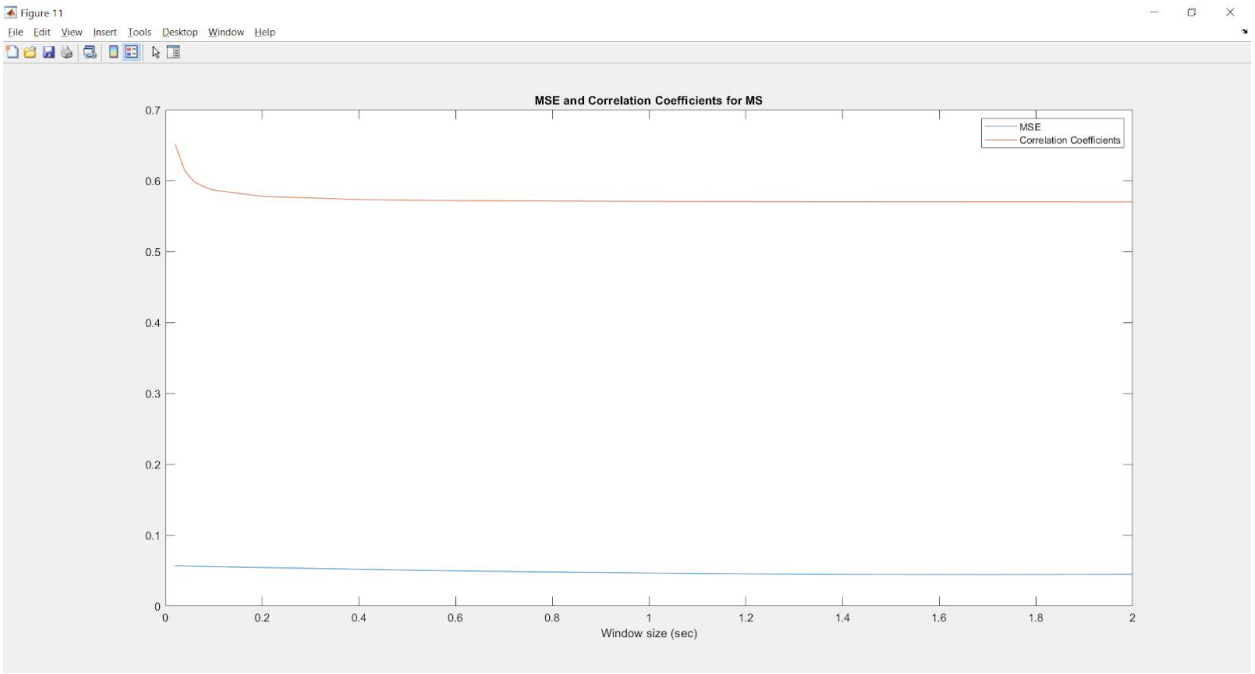

Figure 17. Plot of MSE and correlation coefficients for MS, which contains the values from twenty different window sizes (0.2-2).

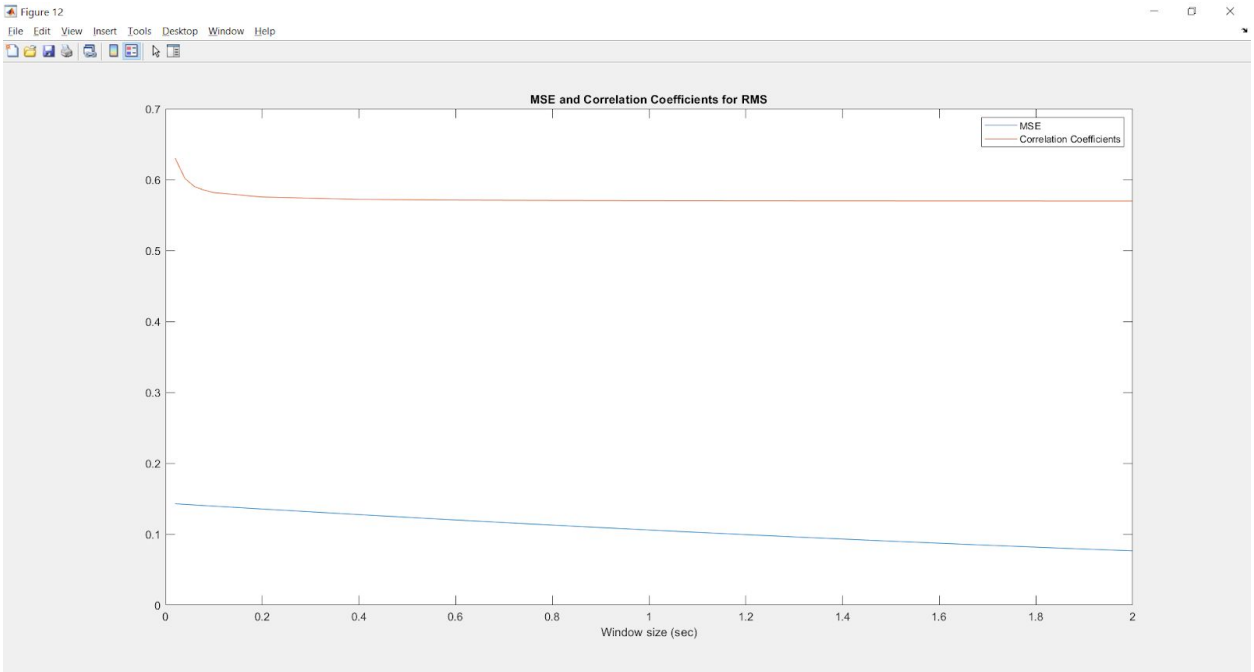

Figure 18. Plot of MSE and correlation coefficients for RMS, which contains the values from twenty different window sizes (0.2-2).

When we look at these three plots we can see that they are different. They have different slopes and y-values, which makes it easier to characterize them. To find the best feature we would have to look at which feature has the highest correlation coefficient and the lower the mean squared error. The higher the correlation coefficient the closer related the values of the windows are, and the lower the MSE the higher the accuracy of prediction. Therefore, when looking at the three plots we can see that the mean squared plot is the best. This conforms again to our previous conclusions in the previous parts of the experiment. It has both a consistent high correlation coefficient and a consistent small MSE. Therefore, with any window size MS plots will be the most accurate at predicting outcomes for us. We know that the smaller window size is best because of previous plots done in this lab, and with these three plots we can see that they are quite consistent all the way. Therefore, we can say that we want a smaller window size when we analyze data.

Experiment 3:

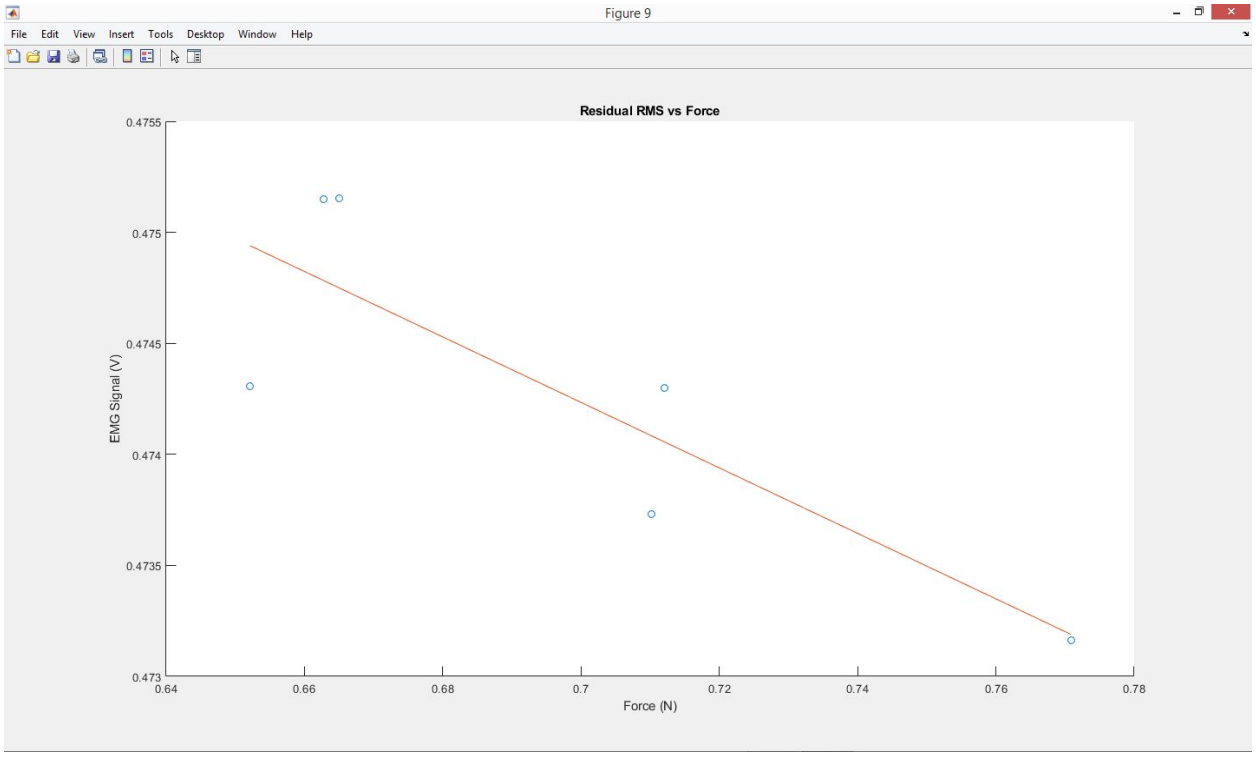

Figure 19: Plot of Residual RMS vs Force

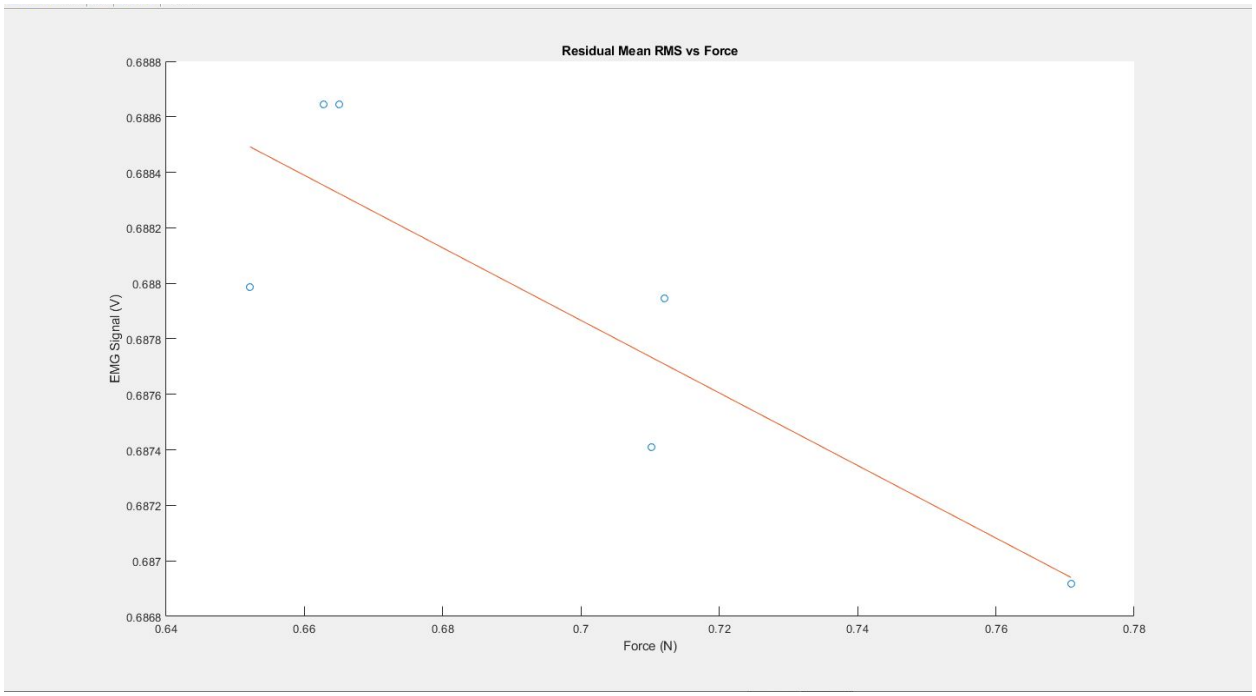

Figure 20: Plot of Residual Mean RMS vs Force

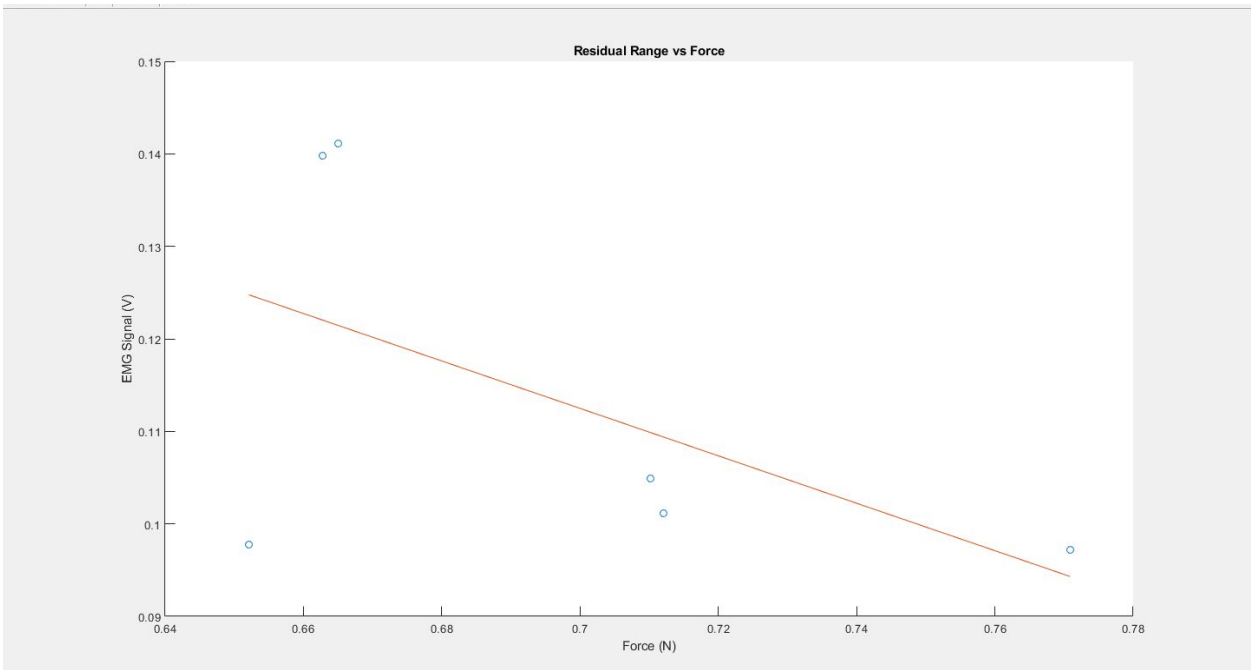

Figure 21: Plot of Residual Range vs Force

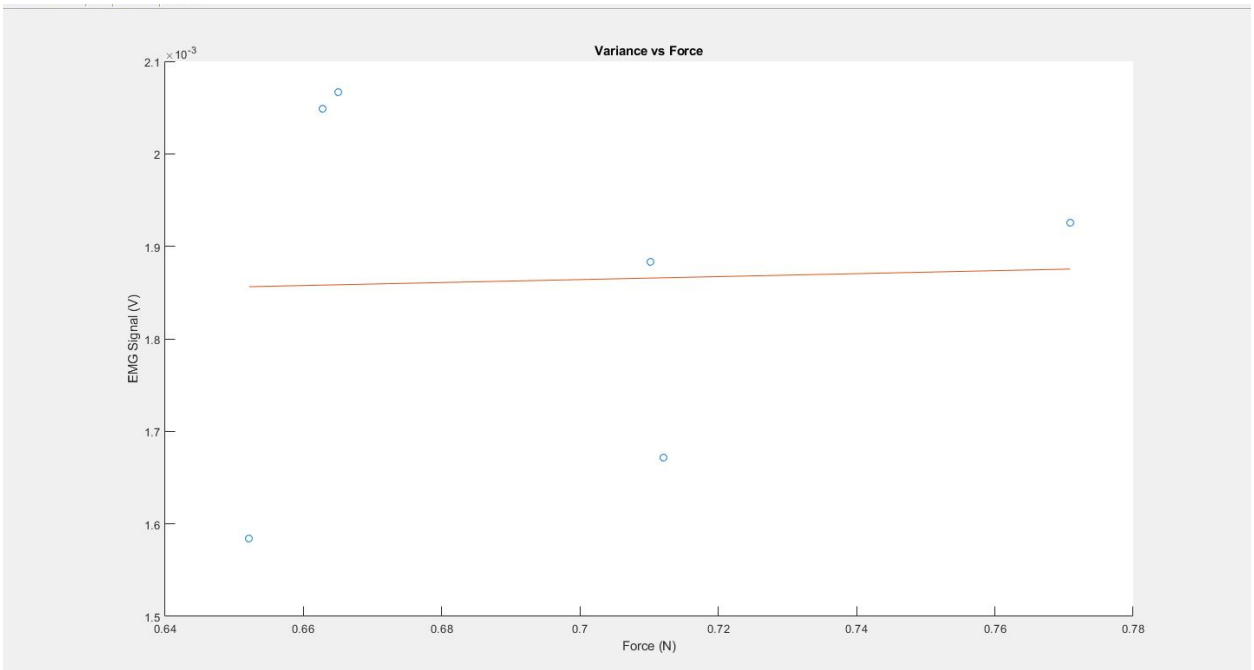

Figure 22: Plot of Residual Variance vs Force

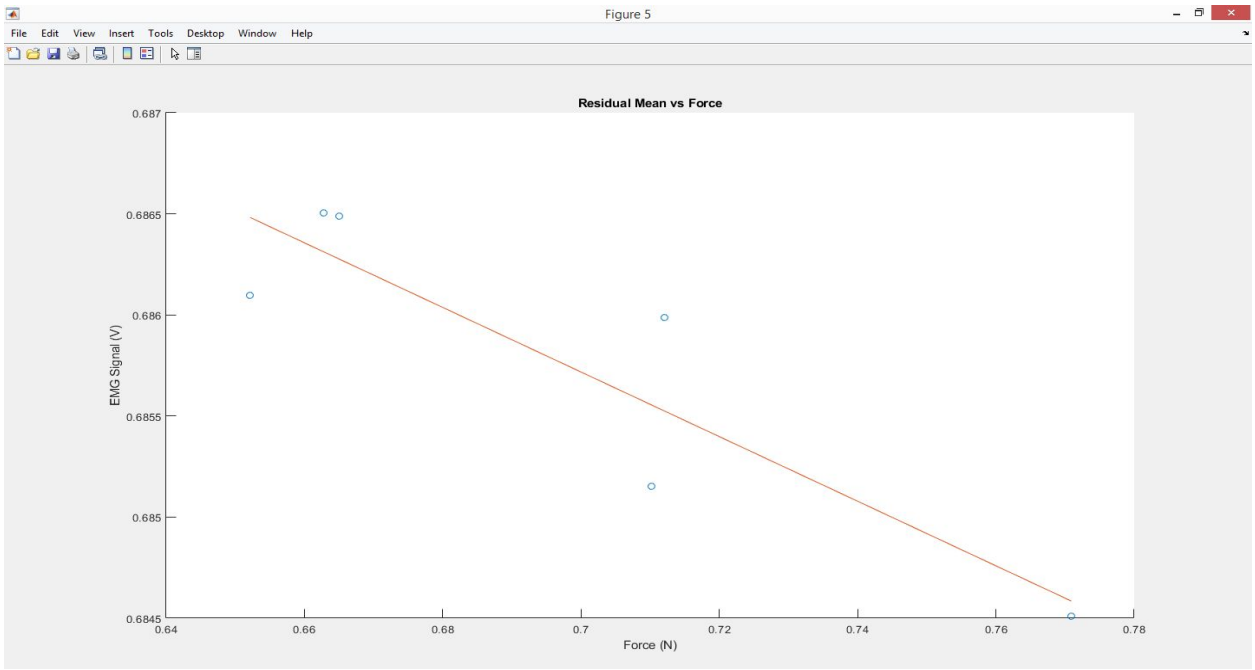

Figure 23: Plot of Residual Mean RMS vs Force

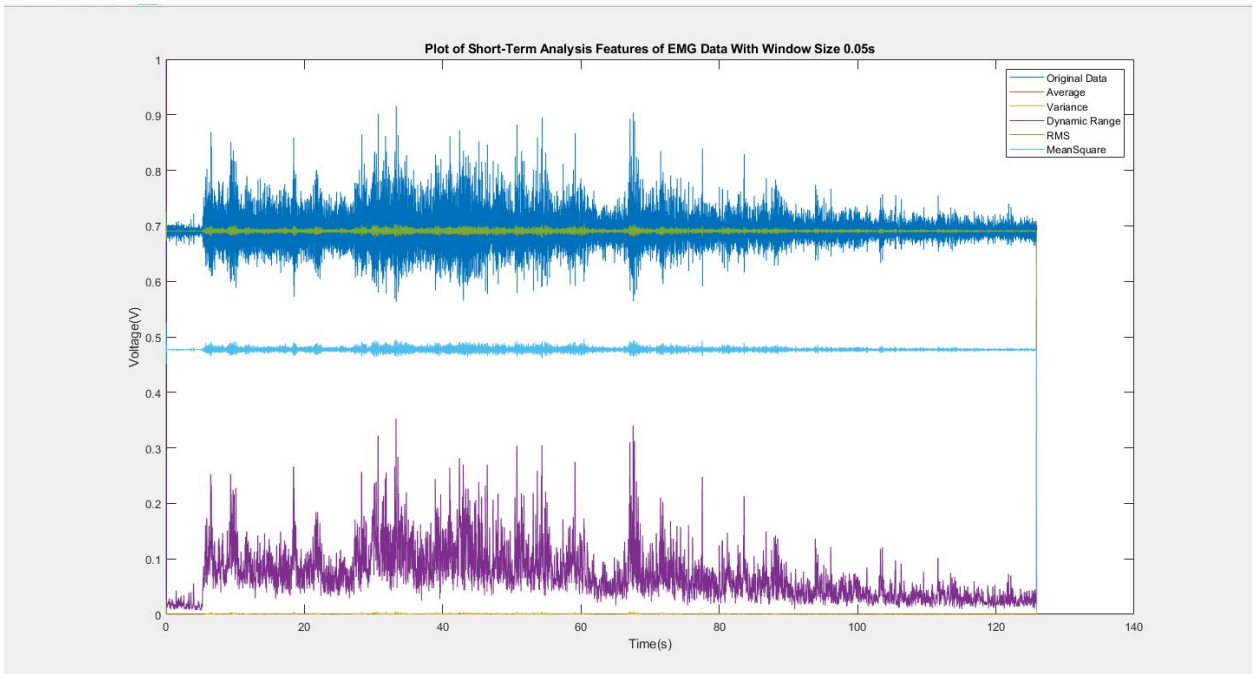

Figure 24: Plot of Short term analysis features of EMG data with window size 0.05

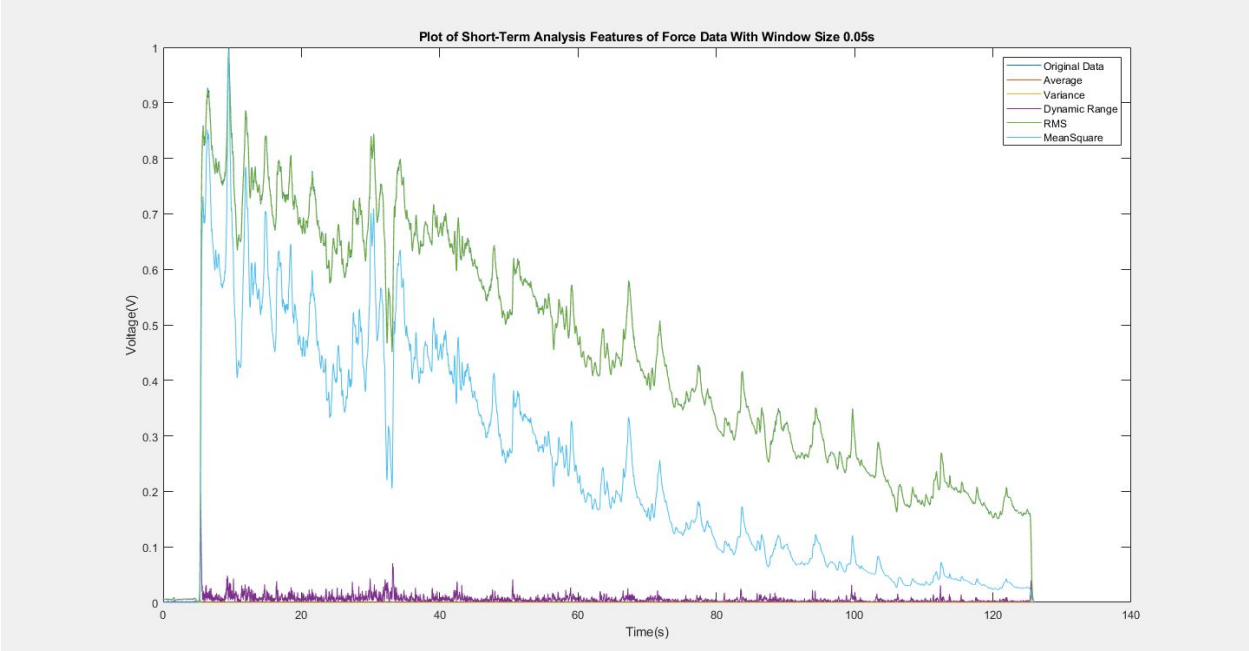

Figure 25: Plot of Short term analysis features of force data with window size 0.05

Part 3 Analysis:

The algorithm used in part 3 was similar to the one used in part 1. Window size of 0.05s set this time to obtain the desired results. The analysis was geared towards the Grasp fatigue data. The plots from Figures 19-21 showed a negative slope (for RMS, Mean RMS, Range RMS vs Force), which were consistent with our theoretical expectations. The plot in Figure 22 (for variance vs force) showed a zero-slope graph, which seemed valid. This is because since the force decreased, it would move towards the x axis and thus away from the mean line. Thus, as the force decreased, the variance increased which is consistent with our results.

It can be seen in Figure 25 that the force decreases with time at around 80 seconds, although the EMG signals continue increasing. In the experiment, the user continued applying force and pressed on the sensor harder after 80 seconds. However, due to physiological activity that the muscles were exhausted or tired after some time thus the force input decreased. This part clearly demonstrates fatigueness as the force input decreases but the force decreases.

Post-Lab Questions:

Note: Although RMS, power, variance, mean are all different methods of quantifying EMG signals, they correlate to force in different ways, which makes them ideal for analysing different cases.

a) Mean: This is the mean force, which can be used during the window functions or over the whole data set. This feature is useful when there are a large amount of data points and a trend wants to be found.
   Variance: This shows us the difference in the mean values. It shows how far the observed values are from the calculated trend line. This is useful when we want to see how far the values stray from the trendline.
   Root mean squared: is a measure of the magnitude of the set of numbers. This gives us a sense of the typical size of the numbers. This can be used instead of standard deviation.
   Mean squared: It calculates and relates the variance of a data set within a group.
   Dynamic range: it shows the maximum and minimum values of a data set. This can help us see the scope/bandwidth of the dataset.
b) EMG signals are datasets which can be very dense and very long. To analyze this whole dataset we would have to segment it, and this means making windows. Having a window which can be translated down the whole data set can make analyzing it simpler. Having a large amount of data being analyzed at once can create unrealistic values. A mean value of a smaller dataset/window is more accurate than the mean of a large dataset. This example can also be related to all the other features used in this lab. Also, as we concluded in the lab, the smaller the window size the more accurate the analysis.
c) As concluded in the experiments, MS is the best feature to analyze the EMG activity. The best way to explain is by using the outputs determined in experiment 2.2. MS had the most consistent low number for MSE and high number for CC.
d) When the window size is large the resolution of the results increases however when the window size is too small it can outlie other significant points. For example in figures 1 and 2, when the the window size is set to 0.25 seconds and 0.05 seconds, the resolution is large for 0.25 seconds, thus it includes more data to process, when compared to a window size of 0.05 seconds. However, with a high volume of data it includes more noise as well to filter which affects the overall accuracy of the analysis. Thus, we would choose smaller window size as optimal as even though the resolution would be a lot smaller, it would exclude redundant data and thus it would result in less noise and more overall accuracy.
e) The linear regression demonstrates the analysis of EMG data compared to the force. In other words, it is a technique to check whether a relationship exists between force and variables (DR, MS, RMS, and so on).
   In this lab, the linear regression demonstrates the relationship between the dependent (othe and independent (force) variable which is modelled by the equation

$y_{est}(F) = mf + b$ [1]

The line of best fit is evident in all graphs in part 2 and 3 where there is a negative slope for most cases i.e. variables when computed against force.

References:

[1] Khademi, April . "Lab Manual" BME 632 - Signals and System II, 16 Feb. 2020, Ryerson University

[2] Khademi, April . "Lab Experiment" BME 632 - Signals and System II, 16 Feb. 2020, Ryerson University

Appendix:

Exercise 1:

```
%% Exercise 1.1
isobicepdata=readtable('isobiceps.csv');
%DATA INPUTS
bicepdata=isobicepdata(:,2);
sampfreq=1000;
secpersamp=1/sampfreq;
time=(0:26039-1);
time=time*secpersamp;
time=time.';
bicepdata=table2array(bicepdata);

%% WINDOW 1

window1=0.25;
window1=window1*sampfreq;

%Short-Term Analysis Variables
[Average1,Variance1,DR1,RMS1,MS1]= WindowData(bicepdata,window1);
%Flip to Vertical
Average1=Average1.';
Variance1=Variance1.';
DR1=DR1.';
RMS1=RMS1.';
MS1=MS1.';

%Plot of Short-Term Analysis Variables over Original
figure(1)
plot(time,bicepdata);
title('Plot of Short-Term Analysis Features of Bicep Data (Window Size 0.25sec)')
xlabel('Time(s)')
ylabel('Voltage(V)')
hold on
plot(time,Average1);
plot(time,Variance1);
plot(time,DR1);
plot(time,RMS1);
plot(time,MS1);
legend('Original Data','Average','Variance','Dynamic Range','RMS','MeanSquare')
```

```
hold off

%% WINDOW 2

window2=0.05;
window2=window2*sampfreq;

%Short-Term Analysis Variables
[Average2,Variance2,DR2,RMS2,MS2]= WindowData(bicepdata,window2);
Average2=Average2.';
Variance2=Variance2.';
DR2=DR2.';
RMS2=RMS2.';
MS2=MS2.';

%Plot of Short-Term Analysis Variables Over Original
figure(2)
plot(time,bicepdata);
title('Plot of Short-Term Analysis Features of Bicep Data (Window Size 0.05sec)')
xlabel('Time(s)')
ylabel('Voltage(V)')
hold on
plot(time,Average2);
plot(time,Variance2);
plot(time,DR2);
plot(time,RMS2);
plot(time,MS2);
legend('Original Data','Average','Variance','Dynamic Range','RMS','MeanSquare')
hold off

%% WINDOW 3

window3=1;
window3=window3*sampfreq;

%Short-Term Analysis Variables
[Average3,Variance3,DR3,RMS3,MS3]= WindowData(bicepdata,window3);
Average3=Average3.';
Variance3=Variance3.';
DR3=DR3.';
RMS3=RMS3.';
MS3=MS3.';
```

```
%Plot of Short-Term Analysis Variables Over Original
figure(3)
plot(time,bicepdata);
title('Plot of Short-Term Analysis Features of Bicep Data (Window Size 1sec)')
xlabel('Time(s)')
ylabel('Voltage(V)')
hold on
plot(time,Average3);
plot(time,Variance3);
plot(time,DR3);
plot(time,RMS3);
plot(time,MS3);
legend('Original Data','Average','Variance','Dynamic Range','RMS','MeanSquare')
hold off
```

Exercise 2:

```
%% Experiment 2

% Opening File
m = readtable('GraspForce.csv');

% Creating vectors
tx2 = m(:,1); %extracting time values from the experiment data
tx2 = tx2{:,:}; %making the matrix useful for calculations
emgx2 = m(:,2); %extracting emg values from the experiment data
emgx2 = emgx2{:,:}; %making the matrix useful for calculations
forcex2 = m(:,3); %extracting force values from the experiment data
forcex2 = forcex2{:,:}; %making the matrix useful for calculations

timex2 = 0 : length(tx2) - 1;
timex2 = timex2 * 0.001; %conversion of time array to take into the account the sampling rate
(1000Hz)

%Normalizing the data
normForce = forcex2 - min(forcex2(:));
normForce = normForce ./ max(normForce(:));

normemg = emgx2 - min(emgx2(:));
normemg = normemg ./ max(normemg(:));

normemg2 = normemg;
normForce2 = normForce;
```

```
%Threshold for the data to find where fore was applied

thresholdind = find(0.12>normForce2); %finding where the force is below the threshold
normForce2(thresholdind) = 0;
normemg2(thresholdind) = 0;

% normForce = normForce .* (normForce>0.12); similar to above code but may

%% Plots showing where force was applied

normemg2(normemg2 == 0) = nan;
figure(1)
plot(timex2,normemg);
title('Sections of Interest on EMG Plot')
xlabel('Time(s)')
ylabel('Voltage(V)')
ylim([0.6 0.8])
hold on;
ylim([0.6 0.8])
plot(timex2,normemg2,'color','r');
legend('Original Signal', 'Sectioned Signal')
hold off;

normForce2(normForce2 == 0) = nan;
figure(2)
plot(timex2,normForce);
title('Sections of Interest on Force Plot')
xlabel('Time(s)')
ylabel('Voltage(V)')
hold on;
plot(timex2,normForce2,'color','r');
legend('Original Signal', 'Sectioned Signal')
hold off;


%% Find range function to find the areas of interest

nanLocations = ~isnan(normForce2); % Get logical array of whether element is NaN or not.
props = regionprops(nanLocations, 'Area', 'PixelIdxList'); % Find all the regions.
% DONE!  Now let's print them out
for k = 1 : length(props)
  fprintf('Region #%d has length %d and starts at element %d and ends at element %d\n',...
```

```
        k, props(k).Area, props(k).PixelIdxList(1), props(k).PixelIdxList(end));
end

%% Window for each section of the signals outlined by the code above

%creating vectors with the respective window sizes based off of above code
forcesec1 = normForce(4949:8720);
forcesec2 = normForce(11873:14696);
forcesec3 = normForce(18009:21180);
forcesec4 = normForce(24753:28580);
forcesec5 = normForce(31773:35976);
forcesec6 = normForce(39069:43324);

emgsec1 = normemg(4949:8720);
emgsec2 = normemg(11873:14696);
emgsec3 = normemg(18009:21180);
emgsec4 = normemg(24753:28580);
emgsec5 = normemg(31773:35976);
emgsec6 = normemg(39069:43324);

%windows
sampfreq=1000;
window1=0.25;
window1=window1*sampfreq;

%Short-Term Analysis Variables
%FSec1
[AverageF1,VarianceF1,DRF1,RMSF1,MSF1]= WindowData(emgsec1,window1);
%Flip to Vertical
AverageF1=mean(AverageF1);
meanF1=mean(forcesec1);
VarianceF1=mean(VarianceF1);
DRF1=mean(DRF1);
RMSF1=mean(RMSF1);
MSF1=mean(MSF1);

%FSec2
[AverageF2,VarianceF2,DRF2,RMSF2,MSF2]= WindowData(emgsec2,window1);
%Flip to Vertical
AverageF2=mean(AverageF2);
meanF2=mean(forcesec2);
VarianceF2=mean(VarianceF2);
DRF2=mean(DRF2);
```

```
RMSF2=mean(RMSF2);
MSF2=mean(MSF2);

%FSec3
[AverageF3,VarianceF3,DRF3,RMSF3,MSF3]= WindowData(emgsec3,window1);
%Flip to Vertical
AverageF3=mean(AverageF3);
meanF3=mean(forcesec3);
VarianceF3=mean(VarianceF3);
DRF3=mean(DRF3);
RMSF3=mean(RMSF3);
MSF3=mean(MSF3);

%FSec4
[AverageF4,VarianceF4,DRF4,RMSF4,MSF4]= WindowData(emgsec4,window1);
%Flip to Vertical
AverageF4=mean(AverageF4);
meanF4=mean(forcesec4);
VarianceF4=mean(VarianceF4);
DRF4=mean(DRF4);
RMSF4=mean(RMSF4);
MSF4=mean(MSF4);

%FSec5
[AverageF5,VarianceF5,DRF5,RMSF5,MSF5]= WindowData(emgsec5,window1);
%Flip to Vertical
AverageF5=mean(AverageF5);
meanF5=mean(forcesec5);
VarianceF5=mean(VarianceF5);
DRF5=mean(DRF5);
RMSF5=mean(RMSF5);
MSF5=mean(MSF5);

%FSec6
[AverageF6,VarianceF6,DRF6,RMSF6,MSF6]= WindowData(emgsec6,window1);
%Flip to Vertical
AverageF6=mean(AverageF6);
meanF6=mean(forcesec6);
VarianceF6=mean(VarianceF6);
DRF6=mean(DRF6);
RMSF6=mean(RMSF6);
MSF6=mean(MSF6);
%% Linear regression values
```

```
AverageFvector = [meanF1, meanF2, meanF3, meanF4, meanF5, meanF6];
%force
meanFvector = [AverageF1, AverageF2, AverageF3, AverageF4, AverageF5, AverageF6];
%Variance
meanVvector = [VarianceF1,VarianceF2,VarianceF3,VarianceF4,VarianceF5,VarianceF6];
%DR
meanDRvector = [DRF1, DRF2, DRF3, DRF4, DRF5, DRF6];
%RMS
meanRMSvector = [RMSF1, RMSF2, RMSF3, RMSF4, RMSF5, RMSF6];
%MS
meanMSvector = [MSF1, MSF2, MSF3, MSF4, MSF5, MSF6];
%% Linear Regression
[meanFaverage,coeffF]=LinearRegression(AverageFvector,meanFvector);
[meanVaverage,coeffV]=LinearRegression(AverageFvector,meanVvector);
[meanDRaverage,coeffDR]=LinearRegression(AverageFvector,meanDRvector);
[meanRMSaverage,coeffRMS]=LinearRegression(AverageFvector,meanRMSvector);
[meanMSaverage,coeffMS]=LinearRegression(AverageFvector,meanMSvector );

%% Linear regression plots
%force
figure(5)
scatter(AverageFvector,meanFvector);
title('Mean Force True Value vs Force')
xlabel('Force Vector')
ylabel('Mean Force Vector')
hold on
plot(AverageFvector,meanFaverage);
text(0.4,0.664,'y = 0.0096F + 0.6622');
hold off;

%variance
figure(6)
scatter(AverageFvector,meanVvector);
title('Variance True Value vs Force')
xlabel('Force Vector')
ylabel('Mean Force Variance')
hold on
plot(AverageFvector,meanVaverage);
text(0.4,0.0055,'y = -0.0021F + 0.0064');
hold off;

%DR
figure(7)
```

```
scatter(AverageFvector,meanDRvector);
title('Dynamic Range True Value vs Force')
xlabel('Force Vector')
ylabel('Mean Force Dynamic Range')
hold on
plot(AverageFvector,meanDRaverage);
text(0.4,0.075,'y = 0.0351F + 0.0653');
hold off;

%RMS
figure(8)
scatter(AverageFvector,meanRMSvector);
title('Root Mean Squared True Value vs Force')
xlabel('Force Vector')
ylabel('Mean Force Root Mean Squared')
text(0.4,0.673,'y = 0.0065F + 0.6714');
hold on
plot(AverageFvector,meanRMSaverage);
hold off;

%MS
figure(9)
scatter(AverageFvector,meanMSvector);
title('Mean Squared True Value vs Force')
xlabel('Force Vector')
ylabel('Mean Force Mean Squared')
text(0.4,0.458,'y = 0.0067F + 0.4568');
hold on
plot(AverageFvector,meanMSaverage);
hold off;
%% Residuals
% force
res1 = meanFvector - meanFaverage;
% variance
res2 = meanVvector - meanVaverage;
% DR
res3 = meanDRvector - meanDRaverage;
% RMS
res4 = meanRMSvector - meanRMSaverage;
%MS
res5 = meanMSvector - meanMSaverage;
%% Residual Plots
%force
```

```
figure(5)
scatter(AverageFvector,res1);
title('Force Residuals')
hold on
scatter(AverageFvector,meanFaverage);
legend('True Values','Residuals');
hold off;

%variance
figure(6)
scatter(AverageFvector,res2);
title('Variance Residuals')
hold on
scatter(AverageFvector,meanVaverage);
legend('True Values','Residuals');
hold off;

%DR
figure(7)
scatter(AverageFvector,res3);
title('Dynamic Range Residuals')
hold on
scatter(AverageFvector,meanDRaverage);
legend('True Values','Residuals');
hold off;

%RMS
figure(8)
scatter(AverageFvector,res4);
title('Root Mean Squared Residuals')
hold on
scatter(AverageFvector,meanRMSaverage);
legend('True Values','Residuals');
hold off;

%MS
figure(9)
scatter(AverageFvector,res5);
title('Mean Squared Residuals')
hold on
scatter(AverageFvector,meanMSaverage);
legend('True Values','Residuals');
hold off;
```

```
%% Mean square error and Correlation Coefficients
[MSEF,coeffF]=MSEandResiduals(AverageFvector,meanFvector);
[MSEV,coeffV]=MSEandResiduals(AverageFvector,meanVvector);
[MSEDR,coeffDR]=MSEandResiduals(AverageFvector,meanDRvector);
[MSERMS,coeffRMS]=MSEandResiduals(AverageFvector,meanRMSvector);
[MSEMS,coeffMS]=MSEandResiduals(AverageFvector,meanMSvector);

%% 20 Windows

windows = [0.02,0.04,0.06,0.08,0.1,0.2,0.4,0.6,0.8,1,1.1,1.2,1.3,1.4,1.5,1.6,1.7,1.8,1.9,2];
for i = 1:20
        window=windows(i);
        window=window*sampfreq;

[msemean2(i),coeffmean2(i),msems2(i),coeffms2(i),mserms2(i),coeffrms2(i)]=exp2p2(AverageF
vector,window,emgsec1,emgsec2,emgsec3,emgsec4,emgsec5,emgsec6);
end

%making data vetors length 20
msemean2=msemean2(1:20);
coeffmean2=coeffmean2(1:20);
msems2=msems2(1:20);
coeffms2=coeffms2(1:20);
mserms2=mserms2(1:20);
coeffrms2=coeffrms2(1:20);

figure(10)
plot(windows,msemean2);
hold on
title('MSE and Correlation Coefficients for Mean');

plot(windows,coeffmean2);
legend('MSE','Correlation Coefficients');
hold off

figure(11)
plot(windows,msems2);
hold on
title('MSE and Correlation Coefficients for MS');
xlabel('Window size (sec)')
plot(windows,coeffms2);
legend('MSE','Correlation Coefficients');
hold off
```

```
figure(12)
plot(windows,mserms2);
hold on
title('MSE and Correlation Coefficients for RMS');
xlabel('Window size (sec)')
plot(windows,coeffrms2);
legend('MSE','Correlation Coefficients');
hold off
```

Window function:

```
function [averagedata,variancedata,dynamicrangedata,rmsdata,meansqrdata] =
WindowData(bicepdata,window)
%Pad
w=zeros(1,window);
w=w.';
padded=vertcat(bicepdata,w);
lengthofdata=length(bicepdata);

%Window
for i = 1:lengthofdata
        windowdata=padded(i:i+window);
        avedat(i)=mean(windowdata);
        vardat(i)=var(windowdata);
        dynamicrangedat(i)=max(windowdata)-min(windowdata);
        RMSdat(i)=rms(windowdata);
        MSdat(i)=(rms(windowdata))^2;
end

averagedata=avedat;
variancedata=vardat;
dynamicrangedata=dynamicrangedat;
rmsdata=RMSdat;
meansqrdata=MSdat;

End
```

Linear regression function:

```
function [output,coeff] = LinearRegression(ForceVector,AnalysisVector)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
coeff=polyfit(ForceVector,AnalysisVector,1);
output=polyval(coeff,ForceVector);
```

```
end
```

MSE and residuals function:

```
function [MSE,CorrCoeff] = MSEandResiduals(AverageForceVector,AnalysisVector)
k=corrcoef(AverageForceVector,AnalysisVector);
%Used to take value out of array for better automation
k=k(1,2);
CorrCoeff=k;
MSE=immse(AverageForceVector,AnalysisVector);
end
```

Exercise 2.2 function for twenty window sizes:

```
function [mse_mean,cc_mean,mse_ms,cc_ms,mse_rms,cc_rms] =
exp2p2(AverageForceVector,windowsize,Section1,Section2,Section3,Section4,Section5,Sectio
n6)

[Mean1,Variance1,DR1,RMS1,MS1]= WindowData(Section1,windowsize);
Average1=mean(Mean1);
RMS1=mean(RMS1);
MS1=mean(MS1);
[Mean2,Variance2,DR2,RMS2,MS2]= WindowData(Section2,windowsize);
Average2=mean(Mean2);
RMS2=mean(RMS2);
MS2=mean(MS2);
[Mean3,Variance3,DR3,RMS3,MS3]= WindowData(Section3,windowsize);
Average3=mean(Mean3);
RMS3=mean(RMS3);
MS3=mean(MS3);
[Mean4,Variance4,DR4,RMS4,MS4]= WindowData(Section4,windowsize);
Average4=mean(Mean4);
RMS4=mean(RMS4);
MS4=mean(MS4);
[Mean5,Variance5,DR5,RMS5,MS5]= WindowData(Section5,windowsize);
Average5=mean(Mean5);
RMS5=mean(RMS5);
MS5=mean(MS5);
[Mean6,Variance6,DR6,RMS6,MS6]= WindowData(Section6,windowsize);
Average6=mean(Mean6);
RMS6=mean(RMS6);
MS6=mean(MS6);
```

```
MeanVector=[Average1,Average2,Average3,Average4,Average5,Average6];
RMSVector=[RMS1,RMS2,RMS3,RMS4,RMS5,RMS6];
MSVector=[MS1,MS2,MS3,MS4,MS5,MS6];

[mse_mean,cc_mean]=MSEandResiduals(AverageForceVector,MeanVector);
[mse_ms,cc_ms]=MSEandResiduals(AverageForceVector,MSVector);
[mse_rms,cc_rms]=MSEandResiduals(AverageForceVector,RMSVector);
End
```

Exercise 3:

```
% Opening File
m = readtable('GraspFatigue.csv');

% Creating vectors
tx2 = m(:,1); %extracting time values from the experiment data
tx2 = tx2{:,:}; %making the matrix useful for calculations
emgx2 = m(:,2); %extracting emg values from the experiment data
emgx2 = emgx2{:,:}; %making the matrix useful for calculations
forcex2 = m(:,3); %extracting force values from the experiment data
forcex2 = forcex2{:,:}; %making the matrix useful for calculations

timex2 = 0 : length(tx2) - 1;
timex2 = timex2 * 0.001; %conversion of time array to take into the account the sampling rate
(1000Hz)

%Normalizing the data
normForce = forcex2 - min(forcex2(:));
normForce = normForce ./ max(normForce(:));

normemg = emgx2 - min(emgx2(:));
normemg = normemg ./ max(normemg(:));



%% Experiment 1 for the full emg and force signals (not needed)
% for force
sampfreq=1000;
window1=0.05;
window1=window1*sampfreq;
```

```
%Short-Term Analysis Variables
[Average1,Variance1,DR1,RMS1,MS1]= WindowData(normForce,window1);
%Flip to Vertical
Average1=Average1.';
Variance1=Variance1.';
DR1=DR1.';
RMS1=RMS1.';
MS1=MS1.';

%Plot of Short-Term Analysis Variables over Original
figure(3)
plot(timex2,normForce);
title('Plot of Short-Term Analysis Features of Force Data With Window Size 0.25s')
xlabel('Time(s)')
ylabel('Voltage(V)')
hold on
plot(timex2,Average1);
plot(timex2,Variance1);
plot(timex2,DR1);
plot(timex2,RMS1);
plot(timex2,MS1);
legend('Original Data','Average','Variance','Dynamic Range','RMS','MeanSquare')
hold off

% for emg

%Short-Term Analysis Variables
[Average2,Variance2,DR2,RMS2,MS2]= WindowData(normemg,window1);
%Flip to Vertical
Average2=Average2.';
Variance2=Variance2.';
DR2=DR2.';
RMS2=RMS2.';
MS2=MS2.';

%Plot of Short-Term Analysis Variables over Original
figure(4)
plot(timex2,normemg);
title('Plot of Short-Term Analysis Features of EMG Data With Window Size 0.25s')
xlabel('Time(s)')
ylabel('Voltage(V)')
hold on
plot(timex2,Average2);
```

```
plot(timex2,Variance2);
plot(timex2,DR2);
plot(timex2,RMS2);
plot(timex2,MS2);
legend('Original Data','Average','Variance','Dynamic Range','RMS','MeanSquare')
hold off


%% Window for each section of the signals outlined by the code above

%creating vectors with the respective window sizes based off of above code
forcesec1 = normForce(4949:8720);
forcesec2 = normForce(11873:14696);
forcesec3 = normForce(18009:21180);
forcesec4 = normForce(24753:28580);
forcesec5 = normForce(31773:35976);
forcesec6 = normForce(39069:43324);

emgsec1 = normemg(4949:8720);
emgsec2 = normemg(11873:14696);
emgsec3 = normemg(18009:21180);
emgsec4 = normemg(24753:28580);
emgsec5 = normemg(31773:35976);
emgsec6 = normemg(39069:43324);

%windows
sampfreq=1000;
window1=0.05;
window1=window1*sampfreq;

%Short-Term Analysis Variables
%FSec1
[AverageF1,VarianceF1,DRF1,RMSF1,MSF1]= WindowData(emgsec1,window1);
%Flip to Vertical
AverageF1=mean(AverageF1);
meanF1=mean(forcesec1);
VarianceF1=mean(VarianceF1);
DRF1=mean(DRF1);
RMSF1=mean(RMSF1);
MSF1=mean(MSF1);

%FSec2
[AverageF2,VarianceF2,DRF2,RMSF2,MSF2]= WindowData(emgsec2,window1);
```

```
%Flip to Vertical
AverageF2=mean(AverageF2);
meanF2=mean(forcesec2);
VarianceF2=mean(VarianceF2);
DRF2=mean(DRF2);
RMSF2=mean(RMSF2);
MSF2=mean(MSF2);

%FSec3
[AverageF3,VarianceF3,DRF3,RMSF3,MSF3]= WindowData(emgsec3,window1);
%Flip to Vertical
AverageF3=mean(AverageF3);
meanF3=mean(forcesec3);
VarianceF3=mean(VarianceF3);
DRF3=mean(DRF3);
RMSF3=mean(RMSF3);
MSF3=mean(MSF3);

%FSec4
[AverageF4,VarianceF4,DRF4,RMSF4,MSF4]= WindowData(emgsec4,window1);
%Flip to Vertical
AverageF4=mean(AverageF4);
meanF4=mean(forcesec4);
VarianceF4=mean(VarianceF4);
DRF4=mean(DRF4);
RMSF4=mean(RMSF4);
MSF4=mean(MSF4);

%FSec5
[AverageF5,VarianceF5,DRF5,RMSF5,MSF5]= WindowData(emgsec5,window1);
%Flip to Vertical
AverageF5=mean(AverageF5);
meanF5=mean(forcesec5);
VarianceF5=mean(VarianceF5);
DRF5=mean(DRF5);
RMSF5=mean(RMSF5);
MSF5=mean(MSF5);

%FSec6
[AverageF6,VarianceF6,DRF6,RMSF6,MSF6]= WindowData(emgsec6,window1);
%Flip to Vertical
AverageF6=mean(AverageF6);
meanF6=mean(forcesec6);
```

```
VarianceF6=mean(VarianceF6);
DRF6=mean(DRF6);
RMSF6=mean(RMSF6);
MSF6=mean(MSF6);

%% Linear regression
AverageFvector = [meanF1, meanF2, meanF3, meanF4, meanF5, meanF6];

%force
meanFvector = [AverageF1, AverageF2, AverageF3, AverageF4, AverageF5, AverageF6];
coeff1 = polyfit(AverageFvector,meanFvector,1);
meanFaverage = polyval(coeff1,AverageFvector);

%Variance
meanVvector = [VarianceF1,VarianceF2,VarianceF3,VarianceF4,VarianceF5,VarianceF6];
coeff2 = polyfit(AverageFvector,meanVvector,1);
meanVaverage = polyval(coeff2,AverageFvector);

%DR
meanDRvector = [DRF1, DRF2, DRF3, DRF4, DRF5, DRF6];
coeff3 = polyfit(AverageFvector,meanDRvector,1);
meanDRaverage = polyval(coeff3,AverageFvector);

%RMS
meanRMSvector = [RMSF1, RMSF2, RMSF3, RMSF4, RMSF5, RMSF6];
coeff4 = polyfit(AverageFvector,meanRMSvector,1);
meanRMSaverage = polyval(coeff4,AverageFvector);

%MS
meanMSvector = [MSF1, MSF2, MSF3, MSF4, MSF5, MSF6];
coeff5 = polyfit(AverageFvector,meanMSvector,1);
meanMSaverage = polyval(coeff5,AverageFvector);

%force
figure(5)
text(0,0,'y = 0.0096F + 0.6622');
scatter(AverageFvector,meanFvector);
hold on
plot(AverageFvector,meanFaverage);
hold off;
title('Force residuals');

%variance
```

```
figure(6)
text(0,0,'y = -0.0021F + 0.0064');
scatter(AverageFvector,meanVvector);
hold on
plot(AverageFvector,meanVaverage);
hold off;
title('Variance residuals');

%DR
figure(7)
text(0,0,'y = 0.0351F + 0.0653');
scatter(AverageFvector,meanDRvector);
hold on
plot(AverageFvector,meanDRaverage);
hold off;
title('Dynamic Range residuals');

%RMS
figure(8)
text(0,0,'y = 0.0065F + 0.6714');
scatter(AverageFvector,meanRMSvector);
hold on
plot(AverageFvector,meanRMSaverage);
hold off;
title('Root Mean Squared residuals');

%MS
figure(9)
text(0,0,'y = 0.0067F + 0.4568');
scatter(AverageFvector,meanMSvector);
hold on
plot(AverageFvector,meanMSaverage);
hold off;
title('Mean Squared residuals');

%% Correlation coefficient

%forece
R1 = corrcoef(AverageFvector,meanFvector);

% Variance
R2 = corrcoef(AverageFvector,meanVvector);
```

```
% DRF
R3 = corrcoef(AverageFvector,meanDRvector);

% RMS
R4 = corrcoef(AverageFvector,meanRMSvector);

%MS
R5 = corrcoef(AverageFvector,meanMSvector);

%% Mean square error

% force
MSE1 = immse(AverageFvector,meanFvector);

%variance
MSE2 = immse(AverageFvector,meanVvector);

% DRF
MSE3 = immse(AverageFvector,meanDRvector);

% RMS
MSE4 = immse(AverageFvector,meanRMSvector);

% MS
MSE5 = immse(AverageFvector,meanMSvector);
```

Windowdata.m

```
function [averagedata,variancedata,dynamicrangedata,rmsdata,meansqrdata] =
WindowData(bicepdata,window)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
%PADDING DATA
c=zeros(1,window);
c=c.';
padded=vertcat(bicepdata,c);
lengthofdata=length(bicepdata);


%CREATE WINDOW
for i = 1:lengthofdata
    %GET WINDOW DATA
```

```
    windowdata=padded(i:i+window);
    %GET SHORT TERM ANALYSIS VARIABLE
    avedat(i)=mean(windowdata);
    vardat(i)=var(windowdata);
    dynamicrangedat(i)=max(windowdata)-min(windowdata);
    RMSdat(i)=rms(windowdata);
    MSdat(i)=(rms(windowdata))^2;
end

averagedata=avedat;
variancedata=vardat;
dynamicrangedata=dynamicrangedat;
rmsdata=RMSdat;
meansqrdata=MSdat;


end
```

Linear regression.m

```
function [output,coeff] = LinearRegression(ForceVector,AnalysisVector)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here
coeff=polyfit(ForceVector,AnalysisVector,1);
output=polyval(coeff,ForceVector);
end
```