**Computer Assignment 1: Simple Statistics**

John Lee

ECE 3522: Stochastic Processes in Signals

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 1912

# Problem Statement

This computer assignment will be distributed in three separate tasks. Each task’s objective is to familiarize ourselves with understanding the purpose of statistical data through the use of the MATLAB software.

**Task 1:** Relates to the simple understanding on plotting a given data set, which is compiled on a excel spreadsheet. We were given a set of numbers and asked to simply plot the given values to observe the entire statistical data. We observed a rugged curve in which contains all of the values from the googles stocks information.

**Task 2:** Our second task simply asked us to compute the values obtained after loading and inputting both google and audio file data information. We needed to find the values for the mean, median, variance, minimum and maximum values.

**Task 3:** We were given a set of numbers of frames and samples, which were used to represent the spectrogram of the audio file given from the assignment. Our purpose was to understand the change of line behavior as we increase/decrease the number of frames. We performed the plotting to compare the accuracy of its values to the ones obtained from Task 2.

# Approach and Results

The first two portions of the computer assignments were based on simple computation and commands that lead to basic results. We found that on **Figure 1**, after plotting the values of the excel data sheet, the plot generated a rugged curved line which represents the values of the google stocks during a given period of time. The first part of the assignments was loading a given data sheets and plot its values to observe statistical data. We computed the result on **Table 1,** the outcome for the *signal values vs. the obtained values* after using the RMS code.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Signal** | Mean | Median | Minimum | Maximum | Variance |
| Google | 286.75 | 264.78 | 47.93 | 61.4 | 16.191e3 |
| Audio | -0.389 | 83 | -14.993e3 | 10104 | 4.139e6 |

 |

 **Table 1.** Signal values.

 The purpose of these values being used, is to compare them to the next task, which asks you to plot multiple windows with respect of a given set of values that will affect the shape and form of the line graph.



**Figure 1.** Google stock prices plot. Time period vs Overall values.

The final task of the assignment was difficult to understand as it was based on last semesters working exercise. The given code which contained functions was not properly working as we tried to load our plots and values for both google stock prices and the raw audio file. Another approach that we used to load our values, was to use the for loop statements. MATLAB, suggested a set of formulas that we used to create these plots manually and to make sure that these values were generating a loop of at least 3-4 windows, based on the given numbers of the M-value.

Since we are given two sets of frame values M and samples N, we could observe from **Figure 2-5** that the graphs may looks alike, although the change of the window duration defines the smoothness of the graph, the increase of these frame values leads to a more stair cased plot. This occurs due to the increase of frames, which gives us more detailed and accurate values taken from the plots. Note that these are testing values, to observe the line behavior under specific frame-value conditions.

 **Figure 2.** M = 1 N=[7,30] **Figure 3.** M = 7 N = [7, 30]



**Figure4.**M = 14 N = [7, 30]

 

**Figure 5.** M = 30 N = [7, 30]

The second portion of the final part of the assignment, shows us that **Figure 6 – 7** are representations of the audio file. We could observe that a given set of frame values for M are tested and compared to the sample values N. We can also observe that the increase of the window duration also affects the smoothness of the graph.



**Figure 6.** M = 40 N = [160, 240]

  **Figure 7.** M = 80 N = [160, 240]



**Figure 8.** M = 160 N = [160, 240]

# MATLAB Code

The MATLAB code below contains a number of statements that generate the plots. Note that we are looking for the overall plot to understand the idea of the entire statistical data. The plot on **Figure 9** shows the statistical data of the google stock market values, based from time interval and all the values. **Figure 10** represents a simple set of statements that generate the signal values: Minimum, Maximum, Mean, Median and Variance.



**Figure 9.** Google Plot, MATLAB Code **Task 1.**



**Figure 10.** Signal values. **Task 2.**

Next, the MATLAB code below, represents the steps to plot a set of windows based on a given set of frame/sample values to later obtain the mean and variance results. At first, we ran through some syntax errors while using the standard RMS code given from last semester. Since MATLAB was constantly disagreeing on using function statements, we decided to go through the use of ‘for’ statements, which was suggested by the software itself. These for loops will allow us to run a multiple set of windows that pertains to the signal values (M,N) and run multiple tests that plots the values based on frame vs sample values. You will need to create a nested for loop of each parameter combination of the frame/sample values. Depending on the number of frame values, the loop will create a set of windows according to the number of M values given. Like any other loop, you will notice that there is a starting point and an ending point to make sure the loop statement does not run forever.







**Figure 11.** Google stocks frame/sample MATLAB code.







**Figure 12.** Audio file frame/sample MATLAB code.

# Conclusions

This assignment was designed to refresh most of our MATLAB skills and familiarize ourselves to understand and read the purpose of frames and sample applied on window plots. You will notice that the change of frame values affect the results of the graphs, as it could change the smoothness or giving a more accurate shape that is easy to determine the frequency values at certain points. Unfortunately, during our process of comparing results, we were not able to obtain the second values of the mean and variance for **Task 3.** Also to the lack of information as well as our inability to conclude this experiment, we experienced some difficulties in the process of figuring out the mean and the variance of the experimental value.

.V. MATLAB CODE II

%% Part 1

clear all;

clc;

% Loading the Excel data sheet

data = xlsread('google\_v00.xlsx');

% Setting the column locations with their respected variables

Date\_column = data(1,:);

Open\_column = data(2,:);

High\_column = data(3,:);

Low\_column = data(4,:);

Close\_column= data(5,:);

% Plot of google stocks data

figure(1);

p1 = plot(data);grid on

title('Google Stocks')

xlabel ('Time period');

ylabel('Overal Values');

%% Part 2

clear all;

clc;

% Min, Max, Mean, Median, Variance values for Google Stocks

data = xlsread('google\_v00.xlsx');

Faudio = fopen('rec\_01\_speech.raw','r');

b = fread(Faudio, inf, 'int16');

Minimum1 = min(data(:));

Maximum1 = max(data(:));

Mean1 = mean(data(:));

Median1 = median(data(:));

Var1 = var(data(:));

Valu1 = sprintf('Minimum=%d Maximum=%d Mean=%d Median=%d Variance=%d', Minimum1, Maximum1, Mean1, Median1, Var1);

display(Valu1);

% Min, Max, Mean, Median, Variance values for Audio file

Minimum2 = min(b(:));

Maximum2 = max(b(:));

Mean2 = mean(b(:));

Median2 = median(b(:));

Variance2 = var(b(:));

Valu2 = sprintf('Minimum=%d Maximum=%d Mean=%d Median=%d Variance=%d', Minimum2, Maximum2, Mean2, Median2, Variance2);

display(Valu2);

%% Part 3 Google Stock prices plot

clear;

clc;

%loading audio file onto matlab

a = xlsread('google\_v00.xlsx');

%defining M and N

M = [1, 7, 14, 30];

N = [7,30];

%beginning of loop over signal M

for m = 1:length(M)

 Fig = figure('name', 'rms plot', 'numbertitle', 'off');

%beginning of loop over signal N

 for n = 1:length(N)

 start = 1;

 %Main loop over signal

 for main = 1:round(length(a)/M(m))

 %Pointer of frame location

 point = round(M(m)\*(main-1)+M(m)/2);

 p\_begin = round(point - N(n)/2);

 p\_end = round(point + N(n)/2-1);

 % when your point starts and ends, your values will always be positive.

 if(p\_begin < 1)

 p\_begin = 1;

 end

 if(p\_end > length(a))

 p\_end = length(a);

 end

 % Generating the windows (Beginning pointer vs Ending pointer)

 winvector = (p\_begin : p\_end);

 Average = sum(a(winvector).^2 / N(n));

 % Creating the repetitions of the M values.

 for main2 = 1:(M(m))

 RMS(start) = sqrt(Average);

 start = start + 1;

 % Since length contains numerous values, we will make sure it does

 % not overpass the values of start

 if start > length(a)

 break;

 end

 end

 end

 while start <= length(a)

 RMS(start) = 0;

 start = start + 1;

 end

 % Generating the plots for all values.

 figure(Fig);

 subplot(1 + length(N), 1, n);

 plot(RMS);

 ValDisp = sprintf('M = %d N = %d',M(m),N(n));

 title(ValDisp);

 xlabel('Time');

 ylabel('RMS');

 % Plot of input signal

 figure(Fig);

 subplot(1 + length(N), 1, n + 1);

 plot(a);

 title('Signal');

 xlabel('Time');

 ylabel('Amplitude');

 end

end

% Obtaining the results of the google docs.

Mean1 = mean(a(:));

Variance1 = var(a(:));

Values = sprintf('Mean = %d Variance = %d', Mean1, Variance1);

display(Values);

%% Part 3. Raw audio plot.

clear all;

clc;

%loading audio file onto matlab

Faudio = fopen('rec\_01\_speech.raw','r');

b = fread(Faudio, inf, 'int16');

fclose(Faudio);

%defining M and N

M = [40, 80, 160];

N = [160, 240];

%beginning of loop over signal M

for m = 1:length(M)

 Fig = figure('name', 'rms plot', 'numbertitle', 'off');

%beginning of loop over signal N

 for n = 1:length(N)

 start = 1;

 %Main loop over signal

 for main = 1:round(length(b)/M(m))

 %Pointer of frame location

 point = round(M(m)\*(main-1)+M(m)/2);

 p\_begin = round(point - N(n)/2);

 p\_end = round(point + N(n)/2-1);

 % when your point starts and ends, your values will always be positive.

 if(p\_begin < 1)

 p\_begin = 1;

 end

 if(p\_end > length(b))

 p\_end = length(b);

 end

 % Generating the windows (Beginning pointer vs Ending pointer)

 winvector = (p\_begin : p\_end);

 Average = sum(b(winvector).^2 / N(n));

 % Creating the repetitions of the M values.

 for main2 = 1:(M(m))

 RMS(start) = sqrt(Average);

 start = start + 1;

 % Since length contains numerous values, we will make sure it does

 % not overpass the values of start

 if start > length(b)

 break;

 end

 end

 end

 while start <= length(b)

 RMS(start) = 0;

 start = start + 1;

 end

 % Generating the plots for all values.

 figure(Fig);

 subplot(1 + length(N), 1, n);

 plot(RMS);

 ValDisp = sprintf('M = %d N = %d',M(m),N(n));

 title(ValDisp);

 xlabel('Time');

 ylabel('RMS');

 figure(Fig);

 subplot(1 + length(N), 1, n + 1);

 plot(b);

 title('Signal');

 xlabel('Time');

 ylabel('Amplitude');

 end

 end

Mean2 = mean(b(:));

Variance2 = var(b(:));

Values = sprintf('Mean2 = %d Variance2 = %d', Mean2, Variance2);

display(Values);