Miles A. Vendetti-Houser

ECE 3522: Stochastics

Department of Electrical and Computer Engineering, Temple University, Philadelphia, PA 19122

# Problem Statement

The assignment is designed to familiarize students with statistical computational analysis using MATLAB since higher level and more difficult tasks will be required to work on as the course moves forward. We are provided with two sets of data; the first is Google’s stock price since its creation. This is served as an excel sheet containing properties such as; date, and closing/open/high/low stock prices for each corresponding date. The second set of data is the ol’ familiar speech signal, seen in Signals & Systems, saved as raw data. The audio clip is sampled at 8 kHz and lasts about 10 seconds.

The first part of the assignment has students find specific calculations that are popular in statistics. Being that this is a statistics and probability related course, one may assume that simple calculations may be expected on the first computer assignment. The computations required are as followed; minimum value, maximum value, mean, median, and variance. Further explanation will be provided regarding finding these values. The second part of the assignment involves implementing the same window-frame we saw in writing assignments from last semester. The program to be written on MATLAB should use frame values for the Google stock data which allows us to view the mean and variance over a day, one week, two weeks, and a month. The frame sizes for the signal relate to 5, 10, and 20 milliseconds. Finally, the output data from the window/frame system will be plotted and compared to original data.

# Approach and Results

In order to get started, the required data has to be imported. Importing the Excel sheet information was accomplished using the built-in function, xlsread(). The information in each column had to go through some simple data conversion processes, yet that was easily solved by using MATLAB’s extensive documentation. The audio signal was read and stored with all existing information from original source.

MATLAB provides users with a Statistical Toolbox which includes functions that calculate simple values such as the mean and maximum values of input data. The attached MATLAB script in section 4 below shows the built-in functions; mean(), median(), max(), min(), variance() all in use in order to determine the desired values. The following table displays all of the results from MATLAB functions.

Table

|  |  |  |
| --- | --- | --- |
|  | Google Stock | Audio File |
| Minimum | 49.95 | -12639 |
| Maximum | 609.5 | 9519 |
| Median | 264.8 | 83 |
| Mean | 286.7 | -1.88 |
| Variance | 1.92\*104 | 5.02\*106 |

We can see that these values make sense in reference to a variety of factors. We would hope to assume that the minimum value would never cross below $0.00 (*investors can lose everything they invested in a stock, but they’ll never need to pay more*) so the resulting minimum value can be assumed correct. The table also shows a large value for the maximum. This tells us that Google went up in demand as a company since its inception which of course can be implied by looking at the various technologies and products available on the market. The large value could have been assumed because nowadays we cannot go a single day without seeing a product designed or created by Google.

The original stock closing data was cut and shortened to contain enough information to plot over a one year period. Smaller data is important because the window and framing technique uses very small frame sizes. If we kept the entire range of Google’s stock values, the difference between plots may be unnoticeable resulting in useless data.

As we can see in figure below, the frame size is set to one day. Day by day, a window size of a week is computed in order to find the mean leaving a smooth curve portraying soft transitions between closing price values. Below, the window size increases to a month where a larger mean is calculated, hence providing even smoother transitions. This plot does remain the smoothest curve out of all of the possible permutations.

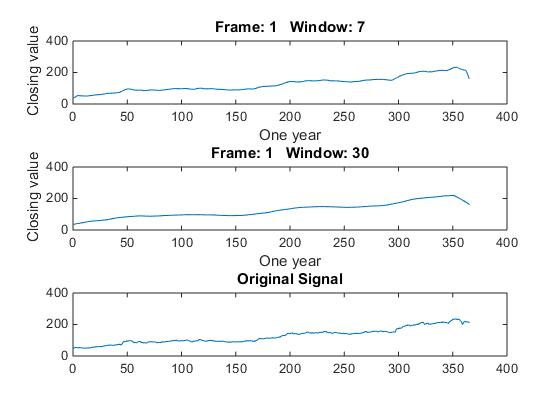


Figure 1:One day Frame for Google Stock

The frame and windows sizes are changed to the span of a single week. From the looks of the figure below, you can see small steps throughout the year.

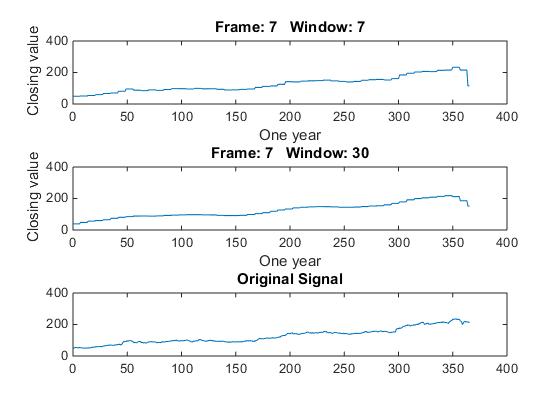


Figure 2: One Week Frame for Google Stock

The differences from each week are beginning to look a little more explicit. However, it seems ideal to continue increasing our frame size if we wish to analyze distinct changes in Google’s first year. In which case, we now try a frame size of both two weeks and one month. Comparing both graphs shown below, we can count the number of steps made with the one month frame, yet this does not show definitive periods in time when the stock could have fell. The two week frame size covers this loss slightly better than the month frame, but not as well as the one week frame size with a one week window.

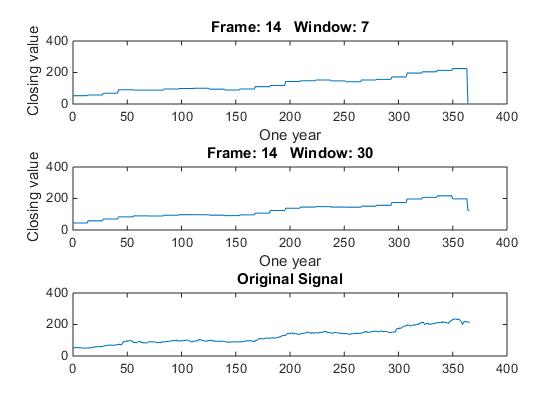


Figure 3: Two Week Frame for Google Closing

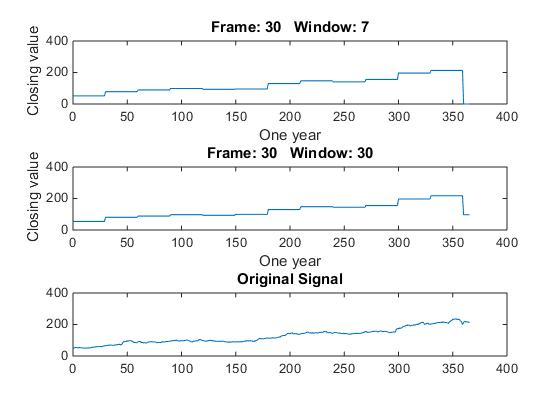


Figure 4: One month Frame size for stock

When dealing with the audio signal, we must know the format of the data before import. The audio file was created using Audacity which defaults to 16-bit integers. Therefore, we required MATLAB to import the data as int16 numeric values. As for plotting the audio signal and computing the mean, the same window-frame was used as the stock information. The assignment required looking for small periods of time (5-20ms). In order to display efforts made regarding the mean calculation, it was key to shrink the original 10 second long clip to about 8000 points of data or 1 second of sample time. With the frame size of 5, 10, and 20ms plotted below with corresponding window sizes of 160 and 240 (20 and 40 milliseconds), we notice a pattern with larger frame sizes. The shape of the curve contains points whose differences are much more noticeable when compared to smaller frames. As referenced in class, the chosen frame/window combination is all relative to the application of the system.

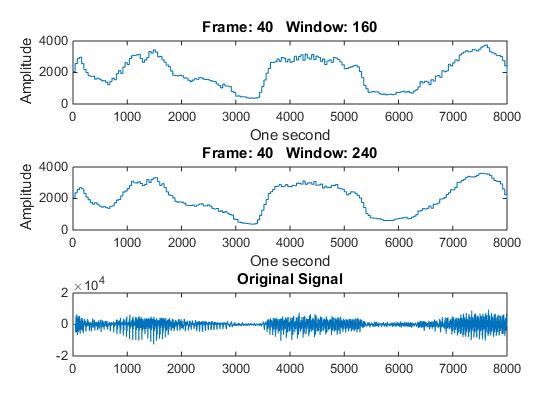


Figure 5: 5ms frame duration

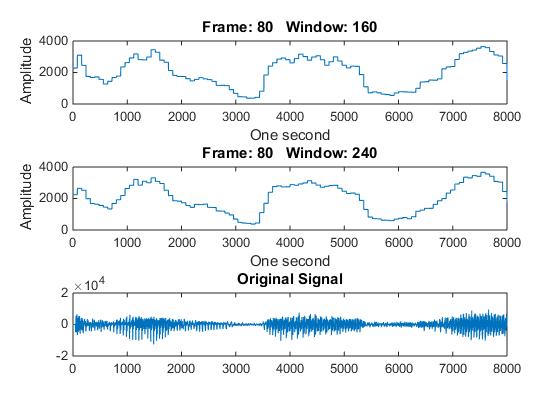


Figure 6: 10ms Frame Duration - Audio

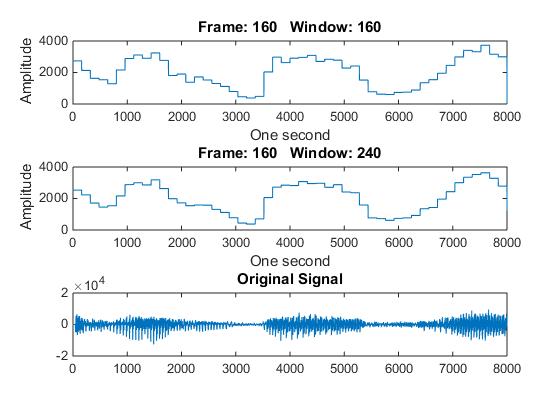


Figure 7: 20ms frame duration

# MATLAB Code

%% Miles A. Vendetti-Houser

% Computer Assignment 01

% Stochastics | Dr. Picone | ECE 3522

%

% 1. Load and plot the data from excel sheet.

%

% 2. Compute global statistics on signals;

%

% a. Minimum value, Maximum value, Mean, Median, and Variance.

%

% 3. Define a window of data to be N samples Define a frame to be M

% samples (e.g., every ten samples). Loop over these signals and

% compute the mean and variance for each window of data. Plot the

% results. Do this for the following parameter combinations:

%

% Google Stock Price: M = 1, 7, 14, 30 (days); N = 7, 30 (days)

% [use the closing value]

% Audio File: M = 40 (5 msec), 80 (10 msec), 160 (20 msec);

% N = 160 (20 msec), 240 (30 msec)

% Compare and contrast these plots to the values obtained

% in step 2.

% Problem One

clear;clc;close all;

%variable declarations

filename = 'google\_v00.xlsx';

audio\_file.location = 'rec\_01\_speech.raw';

%import the data

[~,~,raw] = xlsread(filename);

% import the audio signal

ff1 = fopen(audio\_file.location, 'r');

sig = fread(ff1, inf, 'int16');

fclose(ff1);

%put data into allocated vectors

dateStr = char(raw(:,1));

data.open = cell2mat(raw(2:end,2));

data.high = cell2mat(raw(2:end,3));

data.low = cell2mat(raw(2:end,4));

data.close = cell2mat(raw(2:end,5));

%convert date string to numeric type

formatIn = 'mm/dd/yyyy';

data.date = datenum(dateStr(2:end,:), formatIn);

%plot the data

clf;

figure(1)

plot(data.date, data.close);

title('Close')

% Using built-in functions in order to find the following values for

% Google stock closing values:

%

% mean

% max AND min

% variance

data.min = min(data.close)

data.max = max(data.close)

data.mean = mean(data.close)

data.var = var(data.close)

% define two key parameters:

% M: frame duration in samples - how often we compute an output

% N: window duration in samples - how much data we use in each computation

%

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

N = [ 7, 30 ];

oneYear = 365;

data.shortclose = data.close(1:oneYear);

% create a matrix to store the output

%

rms = zeros(length(M), length(N), length(data.shortclose));

% loop over the a set of frame/window combinations.

%

rows = length(N)+1;

cols = 1;

for j = 1:length(M)

plot\_location = 0;

num = sprintf('%d',j);

h1 =figure('name', num, 'numbertitle', 'off');

for k = 1:length(N)

figure(h1)

plot\_location=plot\_location+1;

rms(j,k,:) = compute\_rms(data.shortclose, M(j), N(k));

subplot( rows, cols, plot\_location );

plot(squeeze(rms(j,k,:)));

str = sprintf('Frame: %d Window: %d', M(j), N(k));

title(str);

xlabel('One year');

ylabel('Closing value');

end

subplot(rows, cols, plot\_location+1 );

plot(data.shortclose);

title('Original Signal');

end

% define two key parameters:

% M: frame duration in samples - how often we compute an output

% N: window duration in samples - how much data we use in each computation

%

M = [ 40 , 80, 160];

N = [ 160, 240 ];

% Take only one second of data!

oneSecond = 8000;

new\_sig = sig(1:oneSecond);

% create a matrix to store the output

%

rms = zeros(length(M), length(N), length(new\_sig));

% loop over the a set of frame/window combinations.

%

rows = length(N)+1;

cols = 1;

for m = 1:length(M)

plot\_location = 0;

num = sprintf('%d',m);

h1 =figure('name', num, 'numbertitle', 'off');

for n = 1:length(N)

figure(h1)

plot\_location=plot\_location+1;

rms(m,n,:) = compute\_rms(new\_sig, M(m), N(n));

subplot( rows, cols, plot\_location );

plot(squeeze(rms(m,n,:)));

str = sprintf('Frame: %d Window: %d', M(m), N(n));

title(str);

xlabel('One second');

ylabel('Amplitude');

end

subplot(rows, cols, plot\_location+1 );

plot(new\_sig);

title('Original Signal');

end

audio\_file.mean = mean(new\_sig);

audio\_file.var = var(new\_sig);

audio\_file.max = max(new\_sig);

audio\_file.min = min(new\_sig);

audio\_file.median = median(sig);

# Conclusions

The assignment provided expanded solidarity about a range of misconceptions and questions that would have otherwise remained unknown or left to post-college experience for discovery. A major breakthrough unmentioned throughout the results and analysis was the uncertainty of how to handle signals in any shape or form when specific computations on the data are asked to be performed. Repeated exposure of assignments that require manipulation on various types of signals conditions students to not only to expand computational skillset, but to allow them unbounded reign in terms of how they should represent the data, how to perform calculations on small-to-large sets of data, and introduce a new method of visualizing and learning concepts considered core in class.

The reuse of code was important here if minimal time was to be wasted. I am speaking of the compute\_rms function from previous semester. Learning the fact that computing the mean of discrete data was the same as finding the energy of a signal allowed efficient results in a timely manner. Also, the window-framing technique from last semester seems to show up frequently in many software applications involving data analysis. When plotting the output from the aforementioned technique, the desired plots must be relative to the frame-window values or very little conclusion can be made regarding a proper choice for the frame-window sizes.