Devin Trejo

ECE 3522: Stochastic Processes in Signals and Systems

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

I. PROBLEM STATEMENT

The purpose of this analysis is to take to see how variance changes if you were reading in real-time data. The analysis simulates for instance live tv broadcasts where the output signal is not completely known. Since we do not have all the data upfront, the variance of the signal will change as more of the signal comes in.

We will calculate our variance using three different approaches. For all analyses we will use both the speech signal and the Google Stock price. The first variance we calculate is the variance given the entire signal. We can find the variance by taking the second moments and subtracting the mean of the signal. This variance will be plotted as a horizontal line in our plots.

The second approach we take is simulating a streaming in data signal (such as our live TV broadcast). We begin by taking the first 10 samples and sequentially compute the variance one sample at a time. Therefore our samples start at 10 then 11, 12, 13, and so on. We plot this variance on the plot as our variance for the entire signal as a whole.

The third approach will use a frame and window method which we have explored in previous assignments. We find the variance for a window of size 30 days at a time. Our frame shifts by a day at a time. We again plot the variance using this method on the same plane to perform comparisons with the previous variances we found. Overall, this assignment asks us to see how a mean/variance changes overtime.

II. APPROACH AND RESULTS

The different methods we proposed are all done for both our Google Stock price and the speech signal. The different methods highlight how the variance and mean change as time progresses for a live signal broadcast. The variance at the beginning of a signal may be largely different compared to the true actual variance.

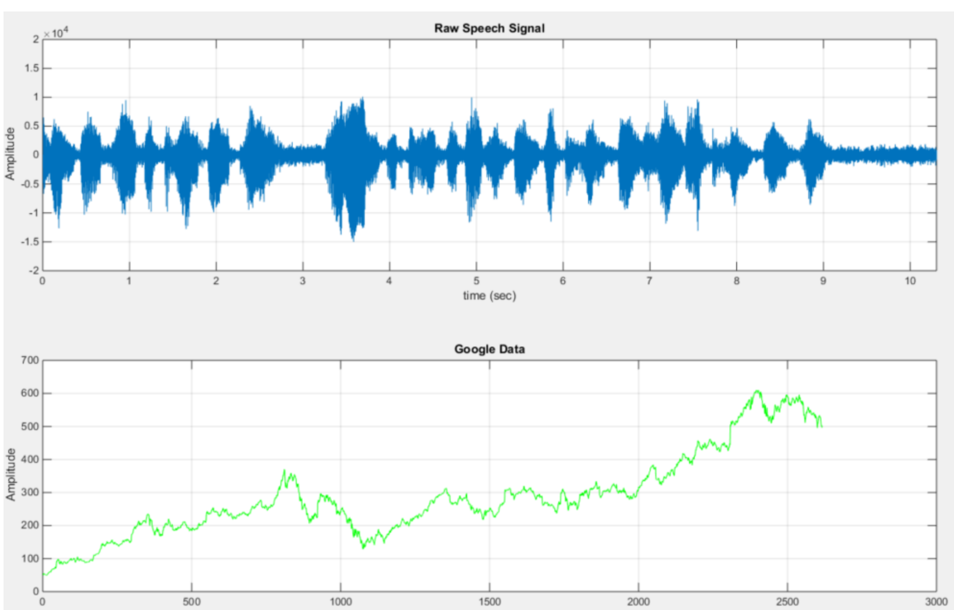

Figure 1: Graphical representation of our Speech and Google Stock price signals.

First for our google stock we obtained the following plot:

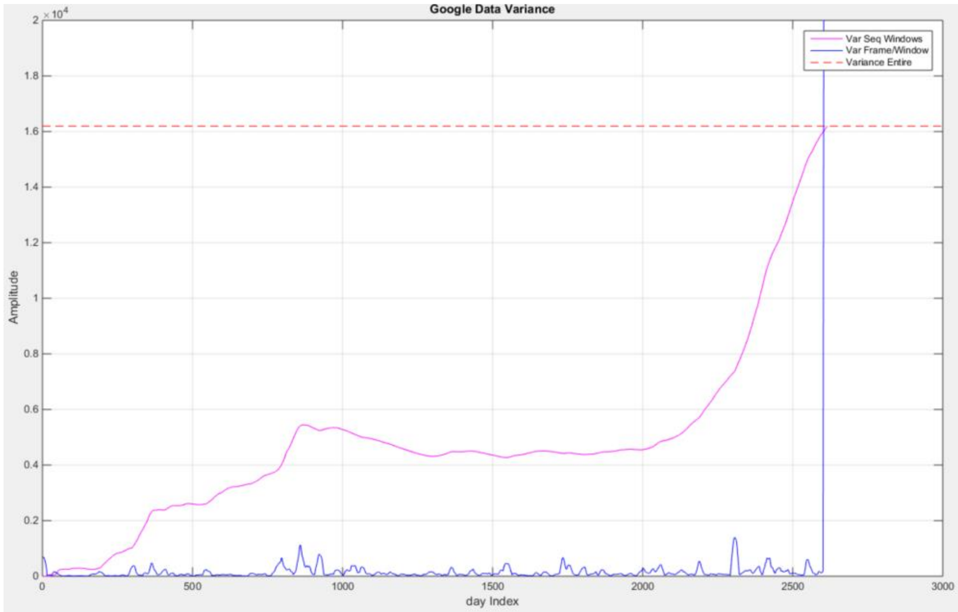

Figure 2: Google Stock Variance using the three methods described

Analyzing this plot shows us the difference in calculating variance. First we see the variance if given the entire signal up front (red dotted line). The variance is rather high indicating that the Google Stock price ranges widely from the expected value (the mean). It is to be expected that we see a high variance since google stock price does not same at the price for an extending period of time. It is always changing.

If we pretend that our Google stock is being read in real time (one day at a time) we see that variance gradually increases (magenta line). As we expand our sample space both the mean and variance changes. For example, for the first ~100 days our variance is rather low since our variance and mean both increase. At around day 400 Google's stock increases rather quickly which corresponds to stark increase in variance. As we incorporate more data into our sample space our values start varying more from the mean. We also should note where the magneta line is pretty steady. This no change in variance portion of the plot indicates that Google's stock price did not change much during this time. Referencing Figure 1 for this range (day 1500 -> 2000) we observed that indeed Google's stock is steady in price. Once we have the entire signal in our sample space [N = length (speech_signal)] we see our variance reach the true variance of the signal.

Our third approach separated our Google stock into smaller windows sizes (blue line). Instead of looking at the signal as a whole we took smaller portions of the signal and calculated variance. As expected with smaller sample spaces the variance is not that high. It is a rather good characteristic for a stock price since it indicates Google's stock price does not fluctuate widely day to day. (Ignore the spike in variance at the end which occurs due to zero stuffing)

We now shift focus to our speech signal:

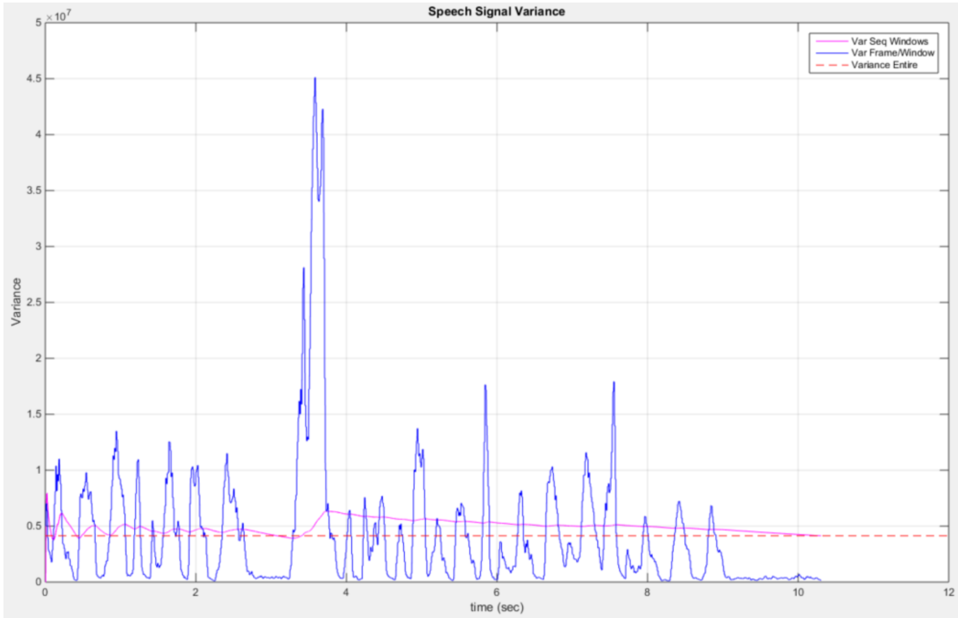

Figure 3: Speech Signal Variance using the three methods described

A speech signal provides us with a different understanding of what variance signifies. The problem is curcial if you are trying normalize the volume of the speech signal since volume is the amplitude of the signal. In an ideal world you have you TV program before you broadcast it which allows you to find the variance in the signal before you broadcast it. Finding the variance of the entire signal is represented by the red dotted line. However, if you are streaming live television you may be taking reevaluating variance at certain time intervals.

The magenta line represents a simulation of finding variance at every sequential sample as if it is being stream in. For the most part our sequential variance stays around true variance. Since our speech signal is composed of a sinusoidal wave we expect the expected value (mean) to be consistently zero. Thus our variance follows the absolute amplitudes of the signal.

The frame and window analysis further exemplifies the portions of the signal when a person is speaking. For example, around a time interval of 3.5 our frame and window shows clearly that speaker talks loudly, or a large spike in variance. Our sequential sample approach of finding variance also increases at this time, but eventually settles back down to the true variance. The random occurrence of when a person talks throws off volume normalization methods. It cannot be predicted when large changes in volume will occur for live television.

III. MATLAB CODE

```
%%
clear; clc; close all;

% Let's first open the raw speech data file and store its values in a
% vector fn
%
fp=fopen('rec_01_speech.raw', 'r');
% Test Sine Wave
    %fp=fopen('rec_01_sine.raw', 'r');
fn=fread(fp,inf,'int16');
fclose(fp);

L_speech = length(fn);
% We are given a sample frequency of 8 kHz
%
fs = 8000;
L_speech = length(fn);
timeL = L_speech/fs;

% We can find the legnth of our signal given our sample frequency
%
t= linspace(0, timeL,L_speech);

% Let's open the xls data file and store its values in avector fn
%
google_v00 = xlsread('google_v00.xlsx');
% google_open = google_v00(:,1);
% google_high = google_v00(:,2);
% google_low = google_v00(:,3);
google_close = google_v00(:,4);
L_googleClose = length(google_close);

clear google_v00

% Let us find the min/max val, mean, median, and variance
%
google_min = min(google_close);
google_max = max(google_close);
google_mean = mean(google_close);
google_median = median(google_close);
google_var = var(google_close);

fn_min = min(fn);
fn_max = max(fn);
fn_mean = mean(fn);
fn_median = median(fn);
fn_var = var(fn);

% Print our findings
%

out = sprintf('Google data: min = %f, max = %f, mean = %f, median = %f, variance = %f
\n'...
    , google_min, google_max, google_mean, google_median, google_var);
disp(out);
```

1 To begin we load in our singal (the same method used in previous assignments). Once we have a data organized into variables inside MATLAB we find the mean and variance of the Google Stock price and speech signal using MATLAB's built in functions.

```
%% Google Stock:

index = 1;
for N = 1:L_googleClose
    windowVar(index) = var(google_close(1:N));
    index = index + 1;
end
```

2 To find the variance simulated as if it was being streamed in we create a for loop that will loop through sequential samples (1, 2, 3). Each time we increment our sample space we find the variance using MATLAB's built in variance function. We stored our variance's for each sample space in a variable $windowVar$.

```
% Now we separate into frames and windows.
%
M = [1];
N = [30];

num_frames = 1+round(L_googleClose / M);
windowMean = zeros(length(N),L_googleClose);
for i=1:num_frames
    sig_wbuf = zeros(1, N);
    Mp = M*(i-1)+M/2;
    windowStart = Mp-N/2;
    windowEnd = windowStart+N-1;

    windowStart = round(windowStart);
    windowEnd = round(windowEnd);

    % transfer the data to this buffer:
    %  note that this is really expensive computationally
    %
    for j = 1:N
        index = windowStart + (j - 1);
        if ((index > 0) && (index <= L_googleClose))
            sig_wbuf(j) = google_close(index);
        end
    end

    windowVar_t = var(sig_wbuf);

   for j = 1:M
        index3 = Mp + (j - 1) - (M/2);
        if ((index3 > 0) && (index3 <= L_googleClose))
            windowVar_FW(index3) = windowVar_t;
        end
   end
end
```

3 Using the same process explained in previous assignments we break our Google stock into separate windows and frames.

```
figure('name','[ECE 3522] Class Assignment [2]');
dDays = 1:1:L_googleClose;
plot(dDays, windowVar, 'm');
hold on
plot(dDays, windowVar_FW, 'b');
plot(xlim, [1 1]*google_var, '--r');
grid on
xlabel('day Index');
ylabel('Amplitude');
title('Google Data Variance');
ylim([0 2E4]);
legend('Var Seq Windows', 'Var Frame/Window', 'Variance Entire');
```

4 Now plot our findings using the three different methods proposed. Using the hold on parameter we can plot each method on the same plot. We limit our y axis so that our zero stuffing doesn't skew our view of the data.

```
%% Speech Signal

% Clear common variables
%
clear windowVar_FW windowVar

% Print our findings
%
out = sprintf('Speech data: min = %f, max = %f, mean = %f, median = %f, variance = %f
\n'...
    , fn_min, fn_max, fn_mean, fn_median, fn_var);
disp(out);

index = 1;
for N = 1:L_speech
    windowVar(index) = var(fn(1:N));
    index = index + 1;
end

% Now we seperate into frames and windows.
%
M = [80];
N = [240];

num_frames = 1+round(L_speech / M);
windowMean = zeros(length(N),L_speech);
for i=1:num_frames
    sig_wbuf = zeros(1, N);
    Mp = M*(i-1)+M/2;
    windowStart = Mp-N/2;
    windowEnd = windowStart+N-1;

    windowStart = round(windowStart);
    windowEnd = round(windowEnd);

    % transfer the data to this buffer:
    %  note that this is really expensive computationally
    %
    for j = 1:N
        index = windowStart + (j - 1);
        if ((index > 0) && (index <= L_speech))
            sig_wbuf(j) = fn(index);
        end
    end

    windowVar_t = var(sig_wbuf);

   for j = 1:M
        index3 = Mp + (j - 1) - (M/2);
        if ((index3 > 0) && (index3 <= L_speech))
            windowVar_FW(index3) = windowVar_t;
        end
   end
end
%%
% Let us plot
%

figure('name','[ECE 3522] Class Assignment [2]');
plot(t, windowVar, 'm');
hold on
plot(t, windowVar_FW, 'b');
plot(xlim, [1 1]*fn_var, '--r');
grid on
xlabel('day Index');
ylabel('Amplitude');
title('Speech Signal Variance');
%ylim([0 ]);
legend('Var Seq Windows', 'Var Frame/Window', 'Variance Entire');

disp('Done Processing');
```

5 We repeat the same process used for the Google stock price for the speech signal.

IV. CONCLUSIONS

Variance corresponds to changes in the signal. The three approaches showcased how variance can change given your method of finding it. Taking the variance of an entire signal is not always possible. The random nature of signals makes it impossible to predict future values of a signal.

The variance using frame and window approach focuses on the variance between a smaller sample set of the entire signal. It shows us the short term changes in the signal. We can zoom in to a portion of our audio signal to better demonstrate the wildly changing nature of the frame and window approach.

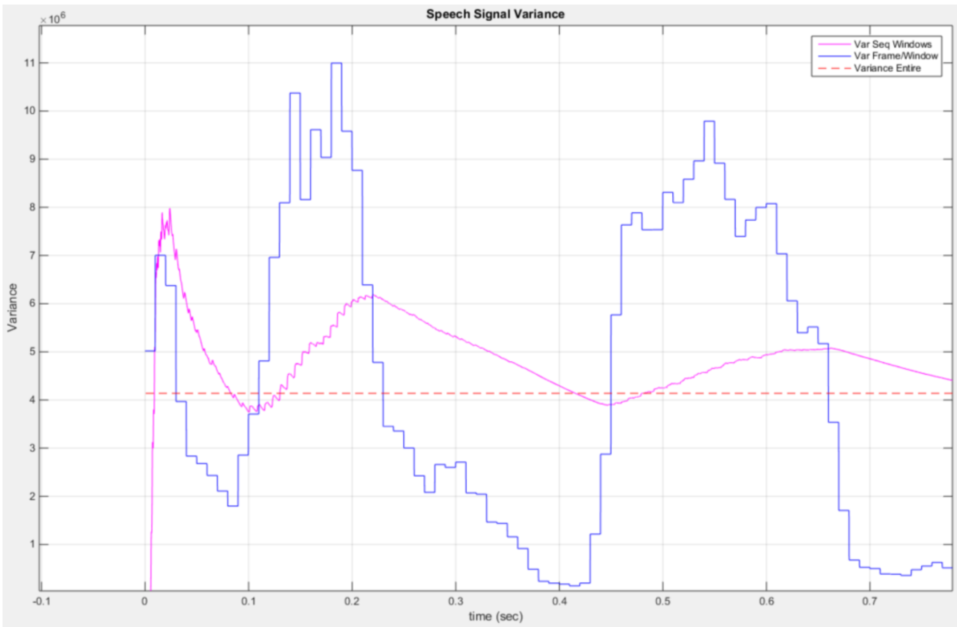

Figure 4: Speech signal variance between 0<t<0.8secs.