Claire Durand

ECE 3522: Stochastic Processes in Signals and Systems

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

# Problem Statement

This assignment explored the concept of variance and the various ways to calculate it using the same Google stock data and speech signal used in the previous assignments. Variance “measures the degree to which a random variable is spread out.” It is defined as the difference between the expected value of the square of x and the square of the expected value of x. It is also related to the standard deviation as variance is the square of the standard deviation. In this assignment, the variance of each entire signal was first found. Then, the “progressive” variance was calculated by starting with the first 10 data points of each signal, and re-calculated for each subsequent additional data point included. Finally, the variance for each signal was calculated using window-frame analysis with a frame of 1 day and a window of 30 days for the Google stock data, and a frame of 10 msec and a window of 30 msec for the speech signal.

# Approach and Results

In order to find the first variance, that of the entire signal, the MATLAB function $var()$ was used. In order to perform a sanity check on the output of this function, the square root of the result was calculated and plotted against the original signal. The square root of the variance is the standard deviation, which should place itself within the min and max values of the original signal. This is the signal shown in Figure 2.

In order to estimate a “progressive” variance, a for-loop was implemented in order to place the variances of an array of growing length into the corresponding indeces of an output array. The first ten values of the output array were all given the variance of the first ten values of the original signal in order to establish the variance of a minimum number of values in order not to start with 0.

For the last method, the window-frame analysis code from the previous assignments was modified so that it returned the variance instead of the mean. In order to obtain the frame duration of 10 msec for the speech signal, the following calculations were performed. Since the signal was sampled at 8 kHz, the inverse was simply the amount of time in seconds in between each data point. The frame was to last 10 msec, therefore, the number of points that fit in that interval was found by dividing the frame duration by the time between samples. The resulting 80 points could then be implemented in the MATLAB code that used number of data points for its frame and window calculations. Once the frame was obtained, it was easy to find the window duration in number of points since 30 msec is simply 3 times the frame duration, 10 msec. Therefore, the 80 points per frame simply needed to be tripled in order to obtain the 240 points per window.

$$\frac{1}{8 kHz}=0.000125 sec=0.125 msec$$

$$\frac{10 msec}{0.125 msec}=80 pts$$

For the sake of speed and detail, the speech signal was shortened to its first 20,000 points. The speech signal was long enough that calculating a new variance for each point -as done in the second method- was computationally too heavy. Moreover, shortening the data allowed a closer look at the data.



*Figure 1 – Variance on Google stock data*



*Figure 2 – Google stock data Standard Deviation*



*Figure 3 – Variance for Speech signal data*

In both Figures 1 and 3, the dotted blue line represents the variance of the total signal. This value is large enough that the original signal looks scaled down. The “progressive” variance line for the Google stock data slowly catches up to the total variance in an over-damped fashion. The same line for the speech signal catches up to its total variance, but in an under-damped fashion. Either way, the “progressive” variance line does ultimately catch up to the total variance which makes sense since toward the end, both lines start to have the same amount of the same data. On the other hand, the last variance obtained through window-frame analysis, in black in Figures 1 and 3, is much more variable since at no point does it ever encompass the entire set of data, only the data immediately around the frame, making the output line more subject to sudden oscillations.

# MATLAB Code

## Computer Assignment No. 3 : Variance (Google Stocks)

clear; clc; clf;
close all

## Part 1 - Estimate the data for the entire data set

fig = 1;

% Load the data
stocks = 'google\_v00.csv'; % Create a variable for the file name
G = csvread(stocks, 1,1); % Read in the data, excluding the text parts.

% From the matrix loaded with the data, extract the signal data to be used
Stocks = G(:,4); % Closing price
Stocks = Stocks';

% Obtain data dimensions
[m,n] = size(G); % Obtain the size of the matrix (how many days of data)
d = 1:m; % Create a time vector d for each day of stock data

% Find the variance in closing price data
total\_var = zeros(1,length(d));

for i = 1:length(Stocks)
 total\_var(i) = var(Stocks, d); % enormous because squared
end

%Variance = Variance';

## Part 2 - "Progressive" estimate of variance

var\_est = zeros(size(Stocks));

for l = 1:length(var\_est)

 if (l <= 10)
 array = Stocks(1:10);
 else
 array = Stocks(1:l);
 end

 var\_est(l) = var(array);
end

## Part 3 - Windows and frame analysis

M = 1; % days
N = 30; % days

% Create a matrix to store the output variance
G\_output\_var = zeros(1,length(Stocks));

for n = 1:length(N)
 for m = 1:length(M)

 % Find the center of the first frame
 Mc = (M(m)+1)/2;

 while (Mc<length(Stocks))
 % Establish the bounds of the window subarray we're looking at
 %
 % Case where M and N are both either even or odd
 % then case where M and N are not both even or odd
 % Note that this will tilt the mean calculation toward the future
 % however we will remain consistent in order to avoid disparities.
 if (mod(M(m),2) == mod(N(n),2))
 Nmin = Mc - ((N(n)-1)/2);
 Nmax = Mc + ((N(n)-1)/2);
 else
 Nmin = Mc - (N(n)/2) - 1;
 Nmax = Mc + (N(n)/2);
 end

 % Handle the beginning and end
 if Nmin < 1
 Nmin = 1;
 end

 if Nmax > length(Stocks)
 Nmax = length(Stocks);
 end

 Window = Stocks(Nmin:Nmax);

 % Establish the bounds of the output subarray in which we're placing results
 Mmin = Mc - ((M(m)-1)/2);
 Mmax = Mc + ((M(m)-1)/2);

 % Don't let your output matrix run larger than our input matrix
 if (Mmax > length(G\_output\_var))
 Mmax = length(G\_output\_var);
 end

 % Place the mean value of the window subarray in every value of the output
 % frame subarray
 for Mi = Mmin:Mmax
 G\_output\_var(n,Mi) = var(Window);
 end

 Mc = Mc + M(m); % Find the new frame center
 end
 end
end

% Plot the data as a function of the days
figure(fig)
plot(d, Stocks),
hold on,
plot(d, total\_var, '-.'), % variance of the entire signal
plot(d, var\_est, 'r'),
plot(d, G\_output\_var, 'k'),
hold off
legend('Closing stock price', 'Variance', 'Updated Variance', 'Windows & frame')
xlabel('Days since inception'), ylabel('Stock price'),
title('Google stock price since inception')

figure(fig+1)
plot(d, Stocks, d, sqrt(total\_var))
xlabel('Days since inception'), ylabel('Stock price'),
title('Google stock price since inception')
legend('Closing Stock Price', 'Standard deviation')

## Computer Assignment No. 3 : Variance (Speech Signal)

clear; clc; clf;
close all

## Part 1 - Estimate the data for the entire data set

fig = 1;

% Open the file
fp = fopen('rec\_01\_speech.raw','r');

% Load the data.
% It is 16-bit sampled data, so we must load it as short integers.
sig = fread(fp,inf,'int16');

% Close the file
fclose(fp);

% Shorten the length of the signal for efficiency's sake
sig = sig(1:20000);

% Obtain data dimensions
[p,q] = size(sig); % Obtain the size of the matrix (how many points of data)
pts = 1:p; % Create a time vector d for each day of stock data

% Find the variance in closing price data
total\_var = zeros(1,length(pts));

for i = 1:length(sig)
 total\_var(i) = var(sig, pts); % enormous because squared
end

%Variance = Variance';

## Part 2 - "Progressive" estimate of variance

var\_est = zeros(size(sig));

for l = 1:length(var\_est)

 if (l <= 10)
 array = sig(1:10);
 else
 array = sig(1:l);
 end

 var\_est(l) = var(array);
end

## Part 3 - Windows and frame analysis

M = 80; % 10 msec
N = 240; % 30 msec

% Create a matrix to store the output variance
G\_output\_var = zeros(1,length(sig));

for n = 1:length(N)
 for m = 1:length(M)

 % Find the center of the first frame
 Mc = (M(m)+1)/2;

 while (Mc<length(sig))
 % Establish the bounds of the window subarray we're looking at
 %
 % Case where M and N are both either even or odd
 % then case where M and N are not both even or odd
 % Note that this will tilt the mean calculation toward the future
 % however we will remain consistent in order to avoid disparities.
 if (mod(M(m),2) == mod(N(n),2))
 Nmin = Mc - ((N(n)-1)/2);
 Nmax = Mc + ((N(n)-1)/2);
 else
 Nmin = Mc - (N(n)/2) - 1;
 Nmax = Mc + (N(n)/2);
 end

 % Handle the beginning and end
 if Nmin < 1
 Nmin = 1;
 end

 if Nmax > length(sig)
 Nmax = length(sig);
 end

 Window = sig(Nmin:Nmax);

 % Establish the bounds of the output subarray in which we're placing results
 Mmin = Mc - ((M(m)-1)/2);
 Mmax = Mc + ((M(m)-1)/2);

 % Don't let your output matrix run larger than our input matrix
 if (Mmax > length(G\_output\_var))
 Mmax = length(G\_output\_var);
 end

 % Place the mean value of the window subarray in every value of the output
 % frame subarray
 for Mi = Mmin:Mmax
 G\_output\_var(n,Mi) = var(Window);
 end

 Mc = Mc + M(m); % Find the new frame center
 end
 end
end

% Plot the data as a function of the days
figure(fig)
plot(pts, sig),
hold on,
plot(pts, total\_var, '-.'), % variance of the entire signal
plot(pts, var\_est, 'r'),
plot(pts, G\_output\_var, 'k'),
hold off
legend('Closing stock price', 'Variance', 'Updated Variance', 'Windows & frame')
xlabel('time'), ylabel('Amplitude'),
title('Speech Signal Variance Analysis')

[*Published with MATLAB® R2014a*](http://www.mathworks.com/products/matlab)

# Conclusions

By virtue of coming from the square of the signal, the variance is enormous. It gets larger as it encompasses more data points since there is more likelihood of differences between the many values. Therefore, window-frame analysis will yield a signal on the same scale as the original signal since it only includes a very limited number of values, presumably within a relatively close range of each other.