Anton LeKang

ECE 3522: Stochastic Processes in Signals and Systems

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

CA3

# Problem Statement

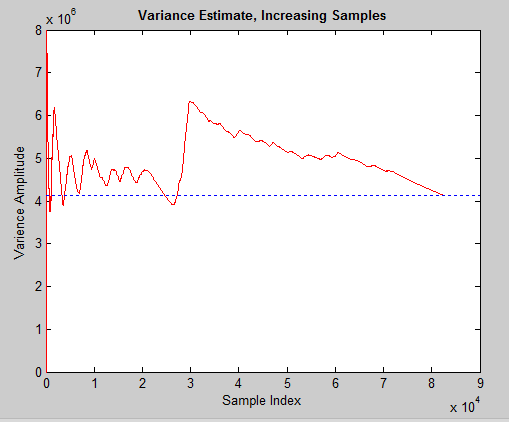
The objective of this lab is to look at the variance or second central moment of a data set, and observe the characteristics of it. The variance is how far a set on numbers is away from the mean, if all of the numbers are the same, then the variance would be zero. This computer assignment will use Google’s stock prices, and an audio file as the main inputted source of the data. This lab will look at the different ways of calculating the variance from its final value, to how this value was processed in real time. MATLAB will be used to compute each of the variances in this computer assignment.

# Approach and Results

Part 1 and 2: Audio File Variance

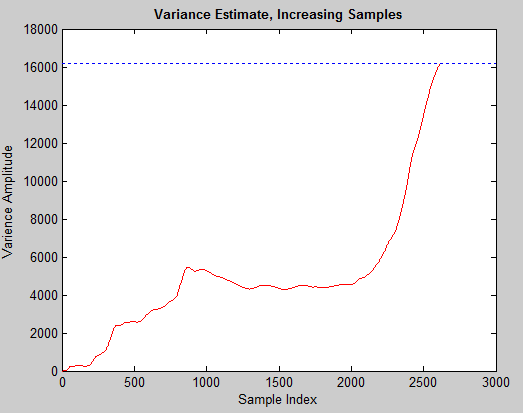
The first part of the lab plots the variance of the data with a dashed horizontal line. Figure 1 represents the audio signal, and the figure 2 represents the google signal.

The second part of the lab had a new part, and was also graphed based on part one, so both of these parts was done together. In this part of the lab the variance is estimated by taking the data points from each individual sample. The first variance was of the first ten points, and then the variance was estimated using each sum from there on out. So the next variance calculated was using the first eleven points, and then twelve after that. As seen in figure 1, the audio file oscillates a lot over the main variance value. This is because this file has negative and positive values, and the values are not consistent with time. Each value could be extremely off from the old value so the variance could be really high, or low depending on the common values. The variance is based on how far that data is from the mean, as time increases with the audio file, the real time variance eventually equals the overall variance.

Figure 1 – Audio File

Part 1 and 2: Google Stock Variance

Figure 2 shoe the real time variance in red, and the overall variance with the blue dotted line. Unlike the audio file, the real time variance of the Google stock does not oscillate. This is because the stock market has a set mean value, and the market has increased since it first went open. The variance at each point is looking how far that data value is, from the actual mean value. The bottom value is 0, since the mean and data values are relative close to each other. Then as time goes on, the mean increases and more data points are added to the scenario. As real time goes on, it eventually catches up to the main variance, which it should do.

Figure 2 – Google Stock Variance

Part 3: Estimating the Variance for Audio File

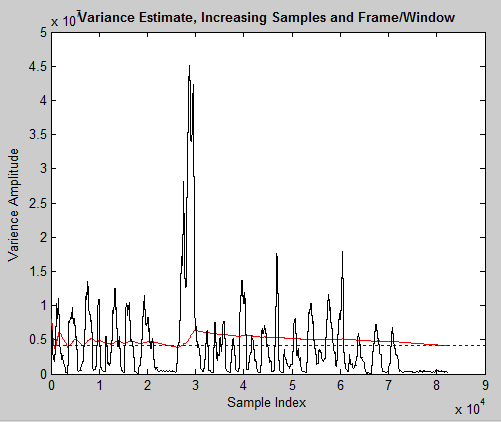
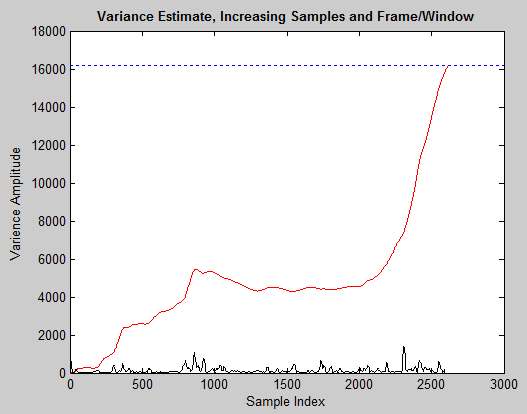
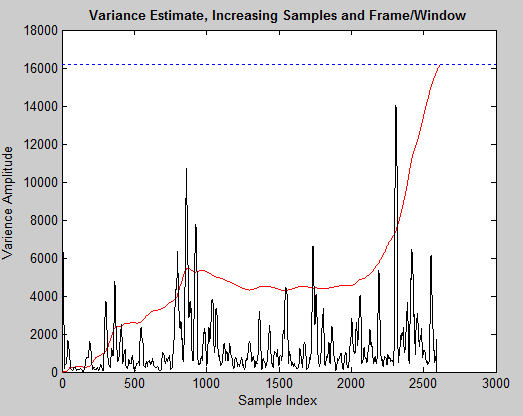
Figure 3 – Estimated Audio File

Figure 3 above shows the audio file variance estimated with a frame duration of 10 msec, and a window of 30msec. This is shown by the black line in the graph above. Relative to the variance calculated in part 2, these value are really far off. A better estimate for the variance would be the variance calculated real time.

Part 3: Estimating the Variance for Google Stock Price

Figure 5 – Estimated Google Prices

Figure 4 – Estimated Google Prices

These figure correlate closer to the variance levels than the audio files did. As the sample index increased the variance amplitude mostly increases in figure 5. Whenever there is a spike in the estimated variance, it correlates nicely with the real time variance. Figure 5 also shows this, but in a smaller scale. The total sum variance of both of the plots eventual add up to the total variance.

# MATLAB Code

Part 1 and 2:

function ca\_3\_1\_gs

clear;clc;

[num, txt, raw] = xlsread('google\_v00.xlsx', 1);

close = num(:, 4);

[z,q] = size(close);

vr = nanvar(close);

%y = linspace(0, z, z);

%figure(1)

%call varience plot for speech file

Q = ca\_3\_final

real\_time(close, z);

hold on

hline = refline(0, vr);

set(hline,'LineStyle',':');

hold on

plot(Q, 'k');

title('Variance Estimate, Increasing Samples and Frame/Window', 'fontweight', 'bold');

xlabel('Sample Index');

ylabel('Varience Amplitude');

end

function real\_time(close, z)

%allocate size of array to store varience values

%N = zeros(z);

%w = z-10;

y = 1:1:z;

y = y';

%seg = close(1:4)

%starting at 10, incrememnt

for i = 10:z

seg = close(1:i);

%compute and store

var = nanvar(seg);

N(i) = var;

end

figure(1)

plot(y, N, 'r');

title('Variance Estimate, Increasing Samples', 'fontweight', 'bold');

xlabel('Sample Index');

ylabel('Varience Amplitude');

end

function ca\_3\_1\_sp

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

speech = fread(fp,inf,'int16');

[z, q] = size(speech);

vr = nanvar(speech);

y = linspace(0, z, z);

figure(1)

%call varience plot for speech file

Q = ca\_3\_final\_ish;

real\_time(speech, z);

hold on

hline = refline(0, vr);

set(hline,'LineStyle',':')

hold on

plot(Q, 'k');

title('Variance Estimate, Increasing Samples and Frame/Window', 'fontweight', 'bold');

xlabel('Sample Index');

ylabel('Varience Amplitude');

end

function real\_time(speech, x)

%allocate size of array to store varience values

y = 1:1:x;

y = y';

%starting at 10, incrememnt

for i = 10:x

seg = speech(1:i);

%compute and store

var = nanvar(seg);

n(i) = var;

end

plot(y, n, 'r');

%

end

This code gets the main variance of the signal and plots it using the refline command. This also is the code for the second part of the lab which sums of the estimates. The program adds the first 10 samples together, and then starts from that point to get the rest of the variances. The hold on command is used to plot the sum of the variance and the reference line of the data together.

Part 3:

function X = ca\_3\_final\_ish

clear; clc;

% close open sessions

%

close all;

% 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 = [ 80];

N = [ 240];

%[sig, txt, raw] = xlsread('google\_v00.xlsx', 1);

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

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

% create a matrix to store the output

X = compute\_rms\_1(sig, M, N);

plot(X, 'k');

end

%{

% function: compute\_rms

%

% arguments:

% sig\_a: the input signal (input)

% fdur\_a: the frame duration in samples (input)

% wdur\_a: the window duration in samples (input)

%

% return:

% rms: a vector of rms values (output)

%

% Note that this function returns the rms counter as a sampled data

% signal that is the same length as the input signal. This is wasteful

% of memory, but makes it easy to produce a time-aligned plot.

%

% This algorithm computes the sum of squares for wdur\_a samples.

%}

function X = compute\_rms\_1(sig\_a, fdur\_a, wdur\_a)

% declare local variables

%

k = 1;

sig\_wbuf = zeros(1, wdur\_a);

num\_samples = length(sig\_a);

num\_frames = 1+round(num\_samples / fdur\_a);

rms\_full = zeros(length(sig\_a),1);

% loop over the entire signal

%

for i = 1:num\_frames

%for indexing

n\_center = (i - 1) \* fdur\_a + (fdur\_a / 2);

n\_left = n\_center - (wdur\_a / 2);

n\_right = n\_left + wdur\_a - 1;

%make sure we're not using points outside the signal

if( (n\_left < 0) || (n\_right > num\_samples) )

sig\_wbuf = zeros(1, wdur\_a);

end

% transfer the data to this buffer:

% note that this is really expensive computationally

for j = 1:wdur\_a

index = n\_left + (j - 1);

if ((index > 0) && (index <= num\_samples))

sig\_wbuf(j) = sig\_a(index);

end

end

% square the signal. divide it by the number of samples used and sum

% the result to build the value for that frame

%

rms = sqrt( (1 / wdur\_a) \* sum(sig\_wbuf.^2));

for j = 1:fdur\_a

index = n\_center + (j - 1) - (fdur\_a/2);

if ((index > 0) && (index <= num\_samples))

rms\_full(index) = rms;

end

end

X(n\_center) = nanvar(sig\_wbuf);

end

for i = 2:length(X)

if X(i) == 0

X(i) = X(i-1);

end

end

end

This program both takes the audio file or the google stock and estimates the variance through a frame and window. The frame and window are design in the top of the program from the N, and M values. The plot of this based on the X value which is the output of the compute\_rms function. This function finds the rms value of the inputted data with respect to the frame and window.

# Conclusions

Variance can be a very useful tool when looking at and observing trends in data. This computer assignment looks at the different ways in which the variance can be found. A simple calculation can be performed that gives the overall variance of data provided. This variance is based on how closely the values are related to the mean. Another way to look at the variance is by calculating the real time value. This can be very useful if the data is positive like the Google stock. If the values are positive and negative the values will jump all over as seen with the audio file. When calculating the real time amplitude, the data always eventually summed to the overall variance. The last way to calculate variance in this assignment was by changing the frame and window duration of a specific signal. Again this was beneficial with the Google stocks since it showed a correlation when there was an increase in variance amplitude, there was also an increase in the real time amplitude.