Stephen Labowski

ECE 3512: Stochastic Processes

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

# Problem Statement

There are two objectives in this assignment. The first is to plot the mean and linear regression of the google stock prices, comparing their plots. The second is to produce histograms of the amplitude of the speech signal. The first representing the normalized data for the PMF and the second being a CDF of the data.

# Approach and Results

For the first part a window of 7 and a frame of 1 was used. The mean was calculated and plotted just as in the first assignment. The linear regression plot was accomplished using the polyfit function of MATLAB. This returns the coefficients of the polynomial to the degree specified in the function input. In this case one to get a linear fit. Linear regression is an approach to model the relationship between a dependent variable and one or more independent variables. It focuses on the conditional probability of y given x.

Figure 1 and 2 below show the plots of the signal, its mean, and the linear regression line. The mean is very near the same fit as the actual signal, just slightly smoothed out. The linear regression in comparison to the mean plot is shows the overall signal path. Meaning that since the line has a positive slope the signal is on an upward trend. For instance if you were looking at buying this stock if the current price is below the regression line it can be a good time to buy, as the stock should be heading back towards the line in the future. Conversely if it were above the line it might be a good time to sell as it is possibly ready to fall back towards the line.



Figure 1: Plot of all stock price data



Figure 2: Start of stock price data

The second objective of the assignment was to create PMF and CDF histograms. Dividing the number of occurrences in that particular bin by the total samples normalizes the PMF histogram (figures 3 and 4). These values will add to one as seen in the CDF histogram (figure5). Due to the large amount of samples and the large range it is expected that the PMF values be relatively low. This is the result as the highest occurring outcome is about .0045. The CDF is also as expected due to the relative shape of the PMF the CDF is extremely steep around the center of the range.



Figure 3: PMF of full range



Figure 4: Zoomed in PMF



Figure 5: CDF histogram

# Conclusions

The results of part one show that the overall trend for google is that its price will continue to rise in the longterm. Linear regression gives us another tool for predicting future outcomes of a signal or system. The second objective of the assignment is a great way to visually interpret the probability of the data. This specific signal shows that its values are concentrated at low amplitudes. This would be expected from speech due to the periods of silence or low noise in-between the words and other sounds.

# MATLAB Code

This first section is the script used for the assignment. It is immediately followed by the function used in the script.

**Appendix A:**

clear;

close all;

M = 1; % Frame

N = 7; % Window

filename = 'google\_v00.xlsx';

%%% Reads in closing prices of google stock

sig = xlsread(filename,'E:E');

%%% Sets up an index array.

i = (1:1:length(sig));

i = i';

%%% Obtains the coefficients for the linear fit regression line.

p = polyfit(i,sig,1);

%%% Creates the linear function to plot.

Y = p(1)\*i + p(2);

sig\_mean = zeros(length(sig));

%%% sets up overlayed plot of the signal its mean and a linear fit

%%% regression line.

h1 = figure('name', 'mean plot', 'numbertitle', 'off');

 sig\_mean = compute\_mean(sig, M, N);

 str = sprintf('frame duration: %d window duration = %d', M, N);

 plot(sig\_mean,'b-');

 hold on

 plot(sig,'r-');

 plot(Y,'g-');

 title(str);

 xlabel('index');

 ylabel('closing price');

 legend('mean', 'sig', 'regression');

 hold off

%%% Use this section for audio file

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

sig2 = fread(fp, inf, 'int16');

fclose(fp);

%%% sets up bins

hist\_bin\_ctrs = -32765:10:32765;

hist\_bin\_ctrs = hist\_bin\_ctrs';

%%% Creates PMF histogram

h2 = figure('name', 'PMF', 'numbertitle', 'off');

 hold on;

 title('PMF');

 xlabel('amplitude');

 histogram(sig2, hist\_bin\_ctrs, 'normalization','probability');

 hold off;

%%% creates CDF histogram

h3 = figure('name', 'CDF', 'numbertitle', 'off');

 hold on;

 title('CDF');

 xlabel('amplitude');

 histogram(sig2, hist\_bin\_ctrs, 'normalization','cdf');

 hold off;

**Appendix B:**

% function: compute\_mean

%

% arguments:

% sig: the input signal (input)

% fdur: the frame duration in samples (input)

% wdur: the window duration in samples (input)

%

% return:

% mean: a vector of mean values (output)

%

function sig\_mean = compute\_mean(sig\_a, fdur\_a, wdur\_a)

% declare local variables

%

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

num\_samples = length(sig\_a);

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

sig\_mean = zeros(length(sig\_a), 1);

% loop over the entire signal

%

for i = 1 : num\_frames

 % generate pointers to the beginning and end of the frame/window:

 % These pointers control how we move through the signal.

 % For example, left and right refer to the beginning and end of

 % a frame/window.

 %

 n\_frame\_left = (i - 1) \* fdur\_a + 1;

 n\_frame\_right = n\_frame\_left + fdur\_a - 1;

 n\_frame\_center = (n\_frame\_right + n\_frame\_left) / 2;

 n\_win\_left = floor(n\_frame\_center - (wdur\_a / 2) + 1);

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

 % zero-stuff the data if necessary:

 % when the pointers exceed the index of the input data we won't be

 % adding enough samples to fill the full window. to solve this zero

 % stuffing will occur to ensure the buffer is always full of the same

 % number of samples

 %

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

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

 end

 % transfer the data to this buffer:

 % note that this is really expensive computationally

 %

 j = 1;

 index = n\_win\_left;

 while (j <= wdur\_a)

 % transfer the signal to the processing buffer:

 % note that we must make sure we do not exceed the

 % limits of the signal for the first and/or last frame

 %

 if ((index >= 1) && (index <= num\_samples))

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

 end

 % increment the counters

 %

 index = index + 1;

 j = j + 1;

 end

 % compute the mean:

 %

 s\_mean = mean(sig\_wbuf);

 for j = 1 : fdur\_a

 index = n\_frame\_left + j - 1;

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

 sig\_mean(index) = s\_mean;

 end

 end

end

end