Vaughn Wiernicki

ECE 3522: Stochastic Processes in Signals and Systems

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

# Problem Statement

This computer assignment revolved around the idea of random numbers. Using a vector of MATLAB generated random numbers we then had to compute the global variance and mean as well as the variance and mean of the signal as a function of the number of points. We also used a histogram and an approximate probability density function to compute the mean-squared error as a function of the number of bins in the histogram.

# Approach and Results

Using MATLAB we could generate an array of random numbers to use as a set of data. The nature of this data set means that no two trials of the code will behave similarly, but regardless, we should see similar behavior overall. For a uniform distribution we figure the mean will be 0.5 and the variance will be 0.083, however the error will be the difference between the overall mean and the mean as a function of the number of data points. We use the same kind of error computation for the variance equation. With a big enough data set the mean and variance of the function as well as the function of the mean and variances will converge and the error would become zero. After plotting the error functions we can see that this is what our functions look like. The following plot is zoomed in on the first 20 data points to see the small oscillations before the convergence:



Figure : Error of Mean and Variance as a function of points

To further illustrate this point I created another figure to zoom in further and display the first few points of the function:



Figure : Zoomed in plot of Figure 1

We can see that the mean and variance functions both have very small values of error until converging further to zero. This can be attributed to the fact that we’re using random numbers and they may not be in a smaller range until there are more points available. After there are more points available to use in the mean and variance calculations, the mean will die down until it gets to the overall value.

We then had to generate three different arrays of random numbers for varying sizes. We then created a histogram for these values with bin sizes 1/100th of the total value of numbers. We also had to normalize the histogram so that we’d have probability of these values appearing. After this we had histogram amplitude values that were then used to calculate the mean-squared error between the histogram’s amplitude and a predicted probability density function’s amplitude. Because the random numbers were of a uniform distribution, the probability density function was created by dividing by the number of data samples in the original array. The first set of data we used was for 10 random values, giving us bin sizes of 1/10th. Using a uniform distribution we expect to see all values to be the same, or at least similar. Below is a figure for the 10 random numbers:



Figure : Mean Squared Error of 10 random values

We see that the curve is exponential. This makes sense as the values further away from the maximum value have such a small chance of appearing. The next value this process was done for was 1,000 random numbers, giving a bin size of 10. Below is the figure received:



Figure : Mean Squared Error of 1,000 random values

Again we see an exponential curve for similar reasons. We do see slight variations in the general trend between the bins. We repeat the process for 1000000 random numbers:



Figure : Mean Squared Error of 1,000,000 random values

The overall trend to observe is that regardless of the number of values and bins we can see a general exponential shape as it converges to a value. In all case the value is typically very small. For a difference of six factors of ten, we see a decrease in amplitude by ten factors of ten.

# MATLAB Code

clear; clc; clf; close all;

%Create Random Numbers

N = 10e6;

R = rand(1, N);

%Computing the mean:

mu = zeros(1, N);

msum = 0;

for i = 1:1:N

 msum = msum + R(1, i);

 mu(1, i) = (1/i)\*msum;

end

m = mean(R);

%Computing the variance:

sigma = zeros(1, N);

vsum = 0;

for j = 1:N

 vsum = vsum + (R(1, j) - mu(1, j)).^2;

 sigma(1, j) = (1/j)\*vsum;

end

v = var(R);

%Getting the Error Functions:

Em = zeros(1, N);

Es = zeros(1, N);

for k = 1:1:N

 Em(1, k) = mu(1, k) - m;

 Es(1, k) = sigma(1, k) - v;

end

%Plotting the Functions:

t = logspace(1, 6, N);

semilogx(t, Em, 'b');

grid on;

xlabel('Number of Points');

ylabel('Error');

hold on;

semilogx(t, Es, 'r');

legend('Error of Mean', 'Error of Variance');

xlim([0, 100]);

----------------------------------------------------------------

clear; clc; clf; close all;

%Create Random Numbers

N = [10e1, 10e3, 10e6];

for a = 1:1:length(N)

 R = rand(1, N(a));

 %Get a Histogram of the vector & Normalize it:

 nbins = N(a)\*0.01;

 plot1 = hist(R, nbins);

 for i = 1:1:length(plot1)

 plot1(1, i) = plot1(1, i)/(nbins \* 10^2);

 end

 %Get an estimate of the PDF (Uniform Distribution):

 pdf = zeros(1, nbins);

 for j = 1:1:length(pdf)

 pdf(1, j) = 1/length(plot1);

 end

 %Get the Mean-Squared Error:

 mse = zeros(1, nbins);

 sumMSE = 0;

 for k = 1:1:length(mse)

 sumMSE = sumMSE + (pdf(1, k) - plot1(1, k))^2;

 mse(1, k) = sumMSE/k;

 end

 %Plot the Functions

 figure;

 if a == 1

 zer = zeros(1, N);

 plot(zer, 'b');

 xlim([0, 1]);

 end

 if a~= 1

 semilogx(mse, 'b');

 end

 grid on;

 hold on;

 % plot(pdf, 'r');

 legend('Mean Squared Error' );%, 'Estimated PDF');

 xlabel('Number of Bins');

 ylabel('Normalized value of Occurences');

 title(['Plot for ' num2str(N(a)) ' Random Numbers'], 'fontweight', 'bold');

end

# Conclusions

From an array of randomly generated numbers we can predict the mean and variance using basic analytic techniques, but MATLAB allows us to easily calculate these values both as a function of the plot of numbers and an overall value. We know that the values of mean and variance when plotted as functions would converge to the global point, as the global mean/variance is a result of all points being considered. For the histogram we also get a similar look where we see a plateau of amplitude as values start showing up with the same commonality. Changing the bin size would however, have tremendous effect on what our mean squared error looks like.