Emilie Doyle

ECE 3522: Stochastic Process in Signals and Systems

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

# Problem Statement

The objective of this assignment is to learn more about hypothesis testing through the implementation of one of the assigned homework problems through MATLAB coding. The assigned problem was an example from the textbook. In this problem, Example 9.9, we are given that the mean lifetime of the lightbulb is 1600 and that the standard deviation is 120. In the assignment we are tasked with creating multiple functions in order to help us with the assignment. We are given what the input and output of the functions should be. We then are tasked with utilizing these functions in application to the example problem, with further applications than asked for in the problem.

# Approach and Results

I first started out with creating three functions: gen\_grv, check\_significance and determine\_significance. Creating each of these was part of the assignment. For gev\_grv, I listed the input as the mean, standard deviation and the number of random variables to be created. For the output, I listed the array X that contained the random variables, and uX, the mean of that array of random variables. I used the normrnd function in order to create the random variables, and I supplied the input of mean and standard deviation and also supplied the dimensions that the array should be. I then used the mean2 function to determine the mean of the array and output that as well. This was not specifically assigned, but I thought it would be helpful later on when calculating things, so I included it in part of the function.

For the check\_significance function, I supplied the input of the original, given mean, the newly calculated mean (from gen\_grv), the standard deviation, the number of random variables and the confidence level. The confidence level must be supplied in a decimal format, and I included that in the documentation of the function. The output of the function is the status. This means if it is statistically the same of different. It essentially would return a Boolean, with 1 meaning it is true, it falls within the range, and 0 meaning that it does not fall within the range. Since the confidence level is one of the inputs, I used a case statement in order to determine the proper k value for the calculations. The k values were taken from an exam problem and the case statement takes the confidence level and determines the corresponding k value. I then calculated the upper and lower limits of the range of accepted values that would be statistically the same. For this I used the equation from class for interval estimation. I then used an if statement to determine if the newly calculated mean fell within the range of accepted values, and assigned a status depending on if it did or did not.

For the determine\_significance function, I supplied the input of the original mean, the calculated mean, the standard deviation and the confidence level. I used an almost identical (only a different internal variable name) case statement to determine the appropriate value of k. I then manipulated the equation used in check\_significance to solve for the necessary value of N. I set limits based on this equation. Next, I looped through a wide range of possible N values to see what fell within the range. At the first value that fit the criterion I broke from the loop and set that value to be N.

In my main code, I set arrays for the given values of n , k and standard deviations. I also set the mean to be 1600. I broke it up into three sections of nested loops: one with standard deviation of 120, one with standard deviation of 240 and one that solved for the maximum variance value. The code for the standard deviations of 120 and 240 are nearly identical, except for variable names and obviously the standard deviation value. In these nested loops, the outer loop cycles through values of n and the inner loop cycles through values of k. In the outer loop I called gen\_grv in order to get the necessary values. Next came the inner loop. I started off by using the check\_significance function and storing the resulting values in a matrix, and then I called the determine\_significance function and stored those values in a different matrix.

Please note that each row corresponds to the different n values and each column corresponds to the different k values.



Figure : Status with StDev= 120



Figure : N Value Significance with StDev=120



Figure : Status with StDev= 240



Figure : N Value Significance with StDev= 240

For determining the maximum variance value I had three nested loops. The outermost loop cycled through values of n and was used to generate the Gaussian random variable data for the inner loops. The middle loop cycles through values of k and literally only contains the innermost loop. The innermost loop sweeps through a wide range of values for the standard deviation and checks the status for each of these values, until it comes across the first value where the status is equal to one (true), which it then proceeds to store in a matrix of standard deviations, and then squares the value and stores in a matrix of variances. Then it breaks from the loop and continues to the next cycle.



Figure : Standard Deviation Break Values



Figure : Variance Break Values

# MATLAB Code

**Functions:**

function [ X, uX ] = gen\_grv( mean, stdev, n )

%UNTITLED Summary of this function goes here

% mean, stdev, n= number of RV, uX= mean of X

X= normrnd(mean, stdev, [1 n]);

uX= mean2(X);

end

function [ status ] = check\_significance( mean1, mean2, stdev, n, confidence\_level )

%UNTITLED2 Summary of this function goes here

% mean1= given mean, mean2= calculated mean, stdev, n= number RV,

% confidence level as decimal

%returned status is as boolean

switch confidence\_level

 case 0.8

 k=1.28;

 case 0.9

 k=1.64;

 case 0.95

 k=1.96;

 case 0.99

 k=2.58;

 case 0.999

 k=3.29;

end

lower\_limit= mean1- k\*(stdev/ sqrt(n));

upper\_limit= mean1+ k\*(stdev/ sqrt(n));

if ((mean2>=lower\_limit)&&(mean2<=upper\_limit))

 status=1; %1= True

else

 status=0; %0=false

end

end

function [ N ] = determine\_significance( mean1, mean2, stdev, confidence )

%UNTITLED4 Summary of this function goes here

% mean1= original, mean2= calculated, stdev, confidence as decimal

switch confidence

 case 0.8

 k=1.28;

 case 0.9

 k=1.64;

 case 0.95

 k=1.96;

 case 0.99

 k=2.58;

 case 0.999

 k=3.29;

end

limit1= ( (k\*stdev)/(mean1-mean2))^2;

limit2= ((k\*stdev)/(mean2-mean1))^2;

for n= 1:1:10e6

 if ((n>= limit1)&&( n>=limit2))

 N=n;

 break

 end

end

%N= ((k\*stdev)/(1-confidence))^2;

end

**Code:**

%Stoch CA 10

n= [10 100 1000];

k= [0.8 0.9 0.95 0.99 0.999];

mean= 1600;

stdev= [120 240];

%StDev120

for i=1:length(n)

 [X uX]=gen\_grv(mean, stdev(1), n(i));

 for j=1:length(k)

 status1(i,j)= check\_significance(mean, uX, stdev(1), n(i), k(j));

 N1(i,j)= determine\_significance(mean, uX, stdev(1), k(j));

 end

end

%StDev240

for i=1:length(n)

 [X uX]=gen\_grv(mean, stdev(2), n(i));

 for j=1:length(k)

 status2(i,j)= check\_significance(mean, uX, stdev(2), n(i), k(j));

 N2(i,j)= determine\_significance(mean, uX, stdev(2), k(j));

 end

end

%Max Var Value

for i=1:length(n)

 [X uX]=gen\_grv(mean, stdev(1), n(i));

 for j=1:length(k)

 for SD= 1:1:10e5

 status= check\_significance(mean, uX, SD, n(i), k(j));

 if (status==1)

 StD(i,j)= SD;

 Variance(i,j)= SD^2;

 break

 end

 end

 end

end

# Conclusions

In conclusion, I really enjoyed this computer assignment. I thought it was very fun and interesting and really helped solidify the learning that we did in class involving confidence intervals. In hindsight, calling gen\_grv multiple times may skew the results a bit, but I don’t think it is enough to be truly worried about, since they are Gaussian variables, and some even have different supplied standard deviations, so they would already be different. This assignment provided great practice for writing functions, which I know is something that I really should be doing much more. I hope that it is alright that I added in another output to the gen\_grv function.

As far as the actual results go, it makes sense that the N value would have to increase over the increase of the confidence levels, as well as increase over the sample size increase. As the confidence level increases, then the value for k will also increase. As the confidence level increases, so too does the range of accepted values. It is very interesting to see the differences between the two standard deviation values and the corresponding N values. Again, it is important to note that these have different generated Gaussian random variables will obviously be different. For the third portion of the code it is interesting to see how the standard deviation needed decreases as the k value increases. This may be because the proportional relationship between the k value, the standard deviation and the range of acceptance.