Submitted by : *SAI KRISHNA W.D.S*
NC STATE ID: 200290148; UNITY ID: swupadr

RANDOM PROCESSES -PROJECT REPORT

1. RESTATEMENT OF THE PROBLEM:

The problem we were given was based on the application of Confidence intervals.

Basically given a communication system as shown below:

$$R = \sqrt{E_b}B + \sqrt{N_0/2}\,N$$

(where Eb is the energy-per-bit, $N0$ is the noise power spectral density, B is the transmitted bit value (+1 or –1) and N is normalized Gaussian noise with mean zero and variance 1.)

The task was to estimate the Bit error rate (BER), by modelling the above communication system, by considering different lengths of input data(n=10;n=100;n=1000), and by conducting the simulations m=100 times. Estimation of the confidence with which the true BER(shown below):

$$\text{BER} = 0.5\ \text{erfc}\left(\sqrt{E_b/N_0}\right)$$

occurs within the interval calculated was asked. The interval deviation and the sample mean were to be calculated by examining the sample vector 'X' ,that was to be designed based on the vector of received values 'R'.

Use *m*=100 and produce results for *n*=10, 100, and 1000. Set Eb/N0 = -3 dB (0.5).

2. EXPERIMENTAL SETUP:

The received vector 'R' was modelled by assuming that only the bits 'B' of magnitude '-1' were transmitted (owing to symmetry, the BER for B=+1, and B=-1 would be the same).

From the given assumption to take Eb/N0=1/2; I have taken Eb=1; and N0=2.

Now, the noise N is Gaussian with 0 mean and variance; and since it is random, I have modelled it using the randn function in Matlab.

I have used a row vector notation to represent the received vector 'R'; with the no. of columns being equal to the number of values received(n1=10 or 100 or 1000)

So, the equation for R actually modifies to be

R= -1 + randn(1,n1);

For this randn function, a random generator function 'rng(seed,'twister') is used.

```
% my name is SAI KRISHNA; SO A=16, E=64, I=256, O = 512, U = 1024)
%     %SEED =16+256+256+16=544
```

Thus, according to the given rubrics, my seed is 544.

Now, essentially for each trial, one row vector R is generated, for which a corresponding vector ‘X’ is generated for assessing the BER.

The vector X consists only of 0s and 1s.

Since the communication system is transmitting only -1 bits, the system has been designed to treat it as a received error if there has been a change in sign, ie., if -1 were transmitted and a positive quantity was received as R.

Similarly, the if the system receives a negative quantity as R, then the system knows that the sign has been maintained and reflects that no error has occurred.

The vector X is actually used to keep track of these above mentioned cases; and is given the value 1 if an error occurs; and a value 0 if no error has occurred.

Now, the sample mean or the estimated BER is calculated by calculating the mean of the vector X. The formula for the confidence interval is:

[M - ((y*sigmax)/(n1^0.5)) , M + ((y*sigmax)/(n1^0.5))];

Where M is the sample mean of the vector X, and y is roughly the number of standard deviations the sample mean is away from the true mean. And since the percentage of 68.3 is given in the problem, it translates to a deviation of one standard deviation on either side of the mean, in the case of a normal deviation.

Therefore, y=1.

Now, there are two cases for the calculation of the confidence interval:

i) Variance is Known:

Here, the X vector is assumed as containing the outputs of a Bernoulli distribution, and hence the variance is calculate as sqrt(p(1-p)); where p is the true BER given.

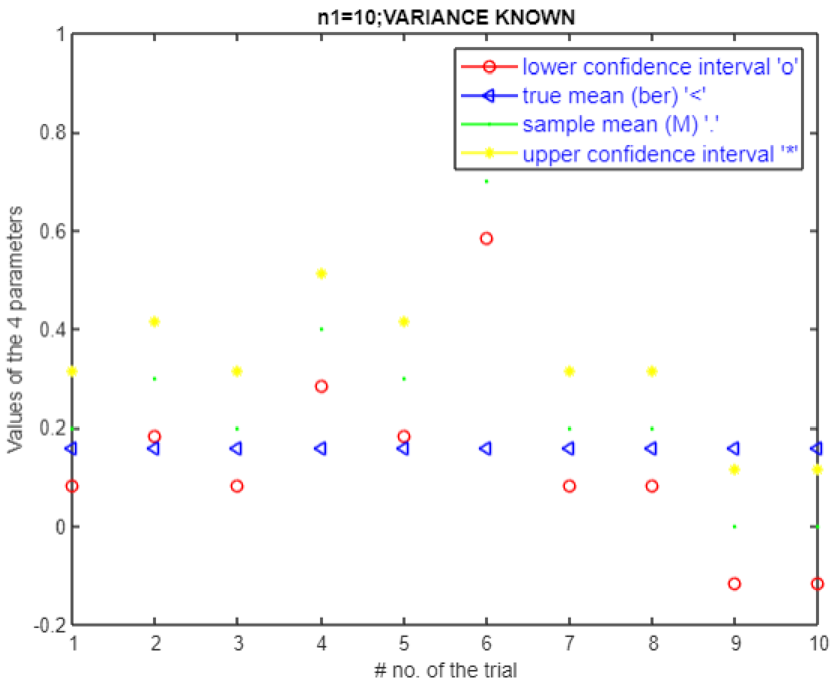
ii) Variance is Unknown:

In this case, the sample variance of the vector X is calculated and used.

The square root of the variance (Standard deviation) obtained in these 2 cases is substituted in place of the variable ‘sigmax’ in the equation above to get the confidence interval for each trial.

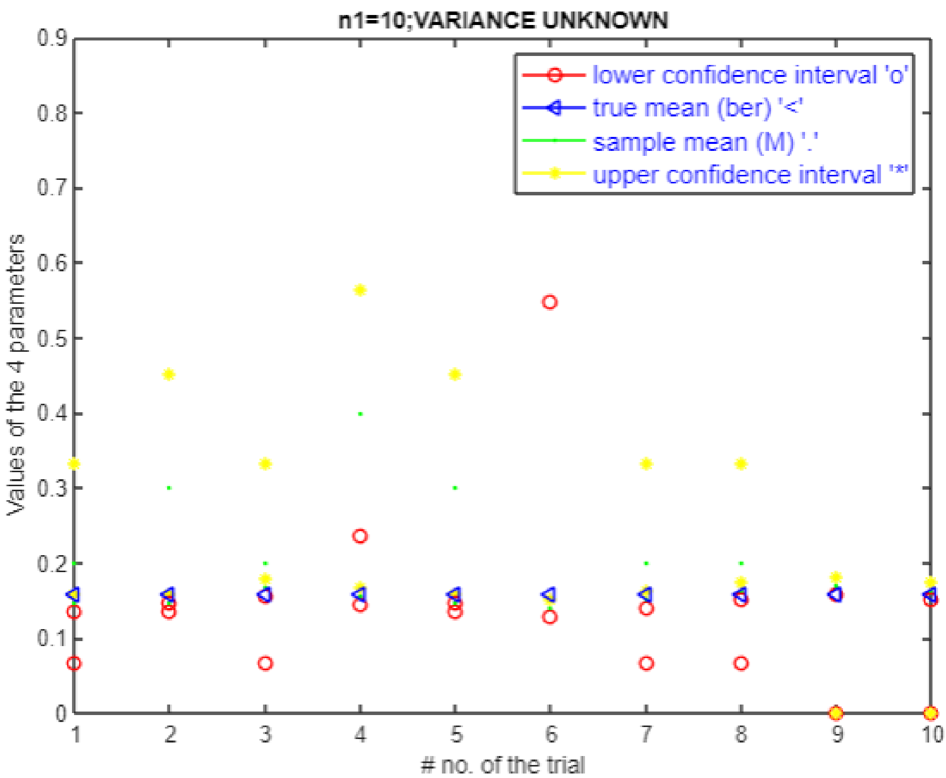
3. RESULTS OBTAINED:

i) n=10; VARIANCE KNOWN



`Percentage of true mean within the confidence interval is 58%.`

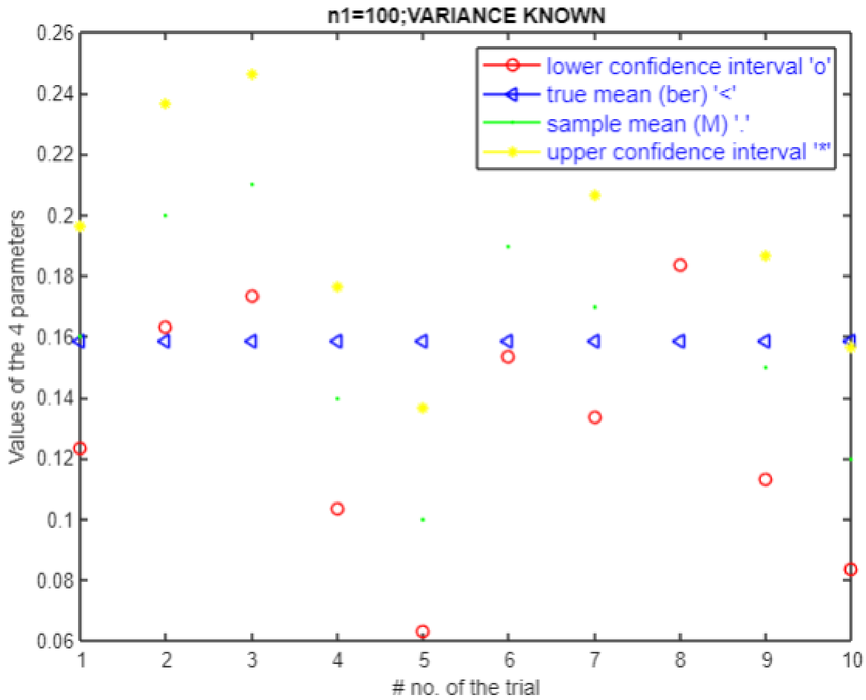
ii) n=10; VARIANCE UNKNOWN



`Percentage of true mean within the confidence interval is 74`

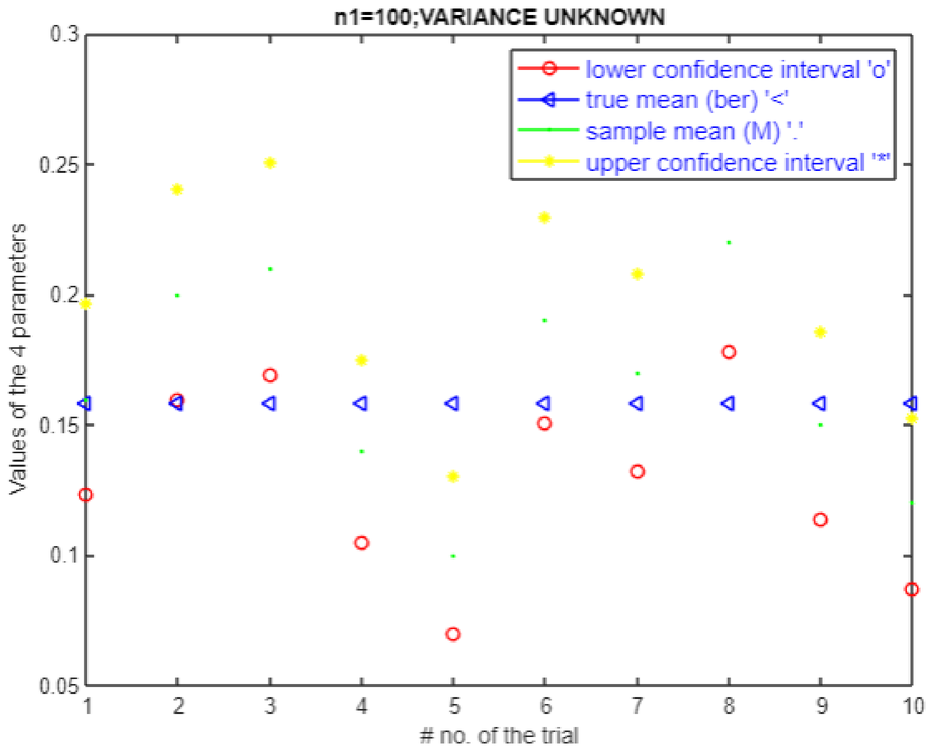
Submitted by : *SAI KRISHNA W.D.S*
NC STATE ID: 200290148; UNITY ID: swupadr

iii) n=100; VARIANCE KNOWN:



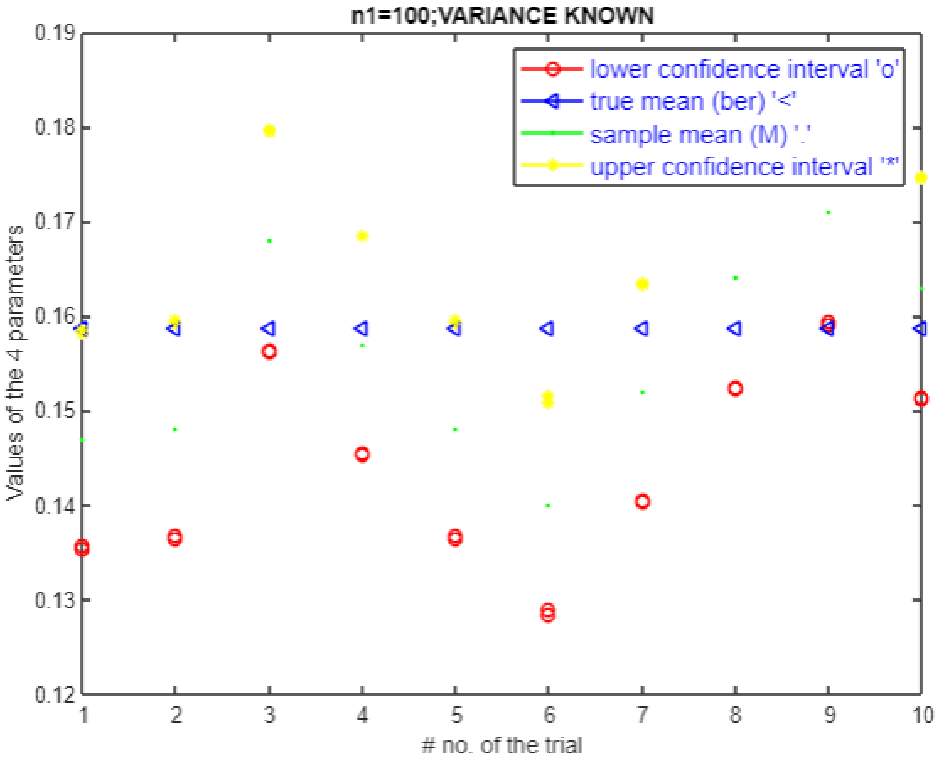
`Percentage of true mean within the confidence interval is 69`

iv) n=100; VARIANCE UNKNOWN:



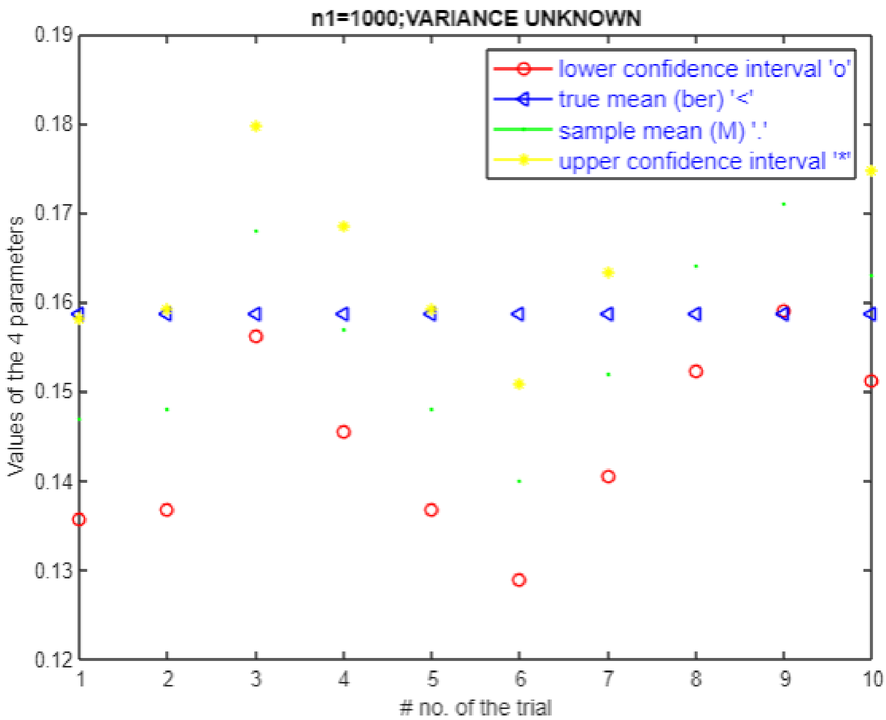
`Percentage of true mean within the confidence interval is 69`

v) n=1000; VARIANCE KNOWN:




```
Percentage of true mean within the confidence interval is 69
```

vi) n=1000, VARIANCE UNKNOWN




```
Percentage of true mean within the confidence interval is 69
```

ANALYSIS OF PLOTS:

The above plots show the distribution of the sample means amongst the confidence intervals for each trial. We also parallelly plot the true ber for each case, and see how the other parameters stock up for each trial. We observe that the confidence interval width decreases as we increase the n value.

For n=10, the plot results show how the difference in terms of the true BER coincides with the interval, whereas for n=100 and n=1000, the BER is the same for both the case of known and unknown variance. I presume this was because n=10 is a small sample value and thus this would have caused a difference in observations.

Now, the same data is presented in the form of a table:

VALUE OF N	VARIANCE KNOWN % CONFIDENCE	VARIANCE UNKNOWN % CONFIDENCE
10	58	74
100	69	69
1000	69	69

ANALYSIS OF TABLE:

The table above shows the percentage of times the true mean falls into the confidence interval. We see that for n1=10 alone, there is a drastic difference, and such a drastic difference was unexpected. But this can be reasoned by using the fact that n=10 is small sample size to find out the estimated variance, which has caused the anomaly. We notice that for bigger sizes of n, the obtained values of confidence tend to agree with both cases, which is intuitive.

4. **CONCLUSION:**

Thus, carrying out these m=100 trials showcase some results that can be cross-checked by intuition.

We observe the following pattern when we observe the impact on the simulated BER on increasing n.

VALUE OF N	Mean(Estimated 'M')	True BER
10	0.1	0.1587
100	0.13	0.1587
1000	0.1590	0.1587

Thus, we see that on increasing the n value, the estimated BER gets closer to the actual BER.

Now, similarly a pattern can be deduced about the width of the confidence interval on increasing the value of n:

VALUE OF N	Width of CI
10	0.2311
100	0.0731
1000	0.0231

Thus, since the estimated BER gets close to the true BER, the confidence interval width also keeps decreasing as we increase 'n'.

Thus, the main points of learning from this project are about realizing the concepts learnt practically in the form of Matlab implementation. During the course of completion of this project,I had to debug my code several times, and that made me grasp how the concept has to be realized in an air-tight manner, as even though the logic implemented is right, the structure of the code is what finally determines the output of the code.

Working with software tools, its easier to analyse for larger data sets with large numbers, which would be a tedious process by hand.

Thus, working on the confidence intervals in a simulated environment has given me an idea of how to tackle actual realistic problems in the industry.

5. APPENDIX:

CODE USED FOR KNOWN VARIANCE:

```
% code for estimating the confidence intervals using known variance
% [ modelling X as a bernoulli random variable; variance =p(1-p)]
clc
m=100;
n1=10;
countn1=0;
% UNCOMMENT THE N1 VALUE ACCORDINGLY
%n1=100;
%n1=1000;
tber=0.5*erfc((0.5)^0.5)
% my name is SAI KRISHNA; SO A=16, E=64, I=256, O = 512, U = 1024)
% %SEED =16+256+256+16=544
rng(544,"twister");




for i=1:m
    % my name is SAI KRISHNA; SO A=16, E=64, I=256, O = 512, U = 1024)
    %SEED =16+256+256+16=544
    x=zeros(1,n1);
    for j=1:n1
        %rng(544,"twister");
        r=-1.+randn(1,n1);
        if r(1,j)>=0    % the bit b is -ve; so if the received value is +ve,
then error occurs, and we set x=1
            x(1,j)=1;
        elseif r(1,j)<0        % the bit b is -ve; so if the received value is
-ve, then no error has occured, and we set x=0
            x(1,j)=0;
        end
    end
    % computing the sample mean M of the X matrix:
     x;
    M=mean(x,'all')
    % Confidence interval is [ M-((y*sigmax)/(n^0.5)) , M+((y*sigmax)/(n^0.5))
]
```

```
    y=1;
    % since we estimate the y for 68.3% confidence, the y=1; as in a standard
    % dev case, the 68.3 % is obtained in + or - one sigma



%     here the X is considered to be a matrix with elements mirroring a
bernoulli random variable;
%     and thus the variance for a bernouli random variable is p(1-p); where
%     p is the probability of success; and here the p is the true BER.
    sigmax=((tber*(1-tber))^0.5)%     % finding the lower and upper limits of
confidence intervals
    lln1=M-((y*sigmax)/(n1^0.5))
    uln1=M+((y*sigmax)/(n1^0.5))
    ciwidth=uln1-lln1
    if tber>=lln1 && tber<=uln1
        countn1=countn1+1
    end
    k=i;
    if k<=10
        plot(k,lln1,"Marker","o","Color","red")
        hold on
        plot(k,tber,"Marker","<","Color","blue")
        hold on
        plot(k,M,"Marker",".","Color","green")
        hold on
        plot(k,uln1,"Marker","*","Color","yellow")
        legend({"lower confidence interval 'o'","true mean (ber) '<'","sample
mean (M) '.' ","upper confidence interval
'*'"},'FontSize',12,'TextColor','blue')
        title("n1=10;VARIANCE KNOWN")
        % I use the appropriate ttile for different n values of 10,100,1000
        % accordingly
        %title("n1=100;VARIANCE KNOWN")
        %title("n1=1000;VARIANCE KNOWN")
        xlabel("# no. of the trial")
        ylabel("Values of the 4 parameters")

    end



end
fprintf("Percentage of true mean within the confidence interval is %d
",countn1)
```

CODE USED FOR UNKNOWN VARIANCE:

```
% code for estimating the confidence intervals when variance is unknown
(sample variance from vector X )
clc
```

Submitted by : *SAI KRISHNA W.D.S*
NC STATE ID: 200290148; UNITY ID: swupadr

```
m=100;
n1=10;
% UNCOMMENT THE N1 VALUE ACCORDINGLY
%n1=100;
%n1=1000;
countn1=0;
tber=0.5*erfc((0.5)^0.5)
% my name is SAI KRISHNA; SO A=16, E=64, I=256, O = 512, U = 1024)
% %SEED =16+256+256+16=544
rng(544,"twister");







for i=1:m
    % my name is SAI KRISHNA; SO A=16, E=64, I=256, O = 512, U = 1024)
    %SEED =16+256+256+16=544
    x=zeros(1,n1);
    for j=1:n1
        %rng(544,"twister");
        r=-1.+randn(1,n1);
        if r(1,j)>=0    % the bit b is -ve; so if the received value is +ve,
then error occurs, and we set x=1
            x(1,j)=1;
        elseif r(1,j)<0        % the bit b is -ve; so if the received value is
-ve, then no error has occured, and we set x=0
            x(1,j)=0;
        end
    end
    % computing the sample mean M of the X matrix:
    M=mean(x,'all')
    % Confidence interval is [ M-((y*sigmax)/(n^0.5)) , M+((y*sigmax)/(n^0.5))
]
    y=1;
    % since we estimate the y for 68.3% confidence, the y=1; as in a standard
    % dev case, the 68.3 % is obtained in + or - one sigma


    sigmax=std( x(:) )
    % finding the lower and upper limits of confidence intervals
    lln1=M-((y*sigmax)/(n1^0.5))
    uln1=M+((y*sigmax)/(n1^0.5))
    if tber>=lln1 && tber<=uln1
        countn1=countn1+1
    end
    k=i;
    if k<=10
        plot(k,lln1,"Marker","o","Color","red")
        hold on
```

```
        plot(k,tber,"Marker","<","Color","blue")
        hold on
        plot(k,M,"Marker",".","Color","green")
        hold on
        plot(k,uln1,"Marker","*","Color","yellow")
        legend({"lower confidence interval 'o'","true mean (ber) '<'","sample
mean (M) '.' ","upper confidence interval
'*'"},'FontSize',12,'TextColor','blue')
        title("n1=10;VARIANCE UNKNOWN")
        % I use the appropriate ttile for different n values of 10,100,1000
        % accordingly
        %title("n1=100;VARIANCE UNKNOWN")
        %title("n1=1000;VARIANCE UNKNOWN")
        xlabel("# no. of the trial")
        ylabel("Values of the 4 parameters")

    end



end


fprintf("Percentage of true mean within the confidence interval is %d
",countn1)
```