University of Southern California

EE511

Simulation Methods for Stochastic Systems

Project #3- Samples and Statistics

BY

Mohan Krishna Thota

USC ID: 6683486728

mthota@usc.edu

Problem 1:

A components manufacturer delivers a batch of 125 microchips to a parts distributor. The distributor checks for lot conformance by counting the number of defective chips in a random sampling (without replacement) of the lot. If the distributor finds any defective chips in the sample, they reject the entire lot. Suppose that there are six defective units in the lot of 125 microchips. Simulate the lot sampling to estimate the probability that the distributor will reject the lot if it tests five microchips. What is the fewest number of microchips that the distributor should test to reject this lot 95% of the time?

SOLUTION:

Hypergeometric Distribution:

- The **Hypergeometric distribution** is used to calculate probabilities when sampling without replacement.
- The distribution describes the probability of 'R' successes in 'N' draws, without replacement from a finite population of size 'N' that contains exactly 'R' successes, where each draw is a success or failure.

Procedure:

- In the problem it is mentioned that the distributor checks in sampling without replacement.
- Let the functioning chips be represented by one and malfunctioned as zero.
- Here a list of 119 one's are created i.e., functioned and a list of six zeroes are created i.e., defective.
- Then the both are combined, and defects are placed at random by using randperm() in MATLAB.
- Now a variable is initialized with False, in order to test the 5 microchips a for loop is iterated.
- A variable 'count' is given a random integer value within 125, which acts as an index pointer.
- Then an if condition is used to check, whether the value pointed by counter is zero.
- If the counter value is zero, then the reject is made 'true'.
- Then another array named 'rejected_1' is used to assign this obtained reject value.
- Then MATLAB's nnz() command is used to check for value of non-zero elements in 'rejected_1' array.
- In order to calculate the fewest number of microchips that the distributor should test to reject this lot 95% of time.

- Firstly, the percentage value is initialized to zero and the number of samples is initialized to get started. We can select any sample value. Let us choose 2.
- While condition is used, to check for 95% of time, and the process is repeated until the while condition is false by increasing the number of samples by one each time.
- The incremented sample value before while loop becomes false is the fewest of the samples required to reject lot 95% of the time.

OUTPUT

```
The number of rejects is

   24

The fewest samples

   52

The number of rejects is

   22

The fewest samples

   51

The number of rejects is

   27

The fewest samples

   58
```

CODE:

```
No_of_chips=125;
defected_mc=zeros(1,6);
non_defected=ones(1,119);
total_mc=[defected_mc,non_defected];
actual_mc=total_mc(randperm(length(total_mc)));
rejected_1=zeros(1,No_of_chips);
rejected_2=zeros(1,No_of_chips);
No_of_sample=3;
percent=0;

%1(a)%
for x = 1:No_of_chips
    intialise_reject = false;
    for j = 1:5
        gen_random = randi(125);
        if(actual_mc(1,gen_random) == 0)
            intialise_reject = true;
            break;
        end
    end
    rejected_1(1,x) = intialise_reject;%assign value of reject to another
array
end
rejections = nnz(rejected_1);

%1(b)%
while(percent<=0.95)
    for num=1:No_of_chips
        initialise_reject=false;
        for j=1:No_of_sample
            count=randi(125);
            if(actual_mc(1,count)==0)
                initialise_reject=true;
                break;
            end
        end
        rejected_2(1,num)=initialise_reject;
    end
    No_of_sample = No_of_sample + 1;
    rejected_mc = nnz(rejected_2);
    percent = rejected_mc/No_of_chips;

end


disp('The number of rejects is');
disp(rejections);
disp('The fewest samples');
disp(No_of_sample)
```

ANALYSIS

Here from the above we could see that, the number of rejects is in the range of 20 to 30. Considering 25 as the median estimated probability that the distributor will reject the lot if it tests five microchip is around 0.25.

The fewest number of microchips that the distributor should test to reject this lot 95% of the time is around 52.

Problem 2:

Suppose that 120 cars arrive at a freeway onramp per hour on average. Simulate one hour of arrivals to the freeway onramp:

1. subdivide the hour into small time intervals(< 1 second) and then
2. perform a Bernoulli trial to indicate a car arrival within each small time-interval. Generate a histogram for the number of arrivals per hour. Repeat the counting experiment by sampling directly from an equivalent Poisson distribution by using the inverse transform method (described in class). Generate a histogram for the number of arrivals per hour using this method. Overlay the theoretical p.m.f. on both histograms. Comment on the results.

SOLUTION:

Here as asked in the question, Simulate one hour of car arrivals to the freeway onramp using Poisson counting

a) Bernoulli Trial Method:

- Here the number of cars arriving per hour i.e., 'lambda'=120 and was asked to divide the hour into smaller time intervals(<1 seconds) therefore dividing it into 1000 sub-intervals say 'No_intervals'=1000.
- Bernoulli trial is used to indicate a car arrival i.e., success probability for Bernoulli trial in each sub-interval is calculated by dividing 'lambda' by 'No_interval'.
- Therefore Bernoulli trial is conducted for each subinterval for a total of N simulations.
- In each simulation a random number is generated using rand() function.
- Success or failure is determined by comparing this Bernoulli success probability with the random number generated.
- Summing up the values for all the Bernoulli trials gives me the desired output.

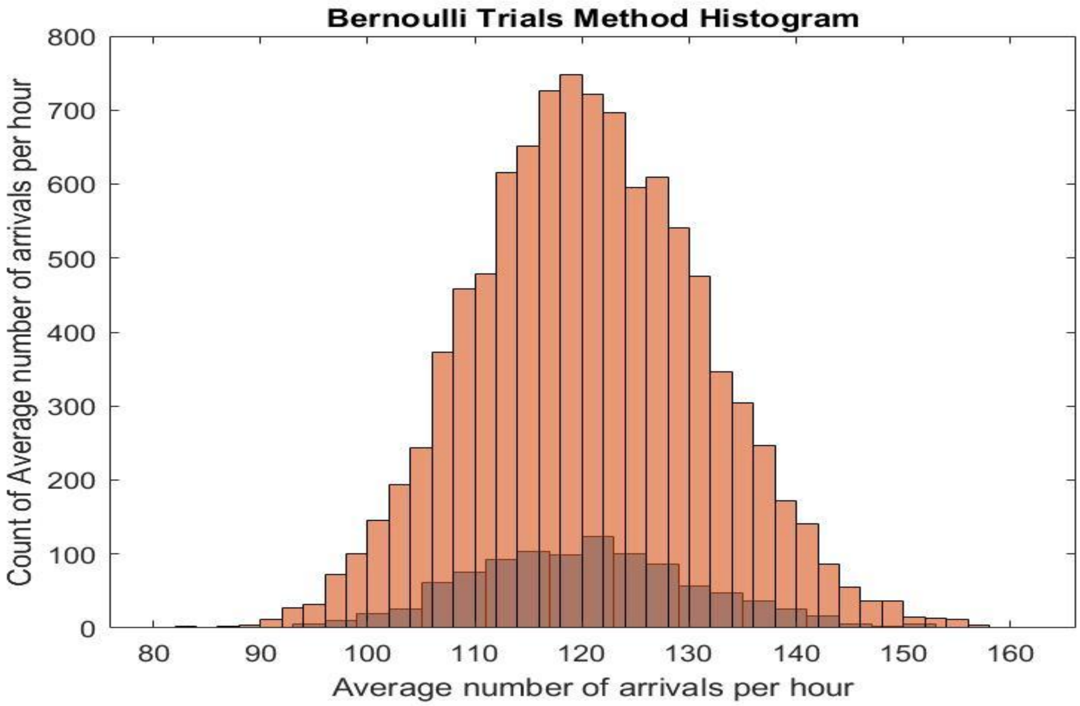
Inverse Transform Method

- As given average of 120 cars arrive at a freeway onramp per hour i.e., 'lambda'=120**.**
- A loop is run for each N. Random number is generated and required initializations are done (i=0, p= negative exponential of lambda and F=p).
- If random number generated is less than F, result is stored and break the loop for next run. Else, F is incremented by p and I by 1, and p is updated.

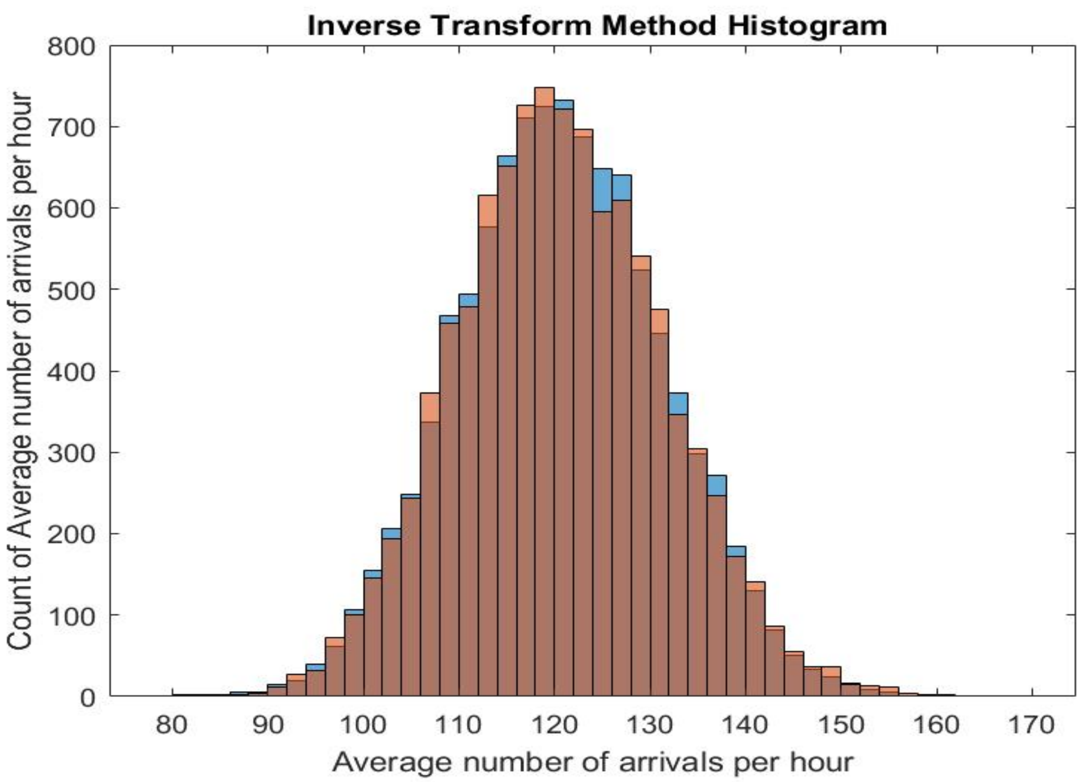
CODE:

```
%Bernoulli Trial Method%
First_approach=zeros(1,[]);
No_of_simulations=1000;
small_time_intervals=10000;
lambda=120;
prob_success=lambda/small_time_intervals;

for i=1:No_of_simulations
    random_number=rand(small_time_intervals,1);
    Bernoulli_success=random_number<prob_success;
    sum_probabilities=sum(Bernoulli_success);
    First_approach=[First_approach,sum_probabilities];
end

%inverse transform Method%
second_approach=zeros(1,small_time_intervals);
for j=1:small_time_intervals
    i=0
    random_number=rand();
    prob=exp(-lambda);
    X=prob
    while(true)
        if(random_number<X)
            second_approach(1,j)=i;
            break;
        else
            prob=(lambda*prob)/(i+1);
            i=i+1;
            X=X+prob;
        end
    end
end

%Displaying%
random_number=rand(small_time_intervals,1);
inv=poissinv(random_number,lambda);

%1st histogram%
figure;
histogram(First_approach);
xlabel('Average number of arrivals per hour');
ylabel('Count of Average number of arrivals per hour');
title('Bernoulli Trials Method Histogram');
hold on
histogram(inv)

%2nd histogram%
figure;
histogram(second_approach);
ylabel('Count of Average number of arrivals per hour');
xlabel('Average number of arrivals per hour');
title(' Inverse Transform Method Histogram');
hold on
histogram(inv)
```

HISTOGRAM:



Bernoulli Trials Method Histogram
Count of Average number of arrivals per hour
Average number of arrivals per hour
0
100
200
300
400
500
600
700
800
80
90
100
110
120
130
140
150
160



Inverse Transform Method Histogram
Count of Average number of arrivals per hour
Average number of arrivals per hour
0
100
200
300
400
500
600
700
800
80
90
100
110
120
130
140
150
160
170

ANALYSIS:

- From the histograms we could see that the calculated theoretical values are similar with that of the obtained values.
- Using Bernoulli trial method, for poisson counting, the approximation is finer when the number of sub-intervals is smaller.
- With the decrease in sub-interval, the approximation is getting better.
- We could see the histogram for the X-axis(average number of cars) and Y-axis(count).
- In the similar manner, for the inverse transform method, the distribution is getting better with the increase in simulations.
- Inverse transform method can be used, as long as when we get an explicit formula for 'prob'
- Poisson distribution refers to when the events occur randomly and independently, at a constant rate (in time), then the count of these events per unit time.
- From the histogram, we could observe that the average number of cars arriving per hour is concentrated around 120.
- The range average number of cars arriving in an hour is between 90 and 155.

Problem 3:

Procedure

- As mentioned in the problem, random samples are generated in such a way that the sum of generated random numbers is equal to four.
- A loop is used to generate random number by rand() function for given number of samples.
- The obtained counts are stored in an array.
- Mean of the stored count for given number of samples is calculated by using Mean() function.
- Histograms of the generated random numbers is obtained using Histogram() command.

Analysis:

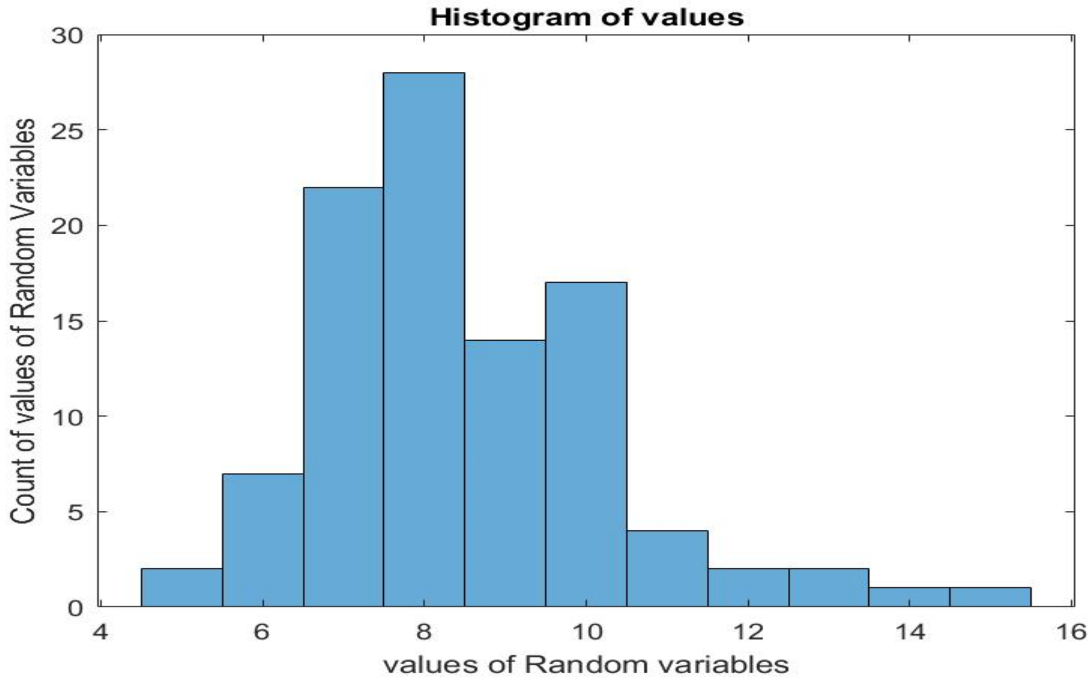
- The given problem is a Geometric random variable.
- From the below figure and the mean value obtained, the mean value is irrespective of the number of samples.
- As Geometric random variables involves independent Bernoulli trials.

$$P[\mathrm{X} = \mathrm{n}] = p(1 - \mathrm{p})n - 1 for all n \Rightarrow$$

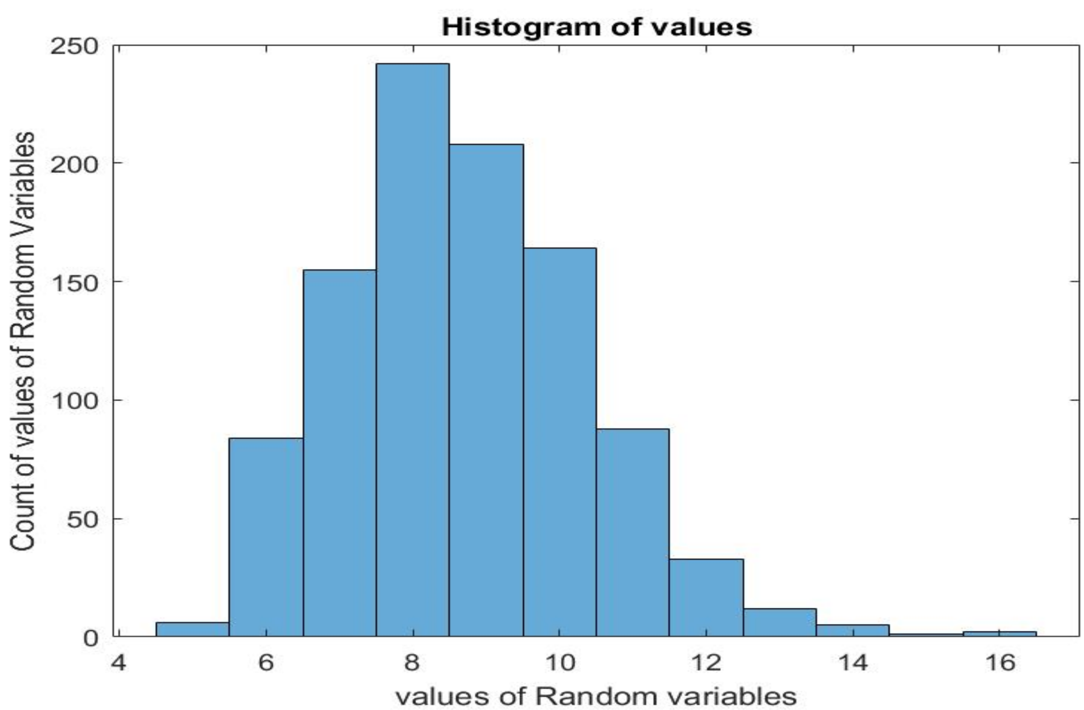
Code:

```
%Problem 3%
clc;
clear;
No_of_samples=100;
array=[]
for k=1:No_of_samples
    sum=0;
    count=0;
    while(sum<=4)
        random_number=rand()
        count=count+1;
        sum=sum+random_number
    end
    array(1,k)=count;
end
Mean_samples=mean(array);
Output=['for a total of ', num2str(No_of_samples),' samples the
mean is ',num2str(Mean_samples)]
disp(Output)
figure;
histogram(array);
title('Histogram of values');
xlabel('values of Random variables');
ylabel('Count of values of Random Variables');
```

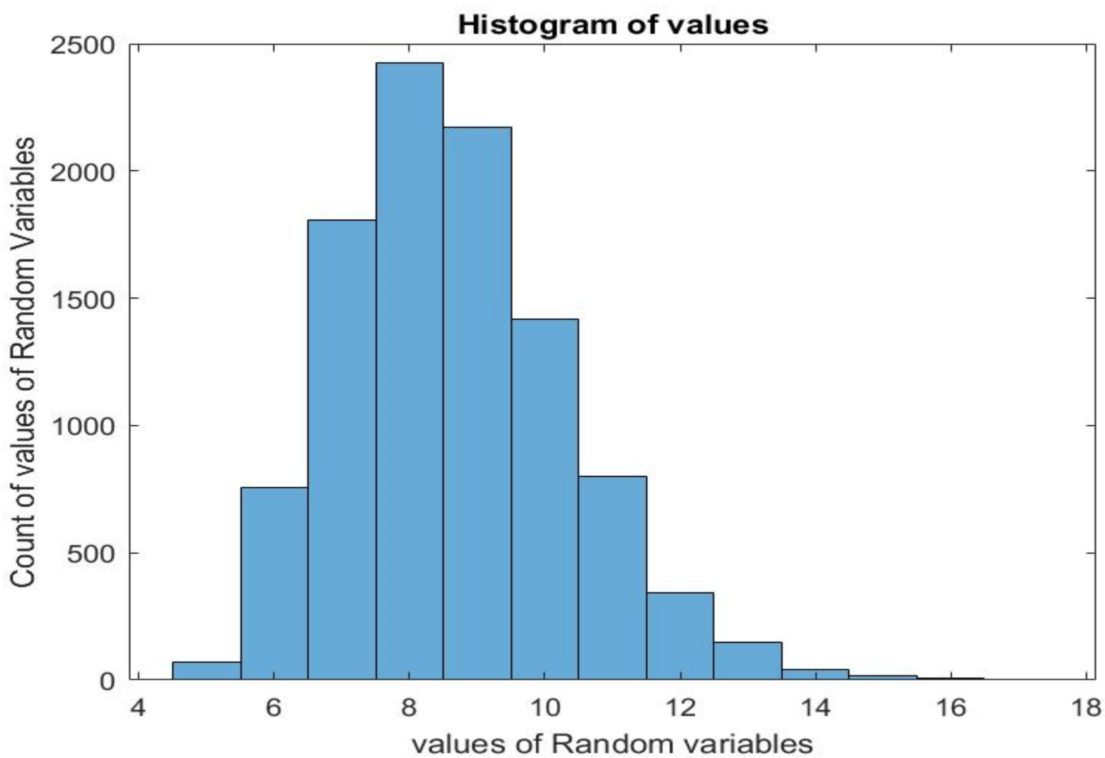
For a total of 100 samples the mean(E[N]) is 8.73



For a total of 1000 samples the mean is 8.726



For a total of 10000 samples the mean is 8.6413



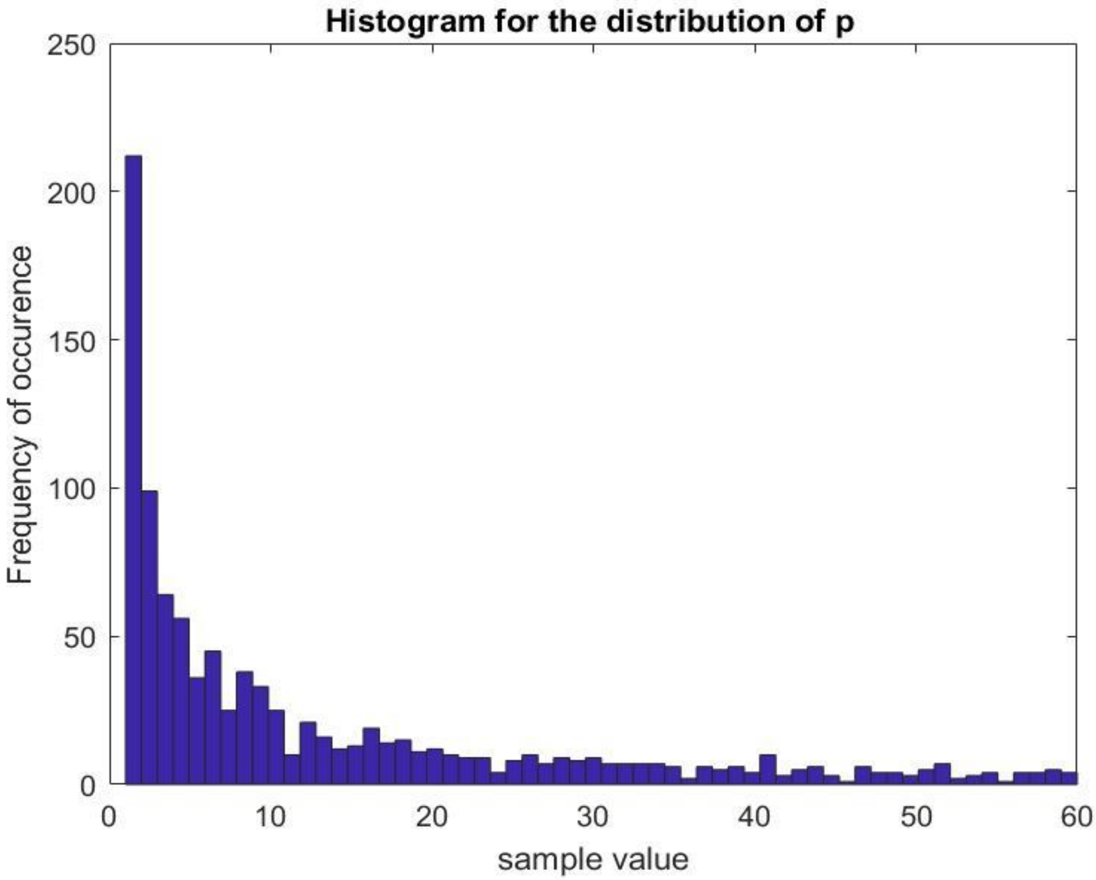
Problem 4:

- Here the p values are found by equating the sum of all values in the sequence to 1.
- By making p=1.p(j) is found out.
- A function say 'K' is designed in such a way that it compares the random variable U with an array where array consists of cumulative sum of all variables of P.
- This function returns the indices of all the values where the cumulative sum is greater than U.
- The obtained indices are saved in X.
- Matlab's find() function is used to generate sequence N60.

CODE:

```
a=0
for i=1:60
    a=a+(1/i);
end
p=1/a;
N60=[];
for i=1:60
    N60(i)=p/i;
end
 sample=[]

for j=1:1000
    m=randsample([1:60],1,true, N60);
    sample(j)=m;
end

hist(sample,60);
title('Histogram for the distribution of p ');
ylabel('Frequency of occurence');
xlabel('sample value');

k=[]
for i=1:1000
    t=0;
    while 1>0
        n=randsample([1,0],1,true, [(p/60),(1-(p/60))]);
        t=t+1;
        if n==1
            k(1,i)=t;
            break;
        end
    end
end
E60_experimental=mean(k);
E60_theotrical=60/p;
V60=(1-(p/60))/(p/60)^2;
output1=['Experimental mean is ',num2str(E60_experimental)];
disp(output1)
output2=['Theotrical mean is ',num2str(E60_theotrical)];
disp(output2)
```

HISTOGRAM



OUTPUT

Experimental mean is 286.849
Theoretical mean is 280.7922
Experimental variance is 4.6782e+04
Theoretical variance is 7.8563e+04

From the above we could see that both the theoretical and experimental values are close to each other.
Also p(60) and p(1) is the most.

PROBLEM 5

Accept-Reject method

It is a classical sampling method which allows one to sample from a distribution which is difficult or impossible to simulate by an inverse transformation. Instead, draws are taken from an *instrumental density* and accepted with a carefully chosen probability. The resulting draw is a draw from the target density.

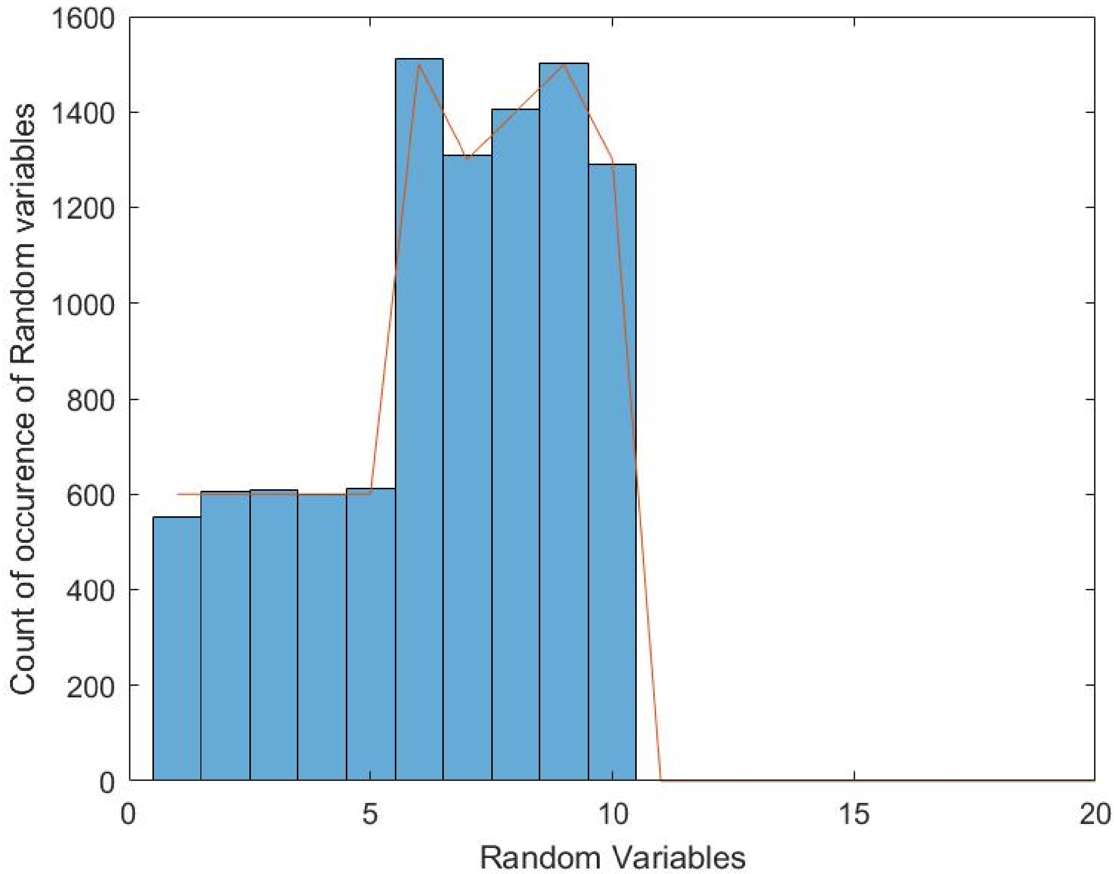
Procedure:

- Here sequence Q is given for 20 values as 0.05 whereas P is given for only for 10 values therefore the remaining ten values are appended with zeroes.
- Here the constant k is obtained by getting max(p/q) values which is 3 i.e., 0.15/0.05=3.
- Therefore, the theoretical efficiency of the sampler is 1/k ->1/3 which is 33%.
- Here random numbers are generated using MATLAB inbuilt rand() function and the generated random numbers are compared to a 'P' of randomly generated Index(j).
- Therefor 3*U<=P(j)/0.05*k*q*j forms overs the plot of p(j) as an envelope.
- Now the variables which falls under the distribution of p(j) are accepted and which does not satisfy the comparison are removed.
- By following the above procedure, the pmf of p(j) is generated.

CODE:

```
Number_of_samples = 10000;
q=[5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5]/100;
p = [6 6 6 6 6 15 13 14 15 13 0 0 0 0 0 0 0 0 0 0]/100;
for i=1:Number_of_samples
    k=0;
    while 1,k = k+1
         j = 1+floor(20*rand);
         if (3*rand())<=p(j)/0.05
             X(i)=j;
             C(i)=k;
             break
         end
    end
end

figure();
histogram(X);
hold on;
plot(p*Number_of_samples)

mean_x = mean(X);
output_mean=['Experimental mean is ',num2str(mean_x)];

E_theotrical=0;
for i=1:20
    E_theotrical=i*p(i)+E_theotrical
end
output=['Theotrical Mean is ',num2str(E_theotrical)];
disp(output)

var_experimental=var(X);
output_var=['Experimental Variance is ',num2str(var_experimental)];
mean_C=mean(C)

output3=['Mean of count array is ',num2str(mean_C)];

sum1=0
for l=1:Number_of_samples
  sum1=sum1+C(1,l);
end
exp_efficiency=(Number_of_samples/sum1);
output4=['Experimental efficiency is ',num2str(exp_efficiency)];

theotrical_eff=1/3
output5=['Theotrical efficiency is ',num2str(theotrical_eff)];
disp(output_mean)
disp(output)

disp(output_var)
disp(output3)
disp(output4)
disp(output5)
```

HISTOGRAM:



OUTPUT:
Experimental mean is 6.4629
Theoretical Mean is 6.48
Experimental Variance is 7.2041
Mean of count array is 3.0018
Experimental efficiency is 0.33313
Theoretical efficiency is 0.33333

ANALYSIS:

Here the theoretical values for the mean and variance are calculated using the following formulae

$$E[X] = \sum nk = 1x \cdot p(x)$$

$$V[X] = E[(X - E[X])2]$$

- From the figure we could see that the histogram and the plot is closer as the experimental values are closer to the theatrical computed values.
- We could also see that both the theatrical and experimental values are closer to 33%.
- From the above findings, we could say that Accept-Reject Method is a good approximation for P(J).