University of Southern California

EE511

Simulation Methods for Stochastic Systems

Project #3

Samples and statistics

BY

Akshay Deepak Hegde

USC ID: 8099460970

hegdeaks@usc.edu

Question 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 it rejects 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?

Description:

The Hypergeometric distribution is a discrete probability distribution that describes the probability of 'k' successes in 'n' draws, without replacement, from a finite population of size N that contains exactly K successes, wherein each draw is either a success or a failure. Here, defective microchips are represented by the value 0 and the non-defective microchips by 1. Then defectmc has been with an array of 6 zeros and nodefectmc with an array of 119 ones. The clubbed array contains 125 values of defective and non-defective chips. Since we need the defect positions to be at random, we reshuffle using the randperm() command in MATLAB.

Now, I initialized a variable called 'reject' with false. And iterated over a for loop to test the 5 microchips. Another variable 'count' is given a random integer value within 125, to act as the index pointer. We then see if the value pointed by 'count' is 0 or not in the microchip array. If it is 0, the 'reject' gets assigned true and the loop breaks for that iteration. We assign the value of reject thus obtained to another array called rejectArray1.

Next, using nnz() command, the value of number of non zero elements are found out. This gives us the number of rejects in the test as the number of ones corresponds to rejects.

Now, to find the fewest number of microchips to be tested such that the lot is rejected 95% of the time, the percent is initialized to 0, and the number of samples to start off with is taken to be 3(can take any small value). The value of samples is incremented by 1 in each iteration until the percent value crosses 95%. The for loop is run again and the same process is repeated until the while condition becomes false. The value of samples before the while condition fails gives us the required fewest number of samples.

Code:

```
 % Akshay Deepak Hegde  USC ID: 8099460970  %
% ------------------------------------------------------------- %
% Project #3-Samples and statistics , EE511: Spring 2017, Due: 7th Feb
% ------------------------------------------------------------- %
% To simulate the lot sampling to estimate the probability
% that distributor will reject lot of it testing 5 chips
% To find fewest number that distributor should check
% ------------------------------------------------------------- %
clc;
clear;
close all;
% ------------------------------------------------------------- %


n=100;%Number of samples
defectmc = zeros(1,6);%initializing the 6 values to zeros(defective)
nodefectmc = ones(1,119);% 119 nondefetive
microchip = [defectmc, nodefectmc];
microchip = microchip(randperm(length(microchip)));%using randperm to randomize he array
rejectArray1 = zeros(1,n);%initializing the array with zeros

for x = 1:n
   reject = false;%initialize reject with false
   for j = 1:5%for loop to test 5 microchips
      count = randi(125);%generates a random int upto 125
      if(microchip(1,count) == 0)%to check at random for defects
         reject = true;%if defect is found assign true(1)
         break;
      end
   end
   rejectArray1(1,x) = reject;%assign value of reject to another array
end

Defective = nnz(rejectArr1);%to count the number of 1's or rejects
disp('The number of rejects for 5 microchip test is');
disp(Defective);
rejectArr2 = zeros(1,n);
percent = 0;%initialize percent to 0
samples = 3;%Assume samples=3 to start off with
while (percent <= 0.95)%to check for 95%
   for x = 1:n
      reject = false;
      for j = 1:samples
         count = randi(125);
```

```
        if(microchip(1,count) == 0)
            reject = true;
            break;
        end
    end
    rejectArr2(1,x) = reject;
  end
  Defective = nnz(rejectArr2);
  percent = Defective/n;%calculate the percent
  samples = samples + 1;%increment samples by 1
end

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

Output:

```
The number of rejects for 5 microchip test is
29
The fewest number of samples is
49

The number of rejects for 5 microchip test is
24
The fewest number of samples is
51

The number of rejects for 5 microchip test is
19
The fewest number of samples is
54
Analysis:
```

The number of rejects when the microchips is sampled in lots of 5 microchips, is around 20-30 as can be seen from the output above. So the median is around 25% of the times. So we can say the probability of rejection is 0.25 for this case.

The fewest number of samples wherein the lot is rejected 95% of the time comes to be around 50 samples as can be seen from the output.

This is a perfect case of Hypergeometric distribution where the probability of ‘k’ successes in ‘n’ draws, without replacement, from a finite population of size N that contains exactly K successes, wherein each draw is either a success or a failure, in this case it is accept or a reject respectively.

Question 2.

Description:

Problem demands the simulation of number of arrivals of car per hour using Poisson counting,

(a) Bernoulli Trial Method
(b) Inverse Transform Method

For (a), average arrival rate, lambda is given as 120. I am dividing the hour into 10000 sub intervals(N) and success probability for Bernoulli trial in each subinterval is calculated by dividing lambda by total number of subintervals. Loop is run for n (number of simulations) times. In each run, a random number is generated using rand() function bernoulli trial is performed. We get either a success or a failure by comparing success probability with the random number generated. Summing up the successes of all Bernoulli trials will give me the desired output.

For (b), we know the average arrival rate, lambda as 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 I break the loop for next run. Else, F is incremented by p and I by 1, and p is updated.

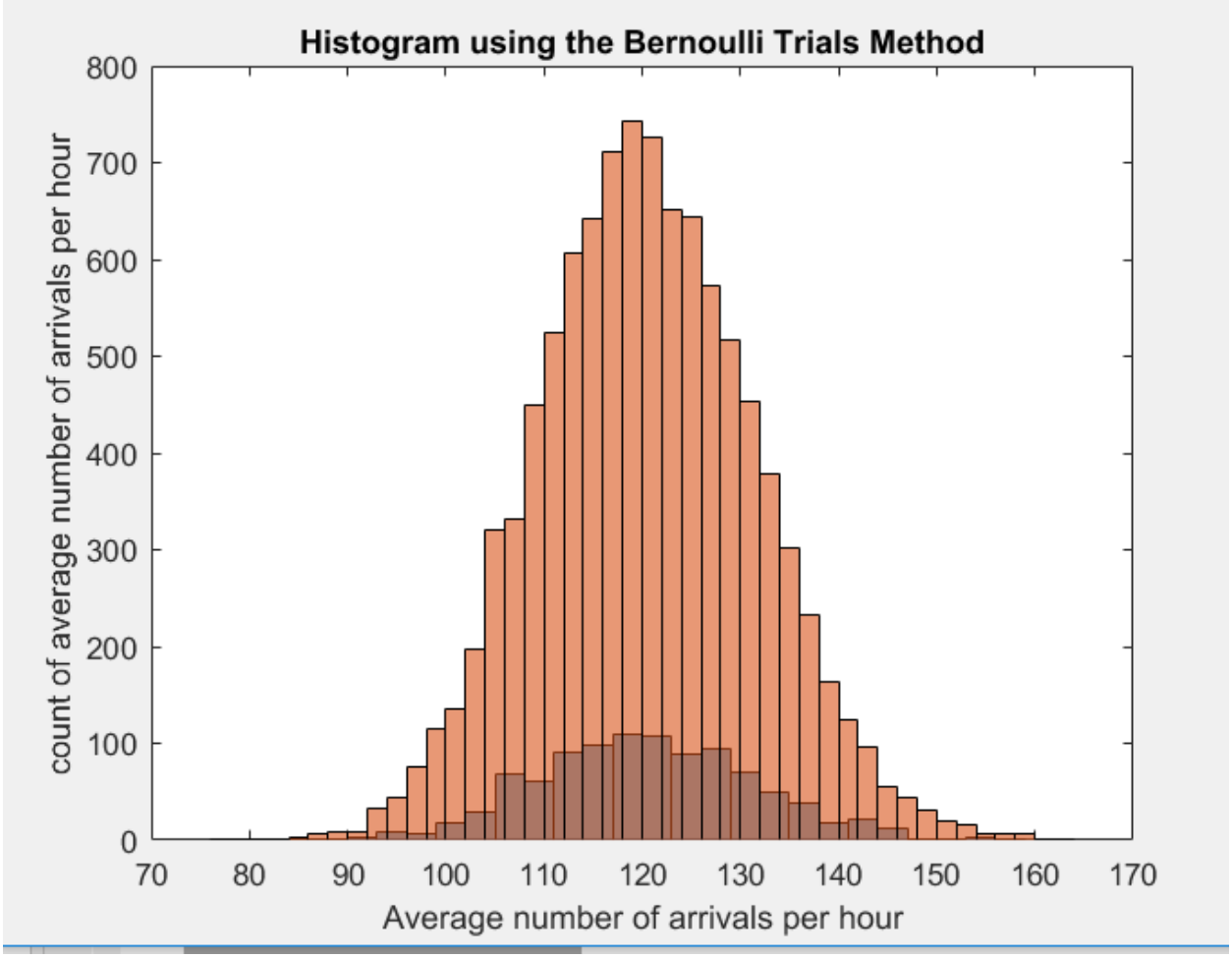
Histograms for each method are generated using histogram() function.

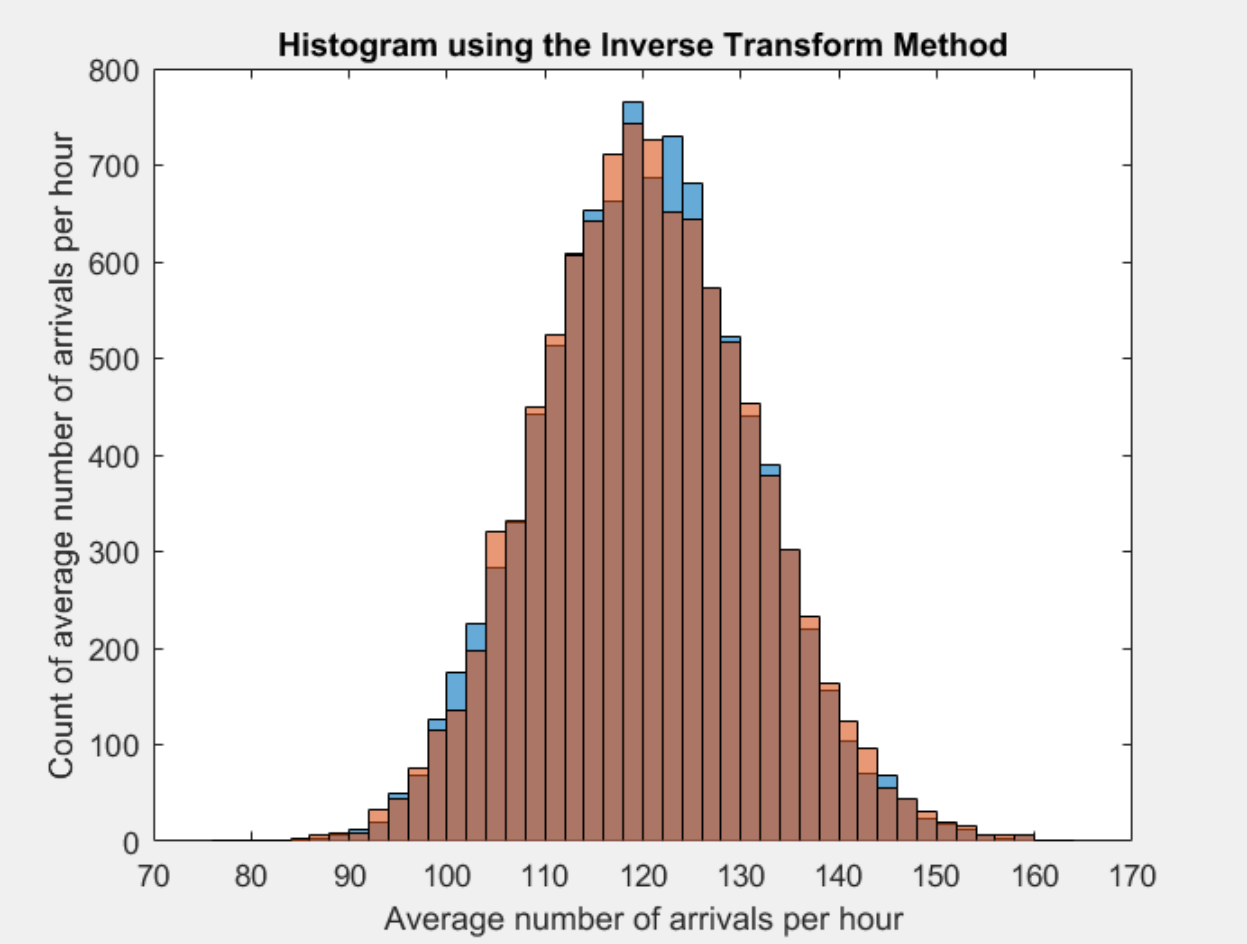
Code:

```
        % Akshay Deepak Hegde  USC ID: 8099460970  %
% ------------------------------------------------------------- %
% Project #3-Samples and statistics , EE511: Spring 2017, Due: 7th Feb
% ------------------------------------------------------------- %
% To simulate one hour of arrivals to the freeway ramp
% (a) sybdividing hour into small time intervals and
%     performing Bernoulli trial within each interval.
% (b) Poisson distribution using inverse transform method
% To generate histograms and overlay theoritical pmfs.
% ------------------------------------------------------------- %
clc;
clear;
close all;
% ------------------------------------------------------------- %
n=1000;%number of simulations
lambda = 120;%120 cars per hour on average
N = 10000;%one hour into 10000 subintervals
p = lambda/N;%Success probability in each Bernoulli random variable
Result1 = zeros(1,[]);
% ------------------------------------------------------------- %
% subdividing and performing Bernoulli trials
for i = 1:n
```

```
    U = rand(N,1);%random number generation
    bernoulliTrials = U<p;%perform Bernoulli trial
    x = sum(bernoulliTrials);%summing up successes
    Result1 = [Result1,x];%store result
end
% ------------------------------------------------------------- %
% Poisson distribution using inverse transform method
lambda = 120;%avearge arrival rate
Result2 = zeros(1,N);%initializing N coulums with 0
% Recursion Method
for k = 1:N
    U = rand();
    i = 0;
    p = exp(-lambda);
    F = p;
    while(true)
        %rand<F, put value and stop, else continue
        if (U < F)
            Result2(1,k) = i;
            break;
        else
            p = (lambda*p)/(i+1);
            F = F + p;
            i = i + 1;
        end
    end
end
% ------------------------------------------------------------- %
% Display outputs.
u=rand(N,1);
z=poissinv(u,lambda);
figure;
histogram(Result1);
title('Histogram using the Bernoulli Trials Method')
ylabel('count of average number of arrivals per hour');
xlabel('Average number of arrivals per hour');
hold on
histogram(z)
figure;
histogram(Result2);
title('Histogram using the Inverse Transform Method')
ylabel('Count of average number of arrivals per hour');
xlabel('Average number of arrivals per hour');
hold on
histogram(z)
```

Output:


Histogram using the Bernoulli Trials Method
count of average number of arrivals per hour
Average number of arrivals per hour
0
100
200
300
400
500
600
700
800
70
80
90
100
110
120
130
140
150
160
170



Discussion and Analysis:

Theoretical values are overlayed with the histograms we got. We can see that, theoretical values are in correspondence with the practical ones.

For Poisson counting using Bernoulli trial method, he approximation is finer when I increase the number of sub intervals from 10000 to100000. Hence, as we make the subinterval smaller and smaller, the approximation gets better and better. From the histogram, X-axis represents the average number of cars arriving per hour and Y-axis represents the count. The inverse transform method can be used in practice as long as we are able to get an explicit formula for F inverse of (y) in closed form. Also, as the number of simulations increases, the distribution tend to perfection(more values for corresponding to 120.)

Poisson model is best suited for sampling the data which is observed its event pattern in time. If events occur randomly and independently, *at a constant rate* (in time), then the *count* of these events per unit time will conform to a Poisson distribution.

From the histograms, it is clear that average number of arrivals of car per hour is distributed around 120 but not too far from 120. Also, there are no bars corresponding to the values below 90 and above

150.This indicates that our values are aligned with respect to the theoretical number of arrivals of cars per hour.

Question 3.

Description:

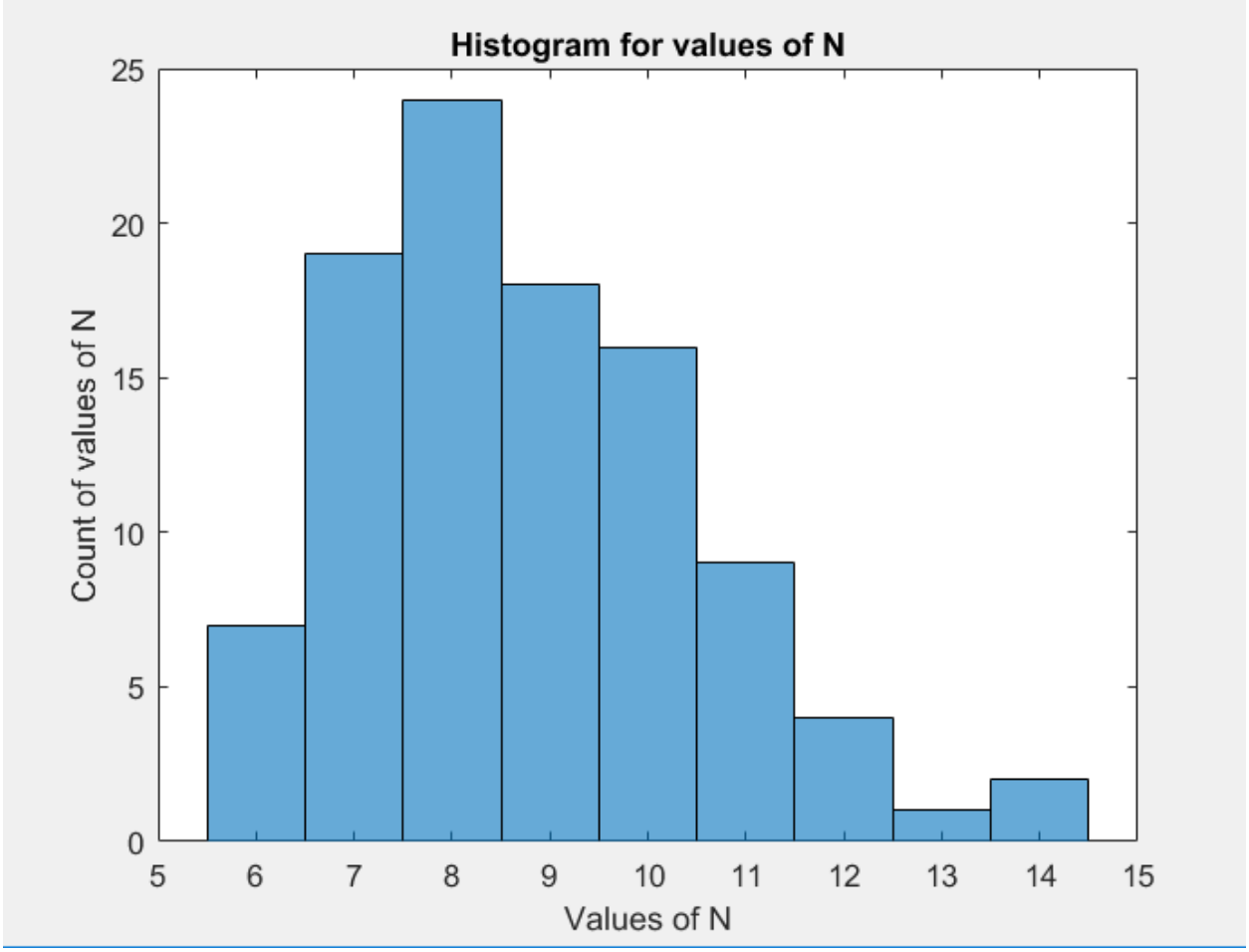
The loop is run for n (number of samples) times. Sum and count are initialized to 0 in each run of the loop. While loop is run until the collective sum of random numbers generated is 4. The counts are being stored in a result array. Histogram of the result is obtained using histogram() command and mean is calculated using mean() function.

CODE:

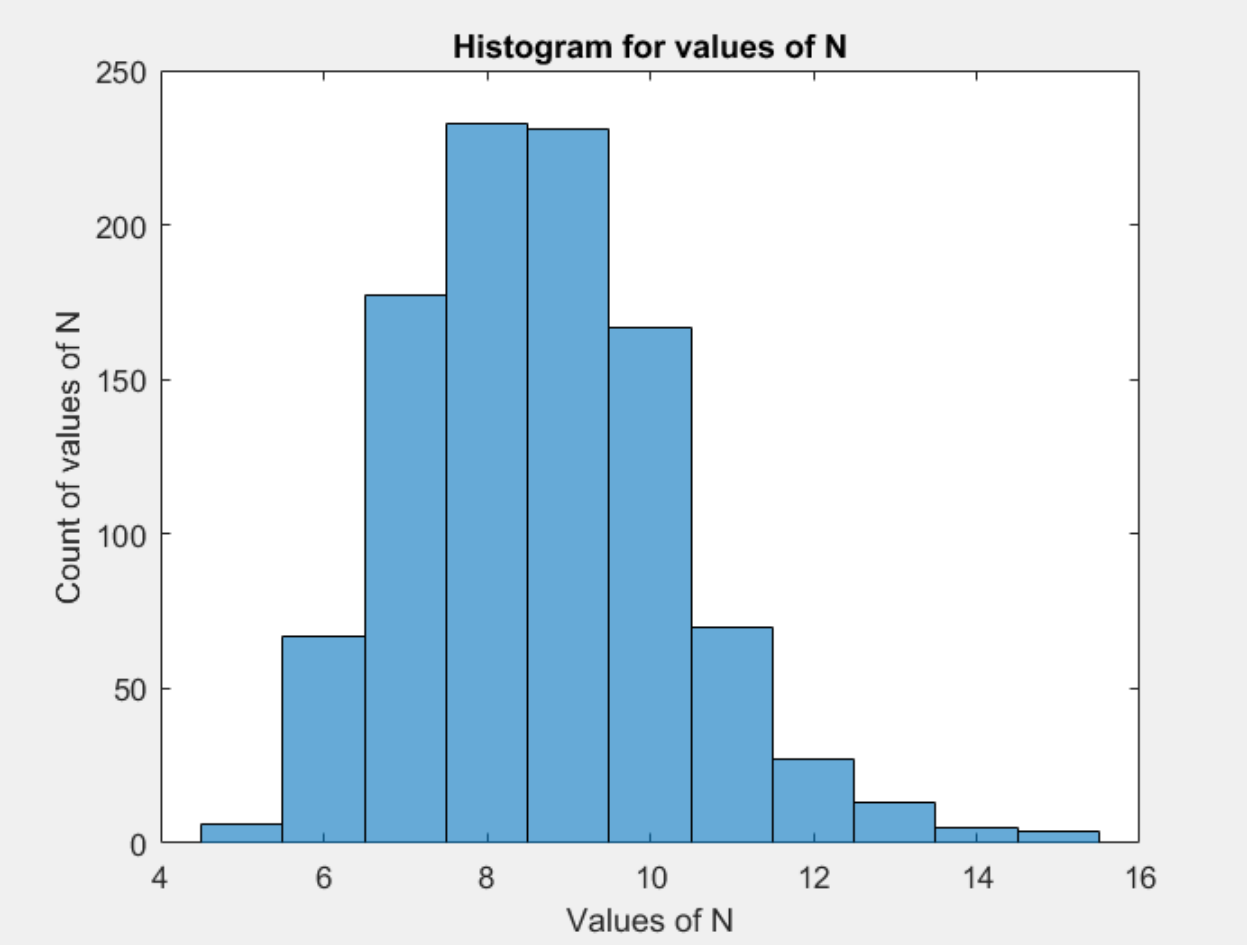
```
% Akshay Deepak Hegde  USC ID: 8099460970  %
% ---------------------------------------------------------------- %
% Project #3-Samples and statistics , EE511: Spring 2017, Due: 7th Feb
% ---------------------------------------------------------------- %
% To define a random variable N = min{n: ?i=1,..,n Xi>4}
% The smallest number of uniform random samples whose sum greater than 4
% To generate histograms using 100,1000,10000 samples for N.
% ---------------------------------------------------------------- %
clc;
clear;
close all;
% ---------------------------------------------------------------- %
n=10000;%number of samples
for i=1:n
    sum=0;%initializations
    count=0;
    while(sum<=4)%loop is run untill collective sum exceeds 4
        count=count+1;
        u=rand();
        sum=sum+u;
    end
    Result(1,i)=count; %Record the result
end
% ---------------------------------------------------------------- %
figure;
histogram(Result);
title('Histogram for values of N');
xlabel('Values of N');
ylabel('Count of values of N');
Mean = mean(Result);
Output=['The mean is ',num2str(Mean),' for ',num2str(n),' samples '];
disp(Output)
```

Output:

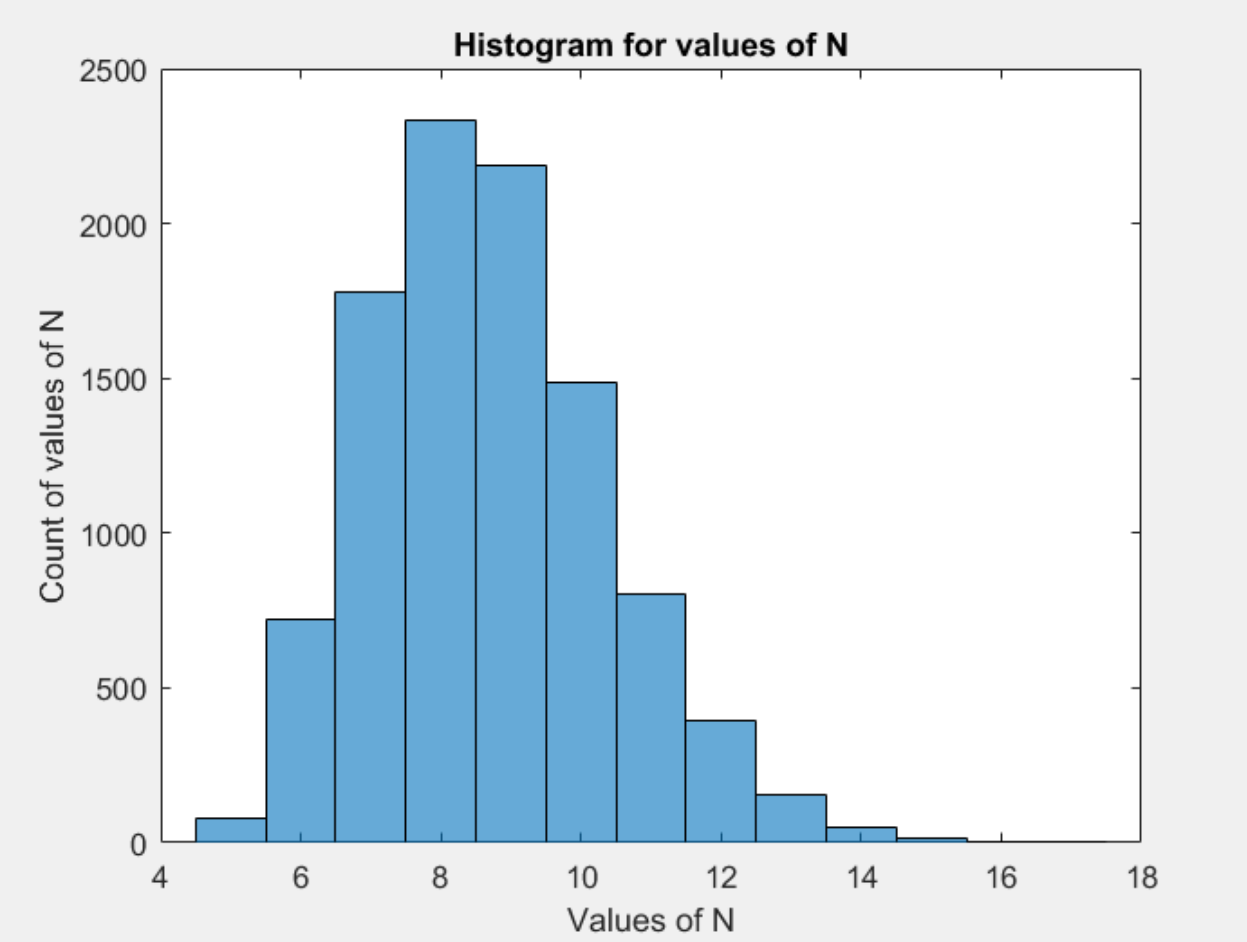
The mean is 8.79 for 100 samples



The mean is 8.677 for 1000 samples



The mean is 8.6902 for 10000 samples



Analysis and Discussion:

E[X] can be calculated using the equation 1/p. Since it assumes the same range irrespective of number of samples, we can say that mean is independent of number of trials.

The above problem is an example of Geometric random variable.

The geometric Random variable involves independent Bernoulli trials with a success probability (say 'p') and if 'X' is the number of trials until the first success,

then: $P[X = n] = p(1 - p)^{n-1}$ for all n=>1

Question 4.

Description:

For finding out the value of p, I equated the sum of all the values in the sequence to 1.

From this, we can say that p=1, through this p(j) is found out.

A function Nj(n,P) is designed which is multiple times in the program. Nj compares the random variable U with the an array containing the cumulative sum of all the variables of P. It returns the indexes of all the values where the cumulative sum is greater than U. Indexes are saved in the X. find() function is used to generate N60.

Code:

```
        % Akshay Deepak Hegde  USC ID: 8099460970  %
% ------------------------------------------------------------- %
% Project #3-Samples and statistics , EE511: Spring 2017, Due: 7th Feb
% ------------------------------------------------------------- %
% To produce a sequence with given condtion
% To generate histograms
% to define r.v and estimate mean and variance
% ------------------------------------------------------------- %
clc;
clear;
close all;
% ------------------------------------------------------------- %
sample = zeros(1,[]);

for j = 1:60
sample = [sample,1/j];
end
s = sum(sample);
p = 1/s;

for j = 1:60
P(j) = p/j;
end
X = Nj(n,P);
% ------------------------------------------------%
histogram(X,60);
title('Histogram for the distribution of p ');
ylabel('Frequency of occurence');
xlabel('Value of sample');
```

```
N60 = find( X == 60,1);
for j = 1:N
X = Nj(n,P);
N60 = [N60 find(X==60,1)];
end

mean = sum(N60)/N;
variance = sum((N60-mean).^2)/1000;
disp(mean); disp(variance); end

function X = Nj(n,P)
X = zeros(1,n);
for i = 1:n
   U = rand();
   index = find(U < cumsum(P));
   X(i) = min(index);
end
```

Output:

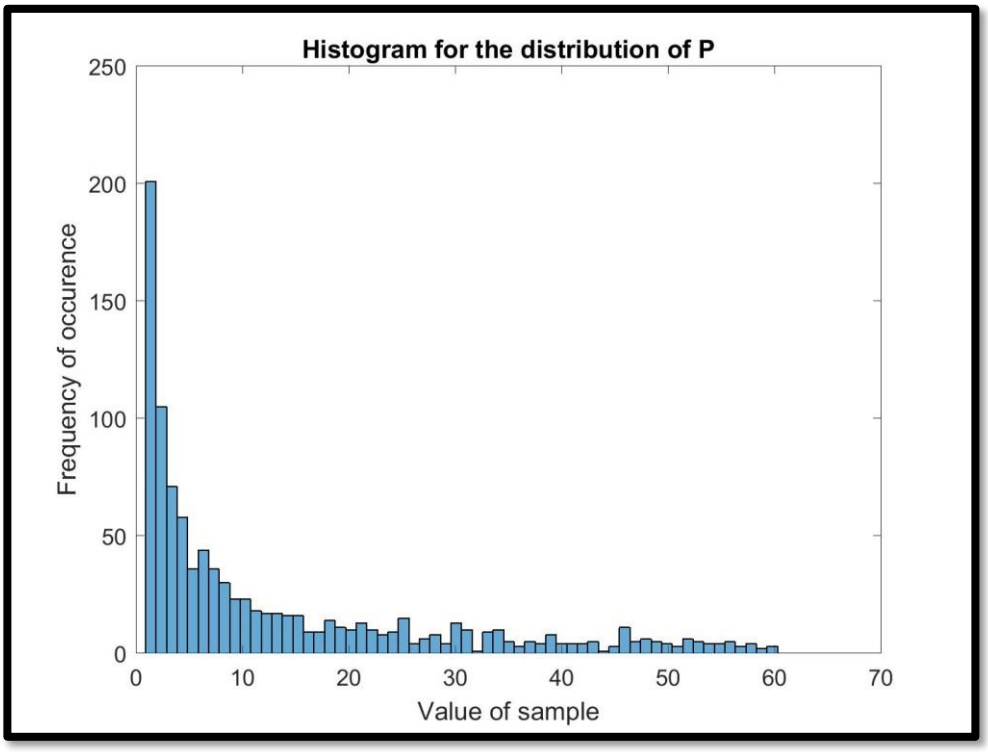
```
>> minutehand(1000,1000)

The Experimental mean is  241.0590


The Theoretical mean is  279.7922


The Experimental variance is   4.6782e+04


The Theoretical variance is   7.8563e+04
```



Analysis:

P(1) is the most and p(60) is the most. Theoritical and experimental values are close to each other.

Question 5.

Accept-Reject method is used for the particular problem.

Sequence P is given for 10 values and 0's are appended for next 10 values. Sequence Q is also given.

The value of constant, c is calculated as max(p/q) = 0.15/0.05 = 3

So, the theoretical efficiency of the system = 1/c = 33%

The approach here is to generate random variables (U) and compare them to a 'P' of randomly generated index (j) i.e. 3*U<=P(j)/0.05.

c*qj forms an envelope over the plot of p(j) and we check if the random variable c*q(j) falls under the distribution of p(j). If it does, we accept the variable and remove the variables that don't satisfy the comparison. This way we can generate the probability mass function for p(j).

Code:

```
            % Akshay Deepak Hegde  USC ID: 8099460970  %
% ------------------------------------------------------------- %
% Project #3-Samples and statistics , EE511: Spring 2017, Due: 7th Feb
% ------------------------------------------------------------- %
% To use accept-reject method for sampling
% To generate histograms and overlay target distribution.
% To calculate sample mean, variance and effeciency
% ------------------------------------------------------------- %
clc;
clear;
close all;
% ------------------------------------------------------------- %
p = [6 6 6 6 6 15 13 14 15 13 0 0 0 0 0 0 0 0 0 0]/100;%initializations
q = [5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5]/100;
N=10000;
for i = 1:N
    k = 0;
    while 1,k = k + 1; % while true, run the loop
        j = 1 + floor(20*rand);  % Get Uniform j
        U = rand();
        if ((3*U) <= p(j)/0.05 ) % Accept p(j) if U<p(j)/c, q(j)= 0.05
            X(i) = j; % recoed the results
            C(i) = k; % count array
            break;
        end
    end
end
% ------------------------------------------------------------- %
% Display the results
figure;
histogram(X);
hold on;
plot(p*N);
ylabel('Count of occurance of random variables');
xlabel('Random variables');
legend('Histogram of Accept-Reject sampling','Distribution of p');
meanX = mean(X);
varX = var(X);
meanC = mean(C);
Output1=['Experimental mean is ',num2str(meanX)];
disp(Output1)
Output2=['Experimental variance is ',num2str(varX)];
disp(Output2)
Output3=['Mean of count array is ',num2str(meanC)];
disp(Output3)
eff = N/sum(C);
Output4=['Experimental effeciency is ',num2str(eff)];
disp(Output4)
eff2=1/3;
Output5=['Theorectical Efficiency (1/c): ',num2str(eff2)];
disp(Output5)
```

Output:

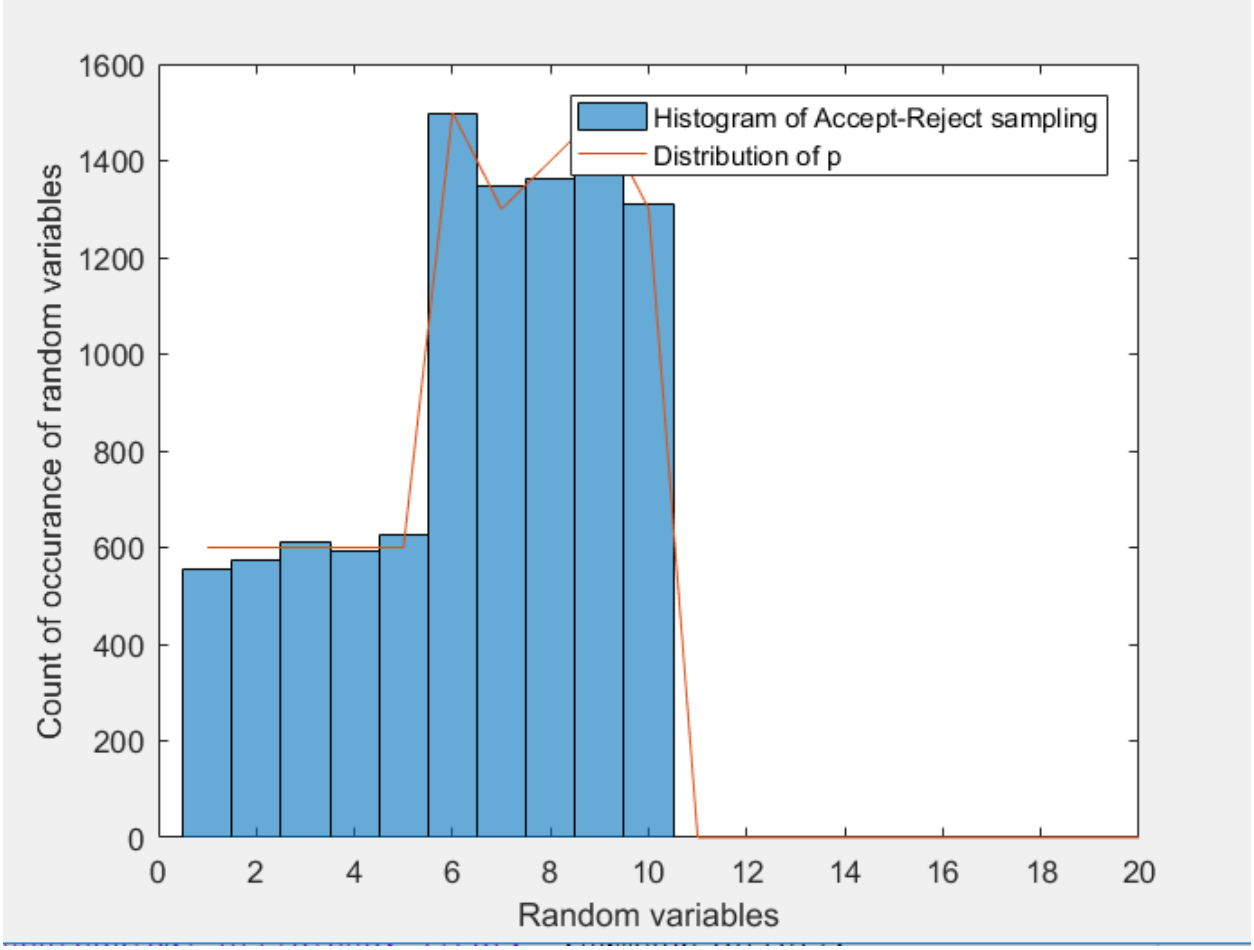
Experimental mean is 6.5195

Experimental variance is 7.0327

Mean of count array is 2.9996

Experimental effeciency is 0.33338

Theorectical Efficiency (1/c): 0.33333



Analysis and Discussion:

From the figure, it is evident that histogram and plot are close.

Theoretical values are calculated using,

$$E[X]=\sum_{k=1}^{n} x \cdot p(x)$$

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

Which gives the values as,

Theoretical mean – 6.4800

Theoretical variance – 7.1896

Above values are close to experimental ones.

Also, both the theoretical and experimental efficiency are close to 33%.

Hence, the accept-reject method is a good approximation of p(j)