Project #2 - Samples and Statistics

Li Yicheng[*]

USCID:7827077047
email: l.y.c.liyicheng@gmail.com
USC Viterbi of Engineering

I. microchips test

I. Problem Description

(a) There is a batch of 125 microchips, if the distributor find any defective microchip in the chosen microchip, the batch of microchips would be rejected. Suppose there are 6 defective units in the lot, what is the probabilty that the distributor will reject the lot if five microchips are tested.
(b) What is the fewest number of microchips that the distributor should test to reject this lot 95% of the time.

II. Experiment(a)

A function called reject_prop is generated to show the probability that the distributor would reject the lot with test_amount microchips were tested.

Algorithm:

*1.Let the defective microchip be No.1 to No.6.
*2.Generate test_amount samples uniformly in 1 to 125
*3.If there exist index form 1 to 6, reject times plus one
*4.Repeat the above routine 20000 times, calculate the amount of rejections in the 20000 times, to derive the rejection probability

III. core code: filename: reject_prop.m

```
function reject_p = reject_prop(test_amount)
%usage: reject_p = reject_prop(test_amount)
%given the test amount, output the reject probability
% input : test_amount : the test amount
% output: reject_p: reject probability
reject = 0;
for i = 1 : 20000 %to be accurate enough, do 20000 trial
    samples = randperm(125,test_amount);%select test amount microchips
    if ( sum(samples < 7) > 0) %let defective chips be No.1 to No.6 , if any of these number is tested just reject
        reject = reject +1; %reject ,detected such defective chip
    end
end
reject_p = reject / 20000;
```

[*] github link: https://github.com/IAMLYCHEE/EE511-PROJECT1

IV. Result(a) & Analysis:

If 5 chips were tested, we query: reject_p = reject_prop(5)
we get:
reject_p = 0.2181
This means, in this simulation, if 5 microchips were tested, the distributor has a probability near 22% to reject the lot.
Let us compare this to the theoretical value. Because the probabilty that the distributor does not reject the lot is easier to compute, we compute that number. Using combination knowledge, we choose 5 from 199 good chips and we choose 5 from all microchips.
$\mathbb{P}(not\ reject) = \frac{\binom{119}{5}}{\binom{125}{5}} = 0.7787$
$\mathbb{P}(reject) = 1 - \mathbb{P}(not\ reject) = 0.2213$
Compare to our result 0.2181, they are very close.

V. Experiment(b)

We make use of the function generated in experiment(a), we just need to change the input parameter, let it increase, until the probabilty reach 95%, then we record the test_amount.

Algorithm:

*1.Calculate the reject probabilty under test_amount
*2.If the probability is less than 95%, increase the test_amount
*3.else record the test_amount

VI. core code: filename: solution1.m

```
test_amount = 5;
while(1)
    p = reject_prop(test_amount);
    if p < 0.95
        test_amount = test_amount + 1;
    else
        break
    end
end
```

VII. Result(b) & Analysis

It is not a function file, just run it.
we get: test_amount = 49
that means, if the distributor test 49 microchips, there is a probability of 95% that he would reject the lot.
Also, let us to compare this to our theoretical value, this time the equation should be:
$\mathbb{P}(reject) = 1 - \frac{\binom{119}{x}}{\binom{125}{x}} = 0.95$

Actually, in matlab, if you compute like $\binom{119}{59}$, it may not give accurate answer,(the number is too large!), so we simplfy the formular.

$$\frac{\binom{119}{x}}{\binom{125}{x}} = \frac{\frac{119!}{(119-x)!x!}}{\frac{125!}{(125-x)!x!}} = \frac{119!(125-x)!}{125!(119-x)!} = \frac{(125-x)(124-x)(123-x)(122-x)(121-x)(120-x)}{125\cdot124\cdot123\cdot122\cdot121\cdot120} = \frac{\binom{125-x}{6}}{\binom{125}{6}}$$

$x = 50, \mathbb{P}(reject) = 0.9571$
$x = 49, \mathbb{P}(reject) = 0.9534$
$x = 48, \mathbb{P}(reject) = 0.9495$
$x = 47, \mathbb{P}(reject) = 0.9452$
Therefore, the theoretical value fit our experiment result.
If the distributor would like to reject the lot 95% of the time, he should take 48 or 49 chips to test.

II. Number of events in specified intervals

I. Probelm Description

(a)Suppose 120 cars arrive at a freeway onramp per hour on average, simulate one hour of arrivals to the freeway ramp.Perform a Bernoulli trial to indicate a car arrival within small time interval.
(b)Sampling directly form an equivalent Possion distribution by using inverse transform method.Generate a histogram for the number of arrivals using this method.

II. Experiment(a)

The requirement is to derive the amount of cars at some position per hour,suppose we divide one hour into 5000 intervals, the probabilty of one car is within this interval is $\mathbb{P}(within\ the\ interval) = \frac{120}{5000}$ and we count whether there is a car in one interval by generating a random number form the u(0,1) distribution.

Algorithm

*1.Generate N random variables from uniform distribution [0,1], N is the number of time interval.
*2.Count the number of these variable whose value is smaller than 120/N. Record the sum as one trial result.
*3.Repeat 1,2 to get more trial results and generate the histogram.

III. Core Code: filename: solution2.m (part of code)

```
function solution2(trial_budget,lambda)
%usage: compute the B-trial result, inverse sampling method, and theoretical value,fora
    possion distribution event
%input: trial times, the expected average of occurence of event (lambda)

%--------------------------------------------------------------------
%theoretical p.m.f of Possion distribution
P_the = zeros(1,lambda*1.5); %record the P theoretical value
x = 1 : lambda*1.5;
for k = 1 : lambda*1.5
    P_the(k) = trial_budget*lambda^k * exp(-lambda)/factorial(k);
end
```

```
%------------------------------------------------------------
%Bernoulli trial method
success_amount = zeros(trial_budget,1);
N = 5000;
P = lambda / N;
for i = 1: trial_budget
    success_amount(i) = sum(rand(N,1) < P);
end
subplot(1,2,1)
histogram(success_amount,lambda*1.5,'BinLimits',[1,lambda*1.5]);
title('Bernoulli trial')
xlabel('number of cars')
ylabel('appear times')
%set the same x-axis [1,lambda*1.5] for comparing
hold on
plot(x,P_the,'r--o');
hold off
legend('B-trial','theoretical');
```

IV. Experiment(b)

The inverse tranform method.Inverse transformation sampling takes uniform samples of a number u u between 0 and 1, interpreted as a probability, and then returns the largest number x x from the domain of the distribution $P(X)$ such that $P(-\infty < X < x) \leq u$. For example, possion has a distribution of $\mathbb{P}(X = i) = \frac{e^{-\lambda}\lambda^{i}}{i!}$, we know that $\sum_{i=1}^{\infty} \mathbb{P}(X = i) = 1$,first generate a random varible u from uniform distribution [0,1], and then we just accumulate $\mathbb{P}$ from $\mathbb{P}(X = 1)$ until $\sum_{1}^{N} \mathbb{P}(X = i)$reach u, record the N.

Algorithm:

*1.Generate a standard uniform distribution variable u, initiate P and accumulate sum F to be P(X=1).
*2.If u > F, F + P_{next} , else break, record the accumulated index i.
*3.Generate i within trial budget.
the above generate one sample using inverse sampling method

V. Core Code :

file: gen_one_possion.m

```
function i = gen_one_possion(lambda)
%usage function x = gen_one_possion(lambda)
u = rand(1);
i = 0;
p = exp(-lambda);
F = p;
while u > F
    p = lambda * p / (i+1);
    F = F + p; %accumulate
    i = i+1;
end
```

file: solution2.m (inverse sample part)

```
%------------------------------------------------------------------------
%inverse tranform method
amount_gened = zeros(trial_budget,1); %generated amount
for i = 1: trial_budget
    amount_gened(i) = gen_one_possion(lambda);
end
subplot(1,2,2)
histogram(amount_gened,lambda*1.5,'BinLimits',[1,lambda*1.5]);
title('inverse-transform method');
xlabel('number of cars')
ylabel('appear times')
hold on
plot(x,P_the,'r--o');
hold off
legend('inverse-sample','theretical');
```

VI. Result (a),(b) & Analysis

query: solution2(10000,120)

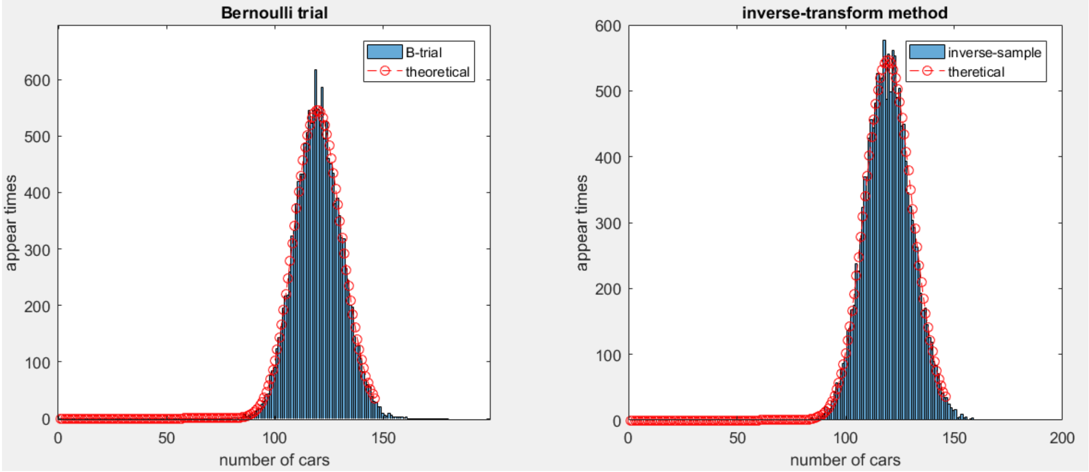


Figure 1: *outcome histogram*

From the figure, it is easy to observe that both the Bernoulli trial and the inverse sampling method reached the theoretical outcome. Both all even had more probability to get 120 cars.

For Bernoulli trial, theoretically,

$\binom{n}{k}p^k(1-p)^{n-k} \to e^{-r}\frac{r^k}{k!}$ when $n \to \infty$

In these case, the n is 5000, and the p is 12/5000, nearly zero. So the above convergence is suitable in this case.

For the inverse sampling method.

Let F be a continuous cumulative distribution function, and let F^{-1} be its inverse function.

$F^{-1}(u) = \inf\{x \mid F(x) \geq u\} \qquad (0 < u < 1).$

Proof:

$\Pr(F^{-1}(U) \leq x)$
$= \Pr(U \leq F(x)) \qquad \text{(applying } F, \text{ to both sides)}$
$= F(x) \qquad\qquad\quad \text{(because } \Pr(U \leq y) = y)$
reference:https://en.wikipedia.org/wiki/Inverse_transform_sampling
Therefore the inverse sampling method is good for generating sample numbers at random from any probability distribution given its cumulative distribution function.

III. AMOUNT OF S.T.D UNIFORM SAMPLE NEEDED TO REACH SUM OF 4

I. Problem Description

(a): Generate a random variable as the smallest number of uniform random samples whose sum is greater than four.
(b): Generate a histogram using 100,1000, and 10000 samples for N.

II. Experiment

To generate random variable N is straigtforward:

Algorithm:

*1:Initiate a zero sum.
*2:Generate a random number from standard uniform distribution.
*3:Add the number to the sum. loop time increase 1
*4: if the sum is larger than four, record the loop times and break,
else goto 2.

filename: generateN.m

```
function N = generateN(thre)
%usage N = generateN(thre)
%input : thre : the threshold for sum, in this experiment , it is 4
%output: the smallest number of sum (u. random samples) > 4
sum = 0;
N = 0;
while sum < thre
    sum = sum + rand(1);
    N = N+1;
end
```

To generate histogram with defined number of N samples: and derive the expectation, just repeat the above function specific times.Use the following function to generate one histogram with specific amount of N.

filename: Nhistogram.m

```
function E = Nhistogram(eval_budget,thre)
%use: E = Nhistogram(eval_budget,thre)
% plot the histogram using eval_budget for N, threshold thre
result = zeros(eval_budget,1);
```

```
for i = 1 : eval_budget
    result(i)= generateN(thre);
end
histogram(result);
xlabel('number of uniform samples')
ylabel('appear times')
E = mean(result);
```

To get the result just use the following querys:

```
subplot(1,3,1)
E_100=Nhistogram(100,4);
 title('100')
subplot(1,3,2)
E_1000=Nhistogram(1000,4);
 title('1000')
subplot(1,3,3)
E_10000=Nhistogram(10000,4);
 title('10000')
```

III. Result(a)(b) & Analysis

Graph:

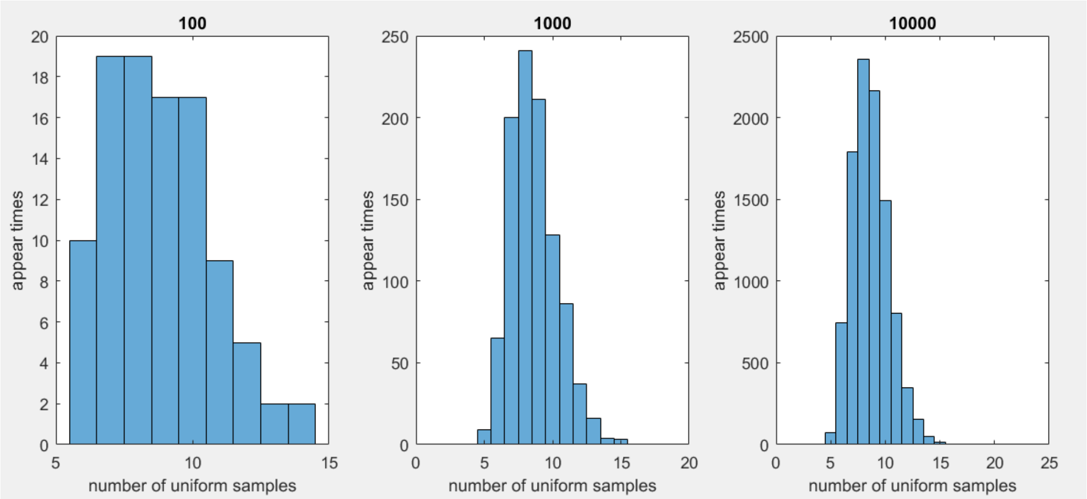


Figure 2: *Histograms for 100,1000,10000 N samples*

Expectations
E_100 = 8.750, E_1000 = 8.6570, E_10000 = 8.6652
Analysis
When it comes to a larger number of N, there tends to exist a convergence in the figure because it is obviously that the 1000 graph and the 10000 graph looks similar. So let us we analyse theoretical why the shape is look like this.
Actually, it is really difficult to generate a curve which can be described in a continous function to fit the histogram, because it only has ten more bars and it is really a discrete distribution. My thought to analyse the problem is as follows:

*1. Assume N numbers from standard uniform distribution have the sum larger than 4
*2. So k numbers is less then 4/N, while others larger than 4/N, but their average is larger than 4/N(else we can not get sum larger than 4).
*3. Therefore, the probabilty for N amount uniform numbers sum larger than 4 is $\mathbb{P}(N) = \binom{N}{k}(\frac{4}{N})^k(1-\frac{4}{N})^{N-k}$
*4. Assume k = N/2, that is half of these number is larger than 4/N, another half is smaller than 4/N
*Take a look at this distribution: $\mathbb{P}(N) = \binom{N}{N/2}(\frac{4}{N})^{N/2}(1-\frac{4}{N})^{N-N/2}$

The following code are generated to analyse the above distribution function:
filename: P_n.m

```
function p = P_n(N)
p = nchoosek(N,floor(N/2))*(4/N)^(N/2)
    *(1-4/N)^(N-(N/2));
end
```

filename: analysis3.m

```
p = zeros(1,20);
for n = 5: 17
    p(n) = P_n(n);
end
stem(p)
title('analysis')
xlabel('N')
ylabel('probability')
```

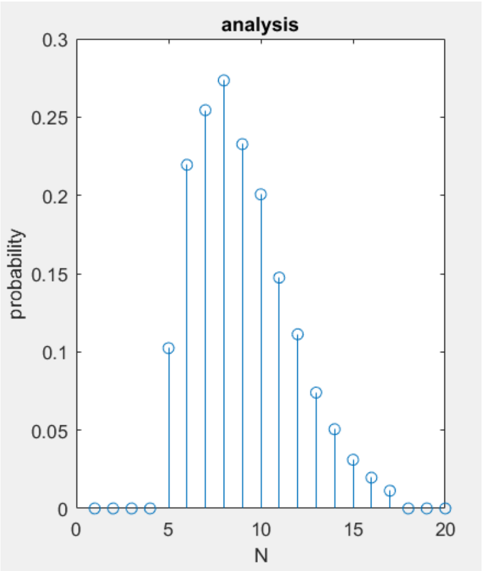


Figure 3: *Theoretical analysis*

Theoretical graph

See, the simulation in figure 2 looks like this kind of distribution, the right side of this distribution is actually binomial distribution. Because from the formular, it is easy to derive it has the same decreasing velocity as the right side of binomial distribution.

IV. Some kind of sequence analysis

I. Problem Description

(a): we have a sequence X_k where $p_j = \frac{p}{j}$ for $j = 1,2,3,\cdots 60$.Generate a historam showing the behaviour of this sequence.
(b): Define the random variable $N_j = min\{k : X_k = j\}$.Simulate sampling from N_60.

II. Experiment(a):

This is like the inverse sampling method. First, [0,1] is devide into 60 parts with length $p/60, p/59, \cdots, p$ seperately. Then we generate a vairable from standard uniform distribution. And then find out what region does this varible locate.It is easier to be understand by the following algorithm.

Algorithm:

*1.Normalize p, let $\sum_{1}^{60} p_j = 1$
*2.Generate a number u from standard uniform distribution
*3.Set accumulate number acc = p/X, X= 60
*4.if u > acc , stop, record X,
else X = X-1,acc=acc+p/X;
goto 4 again.
filename: generateX.m

```
function X = generateX(p)
%usage: result = generateN_4(p)
%input : the constant p
X = 60;
randNumber = rand(1);
acc = p/X;
while (randNumber > acc && X >=1)
    X = X-1;
    acc=acc+p/X;
end
```

filename: solution4.m (X histogram part)

```
%--------------------------------------------------------------
%generate the constant p to make it normalized
sum =0;
for i = 1 : 60
    sum = sum +1/i;
end
p = 1/sum;
%---------------------------------------------------------------
%generate the histogram, do 10000 trials
%basic idea is kind of similar to inverse sampling
result = zeros(10000,1);
for k = 1 : 10000
    result(k) = generateX(p);
end
histogram(result)
title('histogram for {X_k}')
xlabel('index j')
ylabel('appear times in the trial')
```

III. Result(a)&Analysis

figure:

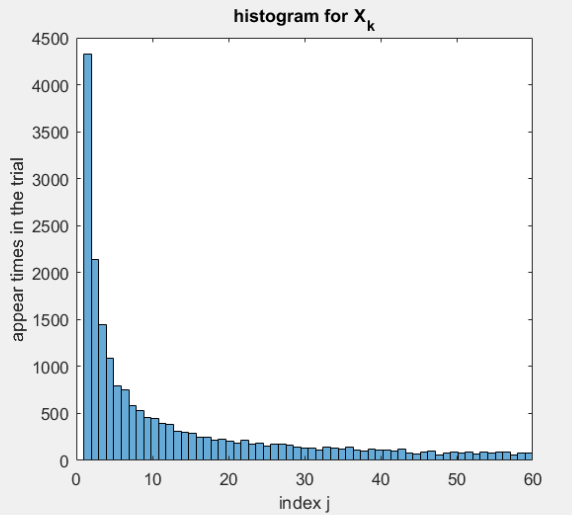


Figure 4: *X distribution*

Analysis Using inverse sampling method works well in this situation, because we can see the second bar has half height as the first bar and the third bar has 1/3 height to the first bar which is corresponding to p,p/2,p/3....p/60. Although there seems to be some 'noise' when it come to p/45 to p/60(not smmoth enough), given their relatively low probability, such noise can be ignored.

IV. Experiment(b)

The above we have already define a function to generate one X, the requirement in this part is straight forward, if the X is not 60, keep generate X, else stop and record the generate times.

Algorithm

*1.Init times be 0
*2.Generate one X
*3.X is 60, stop, else times plus one, goto 2
*4.Repeat 1,2,3 several times.
filename:solution4.m (N_60 part)

```
%-----------------------------------------------------------
% N_60 experiment
result_N60 = zeros(1000,1);
for i = 1 : 1000
    N = 0;
    while generateX(p) ~= 60
        N = N +1;
    end
    result_N60(i) = N;
end
figure
histogram(result_N60)
title('histogram for {N_{60}}')
xlabel('the least amount of trials to reach N_{60}')
ylabel('the number of such amount of trials')
Expectation = mean(result_N60);
variance = var(result_N60);
%-------------------------------------------------------------
```

V. Result(b) & Analysis

Figure & Data

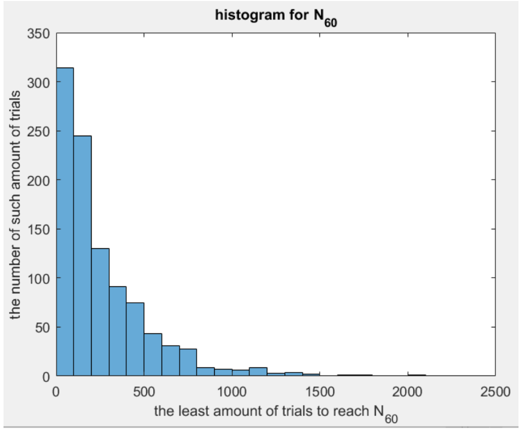


Figure 5: *N_60 Experiment*

Expectation = 280.2020
variance = 7.0729e+04.

Analysis

This event is actually conform to a geomerty distribution with the probability hitting 60 is p/60, not hitting, $\frac{60-p}{60}$. Therefore, theretical expectation is $\frac{1}{p}$, and the variance $\frac{(1-p)}{p^2}$. In this problem, $p = \frac{p_0}{60}$, and p_0 is the normalized one we generated in Experiment(a), which is 0.2137. Therefore, $\mathbb{P}(hit\ 60) = 0.2137/60$.

Theoretical expectation: $1/\mathbb{P}(hit\ 60) = 60/0.2137 = 280.7674$

Experiment expectation: 280.2020.

Theoretical variance: $(1 - \mathbb{P}(hit\ 60))/\mathbb{P}(hit\ 60)^2 = 7.8563e + 04$

Experiment variance: 7.0729e+04

To sum up, in this experiment, the simulation data conform the theoretical data. Therefore we are doing a experiment which conforms the geometry distribution.

V. Accept-reject method

I. Problem Description

Use accept-reject method to sample from p_j by sampling from another uniform distributionq_j. p_j conforms $p_{1\ to\ 5} = 0.06, p_6 = p_9 = 0.15, p_7 = p_10 = 0.13, p_8 = 0.14$.

II. Experiment:

(a): Understanding the accept-reject method. The rejection sampling method generates sampling values from a target X with arbitrary probability density function f(x) by using a proposal distribution Y with probability density g(x). The idea is that one can generate a sample value from X by instead sampling from Y and accepting the sample from Y with probability $f(x)/(Mg(x))$, repeating the draws from Y until a value is accepted.
(b): First, we derive the constant M, M need to be make sure that $Mg(x) > f(x)$ should be true for every x, therefore, we take the max of $(f(x)/g(x))$ to determine the M. Then we using the accept-reject method to derive $f(x)$.
(c:) Overlay the target distribution according to the accepted amount, for example we accept 200 samples in Experiment(b), then the target distribution is [200*0.06,200*0.06,...200*0.14], and compute the efficiency.

Algorithm:

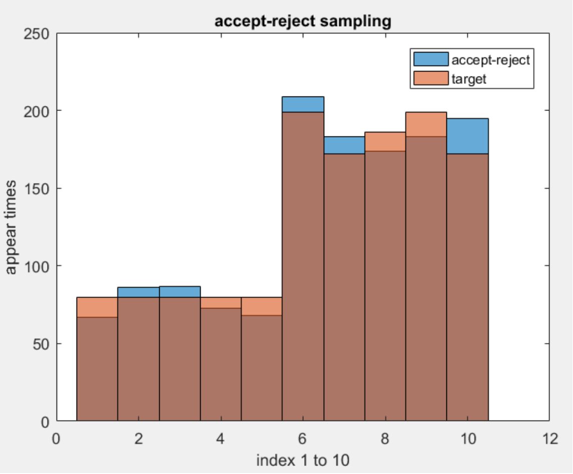
*1. Set distribution q to be $q_1 = 0.1, q_2 = 0.1, \cdots, q_{10} = 0.1$
*2. Compute the constant M
*3. Generate a rand number i from 1 to 10, generate a number u from std U[0,1]
*4. if u< p(i)/Mq(i) , accept and record, else reject.
*5. Repeat 3,4 for trail_budget times.
filename: solution5.m

```
function [mean_exp,mean_the,var_exp,var_the,efficiency] = solution5(trial_budget)
%--------------------------------------------------------------------------
%usage:[mean_exp,mean_the,var_exp,var_the,efficiency] = solution5(trial_budget)
%trial_budget: the number of times you would like to query for generating
%the sample using accept-reject method
%generate the sample mean and variance using accept-reject method
%generate the theoretical mean and variance
%generate the efficiency of the accept-reject method
%Li Yicheng
%--------------------------------------------------------------------------

%accept-reject method
p = [0.06*ones(1,5),0.15,0.13,0.14,0.15,0.13];
%the distribution of p_j
q = ones(1,10)*0.1;
%the distribution to sample from
M = max(p./q);
%derive the coeffience to let all Mq >= p
i=1;
for k = 1 : trial_budget
    y = randi(10);
    u = rand(1);
    if u < p(y)/(M*q(y))
        accept(i) = y;
        i=i+1;
    end
end
%plot part
histogram(accept');
xlabel('index 1 to 10')
ylabel('appear times')
%--------------------------------------------------------------------------
%the target distribution
amount = length(accept);
%to compare the two graph we need to set them have same total sample amount
index = 1;
for i = 1 : 10
    std_distri(index:index + round(amount*p(i))-1)= i*ones(round(amount*p(i)),1);
    %add the 'i's into the sequence, for example in the first loop add
    %(amount * 0.06 ) ones into the sequence from 1 to amount * 0.06;
    index = index + round(amount*p(i));
end
%overlay with the theoretical
hold on
histogram(std_distri);
title('accept-reject sampling')
legend('accept-reject','target')

%the sample mean,variance and the theoretical values
mean_exp = mean(accept);
mean_the = (1:10)*p';
```

```
var_exp = var(accept);
var_the = var(std_distri);
%the efficiency
efficiency = amount/trial_budget;
```

III. Result & Analysis

query: [m_exp,m_the,v_exp,v_the,efficiency]= solution5(2000)



m_exp=6.3273
m_the=6.4800
v_exp=7.7341
v_the=7.2164
efficiency=0.6630
Theoretical efficience $1/M = 1/(max(p_j/q_j)) = 1/(0.15/0.1) = 1/1.5 = 0.6666$

Figure 6: *Experiment result & Target distribution*

All the data derived from experiment is close to theoretical value. Therefore, accept-reject sampling is also a good method to sample from some ditribution by sampling from another distribution. The only drawback may happen is just the efficiency. Because sometimes we may get a lot of unwanted data. So the choice of M should be considered if using this method.