Monte **C**arl**o** **S**imulatio**n**
Assignment-**4**

Shalu Rungta
(140123033)
Department Of Mathematics

March 9, 2016

Q 1 Use the Box-Muller method and Marsaglia-Bray method to do the following:

(a) Generate a sample of 100, 500 and 10000 values from $N(0,1)$. Hence find the sample mean and variance.

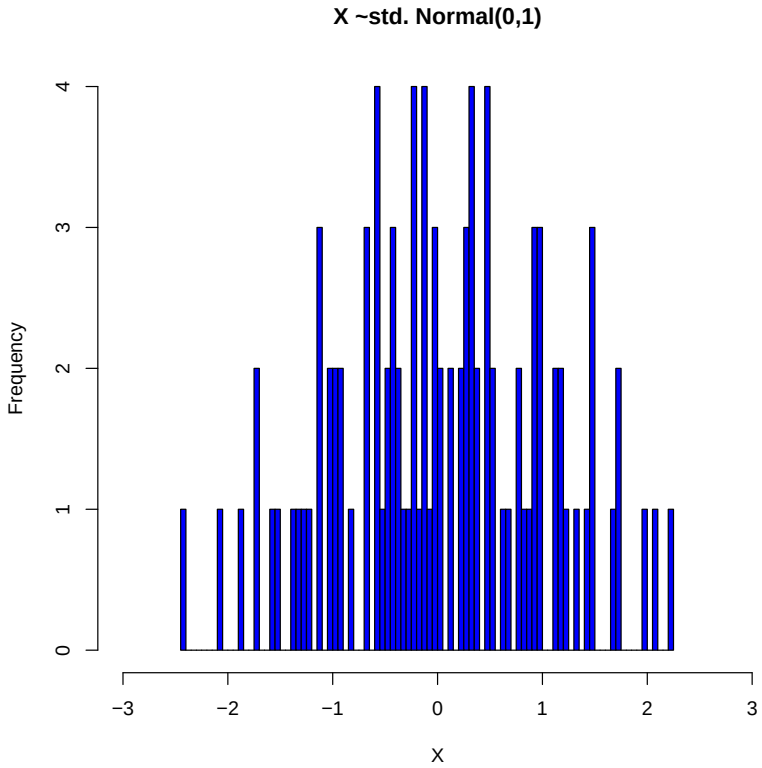
(b) Draw histogram in all cases.

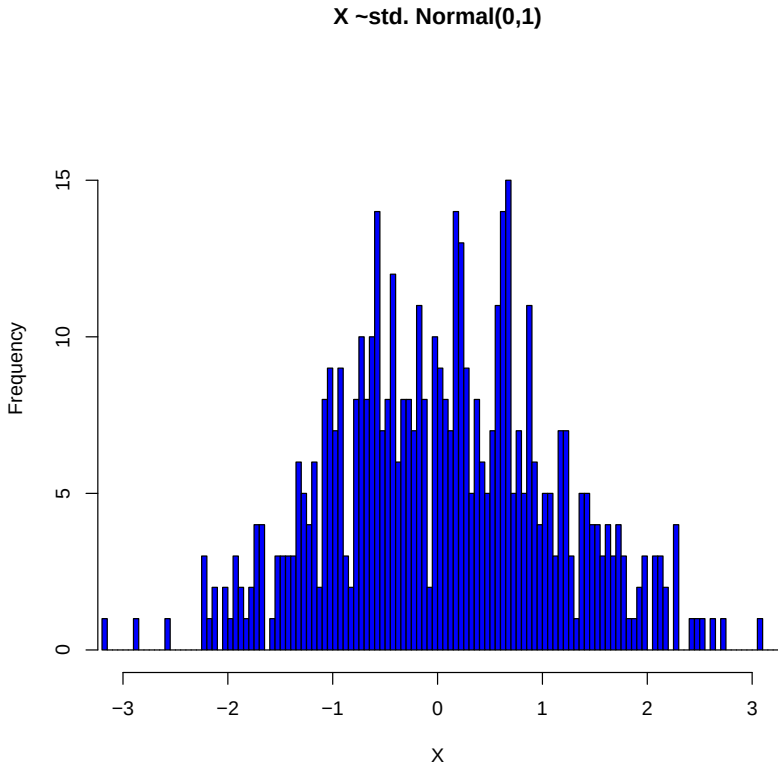
Solution

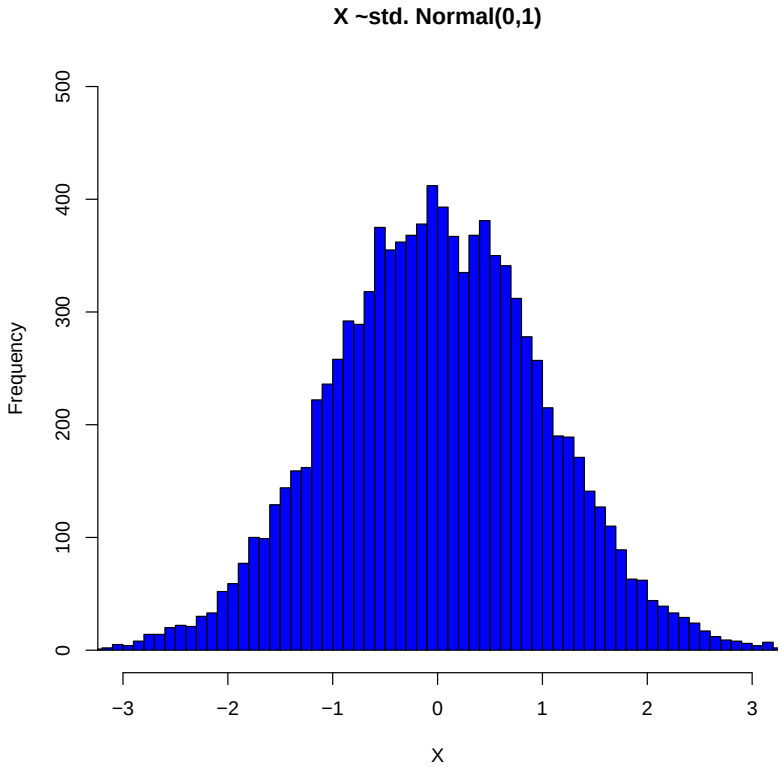
(a)Code **Using Box-Muller method**

```
1
2  x=runif(50);
3  y=runif(50);
4  j=1;
5  z=NULL;
6  for(i in 1:50)
7  {z[j]=((-2*log(x[i]))^0.5)*cos(44*y[i]/7);
8  z[j+1]=((-2*log(x[i]))^0.5)*sin(44*y[i]/7);
9  j=j+2;
10 }
11 print(z);
12 m=mean(z);
13 v=var(z);
14
15 cat("Mean=",m,"\n");
16 cat("variance=",v,"\n");
17
18 hist(z,main="X ~std. Normal(0,1)", xlab="X", ylab="
       Frequency",xlim=c(-3,3), ylim=c(0,4),breaks=100, col
       ="blue")
```

Similarly,sample of 500 and 10,000 can be generated by changing the value of n in the above code.


X ~std. Normal(0,1)
Frequency
0
1
2
3
4
−3
−2
−1
0
1
2
3
X


X ~std. Normal(0,1)
Frequency
0
5
10
15
−3
−2
−1
0
1
2
3
X

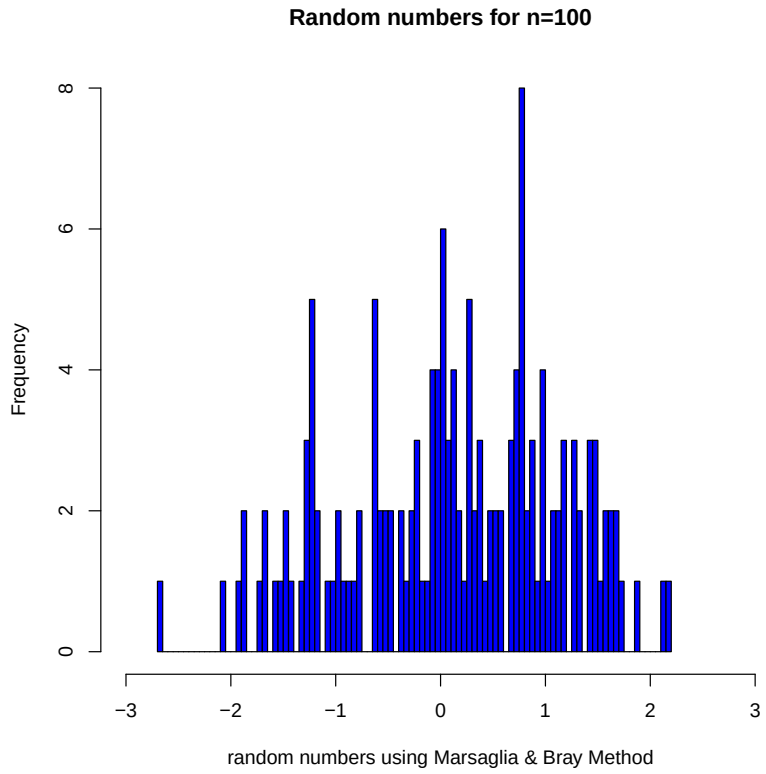


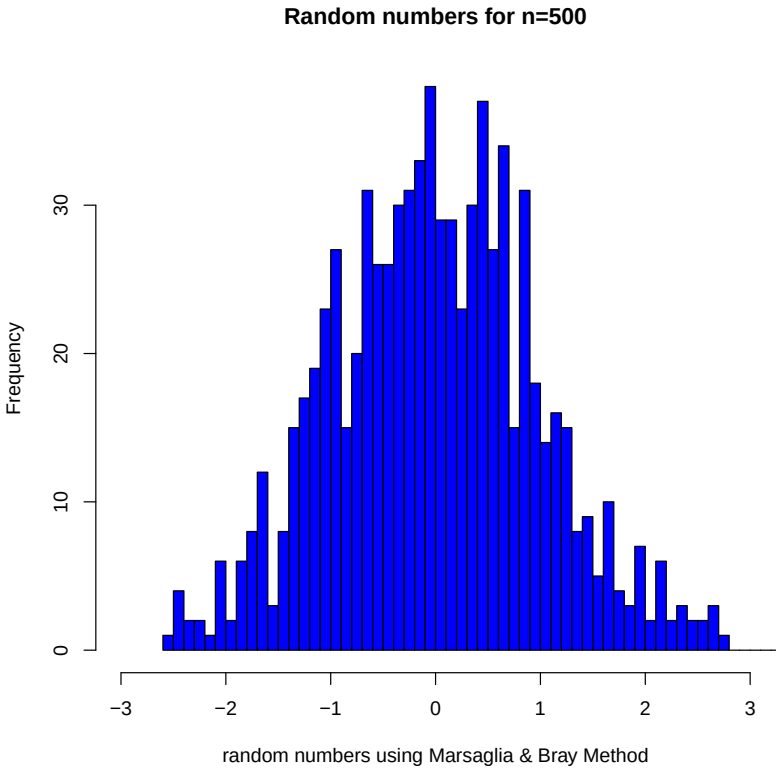
(a)Code **Using Marsaglia-Bray method**

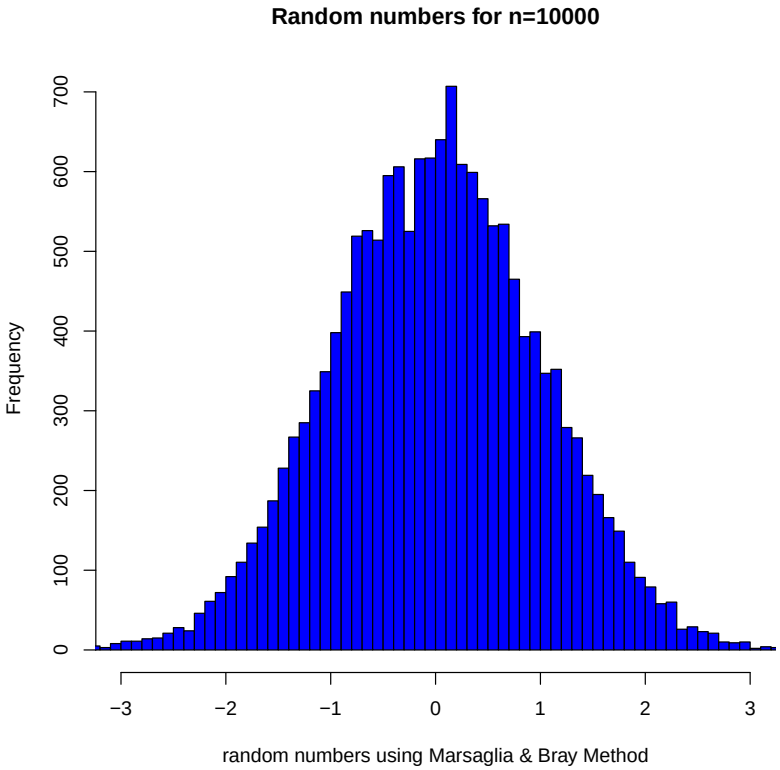
```
f<-function(x)
{
        return (sqrt(((-2)*log(x))/x));
}
n<-500
u1<-runif(n)
u2<-runif(n)
y<-vector('numeric')
z<-vector('numeric')
v1<-vector('numeric')
v2<-vector('numeric')
j<-0
for(i in 1:n)
{
        u1[i]<-(2*u1[i])-1
        u2[i]<-(2*u2[i])-1
        x<-(u1[i]^2)+(u2[i]^2)
        if(x>1)
        {
                next
        }
        j<-j+1
        v<-f(x)
        y<-c(y,v)
        v1<-c(v1,u1[i])
        v2<-c(v2,u2[i])
}
```

```
for(i in 1:j)
{
        z1<-v1[i]*y[i]
        z2<-v2[i]*y[i];
        z<-c(z,z1)
        z<-c(z,z2)
}
m<-mean(z)
v<-var(z)
cat("Mean=",m,"\n")
cat("Variance=",v,"\n")
hist(z,col=c("blue"),breaks=70,xlim=c(-3,3),xlab="
    random numbers using Marsaglia & Bray Method",main=
    paste("Random numbers for n=500"))
```

Similarly,sample of 100 and 10,000 can be generated by changing the value of n in the above code.




Random numbers for n=500
Frequency
0
10
20
30
−3
−2
−1
0
1
2
3
random numbers using Marsaglia & Bray Method


Random numbers for n=10000
Frequency
0
100
200
300
400
500
600
700
−3
−2
−1
0
1
2
3
random numbers using Marsaglia & Bray Method

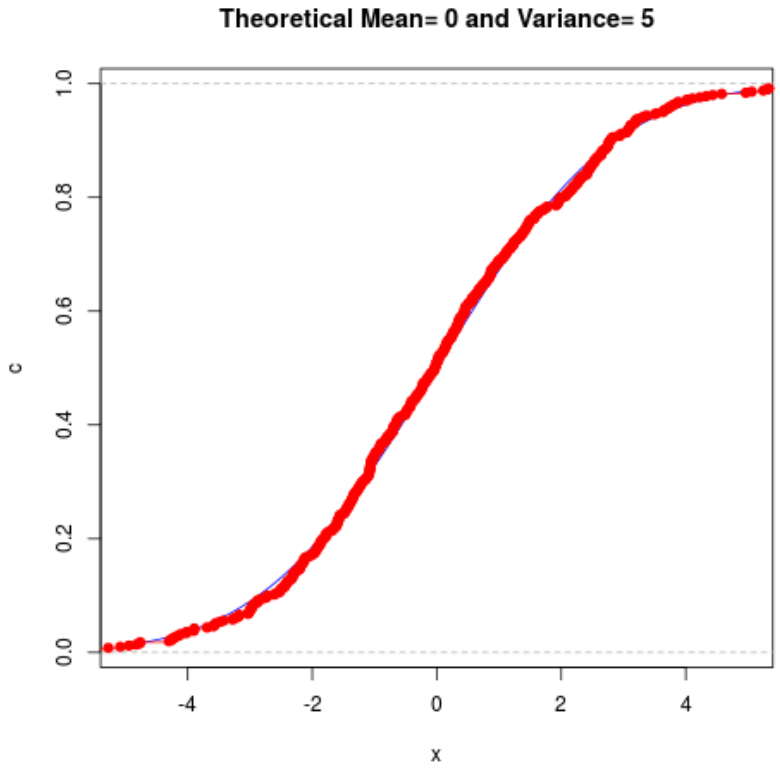
Question 2: Now use the above generated values to generated samples from $N(\mu = 0; \sigma^2 = 5)$ and $N(\mu = 5; \sigma^2 = 5)$. Hence plot the empirical(from sample with size 500) distribution function and theoretical distribution function in the same plot. (Use R/ you should also try making the step function in C)

Solution

Given μ=0

Code **Using Box-Muller method**

```
f<-function(mu,sigma,z)
{
return (mu+sigma*z)
}
x=runif(250);
y=runif(250);
j=1;
z=NULL;
k=NULL;
for(i in 1:250)
{z[j]=((-2*log(x[i]))^0.5)*cos(44*y[i]/7);
z[j+1]=((-2*log(x[i]))^0.5)*sin(44*y[i]/7);
k[j]=f(0,(5)^0.5,z[j]);
k[j+1]=f(0,(5)^0.5,z[j+1]);
j=j+2;
}
x<-seq(-5,5,length=250);
c<-pnorm(x,0,sqrt(5))
png(paste("2a.png"))
plot(x,c,col="blue",type="l",main=paste("Theoretical
    Mean=",0,"and Variance=",5))
lines(ecdf(k),col="red")
```

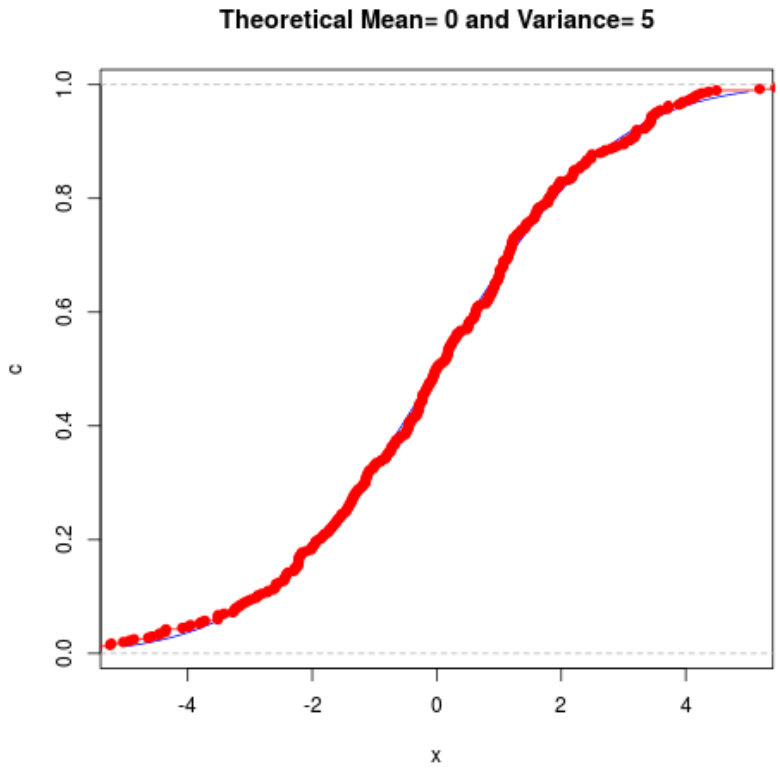


Code **Using Marsaglia-Bray method**

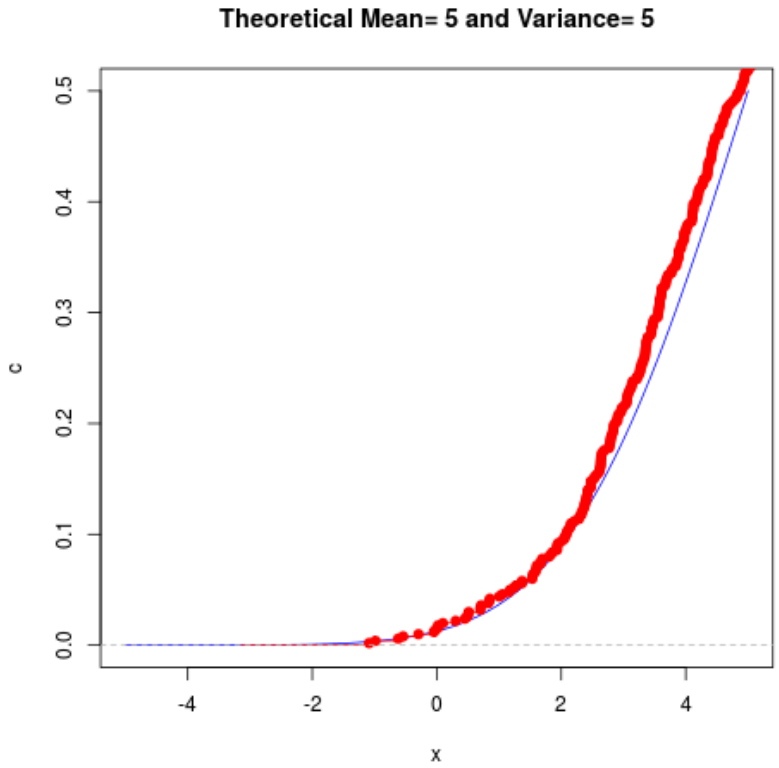
```
f<-function(x)
{
        return (sqrt(((-2)*log(x))/x));
}
g<-function(mu,sigma,z)
{
return (mu+sigma*z)
}
n<-500
u1<-runif(n)
u2<-runif(n)
y<-vector('numeric')
z<-vector('numeric')
v1<-vector('numeric')
v2<-vector('numeric')
k<-vector('numeric')

j<-0
for(i in 1:n)
{
        u1[i]<-(2*u1[i])-1
        u2[i]<-(2*u2[i])-1
        x<-(u1[i]^2)+(u2[i]^2)
        if(x>1)
        {
                next
        }
```

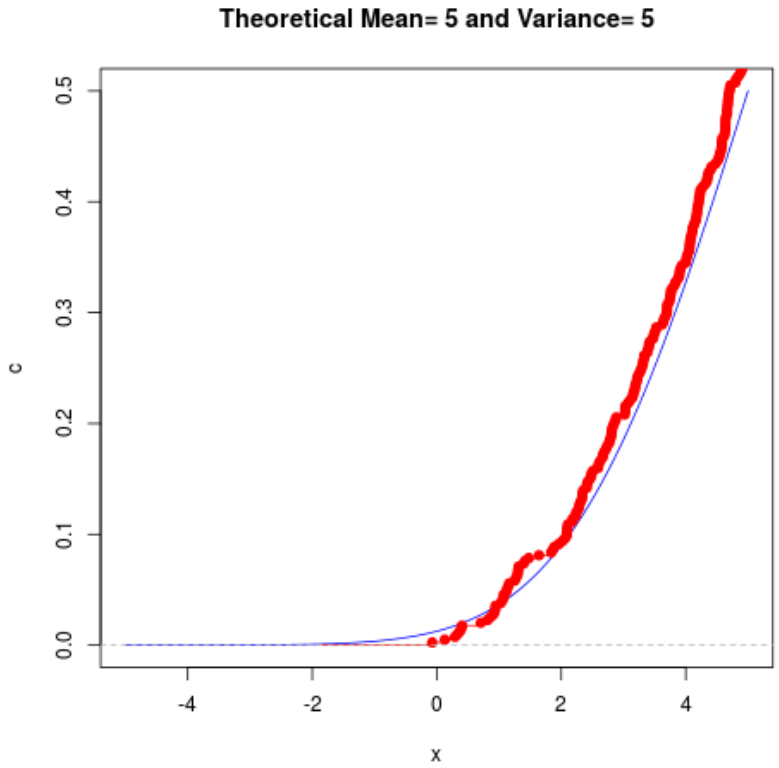
```
28            j<-j+1
29            v<-f(x)
30            y<-c(y,v)
31            v1<-c(v1,u1[i])
32            v2<-c(v2,u2[i])
33 }
34 for(i in 1:j)
35 {
36            z1<-v1[i]*y[i];
37            z2<-v2[i]*y[i];
38            z<-c(z,z1)
39            z<-c(z,z2)
40 }
41 for(i in 1:j)
42 {   s<-g(0,sqrt(5),z[i])
43     k<-c(k,s)
44 }
45 x<-seq(-5,5,length=250);
46 c<-pnorm(x,0,sqrt(5))
47 png(paste("2b.png"))
48 plot(x,c,col="blue",type="l",main=paste("Theoretical
       Mean=",0,"and Variance=",5))
49 lines(ecdf(k),col="red")
```



Given μ=5
Using Box-Muller method



Using Marsaglia-Bray method


Theoretical Mean= 5 and Variance= 5
c
0.0
0.1
0.2
0.3
0.4
0.5
-4
-2
0
2
4
x

Question 3: Keep a track of the computational time required for both the methods. Which method is faster?)

Solution

Code using**Box-Muller method**

```
f<-function()
{
x=runif(250);
y=runif(250);
j=1;
z=NULL;
for(i in 1:250)
{z[j]=((-2*log(x[i]))^0.5)*cos(44*y[i]/7);
z[j+1]=((-2*log(x[i]))^0.5)*sin(44*y[i]/7);
j=j+2;
}

m=mean(z);
v=var(z);

cat("Mean=",m,"\n");
cat("variance=",v,"\n");

hist(z,main="X ~std. Normal(0,1)", xlab="X", ylab="
    Frequency",xlim=c(-3,3), ylim=c(0,15),breaks=100,
    col="blue")
}
t1<-system.time(f(), gcFirst = TRUE);
print(t1);
```

Code using**Marsaglia-Bray Method**

```
g<-function()
{
f<-function(x)
{
        return (sqrt(((-2)*log(x))/x));
}
n<-500
u1<-runif(n)
u2<-runif(n)
y<-vector('numeric')
z<-vector('numeric')
v1<-vector('numeric')
v2<-vector('numeric')
j<-0
for(i in 1:n)
{
        u1[i]<-(2*u1[i])-1
        u2[i]<-(2*u2[i])-1
        x<-(u1[i]^2)+(u2[i]^2)
        if(x>1)
        {
                next
        }
```

```
24            j<-j+1
25            v<-f(x)
26            y<-c(y,v)
27            v1<-c(v1,u1[i])
28            v2<-c(v2,u2[i])
29  }
30  for(i in 1:j)
31  {
32            z1<-v1[i]*y[i]
33            z2<-v2[i]*y[i];
34            z<-c(z,z1)
35            z<-c(z,z2)
36  }
37  m<-mean(z)
38  v<-var(z)
39  cat("Mean=",m,"\n")
40  cat("Variance=",v,"\n")
41  hist(z,col=c("blue"),breaks=70,xlim=c(-3,3),xlab="
        random numbers using Marsaglia & Bray Method",main=
        paste("Random numbers for n=500"))
42  }
43  t1<-system.time(g(), gcFirst = TRUE);
44  print(t1);
```

Observation: We note that for low values of n, Box- Muller seems to perform better than the Marsaglia-Gray method. However for large values of n, this trend falls apart.

This strange behaviour can be explained by looking at the both methods' implementation. We note that the major time consuming factor in Box-Muller is the calculation of trigonometric (sine and cosine) functions while that in case of Marsaglia - Bray (which uses acceptance-rejection principle) is the number of times loop runs (which is always more than the total number of random variables to be generated.)

When n is small , latter factor dominates the former and thus Box-Muller seems to take less time. However in case of large 'n' the scenario is different and the former factor dominates the latter.

We should also keep in mind that the time calculated in case of Marsaglia-Gray is calculated for all the values (irrespective of whether they are accepted or not) which is not the exact time to be speaking precisely. Therefore the correct time observed should be (acceptance probability $\times$ time observed).

However nevertheless, **we can conclude that Marsaglia-Bray method is faster than the Box Muller method**. (Note that the Box-Muller algorithm has been improved by Marsaglia in a way that the use of trigonometric functions can be avoided.It is important, since computation of trigonometric functions is very time-consuming.)

Question 4: For the Marsaglia-Bray method keep track of the proportional of values rejected. How does it compare with 1-$\frac{\pi}{4}$

Solution

Code

```
f<-function(x)
{
        return (sqrt(((-2)*log(x))/x));
}
n<-500
u1<-runif(n)
u2<-runif(n)
y<-vector('numeric')
z<-vector('numeric')
v1<-vector('numeric')
v2<-vector('numeric')
j<-0
count<-0
for(i in 1:n)
{
        u1[i]<-(2*u1[i])-1
        u2[i]<-(2*u2[i])-1
        x<-(u1[i]^2)+(u2[i]^2)
        if(x>1)
        {
      count<-count+1
                next
        }
        j<-j+1
        v<-f(x)
        y<-c(y,v)
        v1<-c(v1,u1[i])
        v2<-c(v2,u2[i])
}
for(i in 1:j)
{
        z1<-v1[i]*y[i]
        z2<-v2[i]*y[i];
        z<-c(z,z1)
        z<-c(z,z2)
}
povr<-count/n

m<-mean(z)
v<-var(z)
cat("Mean=",m,"\n")
cat("Variance=",v,"\n")
cat("proportion of values rejected=",povr,"\n")
hist(z,col=c("blue"),breaks=70,xlim=c(-3,3),xlab="
    random numbers using Marsaglia & Bray Method",main=
    paste("Random numbers for n=500"))
```

Theoretical probability of rejection = 1-$\frac{\pi}{4}$ = 0.2146018
As the number of observations (random numbers generated) in-

creases , the experimental rejection probability gets closer and closer to the theoretical rejection probability.