Documentation: Assignment 6

Abhinav Gupta 150123001

10 March 2017

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.

Code for Box-Muller method (Code for 10,000 values): R

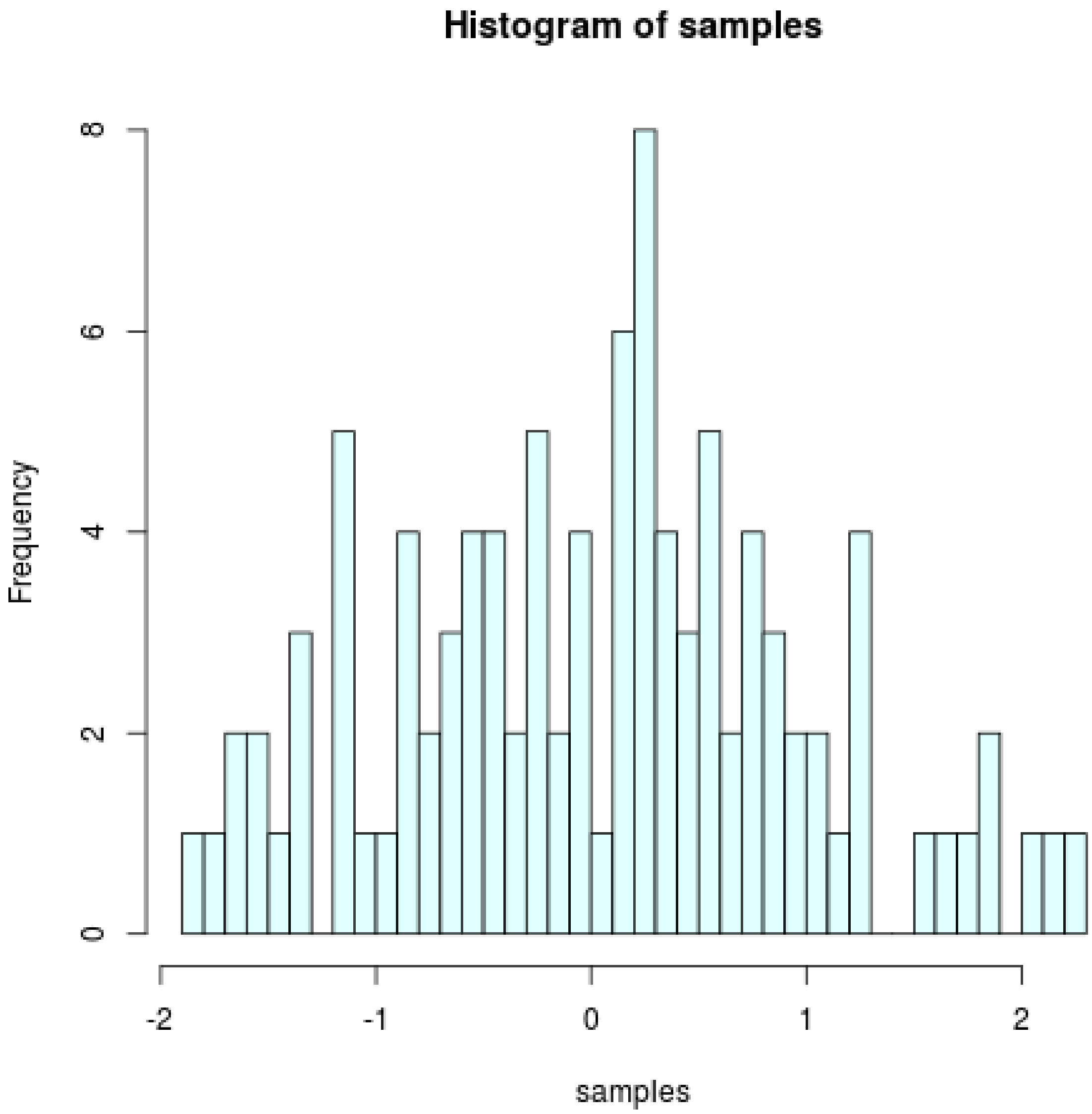
```
radius<-function(u){
    return (-2*log(u))
}
arg<-function(u){
    return (2*pi*u)
}
# change count to 100, 500 and 10,000 as per requirement
count<-10000
samples1<-c()
samples2<-c()
for(i in 1:count/2){
    u1<-runif(1)
    u2<-runif(1)
    samples1[i]<-sqrt(radius(u1))*cos(arg(u2))
    samples2[i]<-sqrt(radius(u1))*sin(arg(u2))
}
samples<-c(samples1,samples2)
png("question1-10000-muller.png")
hist(samples, breaks=50 ,col="light cyan",plot=TRUE)
```

Output:

Mean: 0.002538348

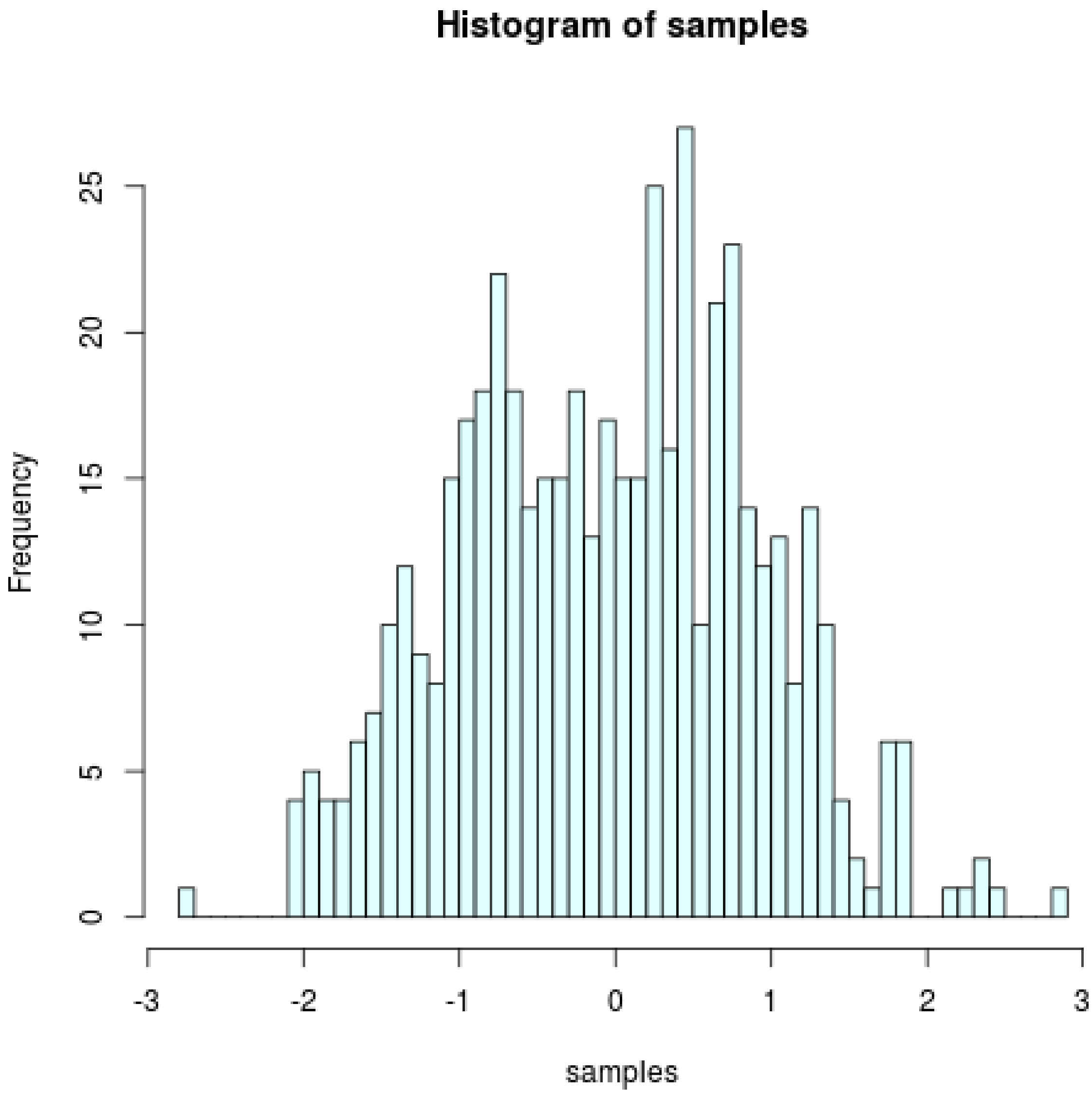
Variance: 1.008891

Observation: **Graph: 100 values:**



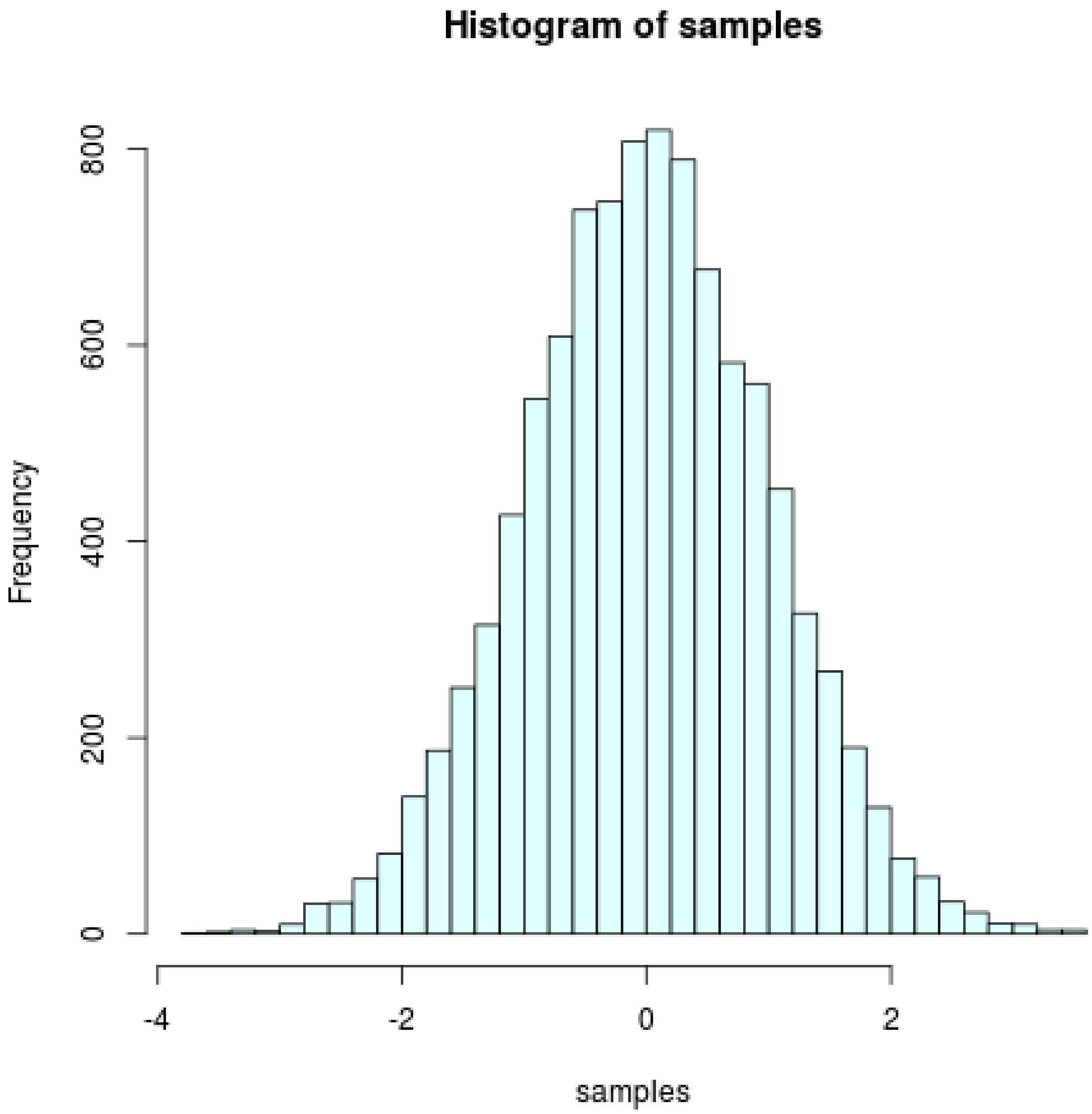
(a) *using R*

Graph: 500 values:



(b) *using R*

Graph: 10000 values:



(c) *using R*

Code for Marsaglia-Bray method (Code for 10,000 values): R

```
square<-function(u1,u2){
    return (u1**2+u2**2)
}
intermediate<-function(x){
    return (sqrt(-2*log(x)/x))
}
# change count to 100, 500 and 10,000 as per requirement
count<-10000

samples1<-c()
```

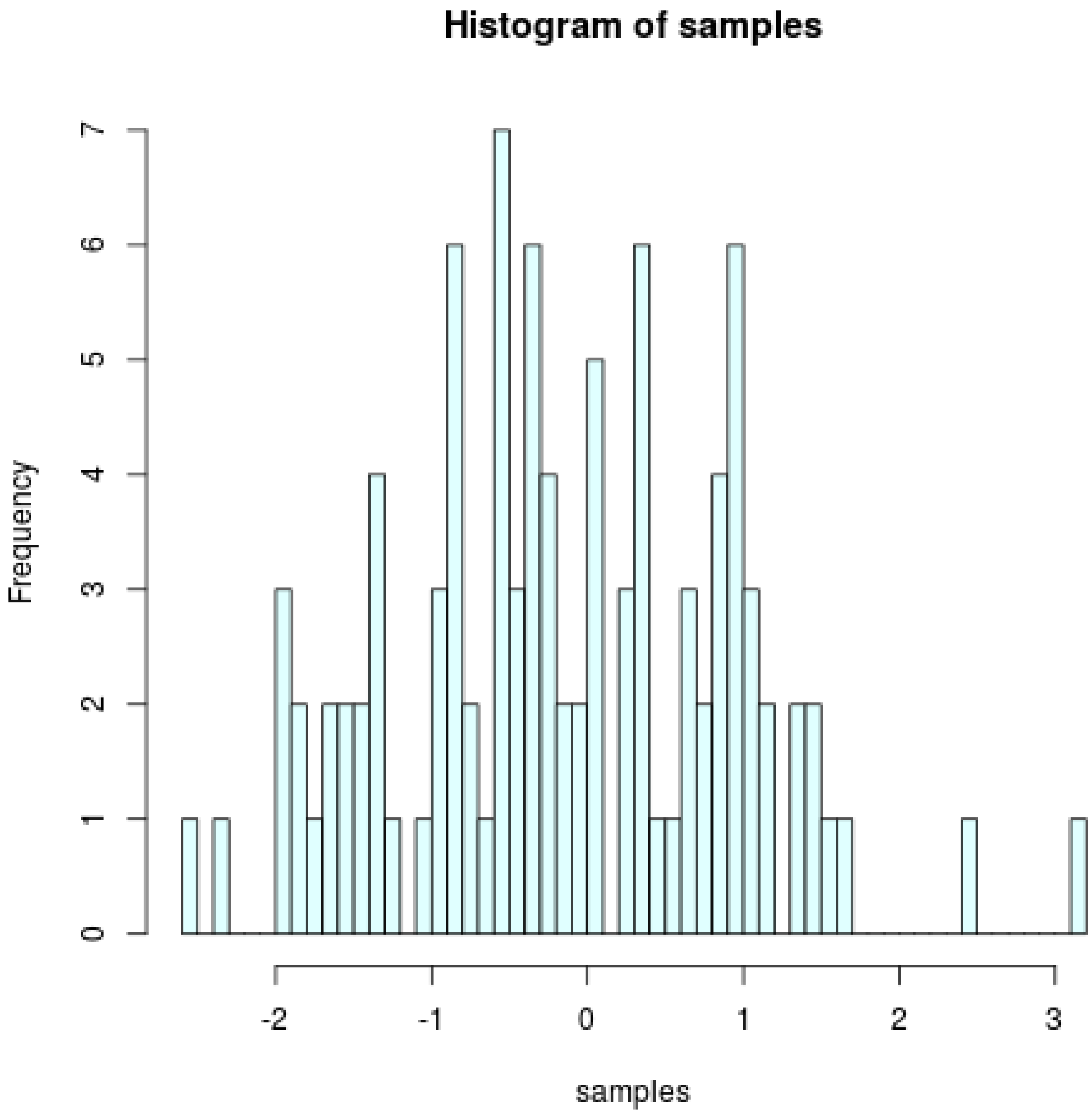
```
samples2<-c()
i<-1
while(1){
    u1<-runif(1)
    u2<-runif(1)
    u1<-2*u1-1
    u2<-2*u2-1
    x<-square(u1,u2)
    if(x>1){
        next
    }
    y<-intermediate(x)
    samples1[i]<-u1*y
    samples2[i]<-u2*y
    i<-i+1
    count<-count-2
    if (count==0){
        break
    }
}
samples<-c(samples1,samples2)
png("question1-10000-bray.png")
hist(samples, breaks=50 ,col="light cyan",plot=TRUE)
```

Output:

Mean: 0.009609266

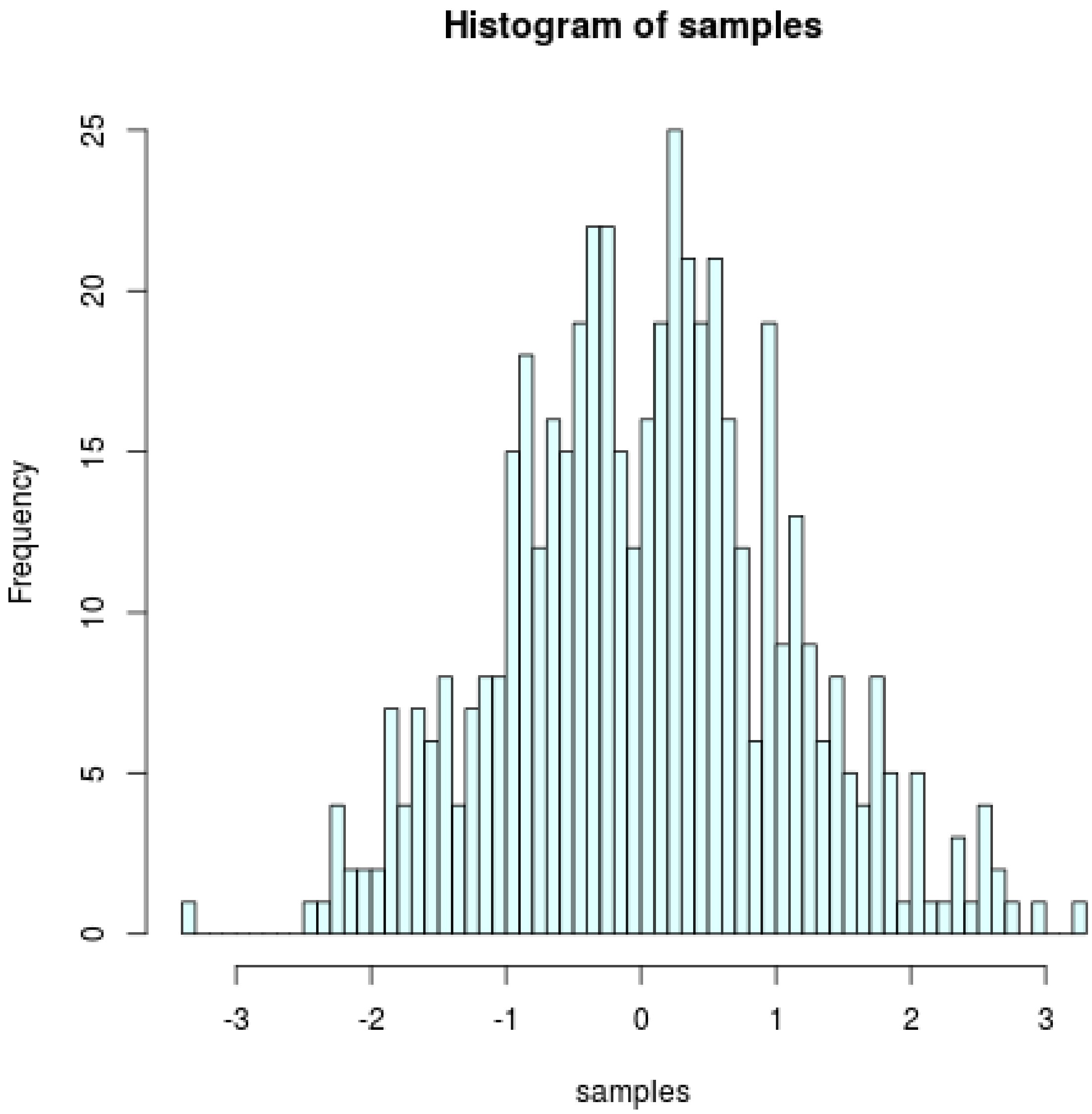
Variance: 1.006028

Observation: **Graph:**



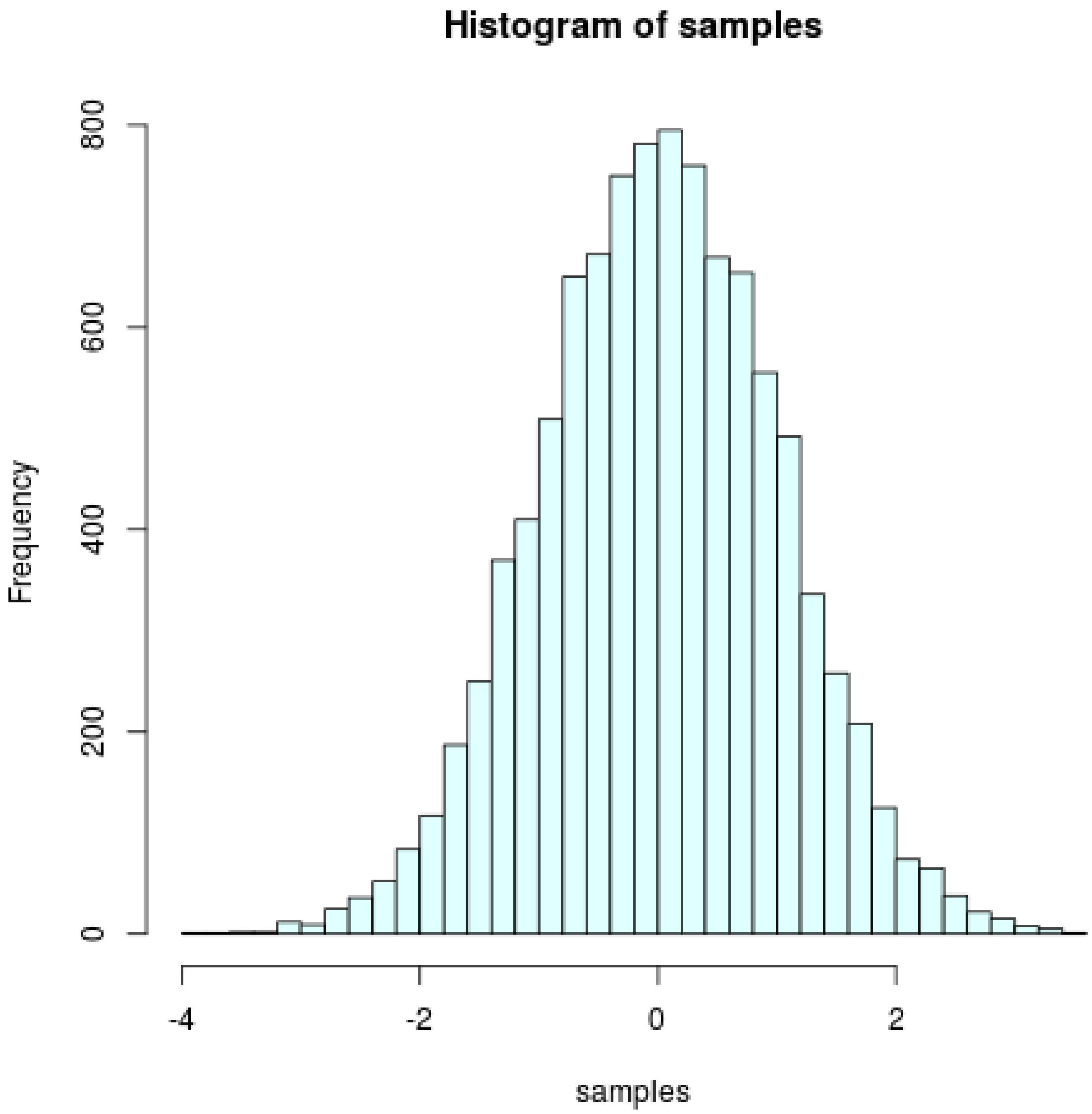
(d) *using R*

Graph: 500 values:



(e) *using R*

Graph: 10000 values:



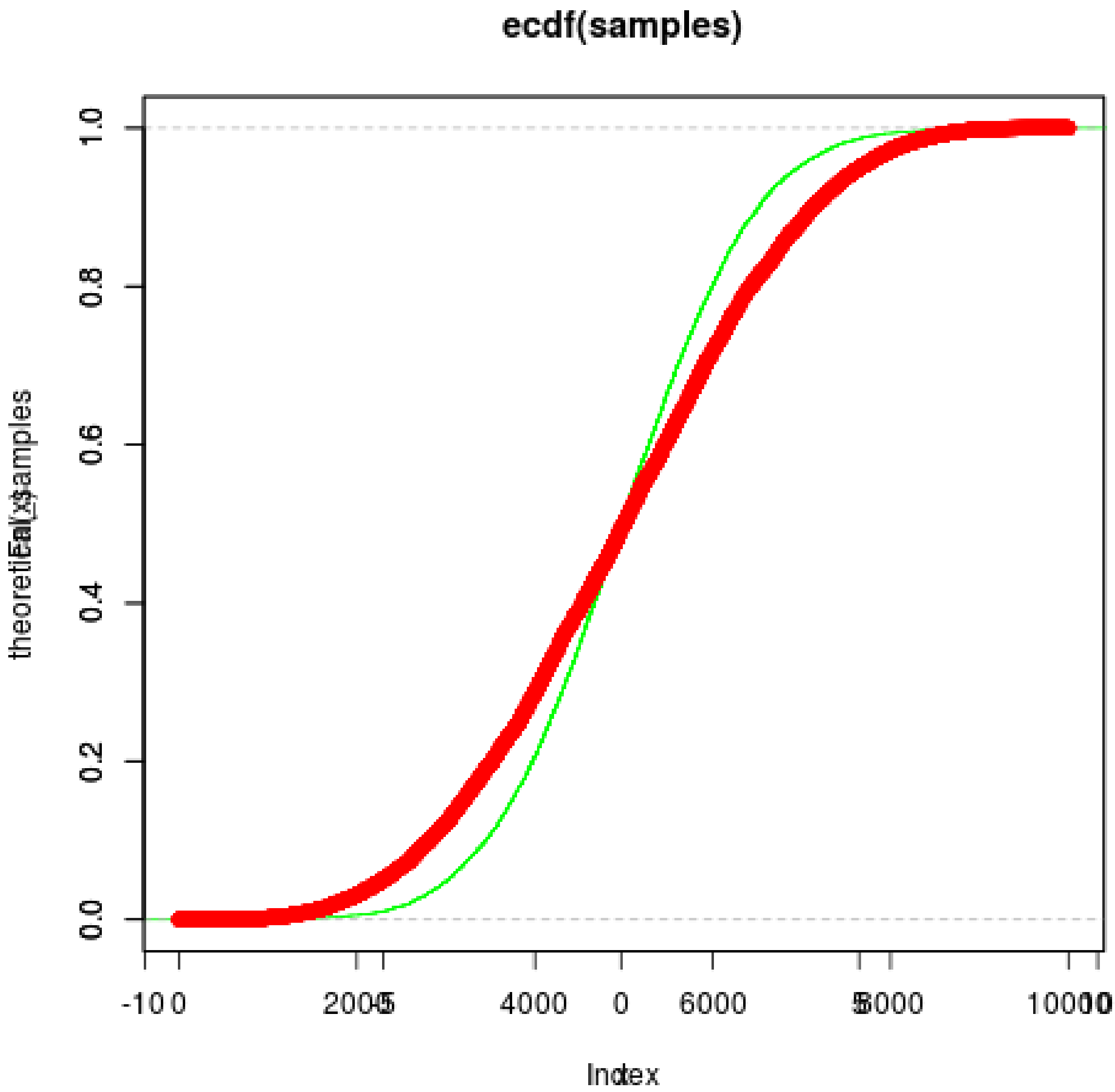
(f) *using R*

Q 2 Now use the above generated values to generated samples from N(= 0, 2 = 5) and N(= 5, 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).

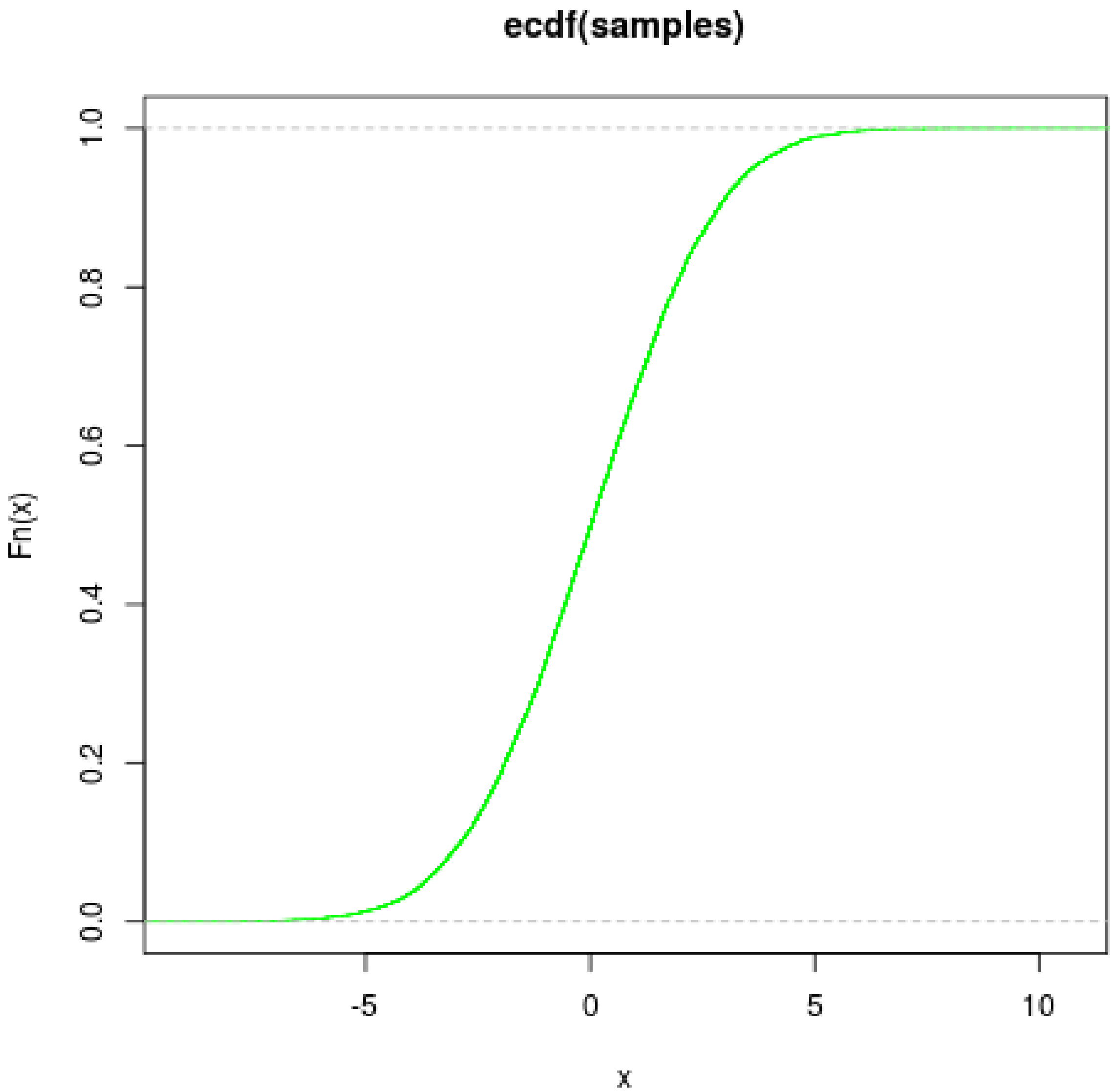
Code for Marsaglia-Bray method mean=0, variance=5 : R

```
square<-function(u1,u2){
    return (u1**2+u2**2)
}
intermediate<-function(x){
```

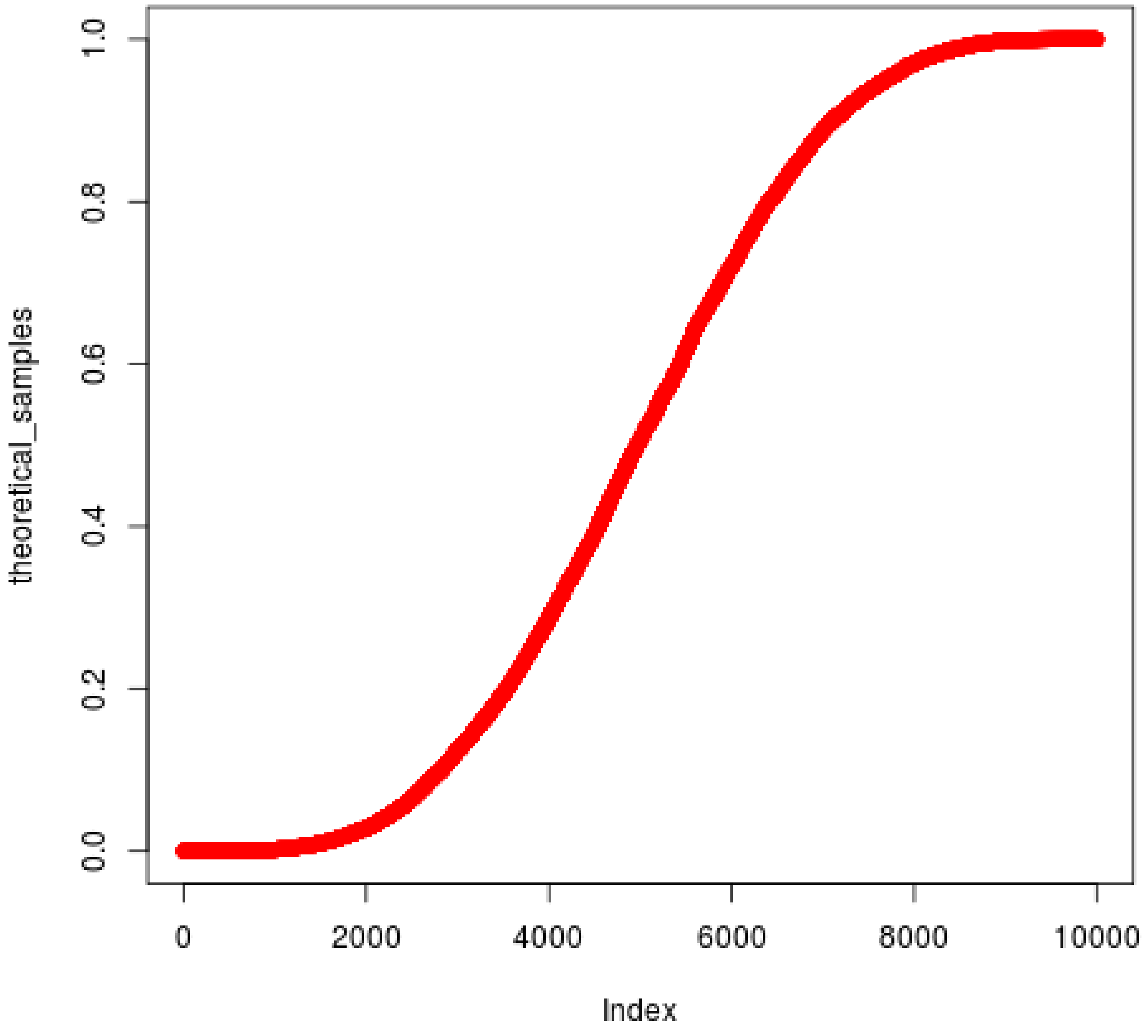
```
    return  (sqrt(-2*log(x)/x))
}
count<-10000
standard_deviation<-sqrt(5)
mean<-0
samples1<-c()
samples2<-c()
i<-1
j<-0
while(1){
    u1<-runif(1)
    u2<-runif(1)
    u1<-2*u1-1
    u2<-2*u2-1
    j<-j+1
    x<-square(u1,u2)
    if(x>1){
        next
    }
    y<-intermediate(x)
    samples1[i]<-u1*y*standard_deviation+mean
    samples2[i]<-u2*y*standard_deviation+mean
    i<-i+1
    count<-count-2
    if (count==0){
        break
    }
}
samples<-c(samples1,samples2)
samples<-sort(samples)
cat("Mean is: ",mean(samples),"\n")
cat("Acceptance Probability: ",(i-1)/j,"\n")
empirical_samples<-ecdf(samples)
png("Empirical(green).png")
plot(empirical_samples,col="green")
#par(new=TRUE)
png("Theoretical(red).png")
theoretical_samples<-pnorm(samples)
plot(theoretical_samples,col="red")
```



(g) *Using R*



(h) *Using R*



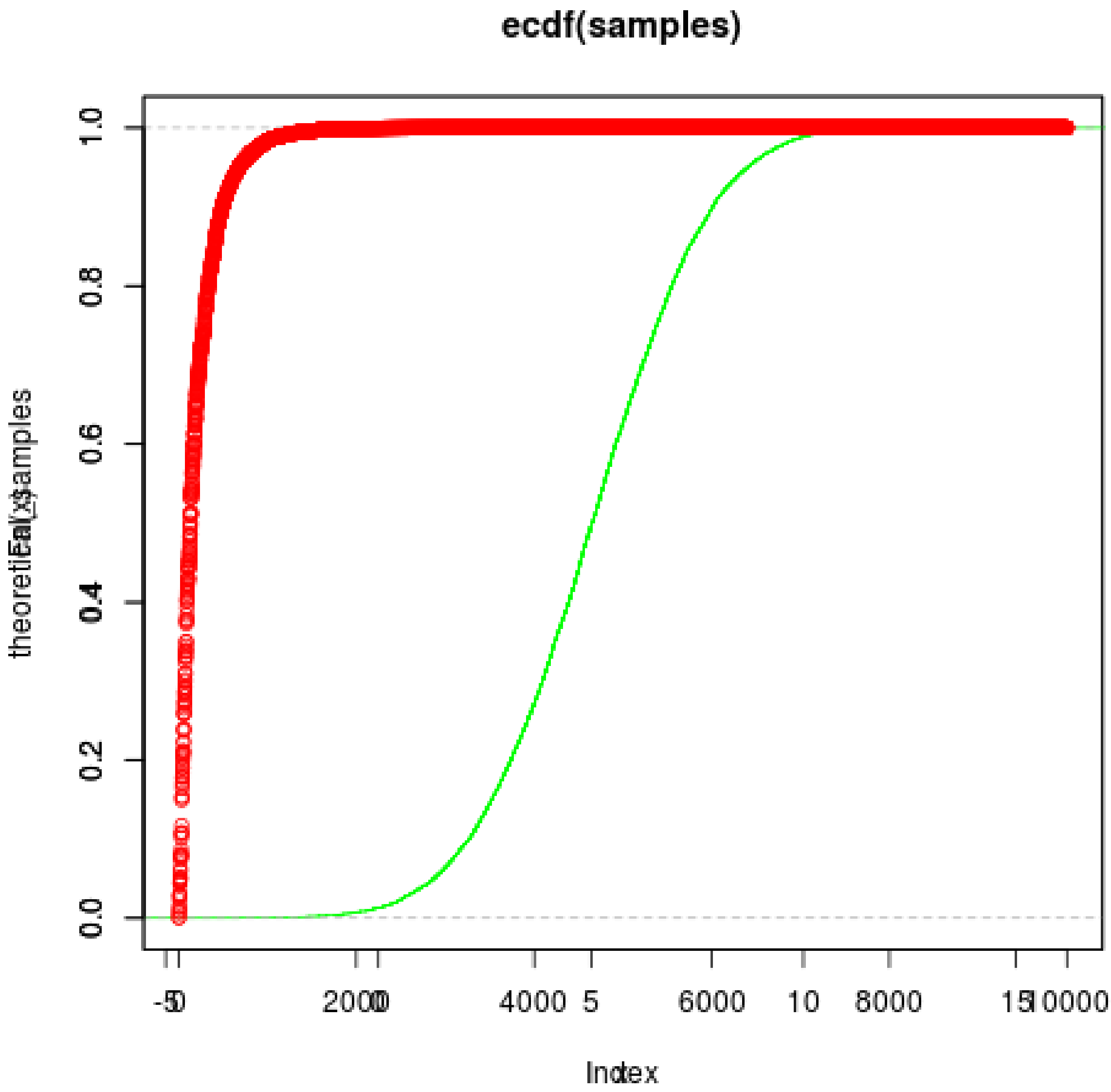
(i) *Using R*

Code for Marsaglia-Bray method mean=5, variance=5 : R

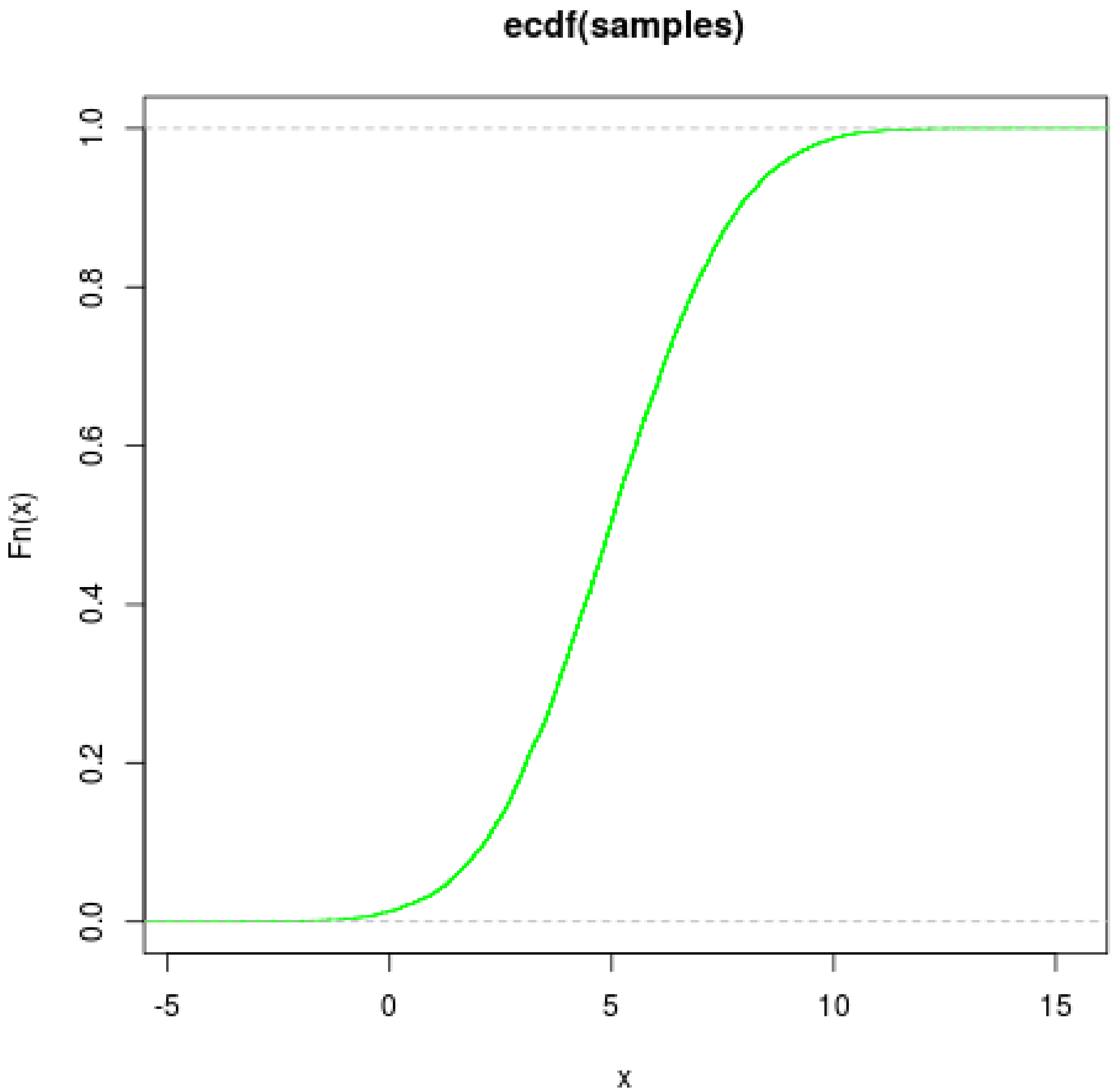
```
square<-function(u1,u2){
    return (u1**2+u2**2)
}
intermediate<-function(x){
    return (sqrt(-2*log(x)/x))
}
count<-10000
standard_deviation<-sqrt(5)
mean<-5
samples1<-c()
```

```
samples2<-c()
i<-1
j<-0
while(1){
    u1<-runif(1)
    u2<-runif(1)
    u1<-2*u1-1
    u2<-2*u2-1
    j<-j+1
    x<-square(u1,u2)
    if(x>1){
        next
    }
    y<-intermediate(x)
    samples1[i]<-u1*y*standard_deviation+mean
    samples2[i]<-u2*y*standard_deviation+mean
    i<-i+1
    count<-count-2
    if (count==0){
        break
    }
}
samples<-c(samples1,samples2)
samples<-sort(samples)
cat("Mean is: ",mean(samples),"\n")
cat("Acceptance Probability: ",(i-1)/j,"\n")
empirical_samples<-ecdf(samples)
png("question2-mean=5.png")
plot(empirical_samples,col="green")
par(new=TRUE)
#png("Theoretical(red)--mean=5.png")
theoretical_samples<-pnorm(samples)
plot(theoretical_samples,col="red")
```

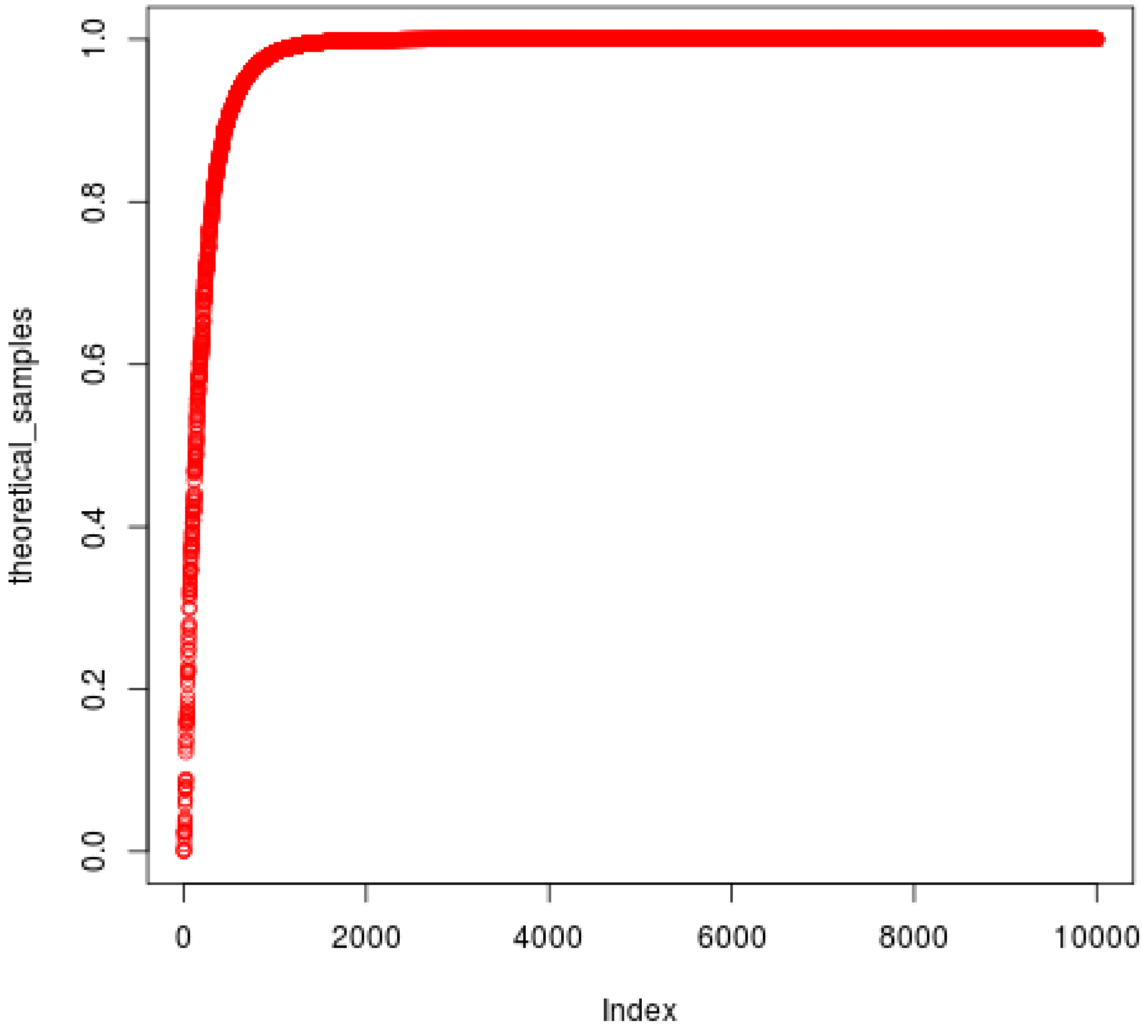
Observation: Graph:


ecdf(samples)
1.0
0.8
0.6
0.4
0.2
0.0
2000
4000
6000
8000
10
10000
Index

(j) *Using R*



(k) *Using R*



(1) *Using R*

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

Code for Box-Muller method: R

```
radius<-function(u){
    return (-2*log(u))
}
arg<-function(u){
    return (2*pi*u)
}
```

```
count<-10000
stime<-Sys.time();
samples1<-c()
samples2<-c()
for(i in 1:count/2){
    u1<-runif(1)
    u2<-runif(1)
    samples1[i]<-sqrt(radius(u1))*cos(arg(u2))
    samples2[i]<-sqrt(radius(u1))*sin(arg(u2))
}


samples<-c(samples1,samples2)
etime<-Sys.time();
cat("Computation Time of Box-Muller Method: ",etime-stime,"\n");
```

Code for Marsaglia-Bray method: R

```
square<-function(u1,u2){
    return (u1**2+u2**2)
}
intermediate<-function(x){
    return (sqrt(-2*log(x)/x))
}
count<-10000
stime<-Sys.time();
samples1<-c()
samples2<-c()
i<-1
while(1){
    u1<-runif(1)
    u2<-runif(1)
    u1<-2*u1-1
    u2<-2*u2-1
    x<-square(u1,u2)
    if(x>1){
        next
    }
    y<-intermediate(x)
    samples1[i]<-u1*y
    samples2[i]<-u2*y
    i<-i+1
```

```
    count<-count-2
    if (count==0){
        break
    }
}
samples<-c(samples1,samples2)
etime<-Sys.time();
cat("Computation Time of Marsaglia-Bray Method: ",etime-stime,"\n");
```

Output:

Computation Time of Box-Muller Method: 0.1807923

Computation Time of Marsaglia-Bray Method: 0.1370828

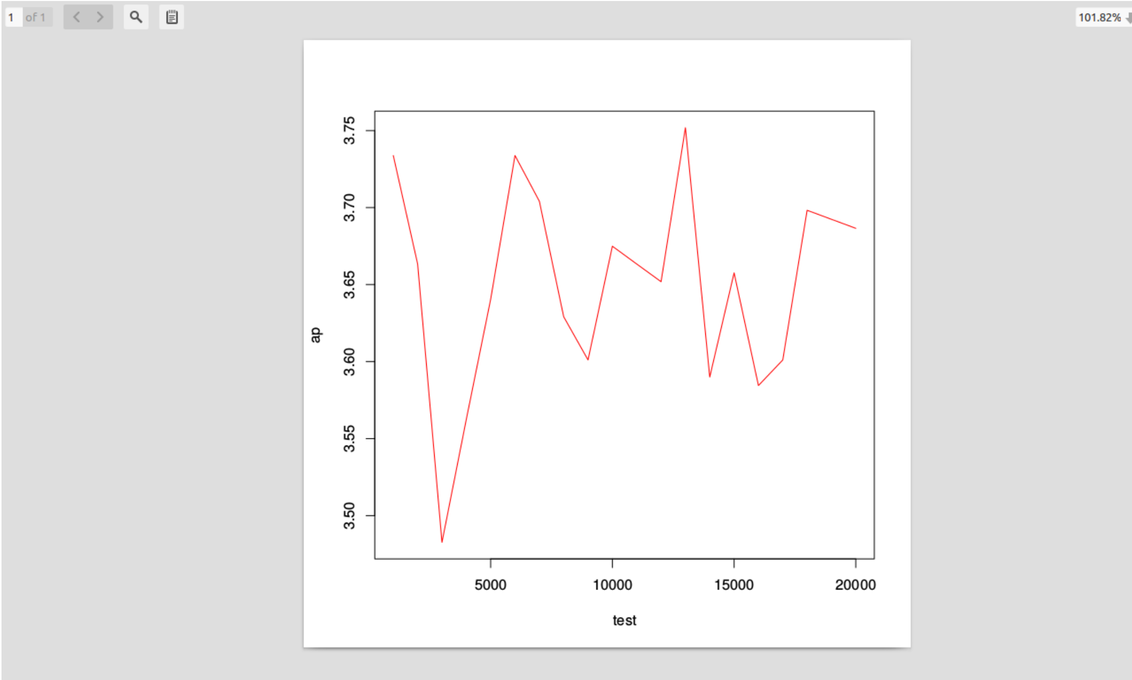
Q 4 For the Marsaglia-Bray method keep track of the proportional of values rejected. How does it compare with 1 pi/4 ?? ?

Code for Marsaglia-Bray method: R

```
square<-function(u1,u2){
    return (u1**2+u2**2)
}
intermediate<-function(x){
    return (sqrt(-2*log(x)/x))
}
count<-1000
ap<-c()
for(k in 1:20){
samples1<-c()
samples2<-c()
i<-1
j<-0
while(1){
    u1<-runif(1)
    u2<-runif(1)
    u1<-2*u1-1
    u2<-2*u2-1
    j<-j+1
    x<-square(u1,u2)
    if(x>1){
        next
```

```
    }
    y<-intermediate(x)
    samples1[i]<-u1*y
    samples2[i]<-u2*y
    i<-i+1
    count<-count-2
    if (count==0){
        break
    }
}
count<-count+1000
ap[k]<-(((i-1)/j)/(1-pi/4))
}
samples<-c(samples1,samples2)
k<-1
test<-c()
constant<-1000
for (k in 1:20){
    test[k]<-constant*k
}
cat("Acceptance Probability: ",(i-1)/j,"\n")
plot(test,ap,type="l",col="red")
```

Observation: **Graph:**

by 1-pi divide 2.png 

(m) *Using R*