# 0330 作業

計財所碩二 108071601 賴冠維

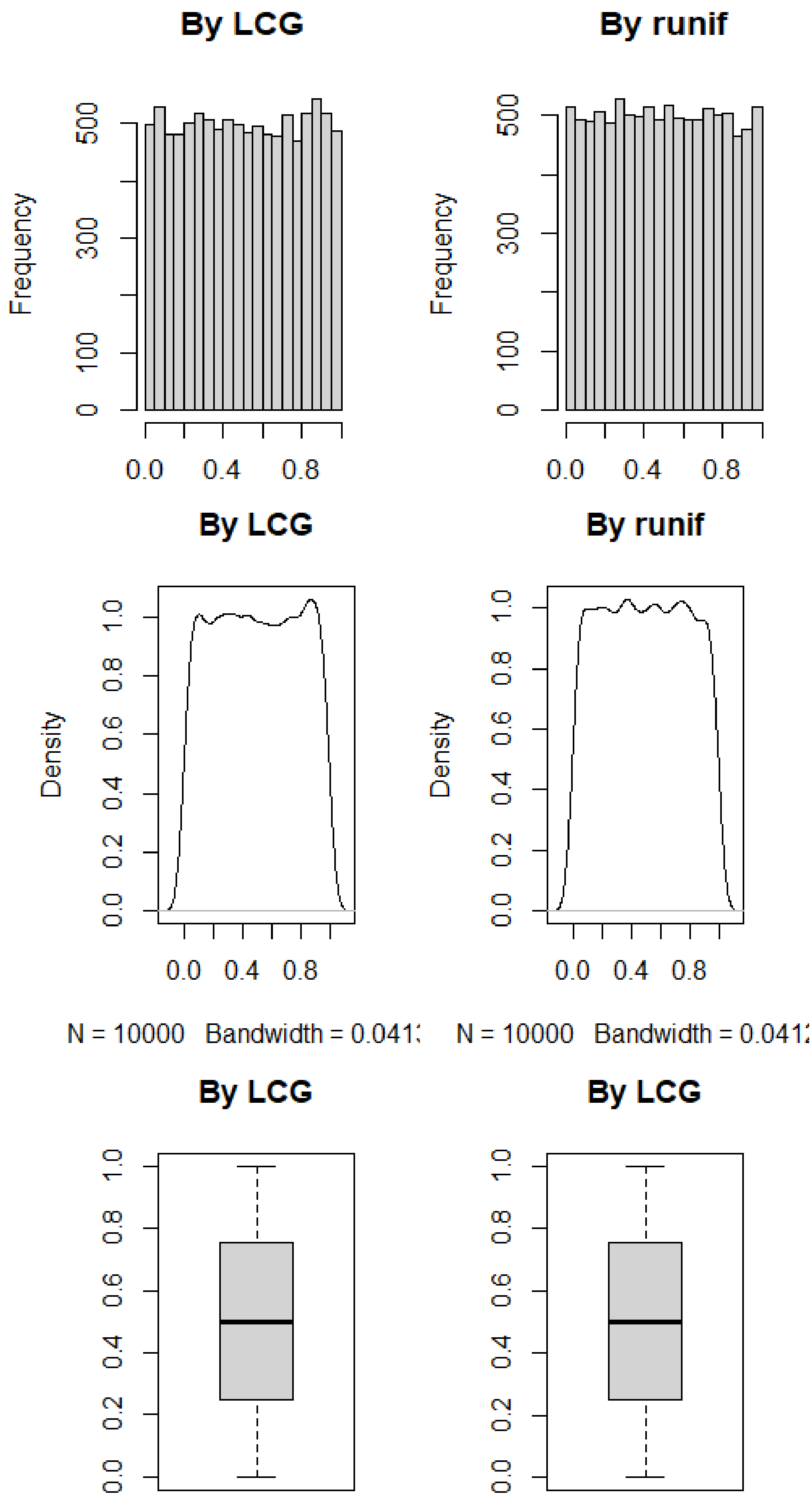
2021/3/29

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --

## √ ggplot2 3.3.2     √ purrr   0.3.4
## √ tibble  3.0.4     √ dplyr   1.0.2
## √ tidyr   1.1.2     √ stringr 1.4.0
## √ readr   1.3.1     √ forcats 0.5.0

## Warning: package 'tibble' was built under R version 4.0.3

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

一開始利用$LCG(Linear\ congruential\ generator)$製造自$Uniform(0,1)$抽樣的樣本。

圖左邊為以 LCG 製造的$Uniform(0,1)$樣本所畫的直方圖及以該樣本所估計的 Density Plot，與右圖$runif()$相比起來十分相似，從盒狀圖來看，也可看出兩組資料十分相近。

```
# 此為 LCG 生成所用之演算法
lcg <- function(a,c,m,run.length,seed) {
  x <- rep(0,run.length)
  x[1] <- seed
  for (i in 1:(run.length-1)) {
    x[i+1] <- (a*x[i] + c) %% m
  }
  U <- x/m
  return(list(x=x,U=U))
}
```


By LCG
By runif
Frequency
0
100
300
500
0.0
0.4
0.8
By LCG
By runif
Density
0.0
0.2
0.4
0.6
0.8
1.0
N = 10000 Bandwidth = 0.041
N = 10000 Bandwidth = 0.041
By LCG
By LCG

# Problem 1

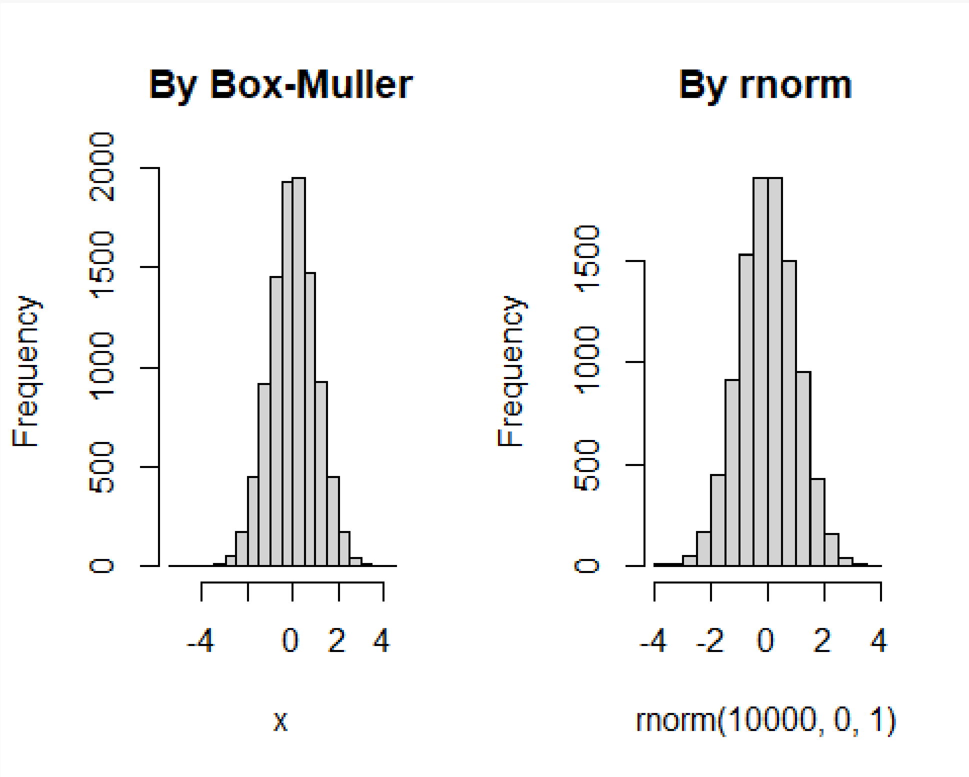
## (a)

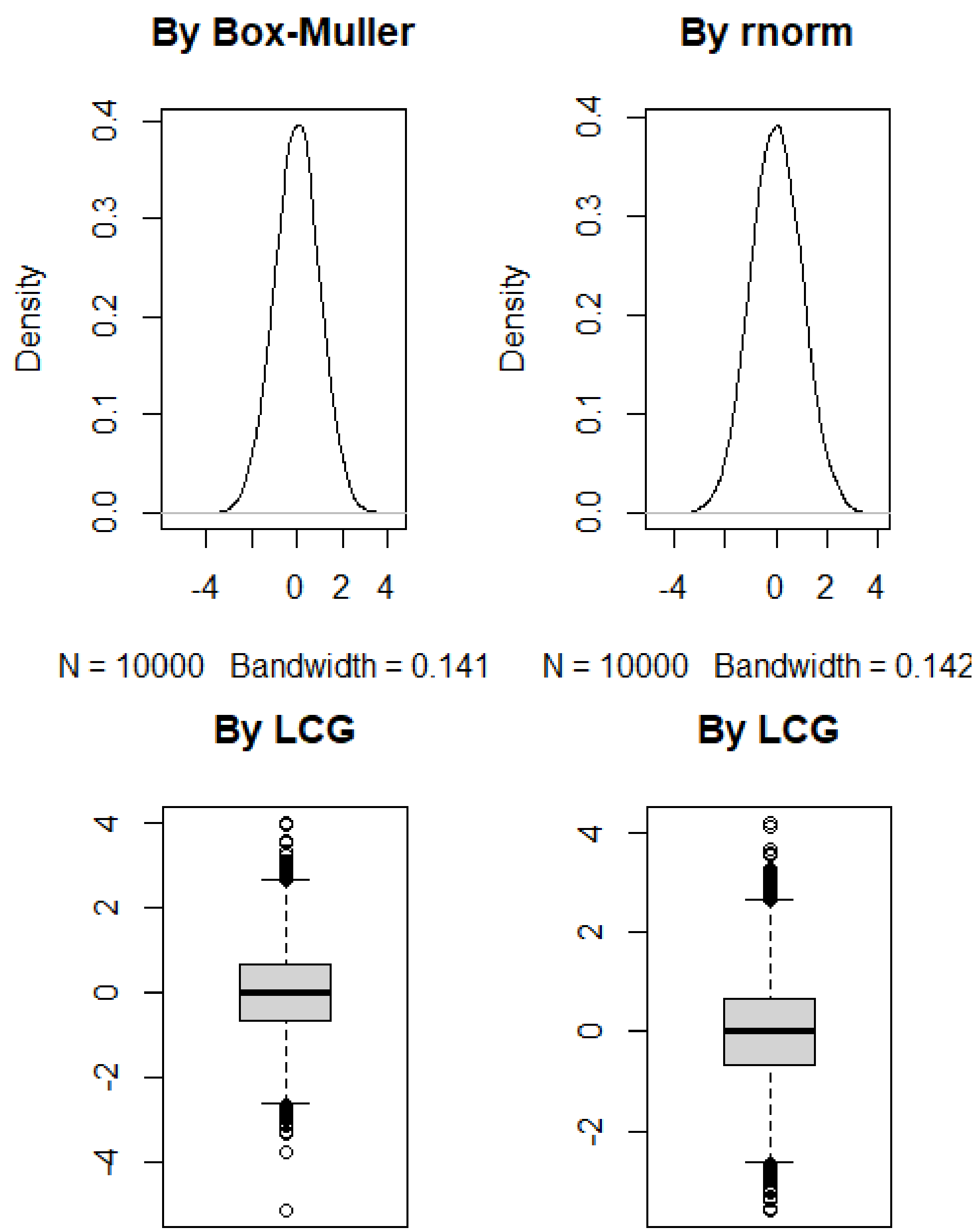
以$Box - Muller$法製造出 10000 個標準常態

圖左邊為以 LCG 製造的$Box - Muller$方法所畫的直方圖及 DensityPlot，與右圖$rnorm()$相比起來十分相似。

從盒狀圖來看，也可看出兩組資料十分相近，僅在 Q1,Q3 以外，$Box - Muller$法較$rnorm()$些微多一些。

```
# 此為 Box-Muller 所用之演算法
fnc = function(U1,U2){
  sqrt(-2*log(U1))*cos(2*pi*U2)
}
```






## (b)

需要用到指數分配，放在第二題結束的地方

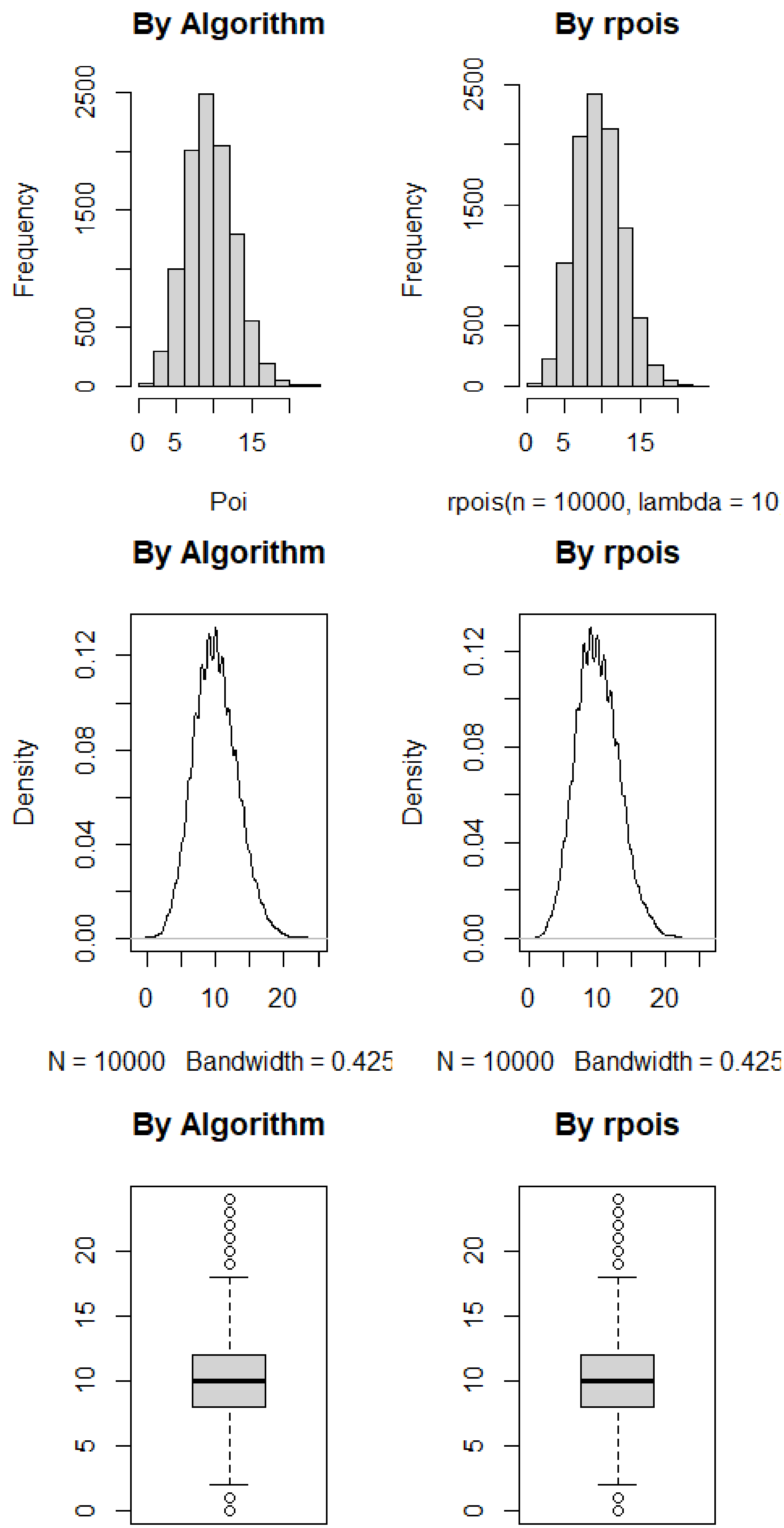
# Problem 2

## (a)

圖左邊為自製以上課所教的演算法所得到$Poisson(10)$的樣本，所畫的直方圖及 DensityPlot，與右圖$rpois()$相比起來十分相似。

從盒狀圖來看，也可看出兩組資料十分相近。

```
# 此生成 Poisson 分配所用之演算法


Poisson = function(mu,n){
  X = sapply(1:n, function(a){
    t = 0 ; X = 0 ; lambda = mu
    while (t<1) {
      U = sample(Uni$U,1)
      t = t - (1/lambda)*log(U)
      X = X+1
    }
    X = X-1
    X
  })

  return(X)
}
```


By Algorithm
By rpois
Frequency
Poi
rpois(n = 10000, lambda = 10
By Algorithm
By rpois
Density
N = 10000 Bandwidth = 0.425
N = 10000 Bandwidth = 0.425
By Algorithm
By rpois

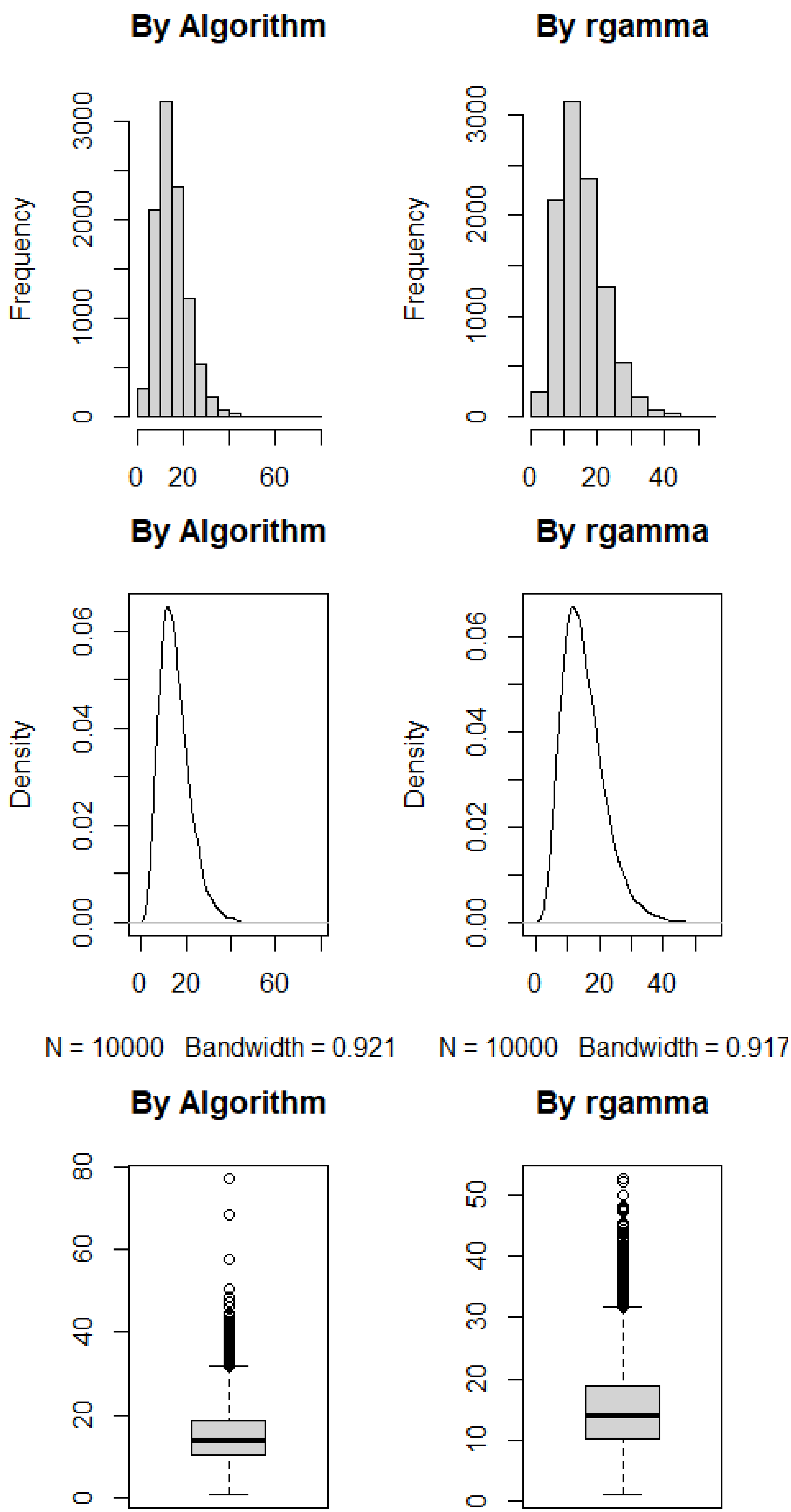
### (b)

先以上課所教的演算法，生成$Exponential\ Distribution$，再將$EXP(1)$連加，即可得到$Gamma(3,1)$的樣本。

圖左所生成樣本畫的直方圖及 DensityPlot，與右圖$rgamma()$相比起來十分相似。

從盒狀圖來看，也可看出兩組資料十分相近，但自己生成的 Gamma 有較易有偏離期望值很多的值產生，也因此自生成樣本的全距較大。

```
# 此為生成 Exponential 分配所用之演算法
EXP = function(mu,n){
 lambda =mu
 N = sapply(1:n, function(a){
  U = sample(Uni$U,1)
  X = -(1/lambda)*log(U)
 })
 N
}

# 此為生成 Gamma 分配所用之演算法
GAM = function(a,b,n){
 i = 1
 x = matrix(0,nrow = n)
 repeat{
  if(i>a) break
  y = EXP(b,n) %>% as.matrix()
  x = x+y
  i = i+1
 }
 return(x)
}
```


By Algorithm
By rgamma
Frequency
0
1000
2000
3000
0
20
60
0
20
40
Density
0.00
0.02
0.04
0.06
N = 10000 Bandwidth = 0.921
N = 10000 Bandwidth = 0.917
0
20
40
60
80
0
10
20
30
40
50

# Problem 1

## (b)

以下為$Acceptance-Rejection\ Approach$的演算法

```
E = EXP(1,10000)

NormalAR = function(c,n){
  f = function(z){        #設定 f 為標準常態分配
    sqrt(2/pi)*exp(-z^2/2)
  }
  g = function(z){        #設定 g 為 exp(1)
    exp(-z)
  }

  f_z = as.vector(rep(0,n))
  i = 0   #設定 i 迭代至所需樣本數
  b = 1
  a = c() #設定 a,b 紀錄該次是否接受，計算 Acceptance Rate
  while (i< n){
    b = b+1
    z = -log(sample(Uni$U,1))
    u1 = sample(Uni$U,1)
    a[b] = ifelse(u1 < f(z)/(c*g(z)),"Y","N")

    if(u1 < f(z)/(c*g(z))){

      u2 = sample(Uni$U,1)
      if(u2 < 0.5) { Z = z } else {Z = -z }
      i = i+1
      f_z[i] = Z
    }
  }
  return(list(f_z,a))
}
```

## Acceptance Rate

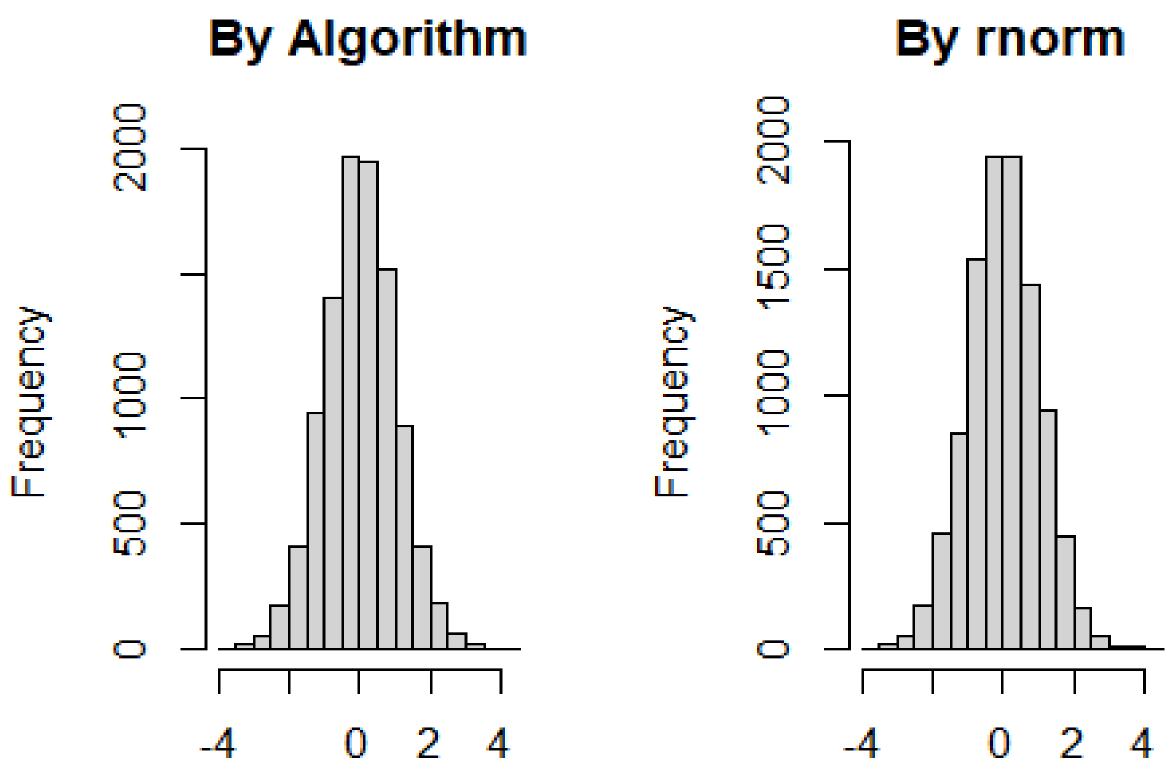
首先設定$c = \frac{1}{\sqrt{2\pi}} exp(\frac{1}{2})$，可得到此時$Acceptance\ Rate$約為 75%。

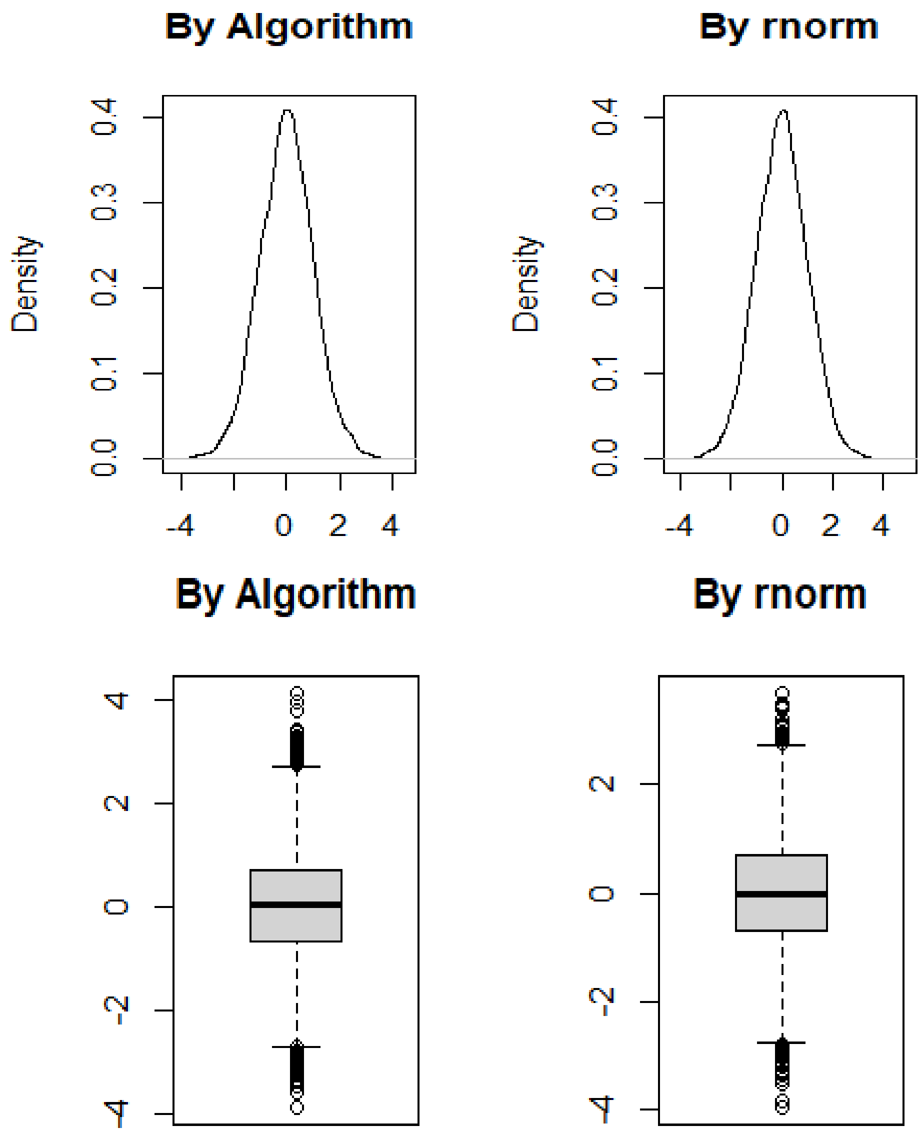
但若是改變 c，令$c^* = \frac{2}{\sqrt{2\pi}} exp(\frac{1}{2}) = 2c$，則此時的$Acceptance\ Rate$下降至約 38%。

```
## [1] "Acceptance_Rate_1 : 74.86%"

## [1] "Acceptance_Rate_2 : 37.83%"
```

先生成$Normal\ Distribution$，圖左所生成樣本畫的直方圖及 DensityPlot，與右圖$rnorm()$相比起來十分相似。

從盒狀圖來看，也可看出兩組資料十分相近。






# Problem 3

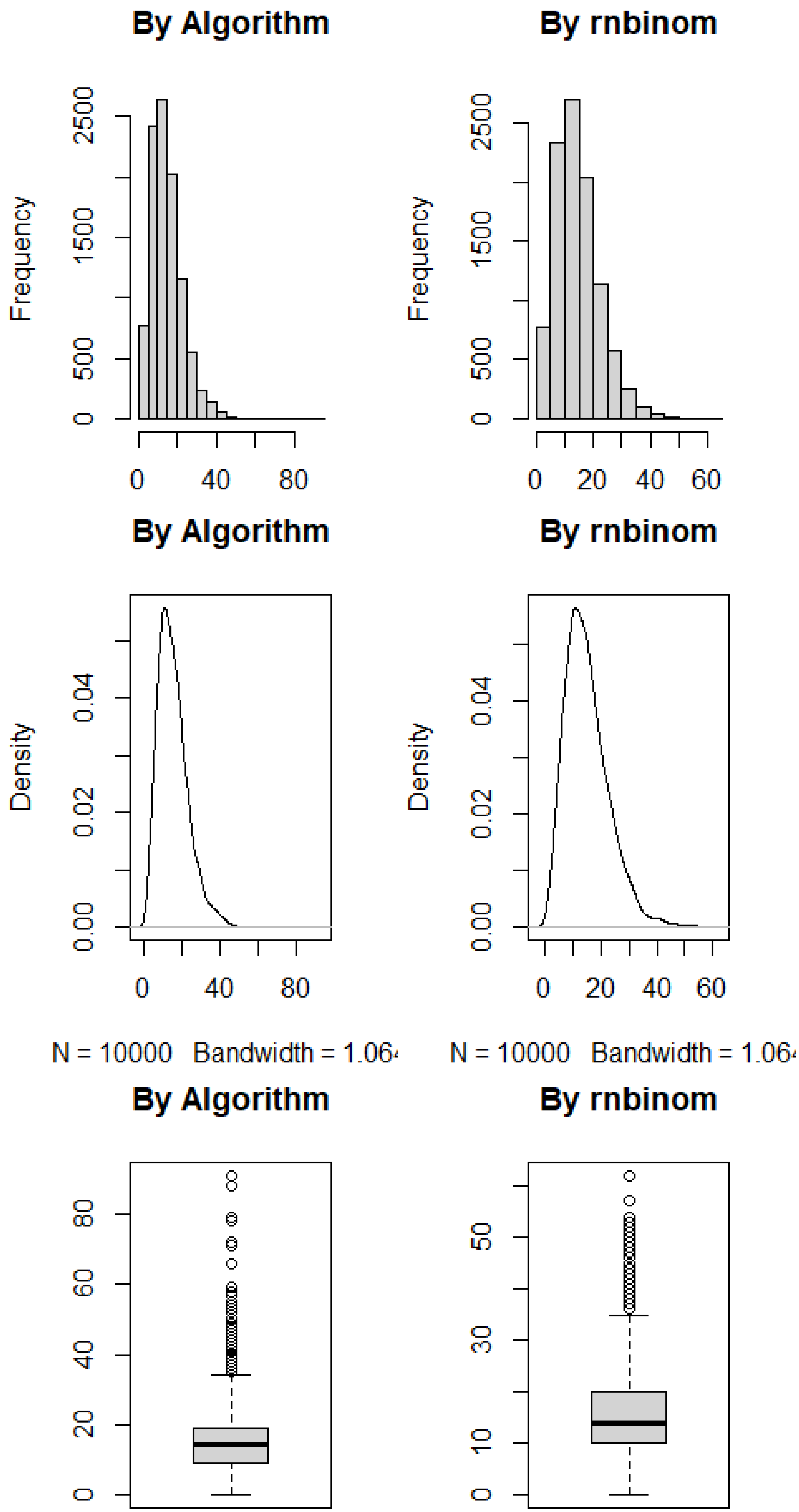
## (c)

```
# 樣本產生方式，將 Gamma 生成樣本帶入 Poisson 分配


G = GAM(a = 5,b = 1/3,n = 10000)
x = matrix(0,nrow=10000)
for (i in 1:10000) {
  x[i,] = Poisson(mu = G[i],n=1)
}
```

由上兩題可得知此混和分配為負二項分配，將生成樣本與*rnbinom*()比較。

圖左所生成樣本畫的直方圖及 DensityPlot，與右圖*rnbinom*()相比起來十分相似。

從盒狀圖來看，也可看出兩組資料十分相近，但自生成的樣本有偏離較大的值出現。



可發現自生成樣本的平均數約為 15，變異數約為 60，

與使用$rnbinom()$所產生的樣本結果相似。

```
## [1] "Mean from the samples by myself: 15.0759"

## [1] "Mean from the samples by rnbinom(): 15.0826"

## [1] "Variance from the samples by myself: 64.649404130413"

## [1] "Variance from the samples by rnbinom(): 59.2290229022902"
```

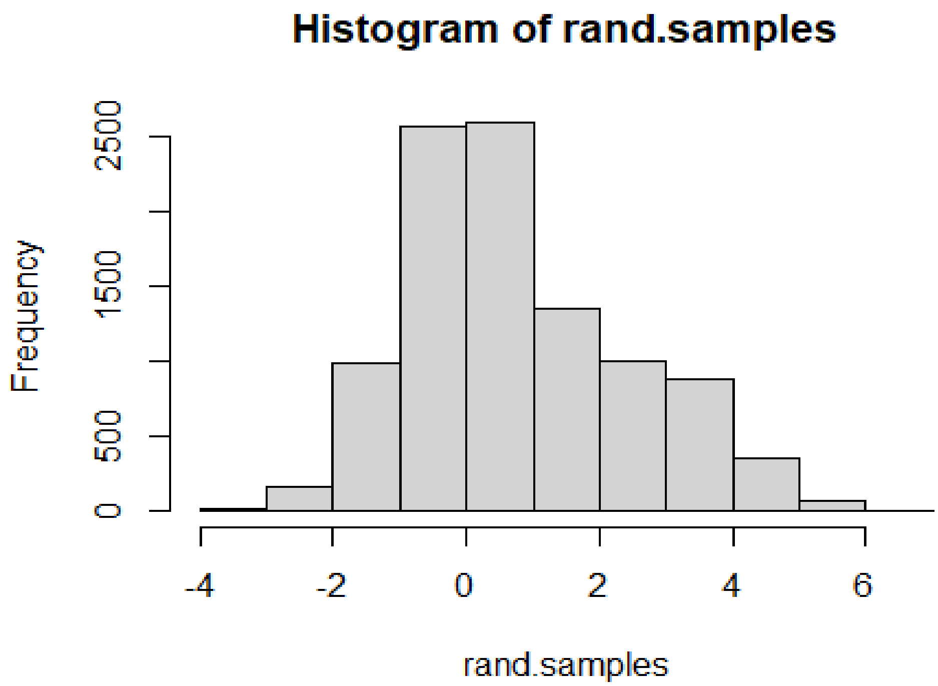
# Problem 4

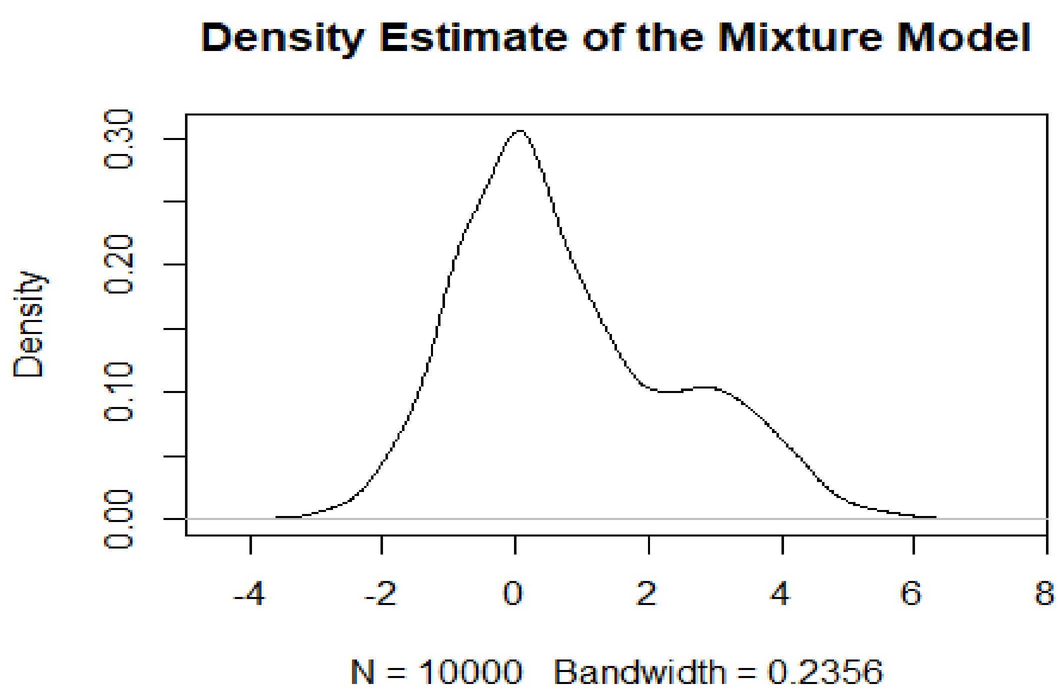
## (b)

由(a)小題可得知其機率密度函數，其生成樣本之演算法解釋如下： 1. 自$Uniform(0,1)$抽一項,$X$ 2. 若$X > 0.75$，則生成$N(3,1)$，若$X <= 0.75$則生成$N(0,1)$

```
vals <- NormalAR(c = sqrt(2/pi)*exp(1/2),n = 10000)


N = 10000
U =runif(N)
rand.samples = rep(NA,N)
for(i in 1:N){
  if(U[i]<0.75){
    rand.samples[i] = sample(vals[[1]],1)
  }else{
    rand.samples[i] = sample(vals[[1]],1)+3 #自 N(0,1)平移 3，變成 N(3,1)
  }
}
```

生成樣本後，畫出該分配的直方圖及 DensityPlot，可發現在$X$在 0,3 的地方有雙峰的現象，符合我們一開始的想像。



附錄(程式碼)

```r
library(tidyverse)

lcg <- function(a,c,m,run.length,seed) {
  x <- rep(0,run.length)
  x[1] <- seed
  for (i in 1:(run.length-1)) {
    x[i+1] <- (a*x[i] + c) %% m
  }
  U <- x/m
  return(list(x=x,U=U))
}

Uni = lcg(a = 7^5,c = 0,m = 2^31-1,run.length = 10000,seed = 5)

par(mfrow=c(1,2))

Uni$U %>% as.numeric() %>% hist(.,main="By LCG")

runif(10000) %>% hist(.,main="By runif")

par(mfrow=c(1,2))

plot(density(Uni$U),main="By LCG")

plot(density(runif(10000)),main="By runif")

boxplot(Uni$U,main="By LCG");boxplot(runif(10000),main="By LCG")

# Problem 1

## (a)

fnc = function(U1,U2){
  sqrt(-2*log(U1))*cos(2*pi*U2)
}

x = fnc(sample(Uni$U,10000),sample(Uni$U,10000))

y = fnc(sample(Uni$U,10000),sample(Uni$U,10000))

par(mfrow=c(1,2))

hist(x,main = "By Box-Muller");hist(rnorm(10000,0,1),main = "By rnorm")

plot(density(x),main="By Box-Muller");plot(density(rnorm(10000,0,1)),main="By rnorm")
```

```r
boxplot(x,main="By LCG");boxplot(rnorm(10000,0,1),main="By LCG")
## (b)
# Problem 2
## (a)
Poisson = function(mu,n){
  X = sapply(1:n, function(a){
    t = 0 ; X = 0 ; lambda = mu
    while (t<1) {
      U = sample(Uni$U,1)
      t = t - (1/lambda)*log(U)
      X = X+1
    }
    X = X-1
    X
  })
  return(X)
}
par(mfrow=c(1,2))
Poi = Poisson(10,10000)
hist(Poi,main="By Algorithm");hist(rpois(n = 10000,lambda = 10),main="By rpois")
plot(density(Poi),main="By Algorithm");plot(density(rpois(n = 10000,lambda = 10)),main="By rpois")
boxplot(Poi,main="By Algorithm");boxplot(rpois(n = 10000,lambda = 10),main="By rpois")
## (b)
EXP = function(mu,n){
  lambda =mu
  N = sapply(1:n, function(a){
    U = sample(Uni$U,1)
    X = -(1/lambda)*log(U)
```

```
  })
  N
}
GAM = function(a,b,n){
  i = 1
  x = matrix(0,nrow = n)
  repeat{
    if(i>a) break
    y = EXP(b,n) %>% as.matrix()
    x = x+y
    i = i+1
  }
  return(x)
}
E = EXP(3,10000)
G = GAM(5,1/3,10000)
par(mfrow=c(1,2))
hist(G,main="By Algorithm");hist(rgamma(n=10000,shape = 5,scale = 3),main="By rgamma")
plot(density(G),main="By Algorithm");plot(density(rgamma(n=10000,shape = 5,scale =
3)),main="By rgamma")
boxplot(G,main="By Algorithm");boxplot(rgamma(n=10000,shape = 5,scale = 3),main="By
rgamma")
## Problem 1
### (b)
E = EXP(1,10000)
NormalAR = function(c,n){
  f = function(z){         #設定 f 為標準常態分配
    sqrt(2/pi)*exp(-z^2/2)
  }
  g = function(z){        #設定 g 為 exp(1)
```

```
    exp(-z)
  }
  f_z = as.vector(rep(0,n))
  i = 0   #設定 i 迭代至所需樣本數
  b = 1
  a = c() #設定 a,b 紀錄該次是否接受，計算 Acceptance Rate
  while (i< n){
    b = b+1
    z = -log(sample(Uni$U,1))
    u1 = sample(Uni$U,1)
    a[b] = ifelse(u1 < f(z)/(c*g(z)),"Y","N")
    if(u1 < f(z)/(c*g(z))){
      u2 = sample(Uni$U,1)
      if(u2 < 0.5) { Z = z } else {Z = -z }
      i = i+1
      f_z[i] = Z
    }
  }
  return(list(f_z,a))
}
## Acceptance Rate
vals <- NormalAR(c = 1*sqrt(2/pi)*exp(1/2),n = 10000)
rate = table(vals[[2]])
r = rate[2]/sum(rate)*100
r = round(r,2)
paste0("Acceptance_Rate_1 : ",r,"%")
vals <- NormalAR(c = 2*sqrt(2/pi)*exp(1/2),n = 10000)
rate = table(vals[[2]])
r = rate[2]/sum(rate)*100
```

```
r = round(r,2)

paste0("Acceptance_Rate_2 : ",r,"%")

par(mfrow=c(1,2))

vals <- NormalAR(c = sqrt(2/pi)*exp(1/2),n = 10000)

hist(vals[[1]],main="By Algorithm");hist(rnorm(n=10000,mean = 0,sd = 1),main="By rnorm")

plot(density(vals[[1]]),main="By Algorithm");plot(density(rnorm(n=10000,mean = 0,sd =
1)),main="By rnorm")

boxplot(vals[[1]],main="By Algorithm");boxplot(rnorm(n=10000,mean = 0,sd = 1),main="By
rnorm")

# Problem 3

## (c)

G = GAM(a = 5,b = 1/3,n = 10000)

x = matrix(0,nrow=10000)

for (i in 1:10000) {

  x[i,] = Poisson(mu = G[i],n=1)

}

par(mfrow=c(1,2))

hist(x,main="By Algorithm");hist(rnbinom(n =10000,size = 5,prob = 0.25),

                                 main="By rnbinom")

plot(density(x),main="By Algorithm");plot(density(rnbinom(n =10000,size = 5,prob = 0.25)),

                                          main="By rnbinom")

boxplot(x,main="By Algorithm");boxplot(rnbinom(n =10000,size = 5,prob = 0.25),

                                       main="By rnbinom")

paste0("Mean from the samples by myself: ",mean(x))

paste0("Mean from the samples by rnbinom(): ",mean(rnbinom(n =10000,size = 5,prob =
0.25)))

paste0("Variance from the samples by myself: ",var(x))

paste0("Variance from the samples by rnbinom(): ",var(rnbinom(n =10000,size = 5,prob =
0.25)))

# Problem 4

## (b)
```

```
vals <- NormalAR(c = sqrt(2/pi)*exp(1/2),n = 10000)
N = 10000
U =runif(N)
rand.samples = rep(NA,N)
for(i in 1:N){
  if(U[i]<0.75){
    rand.samples[i] = sample(vals[[1]],1)
  }else{
    rand.samples[i] = sample(vals[[1]],1)+3 #自 N(0,1)平移 3，變成 N(3,1)
  }
}
hist(rand.samples)
plot(density(rand.samples),main="Density Estimate of the Mixture Model")
```