hw3

Enbo Tian

2022/2/28

Graphical Model 1

a)

```
rm(list = ls())
X <- c("cold", "hot", "mild")
# day 0
day0 = replicate(5,sample(X,size = 1,prob=c(1/3,1/3,1/3)))
# function of day
dayk <- function(day){
  dayk <- rep(0,5)
  for (i in 1:5){
    if (day[i] == "cold"){
    dayk[i] = sample(c("cold", "hot", "mild"),size =1,prob = c(1/2,1/4,
1/4))
    }
    else if (day[i] == "hot"){
    dayk[i] = sample(c("cold", "hot", "mild"),size =1,prob = c(1/3,1/3,
1/3))
    }
    else if(day[i] == "mild"){
      dayk[i]= sample(c("cold", "hot", "mild"),size =1,prob = c(1/4,1/4,
1/2))
    }
  }
  dayk
}
# day 1:5
day1 <- dayk(day0)
day2 <- dayk(day1)
day3 <- dayk(day2)
day4 <- dayk(day3)

day <- data.frame(day0, day1, day2, day3, day4)
day
```

```
##   day0 day1 day2 day3 day4
## 1 mild mild  hot  hot mild
## 2 mild  hot cold  hot  hot
## 3 cold cold mild mild cold
## 4  hot cold  hot mild mild
## 5 mild cold  hot mild mild
```

b)

```
# P(day0)
p0 <- c(1/3,1/3,1/3)
# p (k given k-1)
pgiven <- matrix(c(1/2,1/4,1/4,1/3,1/3,1/3,1/4,1/4,1/2),ncol=3)
# marginal prob
p1 <- pgiven%*%p0
p2 <- pgiven%*%p1
p3 <- pgiven%*%p2
margp <- data.frame(p0,p1,p2,p3)
margp
```

```
##          p0        p1        p2        p3
## 1 0.3333333 0.3611111 0.3634259 0.3636188
## 2 0.3333333 0.2777778 0.2731481 0.2727623
## 3 0.3333333 0.3611111 0.3634259 0.3636188
```

c)

```
# 3|2hot
p3g2 <- c(1/3,1/3,1/3)
p3g2
```

```
## [1] 0.3333333 0.3333333 0.3333333
```

```
# p(1|2="hot") = p(2|1)*p(1)/p(2 = "hot")
p1 <- c(p1)
p2 <- c(p2)
p3 <- c(p3)
p1g2 <- pgiven * p1 / p2
p1g2 <- c(p1g2[,2])
p1g2
```

```
## [1] 0.3312102 0.3389831 0.3312102
```

```
# p(0|1)  = p(1|0)*p(0)/p(1|2="hot")
p0g1 <- pgiven * p0 /p1g2
p0g1
```

```
##           [,1]      [,2]      [,3]
## [1,] 0.5032051 0.3354701 0.2516026
## [2,] 0.2458333 0.3277778 0.2458333
## [3,] 0.2516026 0.3354701 0.5032051
```

d)

```
# give day2 is hot
day2 = "hot"
# get most probable day1
if (max(p1g2) == p1g2[1]){
  day1 = "cold"
  i = 1
}else if(max(p1g2) == p1g2[2]){
  day1 = "hot"
  i = 2
}else if(max(p1g2) == p1g2[3]){
  day1 = "mild"
  i =3
}
# get most probable day0
new_p0g1 <- p0g1[,i]
if (max(new_p0g1) == new_p0g1[1]){
  day0 = "cold"
}else if(max(new_p0g1) == new_p0g1[2]){
  day0 = "hot"
}else if(max(new_p0g1) == new_p0g1[3]){
  day0 = "mild"
}
# Same prob for day3 given hot of day2
c(day0,day1,day2)

## [1] "cold" "hot"  "hot"
```

the most probable report for day 0 to 2 are “cold” “hot” “hot”, and

we have the same probability for day3.

Graphical Model 2

a)

```
mi <- c(-2, 2 ,0)
# height function
height <- function(statep){
  m <- replicate(5,sample(mi,size = 1,prob = statep))
  y <- rep(0,5)
  for (i in 1:5){
    y[i] <- rnorm(1,m,1)
  }
  y
}
# get height
y0 <- height(p0)
y1 <- height(p1)
y2 <- height(p2)
y3 <- height(p3)
datay <- data.frame(y0,y1,y2,y3)
datay
```

```
##          y0        y1          y2          y3
## 1 2.511762 -1.536019  0.03102007 -0.88238828
## 2 2.133691 -2.829667 -0.71771823 -0.03544213
## 3 2.609272 -2.645629 -0.45645577  1.62213386
## 4 1.782848 -1.309067 -0.26663822 -0.52945762
## 5 1.400379 -2.158521 -0.42731199  0.83383134
```

b)

```
m <- c(2,0,-2,-2)
y0 <-replicate(5,rnorm(1,m[1],1))
y1 <-replicate(5,rnorm(1,m[2],1))
y2 <-replicate(5,rnorm(1,m[3],1))
y3 <-replicate(5,rnorm(1,m[4],1))
data2y<- data.frame(y0,y1,y2,y3)
data2y

##          y0         y1         y2         y3
## 1 1.6746606 -0.7783034 -1.6931646 -1.7411468
## 2 1.3366571 -0.6038038  0.1484034 -3.7888110
## 3 0.9875342  1.2656318 -2.1430355 -1.2360920
## 4 1.7316862 -1.1488038 -1.0097901 -0.8994421
## 5 2.4540002 -0.9465946 -1.6004229 -1.1189140
```

c)

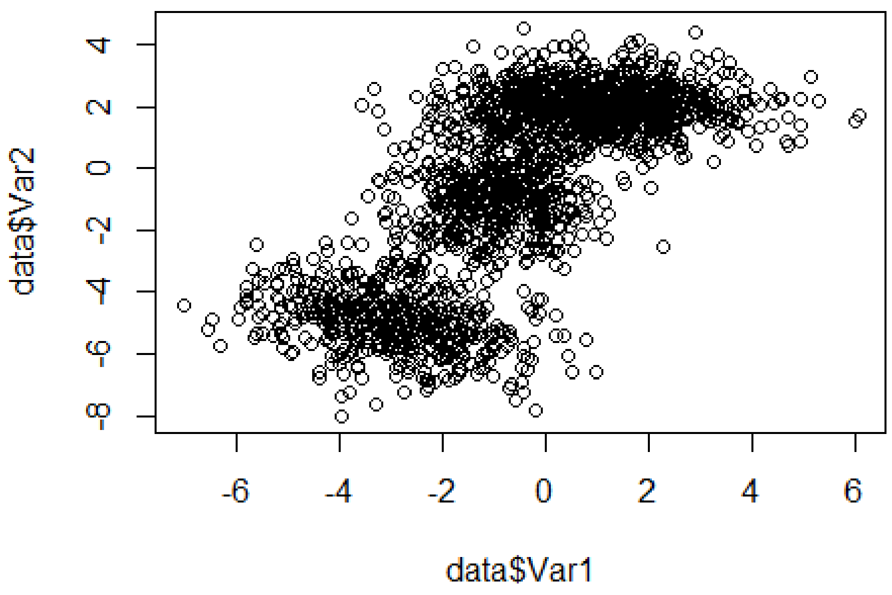
```
Y0 =0.7
Y1 =1.5
Y2 =-1.8
Y3 =-1

#p(y0|x0)
d0 <- rep(0,3)
d0[1] <- dnorm(Y0,-2,1)
d0[2] <- dnorm(Y0,0,1)
d0[3] <- dnorm(Y0,2,1)
#p(y1|x1)
d1 <- rep(0,3)
d1[1] <- dnorm(Y1,-2,1)
d1[2] <- dnorm(Y1,0,1)
d1[3] <- dnorm(Y1,2,1)
#p(y2|x2)
d2 <- rep(0,3)
d2[1] <- dnorm(Y2,-2,1)
d2[2] <- dnorm(Y2,0,1)
d2[3] <- dnorm(Y2,2,1)
#p(y1|x1)
d3 <- rep(0,3)
d3[1] <- dnorm(Y3,-2,1)
d3[2] <- dnorm(Y3,0,1)
d3[3] <- dnorm(Y3,2,1)
# p(x0)*p(y0|x0)*p(x1|x0)*p(y1|x1)*p(x2|x1)*p(y2|x2)*p(x3|x2)*p(y3|x3)
```

```
mp <- p0*d0*pgiven*d1*pgiven*d2*pgiven*d3
sum(mp[,1])

## [1] 4.060199e-06

sum(mp[,2])

## [1] 9.549811e-06

sum(mp[,3])

## [1] 4.031672e-06
```

Gaussian Mixture Model

a)

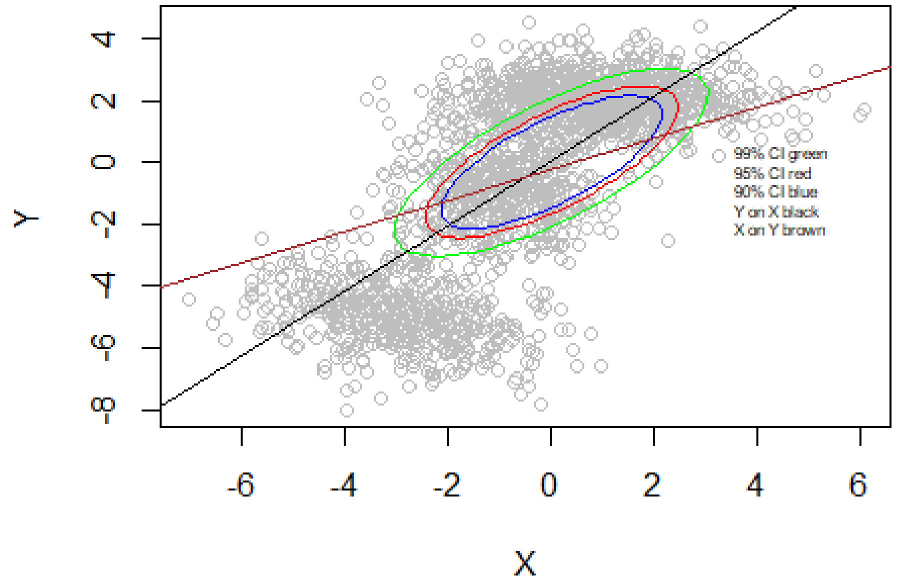
```
rm(list = ls())
library("readxl")
data <- read_excel("gmm_data.xlsx")
plot(data$Var1,data$Var2)
```



the number of clusters is 3.

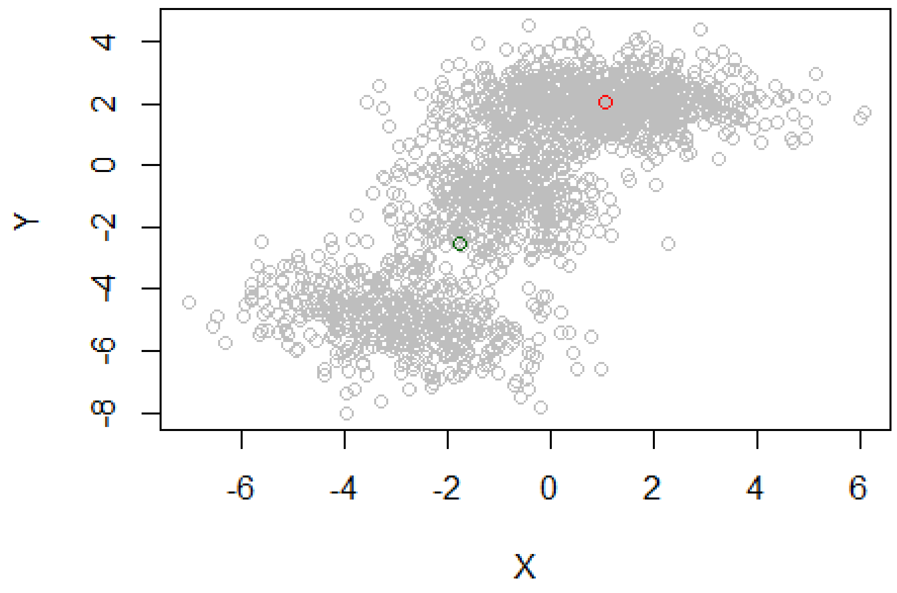
b)

```
library(fMultivar)

## 载入需要的程辑包：timeDate
## 载入需要的程辑包：timeSeries

## 载入需要的程辑包：fBasics

msnFit(data)

##
## Title:
##  Skew Normal Parameter Estimation
##
## Call:
##  msnFit(x = data)
##
## Model:
##  Skew Normal Distribution
##
## Estimated Parameter(s):
## $beta
##          Var1     Var2
## [1,] 1.455257 3.191867
##
## $Omega
##           Var1     Var2
## Var1  8.146794 11.70847
## Var2 11.708474 22.46951
##
## $alpha
##        Var1        Var2
##  -0.8598872 -10.2055688
##
##
## Description:
##  Mon Feb 28 19:07:39 2022 by user: 11193

library(ellipse)

##
## 载入程辑包：'ellipse'

## The following object is masked from 'package:graphics':
##
##     pairs

rho = cor(data)
y_on_x <- lm(data$Var2 ~ data$Var1)
x_on_y <- lm(data$Var1 ~ data$Var2)
plot_legend <- c("99% CI green", "95% CI red","90% CI blue",
                 "Y on X black", "X on Y brown")
plot(data, xlab = "X", ylab = "Y",col = "grey")
lines(ellipse(rho), col="red")
lines(ellipse(rho, level = .99), col="green")
lines(ellipse(rho, level = .90), col="blue")
abline(y_on_x)
abline(x_on_y, col="brown")
legend(3,1,legend=plot_legend,cex = .5, bty = "n")
```



c)

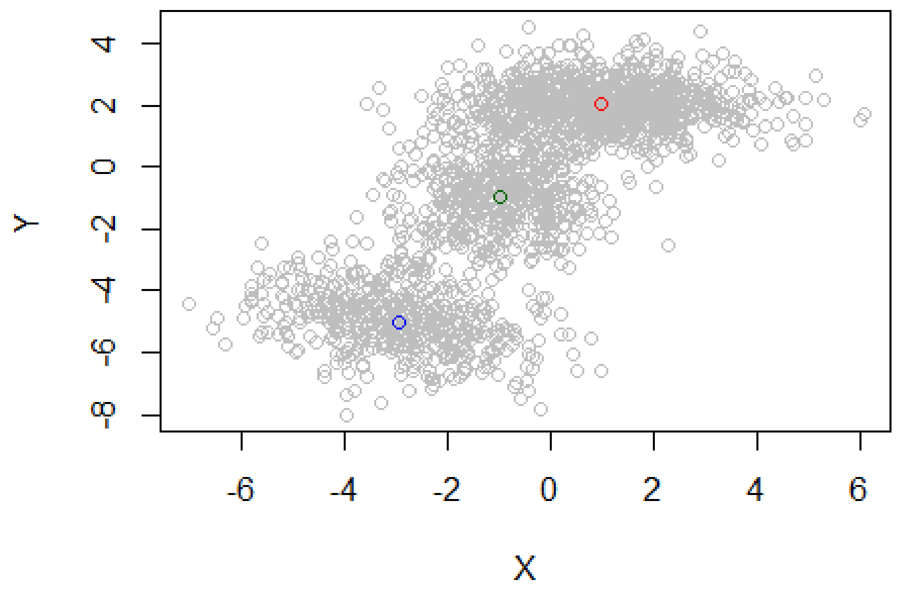
```
library(MGMM)
d <- as.matrix(data)
K2GMM <- FitGMM(d,k=2)

## Objective increment:  10.8
## Objective increment:  0.793
## Objective increment:  0.21
## Objective increment:  0.184
## Objective increment:  0.218
## Objective increment:  0.27
## Objective increment:  0.336
## Objective increment:  0.418
## Objective increment:  0.519
## Objective increment:  0.644
## Objective increment:  0.799
## Objective increment:  0.99
## Objective increment:  1.23
## Objective increment:  1.52
## Objective increment:  1.89
## Objective increment:  2.35
## Objective increment:  2.93
## Objective increment:  3.67
## Objective increment:  4.6
## Objective increment:  5.77
## Objective increment:  7.21
## Objective increment:  8.89
## Objective increment:  10.6
## Objective increment:  12
## Objective increment:  12.2
## Objective increment:  10.8
## Objective increment:  8.22
## Objective increment:  5.53
## Objective increment:  3.51
## Objective increment:  2.22
## Objective increment:  1.42
## Objective increment:  0.928
## Objective increment:  0.614
## Objective increment:  0.411
## Objective increment:  0.276
## Objective increment:  0.187
## Objective increment:  0.127
## Objective increment:  0.0864
## Objective increment:  0.059
## Objective increment:  0.0403
## Objective increment:  0.0276
## Objective increment:  0.0189
## Objective increment:  0.0129
## Objective increment:  0.00888
## Objective increment:  0.0061
## Objective increment:  0.00419
## Objective increment:  0.00288
## Objective increment:  0.00198
## Objective increment:  0.00136
## Objective increment:  0.000935
## Objective increment:  0.000643
## Objective increment:  0.000442
## Objective increment:  0.000304
## Objective increment:  0.000209
## Objective increment:  0.000144
## Objective increment:  9.91e-05
## Objective increment:  6.82e-05
## Objective increment:  4.69e-05
## Objective increment:  3.23e-05
## Objective increment:  2.22e-05
## Objective increment:  1.53e-05
## Objective increment:  1.05e-05
## Objective increment:  7.25e-06
## Objective increment:  4.99e-06
## Objective increment:  3.43e-06
## Objective increment:  2.36e-06
## Objective increment:  1.63e-06
## Objective increment:  1.12e-06
## Objective increment:  7.71e-07
## 68 update(s) performed before reaching tolerance limit.

plot(data, xlab = "X", ylab = "Y",col = "grey")
points(K2GMM@Means[[1]][1],K2GMM@Means[[1]][2],col = "red")
points(K2GMM@Means[[2]][1],K2GMM@Means[[2]][2],col = "dark green")
```



d)

```
K3GMM <- FitGMM(d,k=3)

## Objective increment:  17.1
## Objective increment:  3.64
## Objective increment:  1.16
## Objective increment:  0.414
## Objective increment:  0.163
## Objective increment:  0.0716
## Objective increment:  0.0356
## Objective increment:  0.02
## Objective increment:  0.0126
## Objective increment:  0.00851
## Objective increment:  0.00604
## Objective increment:  0.0044
## Objective increment:  0.00325
## Objective increment:  0.00241
## Objective increment:  0.0018
## Objective increment:  0.00134
## Objective increment:  0.000998
## Objective increment:  0.000743
## Objective increment:  0.000553
## Objective increment:  0.000412
## Objective increment:  0.000306
## Objective increment:  0.000228
## Objective increment:  0.000169
## Objective increment:  0.000126
## Objective increment:  9.32e-05
## Objective increment:  6.91e-05
## Objective increment:  5.13e-05
## Objective increment:  3.8e-05
## Objective increment:  2.82e-05
## Objective increment:  2.09e-05
## Objective increment:  1.55e-05
## Objective increment:  1.15e-05
## Objective increment:  8.52e-06
## Objective increment:  6.32e-06
## Objective increment:  4.68e-06
## Objective increment:  3.47e-06
## Objective increment:  2.57e-06
## Objective increment:  1.9e-06
## Objective increment:  1.41e-06
## Objective increment:  1.05e-06
## Objective increment:  7.74e-07
## 40 update(s) performed before reaching tolerance limit.
```

```r
plot(data, xlab = "X", ylab = "Y",col = "grey")
points(K3GMM@Means[[1]][1],K3GMM@Means[[1]][2],col = "red")
points(K3GMM@Means[[2]][1],K3GMM@Means[[2]][2],col = "dark green")
points(K3GMM@Means[[3]][1],K3GMM@Means[[3]][2],col = "blue")
```



e)

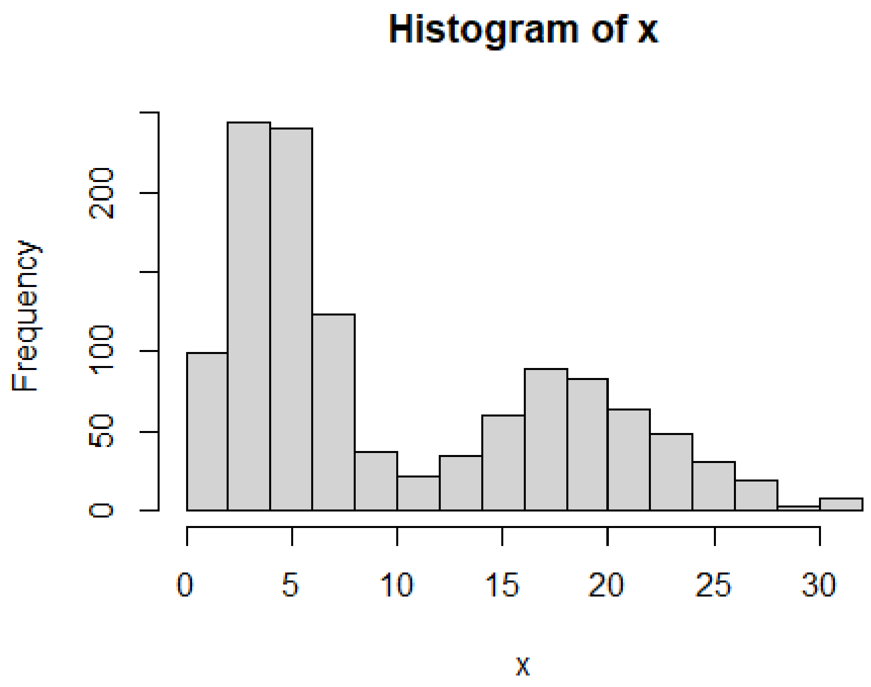
```
# Sigma_i = dx * d covariance matrix
si <- diff(data$Var1) %*% diff(rho)
si[1:20,]
```

```
##             Var1       Var2
##  [1,]  0.2714496 -0.2714496
##  [2,]  0.7881399 -0.7881399
##  [3,]  0.7121274 -0.7121274
##  [4,] -0.8031689  0.8031689
##  [5,] -0.6439405  0.6439405
##  [6,] -0.3233667  0.3233667
##  [7,]  0.8456699 -0.8456699
##  [8,] -1.2999374  1.2999374
##  [9,]  1.1007313 -1.1007313
## [10,]  0.4108798 -0.4108798
## [11,]  0.1088834 -0.1088834
## [12,] -0.4082431  0.4082431
## [13,]  0.5513425 -0.5513425
## [14,] -0.1791793  0.1791793
## [15,]  0.5668767 -0.5668767
## [16,] -1.3834752  1.3834752
## [17,]  0.2978884 -0.2978884
## [18,] -0.1558065  0.1558065
## [19,]  1.0285411 -1.0285411
## [20,] -0.3159542  0.3159542
```

Poisson Mixture Model

1)

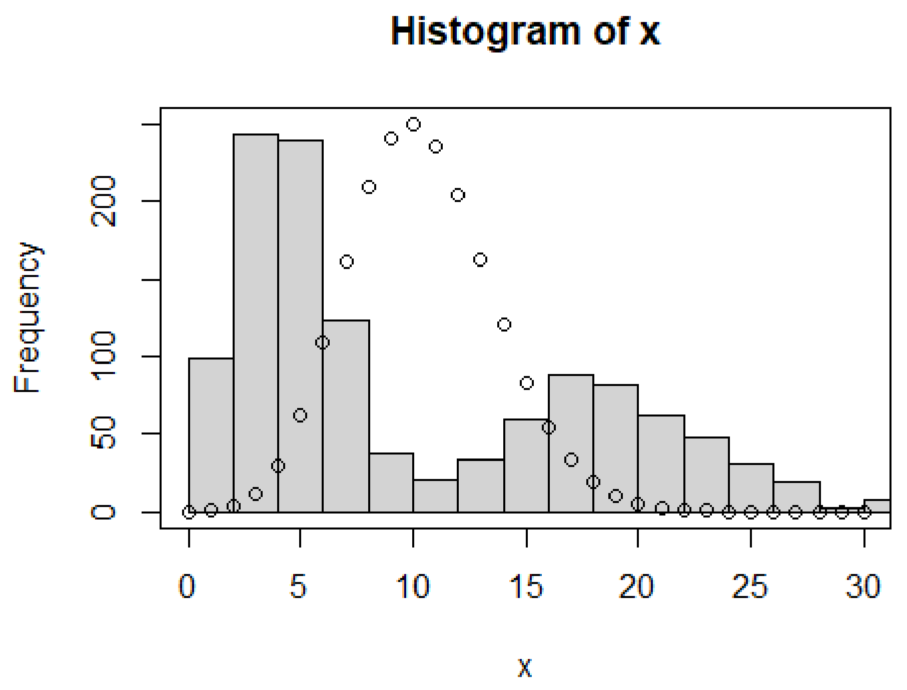
```
rm(list = ls())

library("readxl")
data <- read_excel("poisson_data.xlsx")
x <- data$X
hist(x)
```



From the plot it may fit two distribution from 1-10 and 10-30

b)

```
library(MASS)
lambda <- fitdistr(x,densfun="Poisson")
lambda

##      lambda
##   10.37833333
##  ( 0.09299791)

a <- 0:30
b <- dpois(a,lambda$estimate)
hist(x,xlim=c(0,30),ylim= c(0,250))
par(new=TRUE)
plot(a,b,yaxt="n",xaxt="n",xlab="",ylab="")
```



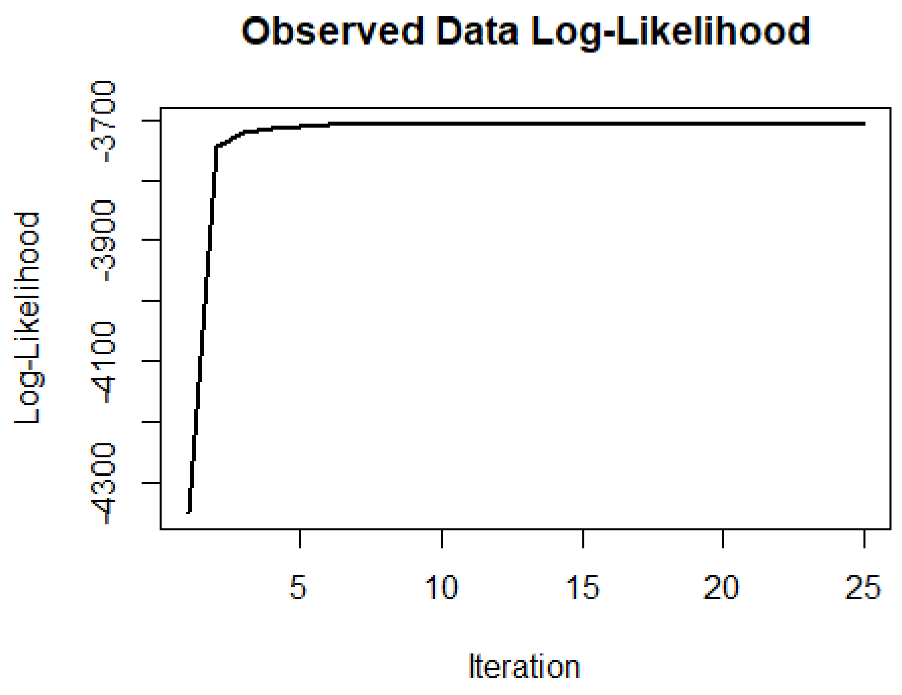
a simple poisson distribution is not a good fit.

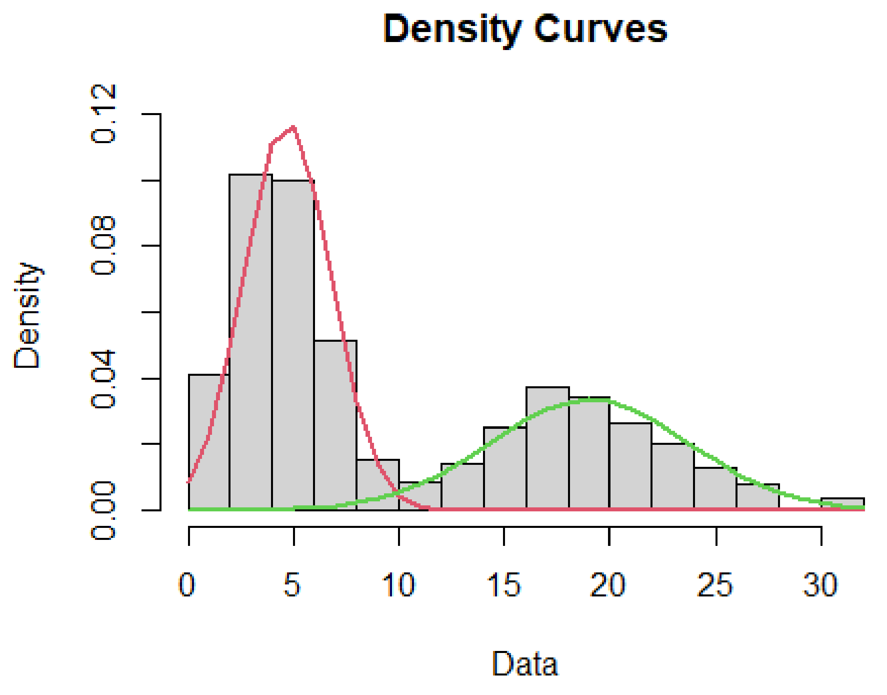
c)

```
library(mixtools)

## mixtools package, version 1.2.0, Released 2020-02-05
## This package is based upon work supported by the National Science Fo
undation under Grant No. SES-0518772.

##
## 载入程辑包：'mixtools'

## The following object is masked from 'package:ellipse':
##
##     ellipse

mixture <- normalmixEM(x,lambda = 0.092997,k=2)

## number of iterations= 24

summary(mixture)

## summary of normalmixEM object:
##           comp 1    comp 2
## lambda 0.604878  0.395122
## mu     4.722448 19.036712
## sigma  2.052662  4.708305
## loglik at estimate:  -3707.207
```

d)

```
plot(mixture, density=TRUE)
```





e)

the single poisson distribution can not fit the data very well, the mixture model give two poisson distribution and separate the data into two modle, which fit the data better.