Assignment III - Poisson Distribution and Hidden Markov Chains

106218 - Getrude Gichuhi

2022-07-28

Question 1

Let X be a random variable which is distributed as a mixture of two distributions with expectations μ1, μ2, and variances σ12 and σ22 , respectively, where the mixing parameters are δ1 and δ2 with δ1 + δ2 = 1.

a) Show that

$$Var(X) = \delta 1\sigma 1^2 + \delta 2\sigma 2^2 + \delta 1\delta 2(\mu 1 - \mu 2)^2$$

$$Var(X) = E(X^2) - \left(E(X)\right)^2$$

$$E(X^2) = Var(X) + \left(E(X)\right)^2$$

$$E(x^2) = (\delta 1(\sigma 1^2 + \mu 1^2) + \delta 2(\sigma 2^2 + \mu 2^2)$$

$$\left(E(X)\right)^2 = (\sigma 1\mu 1 + \sigma 2\mu 2)^2$$

$$= \sigma 1^2\mu 1^2 + 2\sigma 1\sigma 2\mu 1\mu 2 + \sigma 2^2\mu 2^2$$

$$Var(X) = \delta 1(\sigma 1^2 + \mu 1^2) + \delta 2(\sigma 2^2 + \mu 2^2) - (\sigma 1^2\mu 1^2 + 2\sigma 1\sigma 2\mu 1\mu 2 + \sigma 2^2\mu 2^2)$$

$$= \delta 1\sigma 1^2 + \delta 2\sigma 2^2 + \delta 1\mu 1^2(1 - \delta 1) + \delta 2\mu 2^2(1 - \delta 2) - 2\delta 1\delta 2\mu 1\mu 2$$

$$\delta 1 + \delta 2 = 1 \Rightarrow \delta 1(1 - \delta 1) = \delta 2(1 - \delta 2) = \delta 1\delta 2$$

$$Var(x) = \delta 1\sigma 1^2 + \delta 2\sigma 2^2 + \delta 1\delta 2\mu 1^2 + \delta 1\delta 2\mu^2 - 2\delta 1\delta 2\mu 1\mu 2$$

$$\delta 1\sigma 1^2 + \delta 2\sigma 2^2 + \delta 1\delta 2(\mu 1^2 + \mu^2 - 2\mu 1\mu 2)$$

$$\delta 1\sigma 1^2 + \delta 2\sigma 2^2 + \delta 1\delta 2(\mu 1 - \mu 2)^2$$

Show that a mixture of two Poisson distributions, Po(λ1) and Po(λ2), with λ1 6= λ2, is over dispersed, that is Var(X) > E(X).

$$\mu = \sigma^2 = \lambda$$

$$Var(X) = \delta 1\lambda 1 + \delta 2\lambda 2 + \delta 1\delta 2(\lambda 1 - \lambda 2)^2$$

$$= E(X) + \delta 1\delta 2(\lambda - \lambda 2)^2$$

Therefore

$$Var(X) > E(X) = \lambda 1 = \lambda 2$$

Question 2

Write a set of R functions that generates and executes the scripts: dpoismix(x,lambda,delta), ppoismix(q,lambda,delta), qpoismix(p,lambda,delta), rpoismix(n,lambda,delta), You may use any of the available R functions, such as dpois() and ppois() to construct your functions. The tricky one to do is qpoismix(p,lambda,delta). This should compute the quantile, defined as the smallest non-negative integer x which is such that F(x) ≥ p. For experienced R users: Write qpoismix() so that it works when p is a vector. (b) Use graphics to check and illustrate your functions. In particular verify that the random samples generated using rpoismix() have the required properties.

$$dpois(x, lambda, log = FALSE)$$

```
dpois(0:10,
      lambda =5,
      log =FALSE)

##  [1] 0.006737947 0.033689735 0.084224337 0.140373896 0.175467370
0.175467370
##  [7] 0.146222808 0.104444863 0.065278039 0.036265577 0.018132789
```

$$ppois(q, lambda, lower.tail = TRUE, log.p = FALSE)$$

```
ppois(5,
      lambda =10)

## [1] 0.06708596
```

$$qpois(p, lambda, lower.tail = TRUE, log.p = FALSE)$$

```
qpois(0.5,
      lambda =10)

## [1] 10
```

$$rpois(n, lambda)$$

Question 3

Describe how to use the following R commands: a) %*% (matrix multiplication) - Used to calculate the power of matrix

b)t() (transpose a matrix),Given a matrix of y or x, the t returns the opposite of y or x.

c)solve() (solve a system of linear equations, or invert a matrix)- Used to calculate the inverse of a matrix

d)diag() (extract or replace the diagonal of a matrix, or construct a diagonal matrix) - used to extract or replace the diagonal of a matrix

Then, write a R function statdist(gamma) that computes the stationary distribution, δ, of a stationary m-state Markov chain with transition probability matrix gamma.

```
# With a matrix!
statdist <- function(gamma) {
onesMatrix <- round(matrix(1, nrow(gamma), ncol(gamma)))
onesDiagMatrix <- diag(nrow(gamma))
t(rep(1, nrow(gamma)))%*%solve(onesDiagMatrix - gamma + onesMatrix)
}
# Example
testMatrix <- matrix(c(0.6, 0.2, 0.4, 0.35, 0.35, 0.3, 0.5, 0.2, 0.1), 3, 3,
byrow=T)

statdist(testMatrix)

##           [,1]      [,2]      [,3]
## [1,] 0.5103169 0.2210759 0.2837141
```

QUestion 4

Find out how to use the following R commands:

for() (used for looping) sample() (a very useful function for drawing random samples).

a) Then, write a R function genPoisHMM(n,gamma,lambda) that generates a series of length n from a stationary m-state Poisson HMM with transition probability matrix gamma and Poisson parameters lambda. Regard the following notes and specifications. The function should determine the number of states, m, e.g. by using m <- length(lambda).

```
genPoisHMM <- function(n, gamma, lambda) {
  mvect <- 1: length(lambda)
  state <- numeric(n)
  delta <- statdist(gamma)
  state[1]<- sample(mvect, 1, prob =delta)
  for (i in 2: n) {
    state[i] <- sample(mvect, 1,
                       prob= gamma[state[i-1],])
  }
  rpois(n, lambda=lambda[state])
}
```

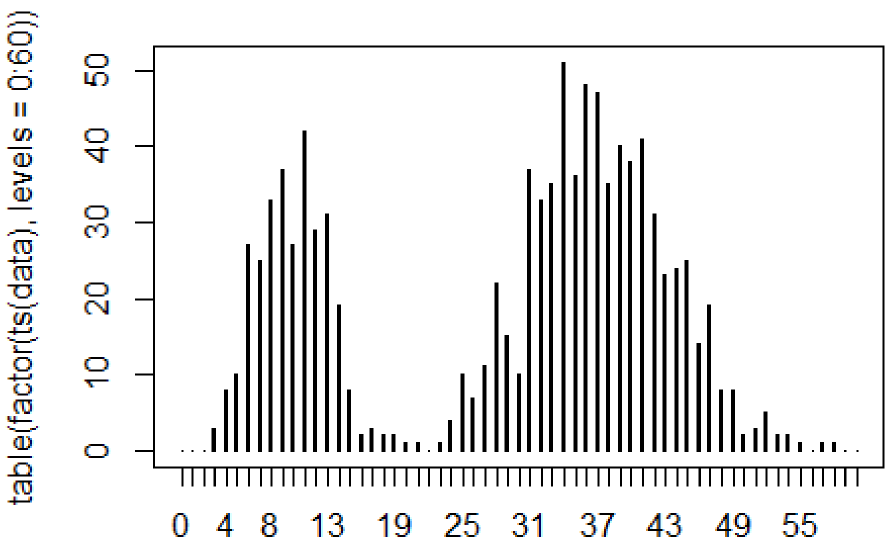
b) To generate the first observation, you will need to compute the stationary distribution, δ. You can use the function statdist() to do this (see Problem 2.5).

c) Try to avoid using if() statements; rather use the function sample() in this application.

d) Test your function by generating a long sequence of observations (say n = 1000) and then check whether the sample mean, variance, histogram, etc. correspond to what you should be getting

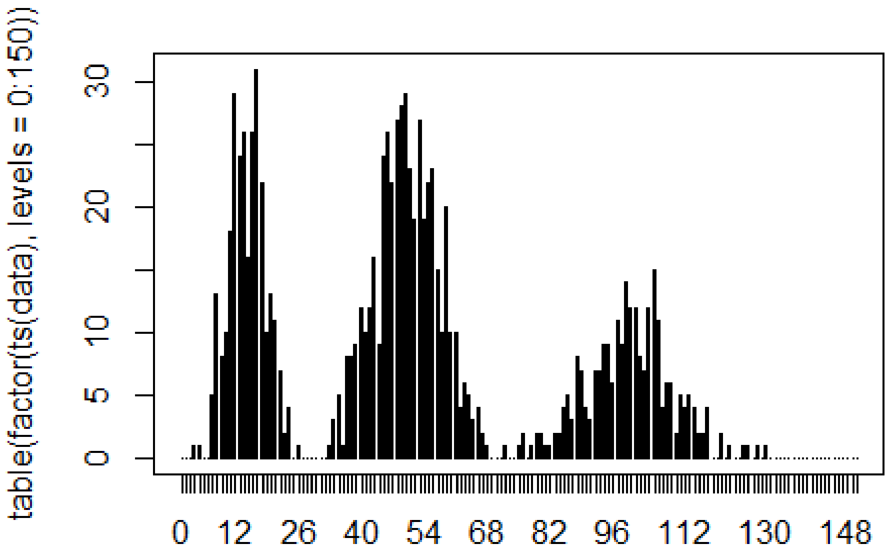
```r
data <- ts(genPoisHMM(
  1000,
  matrix(c(0.5, 0.5,
           0.2, 0.8),
         2, 2, byrow=T),
  c(10, 37)))

plot(table(factor(ts(data), levels = 0:60)))
```




```r
data <- ts(genPoisHMM(
  1000,
  matrix(c(0.4, 0.4, 0.2,
           0.1, 0.6, 0.3,
           0.4, 0.3, 0.3),
         3, 3, byrow=T),
  c(15, 50, 100)))

plot(table(factor(ts(data), levels = 0:150)))
```


table(factor(ts(data), levels = 0:150))
0
5
10
20
30
0
12
26
40
54
68
82
96
112
130
148