短学期作业四

汪利军 *3140105707*

July 7, 2017

Contents

1 MB 3.4 **2**
1.1 (a) . . . 2
1.2 (b) . . . 2
1.3 (c) . . . 3

2 MB 3.7 **4**

3 MB 3.9 **4**

4 MB 3.11 **5**

5 MB 3.13 **6**
5.1 (a) . . . 6
5.2 (b) . . . 7

6 MDL Chapter 10 Worksheet A **8**
6.1 Problem 10.1 . . . 8
6.2 Pronlem 10.2 . . . 9
6.3 Problem 10.3 . . . 9
6.4 Problem 10.4 . . . 9
6.5 Problem 10.5 . . . 10
6.6 Problem 10.6 . . . 10

7 MDL Chapter 12 Worksheet B **10**
7.1 Problem 12.1 . . . 10
7.2 Problem 12.2 . . . 11
7.3 Problem 12.3 . . . 11
7.4 Problem 12.4 . . . 11
7.5 Problem 12.5 . . . 11
7.6 Problem 12.6 . . . 11
7.7 Problem 12.7 . . . 12

1 MB 3.4

1.1 (a)

```r
# generate a random sample
y <- rnorm(100)
# calculate the mean
mean(y)
```

```
## [1] 0.1561251
```

```r
# calculate the standard deviation
sd(y)
```

```
## [1] 1.056849
```

1.2 (b)

```r
av = numeric()
asd = numeric()
for (i in 1:25)
{
  y = rnorm(100)
  av = c(av, mean(y))
  asd = c(asd, sd(y))
}
av
```

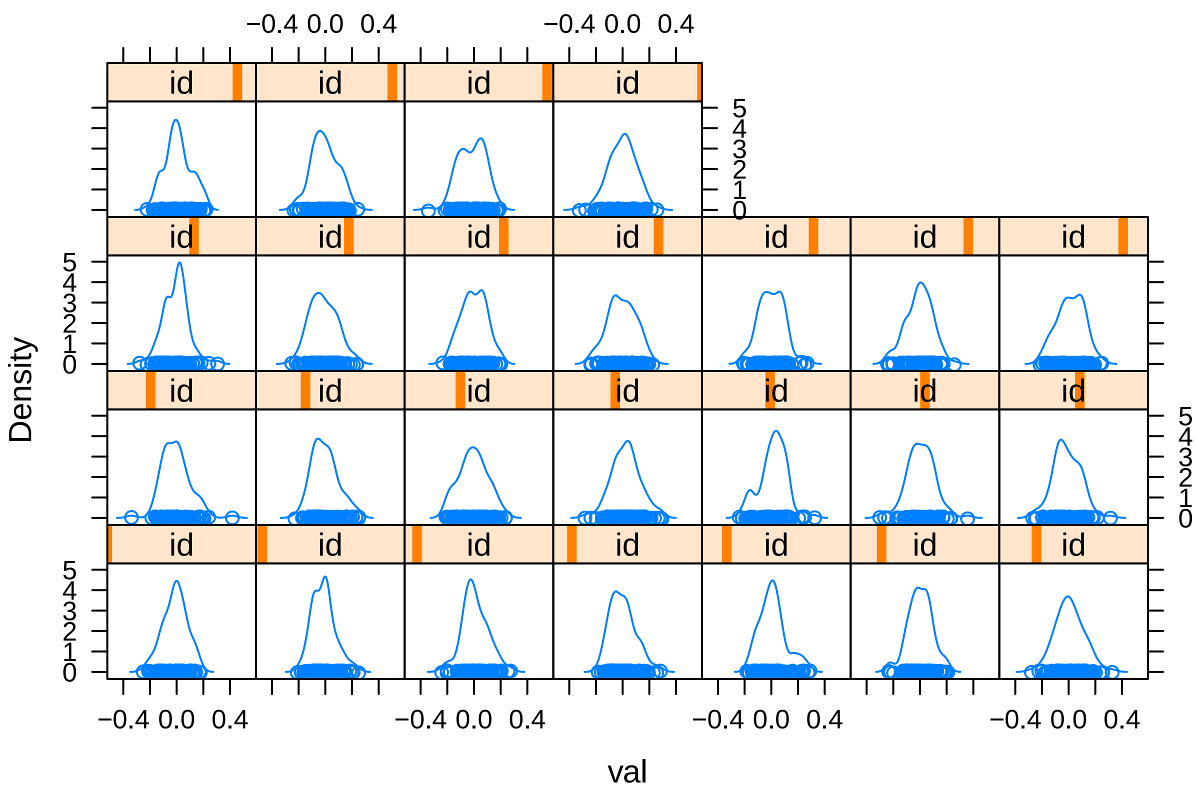
```
##  [1]  0.083072205 -0.036430913  0.041405515 -0.112206390 -0.136894857
##  [6]  0.002343730 -0.146022950 -0.070910922 -0.044158584 -0.118338341
## [11] -0.052574183 -0.063632785 -0.070896875 -0.005993371  0.005133911
## [16] -0.076235705  0.122062588  0.179425263 -0.079339141  0.165387758
## [21]  0.116786871  0.086305712 -0.153162290 -0.138459264  0.260840722
```

```r
asd
```

```
##  [1] 0.8918567 1.0284505 0.9099016 0.9865539 0.9675233 1.0446057 0.9444570
##  [8] 1.0364084 1.0310840 1.1521701 0.9999679 0.8514481 0.9607416 1.0092205
## [15] 1.0632270 0.9652709 0.9297874 0.9724965 0.9630311 0.9496515 0.9343910
## [22] 0.8431814 1.0194252 0.8271572 0.9829843
```

1.3 (c)

```
f <- function()
{
  av = numeric()
  asd = numeric()
  for (i in 1:25)
  {
    y = rnorm(100)
    av = c(av, mean(y))
    asd = c(asd, sd(y))
  }
  res = list(av = av, asd = asd)
  return(res)
}
## 对每个均值重复 nrep 次
nrep = 100
data = sapply(1:nrep, function(x) f()$av)
df = data.frame(val = as.numeric(data), id = rep(1:25, nrep))
library(lattice)
## 在同一张图中作出密度图
densityplot(~val | id, data = df)
```



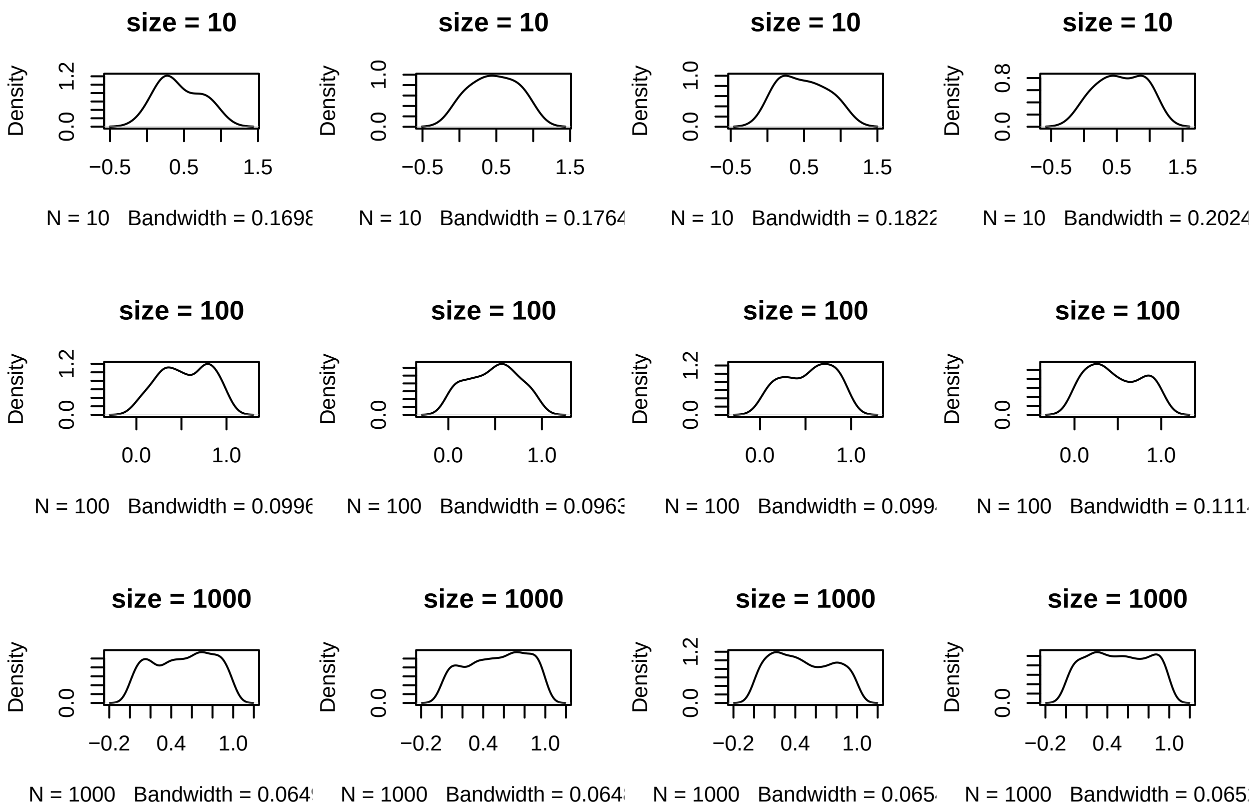
2 MB 3.7

```
x <- runif(10)
x
```

```
##  [1] 0.4956813 0.7677899 0.3945944 0.8842183 0.7478047 0.5714368 0.9914715
##  [8] 0.9865902 0.7722946 0.6449675
```

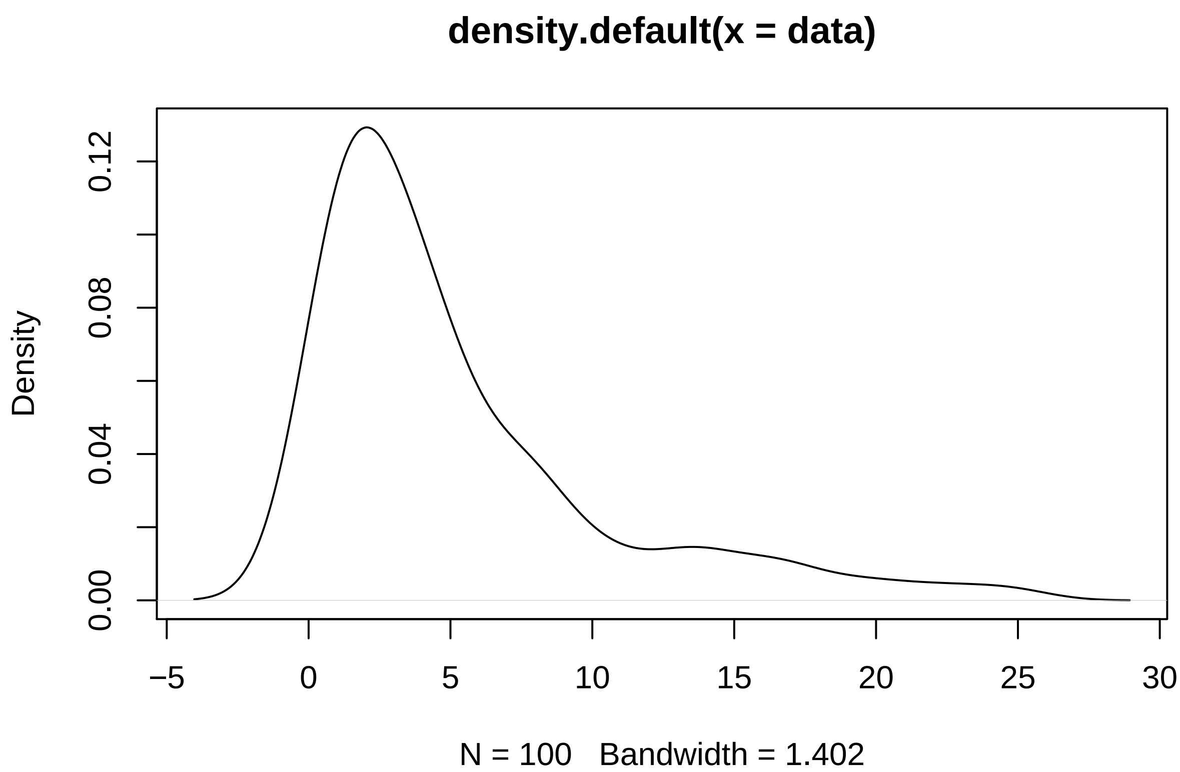
注意到 Exercise 5 并没有涉及 plots，而 Exercise 6 有涉及 plots，猜测应该是重复 Exercise 6 的操作。

```
par(mfrow = c(3, 4))
tmp = sapply(1:4, function(x) plot(density(runif(10)), main = "size = 10"))
tmp = sapply(1:4, function(x) plot(density(runif(100)), main = "size = 100"))
tmp = sapply(1:4, function(x) plot(density(runif(1000)), main = "size = 1000"))
```



3 MB 3.9

```
data = rexp(100, 0.2)
plot(density(data))
```




```r
## 样本均值
mean(data)
```

```
## [1] 5.191844
```

```r
## 总体均值为
1/0.2
```

```
## [1] 5
```

```r
## 相差
mean(data) - 1/0.2
```

```
## [1] 0.1918438
```

4 MB 3.11

```r
data <- c(87, 53, 72, 90, 78, 85, 83)
## 样本均值
mean(data)
```

```
## [1] 78.28571
```

```r
## 样本方差
var(data)
```

```
## [1] 159.9048
```

对于 poisson 分布

```r
x <- rpois(7, 78.3)
mean(x)
```

```
## [1] 81.57143
```

```r
var(x)
```

```
## [1] 76.61905
```

比较均值和方差可以看出，两者非常接近，所以采用 $\lambda = 78.3$ 的 poisson 分布能够很好地模拟这些样本数据。

5 MB 3.13

```r
Pb = matrix(c(0.6, 0.2, 0.2,
              0.2, 0.4, 0.4,
              0.4, 0.3, 0.3), 3, 3, byrow = TRUE)
```

题中给出的平稳分布不满足 $\pi P = \pi$，而且矩阵 Pb 不可逆，则其给出的理论平稳分布并不对。

5.1 (a)

编写 Markov(n, start, transition) 函数，其中 n 是生成的 Markov 链的长度，start 是初始状态，transition 为转移矩阵，返回值是每次访问的状态，由此可以统计每个状态访问的次数。

```r
Markov <- function(n, start = 0, transition)
{
  ## states: 0 1 2 ... size-1
  size = dim(transition)[1]
  p0 = numeric(size)
  p0[start + 1] = 1
  p = t(p0)
  res = c(n)
  for (i in 1:n)
  {
    p = p %*% Pb
    # 产生 [0,1] 随机数
    r = runif(1, 0, sum(p))
    pp = cumsum(p)
```

```
    res[i] = which(r < pp)[1] - 1
  }
  return(res)
}
```

```
xx = Markov(1000, 0, Pb)
freq.tab = table(xx)
dimnames(freq.tab) = list(xx = c("sun", "cloud", "rain"))
prop.table(freq.tab)
```

```
## xx
##   sun cloud  rain
## 0.449 0.273 0.278
```

5.2 (b)

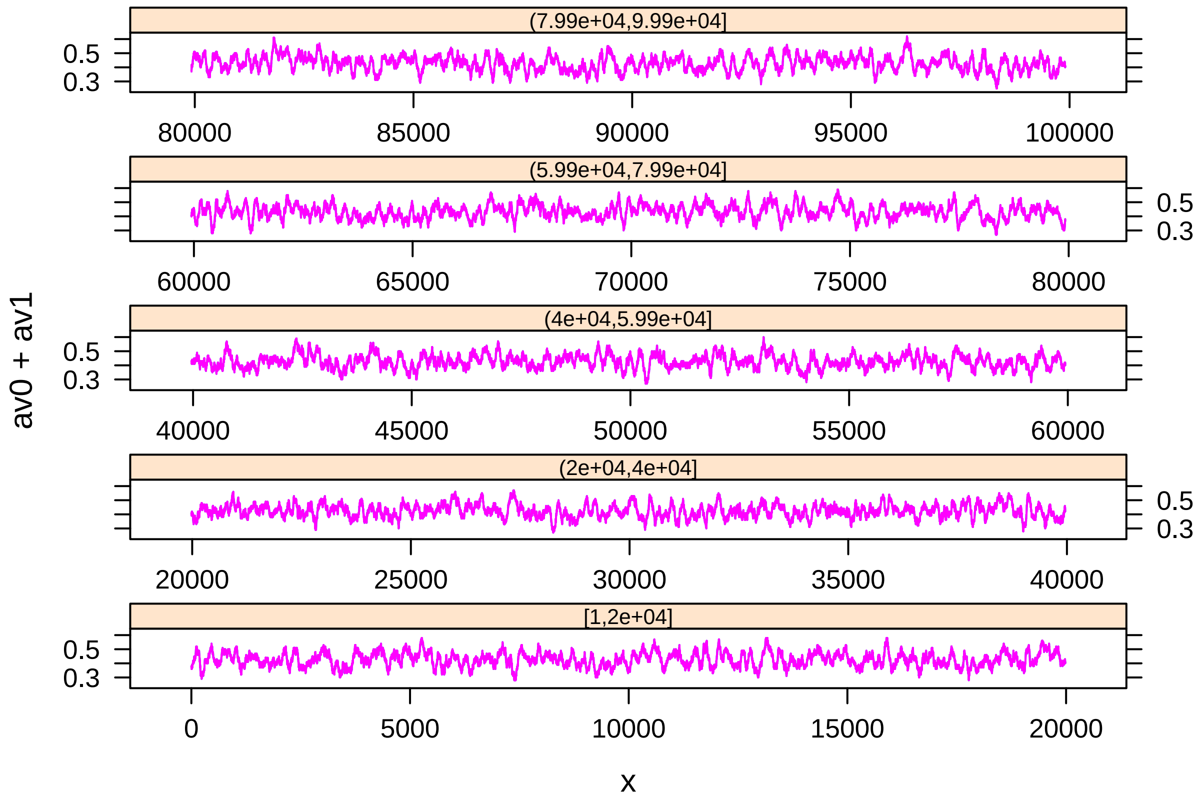
```
library(zoo)
```

```
##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
```

```
plotmarkov <-
  function(n=10000, start=0, window=100, transition=Pb, npanels=5){
    xc2 <- Markov(n, start, transition)
    mav0 <- rollmean(as.integer(xc2==0), window)
    mav1 <- rollmean(as.integer(xc2==0), window)
    npanel <- cut(1:length(mav0), breaks=seq(from=1, to=length(mav0),
                                              length=npanels+1),
                  include.lowest=TRUE)
    df <- data.frame(av0=mav0, av1=mav1, x=1:length(mav0),
                     gp=npanel)
    print(xyplot(av0+av1  ~ x | gp, data=df, layout=c(1,npanels),
                 type="l", par.strip.text=list(cex=0.65),
                 scales=list(x=list(relation="free"))))
}
```

```r
plotmarkov(n = 100000, start = 0, window = 100, transition = Pb, npanels = 5)
```



因给出的转移矩阵不可逆，该问题的平稳性问题无法回答，但是可以看出通过改变 window 的大小对图像会有一些影响。

6 MDL Chapter 10 Worksheet A

```r
A = matrix(c(2, 5,
             3, 4), 2, 2, byrow = T)
I = matrix(c(1, 0,
             0, 1), 2, 2)
```

6.1 Problem 10.1

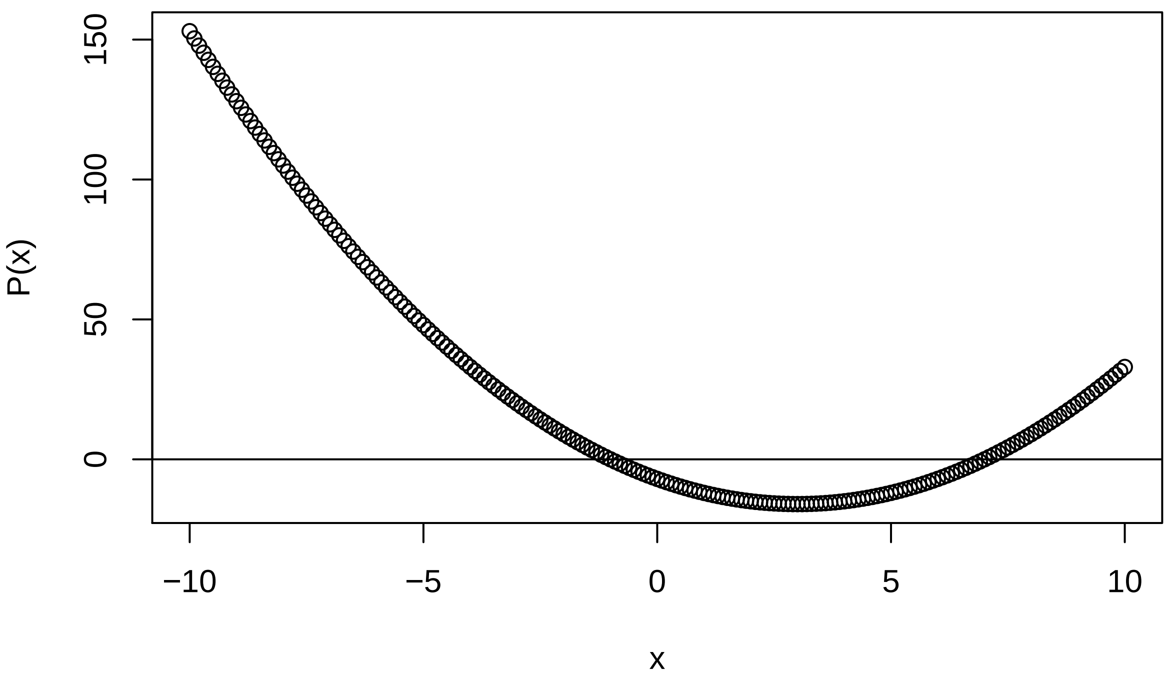
```r
myf <- function(x)
{
  return(det(A-x*I))
}
```

6.2 Pronlem 10.2

```
myf <- function(x)
{
  res = sapply(x, function(xx) det(A-xx*I))
  return(res)
}
```

6.3 Problem 10.3

```
xrange = pretty(c(-10, 10), 200)
Px = myf(xrange)
plot(xrange, Px, xlab = "x", ylab = "P(x)")
abline(h = 0)
```



6.4 Problem 10.4

由上图可以看出两个根位于 $[-5, 0]$ 和 $[5, 10]$ 中

```
root1 = uniroot(myf, interval = c(-5, 0))$root
root2 = uniroot(myf, interval = c(5, 10))$root
c(root1, root2)
```

```
## [1] -0.9999898  6.9999758
```

则两个根为-1.0 和 7.0

6.5 Problem 10.5

$$P(x) = x^2 - 6x - 7$$

```
polyroot(c(-7, -6, 1))
```

```
## [1] -1+0i  7-0i
```

计算得到根为-1 和 7.

6.6 Problem 10.6

```
eigen(A)
```

```
## eigen() decomposition
## $values
## [1]  7 -1
##
## $vectors
##            [,1]       [,2]
## [1,] -0.7071068 -0.8574929
## [2,] -0.7071068  0.5144958
```

特征值为 7 和-1，与前面两问的结果都一致。

7 MDL Chapter 12 Worksheet B

7.1 Problem 12.1

```
rgp <- function(n, mu, sigma, xi)
{
   U = runif(n, 0, 1)
   X = mu + sigma*(U^{-xi}-1)/xi
   return(X)
}
```

7.2 Problem 12.2

```r
data = rgp(1000, 0, 1, 1/4)
head(data)
```

```
## [1] 4.1756502 1.7072699 0.9683411 0.3651328 0.2157702 1.0259134
```

7.3 Problem 12.3

```r
## empirical mean and variance
empirical.mean = mean(data)
empirical.var = var(data)
c(empirical.mean, empirical.var)
```

```
## [1] 1.379293 3.025691
```

7.4 Problem 12.4

```r
## theoretical values
theoretical.mean = 0 + 1/(1-1/4)
theoretical.var = 1^2/((1-1/4)^2*(1-2*1/4))
c(theoretical.mean, theoretical.var)
```

```
## [1] 1.333333 3.555556
```

7.5 Problem 12.5

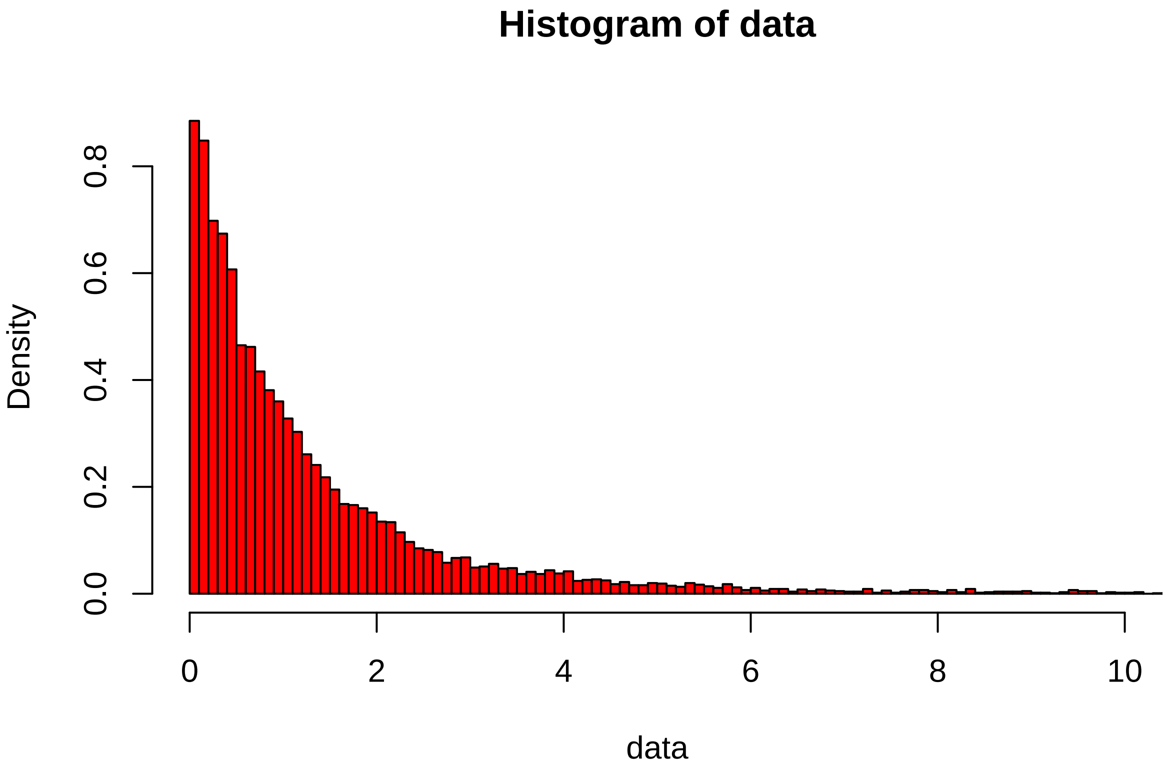
```r
data = rgp(10000, 0, 1, 1/4)
## empirical mean and variance
empirical.mean = mean(data)
empirical.var = var(data)
c(empirical.mean, empirical.var)
```

```
## [1] 1.357386 3.411851
```

7.6 Problem 12.6

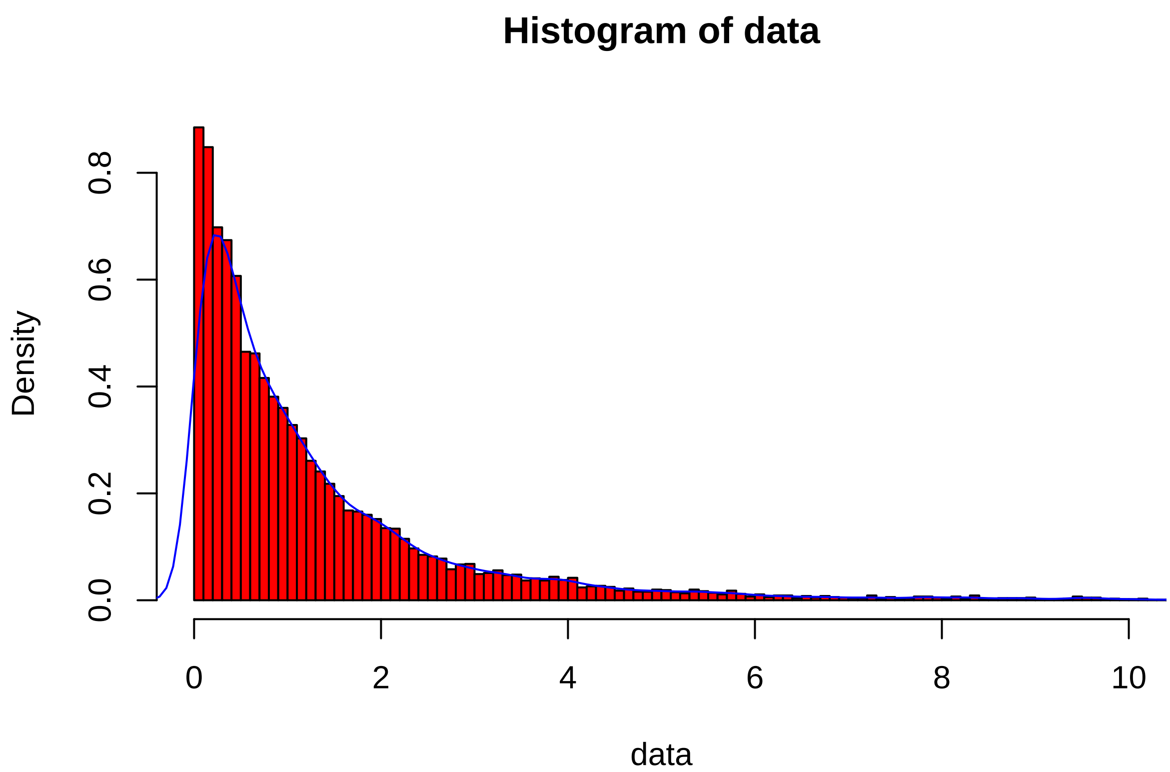
采用上一问中 $n = 10000$ 的样本

```r
hist(data, freq = F, col = "red", xlim = c(0, 10), nclass = 500)
```



7.7 Problem 12.7

```r
hist(data, freq = F, col = "red", xlim = c(0, 10), nclass = 500)
lines(density(data), col = "blue")
```


Histogram of data
Density
0.0
0.2
0.4
0.6
0.8
0
2
4
6
8
10
data