Assignment 1

12111603谭致恒

2023-09-24

R Markdown

Question 1

(a)

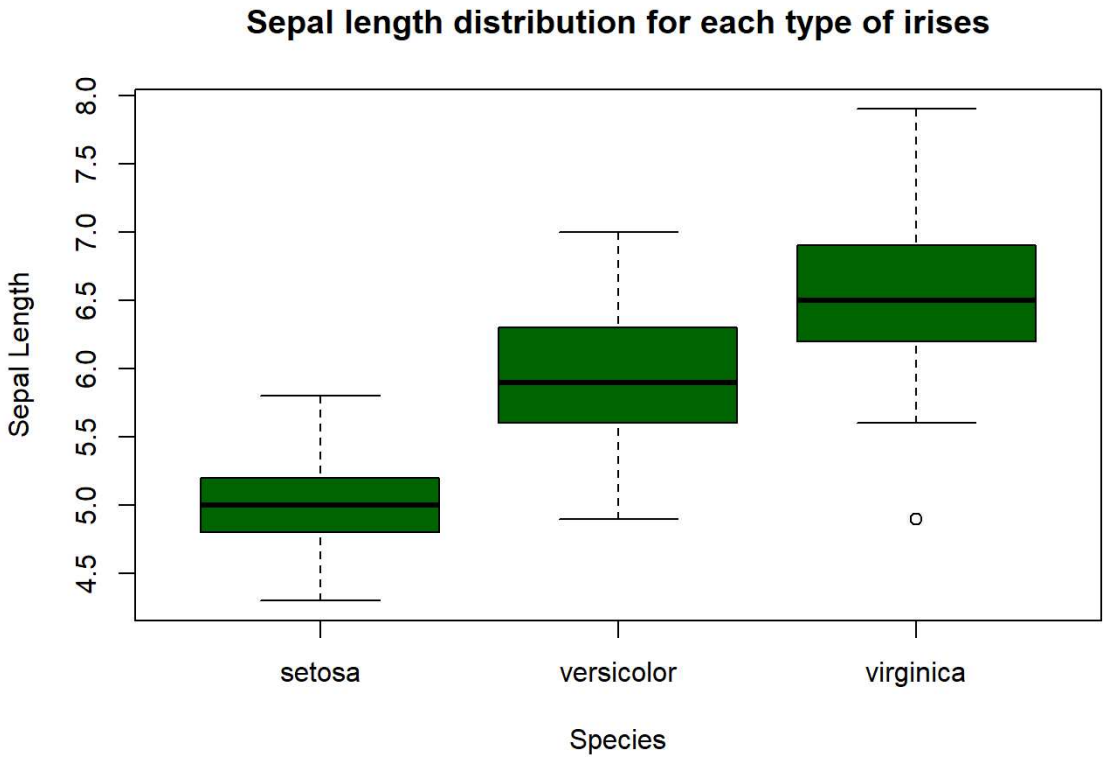
```
#打开iris数据集
library(datasets)
data("iris")
iris_species <- iris$Species
#运用factor
fdata <- factor(iris_species,labels = c("a","b","c"))
table(fdata)
```

```
## fdata
##  a  b  c
## 50 50 50
```

每种花都是50朵。

(b)

```
colors = c("darkgreen","darkgreen","darkgreen")
boxplot(Sepal.Length ~ Species, data = iris,
        main = "Sepal length distribution for each type of irises",
        xlab = "Species", ylab = "Sepal Length",
        col = colors)
```



(c)

首先筛选出长度大于5.5的花

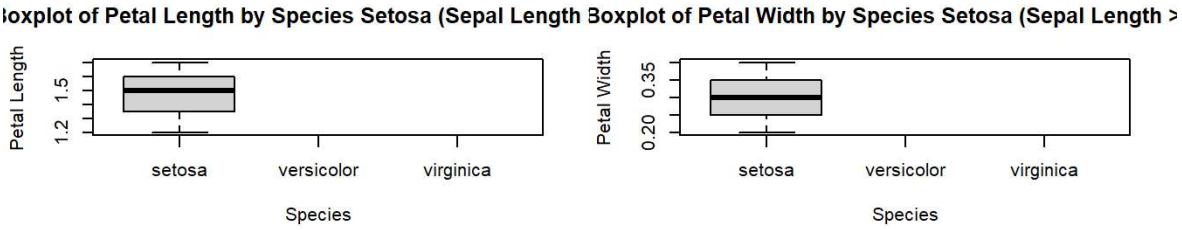
```
filtered_data <- subset(iris, Sepal.Length > 5.5) #筛选＞5.5
filtered_data
```

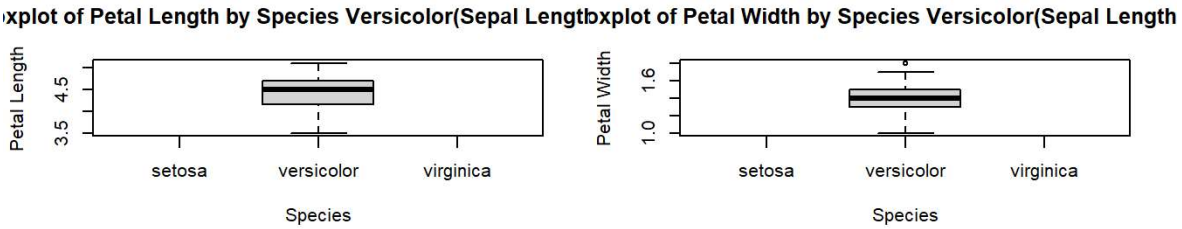
```
##     Sepal.Length Sepal.Width Petal.Length Petal.Width    Species
## 15           5.8         4.0          1.2         0.2     setosa
## 16           5.7         4.4          1.5         0.4     setosa
## 19           5.7         3.8          1.7         0.3     setosa
## 51           7.0         3.2          4.7         1.4 versicolor
## 52           6.4         3.2          4.5         1.5 versicolor
## 53           6.9         3.1          4.9         1.5 versicolor
## 55           6.5         2.8          4.6         1.5 versicolor
## 56           5.7         2.8          4.5         1.3 versicolor
## 57           6.3         3.3          4.7         1.6 versicolor
## 59           6.6         2.9          4.6         1.3 versicolor
## 62           5.9         3.0          4.2         1.5 versicolor
## 63           6.0         2.2          4.0         1.0 versicolor
## 64           6.1         2.9          4.7         1.4 versicolor
## 65           5.6         2.9          3.6         1.3 versicolor
## 66           6.7         3.1          4.4         1.4 versicolor
## 67           5.6         3.0          4.5         1.5 versicolor
## 68           5.8         2.7          4.1         1.0 versicolor
## 69           6.2         2.2          4.5         1.5 versicolor
## 70           5.6         2.5          3.9         1.1 versicolor
## 71           5.9         3.2          4.8         1.8 versicolor
## 72           6.1         2.8          4.0         1.3 versicolor
## 73           6.3         2.5          4.9         1.5 versicolor
## 74           6.1         2.8          4.7         1.2 versicolor
## 75           6.4         2.9          4.3         1.3 versicolor
## 76           6.6         3.0          4.4         1.4 versicolor
## 77           6.8         2.8          4.8         1.4 versicolor
## 78           6.7         3.0          5.0         1.7 versicolor
## 79           6.0         2.9          4.5         1.5 versicolor
## 80           5.7         2.6          3.5         1.0 versicolor
## 83           5.8         2.7          3.9         1.2 versicolor
## 84           6.0         2.7          5.1         1.6 versicolor
## 86           6.0         3.4          4.5         1.6 versicolor
## 87           6.7         3.1          4.7         1.5 versicolor
## 88           6.3         2.3          4.4         1.3 versicolor
## 89           5.6         3.0          4.1         1.3 versicolor
## 92           6.1         3.0          4.6         1.4 versicolor
## 93           5.8         2.6          4.0         1.2 versicolor
## 95           5.6         2.7          4.2         1.3 versicolor
## 96           5.7         3.0          4.2         1.2 versicolor
## 97           5.7         2.9          4.2         1.3 versicolor
## 98           6.2         2.9          4.3         1.3 versicolor
## 100          5.7         2.8          4.1         1.3 versicolor
## 101          6.3         3.3          6.0         2.5  virginica
## 102          5.8         2.7          5.1         1.9  virginica
## 103          7.1         3.0          5.9         2.1  virginica
## 104          6.3         2.9          5.6         1.8  virginica
## 105          6.5         3.0          5.8         2.2  virginica
## 106          7.6         3.0          6.6         2.1  virginica
## 108          7.3         2.9          6.3         1.8  virginica
## 109          6.7         2.5          5.8         1.8  virginica
## 110          7.2         3.6          6.1         2.5  virginica
## 111          6.5         3.2          5.1         2.0  virginica
## 112          6.4         2.7          5.3         1.9  virginica
## 113          6.8         3.0          5.5         2.1  virginica
```

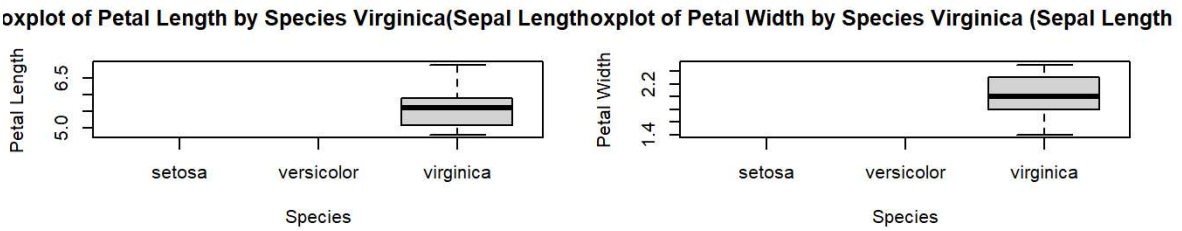
```
## 114          5.7         2.5          5.0         2.0  virginica
## 115          5.8         2.8          5.1         2.4  virginica
## 116          6.4         3.2          5.3         2.3  virginica
## 117          6.5         3.0          5.5         1.8  virginica
## 118          7.7         3.8          6.7         2.2  virginica
## 119          7.7         2.6          6.9         2.3  virginica
## 120          6.0         2.2          5.0         1.5  virginica
## 121          6.9         3.2          5.7         2.3  virginica
## 122          5.6         2.8          4.9         2.0  virginica
## 123          7.7         2.8          6.7         2.0  virginica
## 124          6.3         2.7          4.9         1.8  virginica
## 125          6.7         3.3          5.7         2.1  virginica
## 126          7.2         3.2          6.0         1.8  virginica
## 127          6.2         2.8          4.8         1.8  virginica
## 128          6.1         3.0          4.9         1.8  virginica
## 129          6.4         2.8          5.6         2.1  virginica
## 130          7.2         3.0          5.8         1.6  virginica
## 131          7.4         2.8          6.1         1.9  virginica
## 132          7.9         3.8          6.4         2.0  virginica
## 133          6.4         2.8          5.6         2.2  virginica
## 134          6.3         2.8          5.1         1.5  virginica
## 135          6.1         2.6          5.6         1.4  virginica
## 136          7.7         3.0          6.1         2.3  virginica
## 137          6.3         3.4          5.6         2.4  virginica
## 138          6.4         3.1          5.5         1.8  virginica
## 139          6.0         3.0          4.8         1.8  virginica
## 140          6.9         3.1          5.4         2.1  virginica
## 141          6.7         3.1          5.6         2.4  virginica
## 142          6.9         3.1          5.1         2.3  virginica
## 143          5.8         2.7          5.1         1.9  virginica
## 144          6.8         3.2          5.9         2.3  virginica
## 145          6.7         3.3          5.7         2.5  virginica
## 146          6.7         3.0          5.2         2.3  virginica
## 147          6.3         2.5          5.0         1.9  virginica
## 148          6.5         3.0          5.2         2.0  virginica
## 149          6.2         3.4          5.4         2.3  virginica
## 150          5.9         3.0          5.1         1.8  virginica
```

然后我们绘制3×2的箱线图（每个品种的两个箱线图并排放置）

```
data_setosa <- subset(filtered_data, Species %in% c("setosa"))

data_versicolor <- subset(filtered_data,Species %in% c("versicolor"))

data_virginica <- subset(filtered_data,Species %in% c("virginica"))


# 3×2的图形布局
par(mfrow = c(3, 2))

boxplot(Petal.Length ~ Species, data = data_setosa ,
        main = "Boxplot of Petal Length by Species Setosa (Sepal Length > 5.5)",
        xlab = "Species", ylab = "Petal Length")

boxplot(Petal.Width ~ Species, data = data_setosa,
        main = "Boxplot of Petal Width by Species Setosa (Sepal Length > 5.5)",
        xlab= "Species", ylab = "Petal Width")
boxplot(Petal.Length ~ Species, data = data_versicolor ,
        main = "Boxplot of Petal Length by Species Versicolor(Sepal Length > 5.5)",
        xlab = "Species", ylab = "Petal Length")

boxplot(Petal.Width ~ Species, data =data_versicolor,
        main = "Boxplot of Petal Width by Species Versicolor(Sepal Length > 5.5)",
        xlab= "Species", ylab = "Petal Width")
boxplot(Petal.Length ~ Species, data = data_virginica  ,
        main = "Boxplot of Petal Length by Species Virginica(Sepal Length > 5.5)",
        xlab = "Species", ylab = "Petal Length")

boxplot(Petal.Width ~ Species, data = data_virginica ,
        main = "Boxplot of Petal Width by Species Virginica (Sepal Length > 5.5)",
        xlab= "Species", ylab = "Petal Width")
```







Question 2

(a)

```
data = read.csv("C:/Users/Lenovo/Desktop/R/Chapter 1/F2000.csv")
value = data$marketvalue


# 写一个计算skewness的函数
skewness <- function(input){

average <- mean(input)
for(i in 1:2000){
  input[[i]] <- input[[i]] - average
  #求出x-xbar
}

cubic_sum = 0
for(j in 1:2000){
  cubic_sum = cubic_sum + input[[j]]^3
  #计算立方和
}

square_sum = 0
for(k in 1:2000){
  square_sum = square_sum + input[[k]]^2
  #计算平方和
}

#代入skewness公式进行计算
skewness2 = cubic_sum*sqrt(1999)/square_sum^(3/2)

return(skewness2)

}
 skewness(value)
```

```
## [1] 6.430443
```

```
#经检验，代入skewness公式进行计算得到的偏度与调包得到的结果高度一致，误差小于0.01
```

这2000家公司的marketvalue的偏度约为6.430443。

(b)

```
# taking log:
log_value <- log(value)
skewness_log <- skewness(log_value)
skewness_log
```

```
## [1] 0.03204195
```

故，取对数后的偏度为0.0320419

```
# taking inverse:
inverse_value <- 1/value
skewness_inverse <- skewness(inverse_value)
skewness_inverse
```

```
## [1] 19.92494
```

故，取倒数后的偏度为19.9249364

```
# taking Box&Cox:
Box <- function(x){(x^0.25 - 1)/0.25}
box_value <- Box(value)
skewness_box <- skewness(box_value)
skewness_box
```

```
## [1] 1.381235
```

故，经Box&Cox后的偏度为1.3812352

(c)

```
#确定x轴范围
summary(value)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    0.02    2.72    5.15   11.88   10.60  328.54
```

```
summary(log_value)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -3.912   1.001   1.639   1.690   2.361   5.795
```

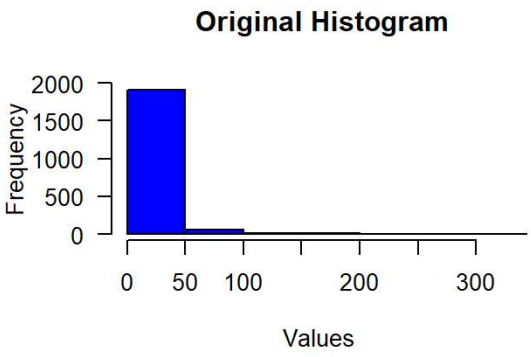
```
summary(inverse_value)
```

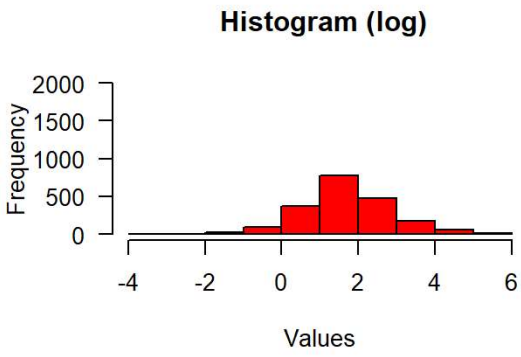
```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##  0.00304  0.09432  0.19417  0.44542  0.36765 50.00000
```

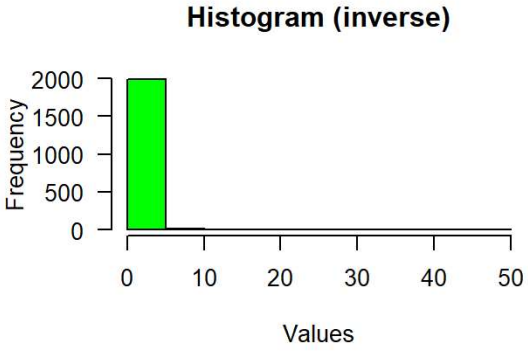
```
summary(box_value)
```

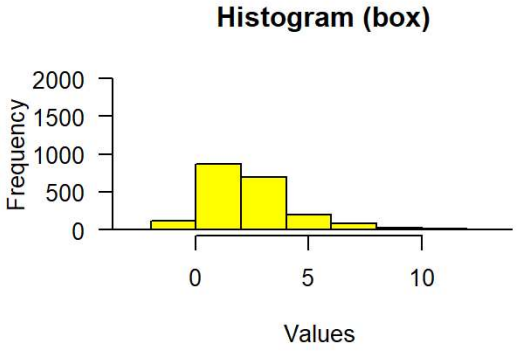
```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  -2.496   1.137   2.026   2.384   3.218  13.030
```

```
#2×2的图
par(mfrow = c(2, 2))

# 绘制直方图
hist(value,
     col = "blue",                  # 设置柱子颜色为蓝色
     border = "black",              # 设置边界线颜色为黑色
     main = " Original Histogram",            # 设置标题
     xlab = "Values",               # 设置x轴标签
     ylab = "Frequency",            # 设置y轴标签
     breaks = 10,                    # 设置柱子的数量
     xlim = c(0, 330),               # 设置x轴范围
     ylim = c(0, 2000),                # 设置y轴范围
     axes = TRUE,                   # 显示坐标轴
     las = 1)                       # 设置刻度标签方向为水平

# 绘制直方图，带有自定义设置
hist(log_value,
     col = "red",                 # 设置柱子颜色为蓝色
     border = "black",              # 设置边界线颜色为黑色
     main = " Histogram (log)",            # 设置标题
     xlab = "Values",               # 设置x轴标签
     ylab = "Frequency",            # 设置y轴标签
     breaks = 10,                    # 设置柱子的数量
     xlim = c(-4, 6),               # 设置x轴范围
     ylim = c(0, 2000),                # 设置y轴范围
     axes = TRUE,                   # 显示坐标轴
     las = 1)                       # 设置刻度标签方向为水平

hist(inverse_value,
     col = "green",                 # 设置柱子颜色为蓝色
     border = "black",              # 设置边界线颜色为黑色
     main = "Histogram (inverse)",            # 设置标题
     xlab = "Values",               # 设置x轴标签
     ylab = "Frequency",            # 设置y轴标签
     breaks = 10,                    # 设置柱子的数量
     xlim = c(0, 50),               # 设置x轴范围
     ylim = c(0, 2000),                # 设置y轴范围
     axes = TRUE,                   # 显示坐标轴
     las = 1)                       # 设置刻度标签方向为水平

hist(box_value,
     col = "yellow",                 # 设置柱子颜色为蓝色
     border = "black",              # 设置边界线颜色为黑色
     main = " Histogram (box)",            # 设置标题
     xlab = "Values",               # 设置x轴标签
     ylab = "Frequency",            # 设置y轴标签
     breaks = 10,                    # 设置柱子的数量
     xlim = c(-3, 14),               # 设置x轴范围
     ylim = c(0, 2000),                # 设置y轴范围
     axes = TRUE,                   # 显示坐标轴
     las = 1)                       # 设置刻度标签方向为水平
```









(d)

```
missing = is.na(data$profits) #找到缺失profits的公司
missing2 <- data[missing,c("sales")] #调出他们的sales
missing2
```

```
## [1] 5.40 5.68 3.50 5.50 4.48
```

```
summary(missing2) #fivenumber
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   3.500   4.480   5.400   4.912   5.500   5.680
```

(e)

```
country <- factor(data$country)



for (i in 1:2000) {
  if(country[i] == "Hong Kong/China"){
    country[i] <- "China"
  }
  if(country[i] == "Taiwan"){
    country[i] <- "China"
  }
  if(country[i] == "Kong/China"){
    country[i] <- "China"
  }
}

nlevels(country)#共有多少个国家
```

```
## [1] 61
```

这里我们发现有“Taiwan，Hongkong/China 和 Kong/China”三个地区都应归为China这个国家。因此实际的国家数量应减去三个，即一共有58个国家。（此时上述三个地区的公司数量均为0个）

```
table(country)#每个国家分别有多少公司
```

```
## country
##                       Africa                    Australia
##                            2                           37
##    Australia/ United Kingdom                      Austria
##                            2                            8
##                      Bahamas                      Belgium
##                            1                            9
##                      Bermuda                       Brazil
##                           20                           15
##                       Canada                Cayman Islands
##                           56                            5
##                        Chile                        China
##                            4                           84
##               Czech Republic                      Denmark
##                            2                           10
##                      Finland                       France
##                           11                           63
##       France/ United Kingdom                      Germany
##                            1                           65
##                       Greece                Hong Kong/China
##                           12                            0
##                      Hungary                        India
##                            2                           27
##                    Indonesia                      Ireland
##                            7                            8
##                      Islands                       Israel
##                            1                            8
##                        Italy                        Japan
##                           41                          316
##                       Jordan                   Kong/China
##                            1                            0
##                        Korea                      Liberia
##                            4                            1
##                   Luxembourg                     Malaysia
##                            2                           16
##                       Mexico                  Netherlands
##                           17                           28
##  Netherlands/ United Kingdom                  New Zealand
##                            2                            1
##                       Norway                     Pakistan
##                            8                            1
##       Panama/ United Kingdom                         Peru
##                            1                            1
##                  Philippines                       Poland
##                            2                            1
##                     Portugal                       Russia
##                            7                           12
##                    Singapore                 South Africa
##                           16                           15
##                  South Korea                        Spain
##                           45                           29
##                       Sweden                  Switzerland
##                           26                           34
##                       Taiwan                     Thailand
##                            0                            9
```

```
##                       Turkey               United Kingdom
##                           12                          137
##    United Kingdom/ Australia  United Kingdom/ Netherlands
##                            1                            1
## United Kingdom/ South Africa                United States
##                            1                          751
##                    Venezuela
##                            1
```

```
total = numeric(58) #创建一个58个0的数组以用来存储每个国家有多少公司

mean = array() #创建一个空数组以用来存储每个国家的资产均值
median = array() #创建一个空数组以用来存储每个国家资产的中位数

name <- unique(country) #找到58个国家的名字
name
```

```
##  [1] United States                United Kingdom
##  [3] Japan                        Switzerland
##  [5] Netherlands                  Netherlands/ United Kingdom
##  [7] France                       Germany
##  [9] Italy                        Spain
## [11] South Korea                  China
## [13] Bermuda                      Canada
## [15] Finland                      Australia
## [17] Russia                       Belgium
## [19] Norway                       Australia/ United Kingdom
## [21] Brazil                       Denmark
## [23] Sweden                       United Kingdom/ Australia
## [25] Ireland                      India
## [27] Panama/ United Kingdom       Singapore
## [29] South Africa                 Austria
## [31] Mexico                       Portugal
## [33] Thailand                     Malaysia
## [35] Turkey                       Greece
## [37] Luxembourg                   Islands
## [39] Liberia                      Indonesia
## [41] Israel                       United Kingdom/ Netherlands
## [43] New Zealand                  Africa
## [45] Chile                        Jordan
## [47] Hungary                      Korea
## [49] Poland                       Cayman Islands
## [51] Czech Republic               Pakistan
## [53] United Kingdom/ South Africa France/ United Kingdom
## [55] Philippines                  Bahamas
## [57] Venezuela                    Peru
## 61 Levels: Africa Australia Australia/ United Kingdom Austria ... Venezuela
```

这里的name虽然有61个level，但是实际只有58个有值，因此Rstutio只输出了58个。

```
for(i in 1:58){
  asset = array()
  for (j in 1:2000) {
    if (country[j] == name[i]){
      asset <- append(asset,data$assets[j])
      total[i] <- total[i]+1
    }
  }
  asset <- asset[-1] #去掉asset数组里面的第一个数（NA）
  mean <- append(mean,mean(asset))
  median <- append(median,median(asset))
}

mean <- mean[-1]
median <- median[-1]

result = data.frame(country = name, num_of_companies = total,
                    mean_assets = mean,median_assets = median)
result
```

```
##                         country num_of_companies mean_assets median_assets
## 1                 United States             751   30.335406         8.220
## 2                United Kingdom             137   45.495693         7.340
## 3                         Japan             316   31.864304        13.135
## 4                   Switzerland              34   73.046765        12.685
## 5                   Netherlands              28   89.156786         8.520
## 6   Netherlands/ United Kingdom               2   73.105000        73.105
## 7                        France              63   62.345238        14.960
## 8                       Germany              65   81.388615        12.670
## 9                         Italy              41   54.887805        20.210
## 10                        Spain              29   40.188621         8.660
## 11                  South Korea              45   21.796222         9.190
## 12                        China              84   15.001190         7.750
## 13                      Bermuda              20   13.845500         7.470
## 14                       Canada              56   32.971786         9.050
## 15                      Finland              11   14.315455        13.350
## 16                    Australia              37   28.827297         5.320
## 17                       Russia              12   19.197500        11.950
## 18                      Belgium               9   79.911111        11.370
## 19                       Norway               8   20.880000        17.200
## 20    Australia/ United Kingdom               2   17.370000        17.370
## 21                       Brazil              15   18.376667         7.460
## 22                      Denmark              10   39.065000         6.160
## 23                       Sweden              26   34.454615         7.540
## 24    United Kingdom/ Australia               1   24.080000        24.080
## 25                      Ireland               8   52.710000        32.310
## 26                        India              27   11.906296         6.080
## 27       Panama/ United Kingdom               1   23.400000        23.400
## 28                    Singapore              16   18.664375         8.470
## 29                 South Africa              15   11.906000         3.540
## 30                      Austria               8   24.205000        10.235
## 31                       Mexico              17    7.080000         5.790
## 32                     Portugal               7   19.842857        12.640
## 33                     Thailand               9   13.014444        11.270
## 34                     Malaysia              16   11.509375         7.885
## 35                       Turkey              12   10.556667         8.055
## 36                       Greece              12   19.153333        16.475
## 37                   Luxembourg               2   36.715000        36.715
## 38                      Islands               1    4.040000         4.040
## 39                      Liberia               1   11.320000        11.320
## 40                    Indonesia               7    9.312857         4.740
## 41                       Israel               8   22.895000        15.010
## 42  United Kingdom/ Netherlands               1    2.510000         2.510
## 43                  New Zealand               1    4.550000         4.550
## 44                       Africa               2   11.220000        11.220
## 45                        Chile               4    6.820000         5.105
## 46                       Jordan               1   22.780000        22.780
## 47                      Hungary               2   10.390000        10.390
## 48                        Korea               4    4.837500         4.510
## 49                       Poland               1    3.990000         3.990
## 50                Cayman Islands               5    4.658000         3.190
## 51                Czech Republic               2    6.445000         6.445
## 52                      Pakistan               1    2.350000         2.350
## 53 United Kingdom/ South Africa               1   23.460000        23.460
## 54       France/ United Kingdom               1   14.880000        14.880
```

```
## 55                  Philippines               2    5.365000         5.365
## 56                      Bahamas               1    3.620000         3.620
## 57                    Venezuela               1    6.690000         6.690
## 58                         Peru               1    0.620000         0.620
```

得到了结果。

```
write.table(result,"countries.txt",sep = ",") # 导出一个.txt文件用于存这个dataframe
```

(f)

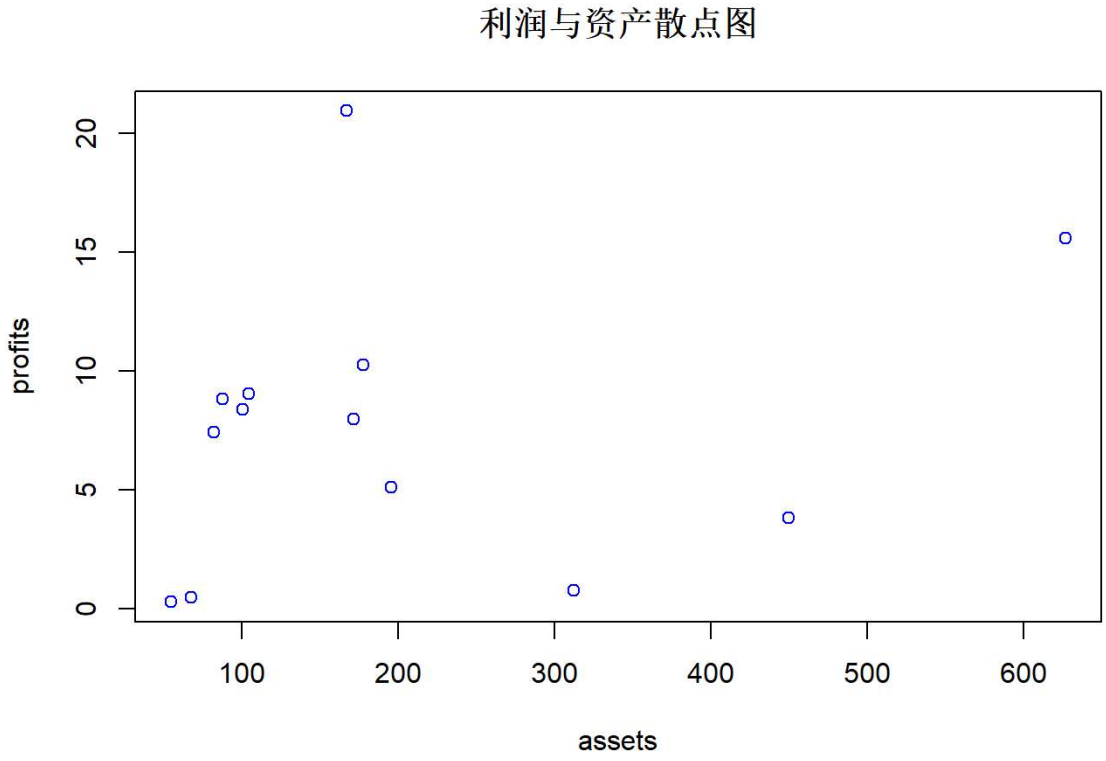
```
#方法一
selected = subset(data,data$sales > 100,c("rank","name","sales",
                                          "profits","assets"))


#方法二
keep = rep(TRUE,2000) #keep表示哪些行要被保留，初始化为全是TRUE

for(i in 1:2000){
  if(data$sales[i] <= 100)
    keep[i] <- FALSE #如果销售额小于100，则改行将被删掉
}

selected2 = data[keep,,drop = FALSE]
selected2 = selected2[c("rank","name","sales","profits","assets")]


sorted_1 = selected[order(-selected$sales),]#排序
sorted_2 = selected[order(selected$assets),]#排序

sorted_1 #输出按sales降序排列
```

```
##     rank                    name  sales profits assets
## 10    10         Wal-Mart Stores 256.33    9.05 104.91
## 5      5                      BP 232.57   10.27 177.57
## 4      4            ExxonMobil 222.88   20.96 166.99
## 29    29          General Motors 185.52    3.82 450.00
## 75    75              Ford Motor 164.20    0.76 312.56
## 21    21         DaimlerChrysler 157.13    5.12 195.58
## 8      8            Toyota Motor 135.82    7.99 171.71
## 2      2        General Electric 134.19   15.59 626.93
## 13    13 Royal Dutch/Shell Group 133.50    8.40 100.72
## 17    17                   Total 131.64    8.84  87.84
## 23    23           ChevronTexaco 112.94    7.43  82.36
## 156  156              Mitsubishi 112.76    0.46  67.69
## 225  225            Mitsui & Co 111.98    0.28  54.88
```

```
sorted_2 #输出按assets升序排列
```

```
##     rank                    name  sales profits assets
## 225  225             Mitsui & Co 111.98    0.28  54.88
## 156  156              Mitsubishi 112.76    0.46  67.69
## 23    23           ChevronTexaco 112.94    7.43  82.36
## 17    17                   Total 131.64    8.84  87.84
## 13    13 Royal Dutch/Shell Group 133.50    8.40 100.72
## 10    10         Wal-Mart Stores 256.33    9.05 104.91
## 4      4              ExxonMobil 222.88   20.96 166.99
## 8      8            Toyota Motor 135.82    7.99 171.71
## 5      5                      BP 232.57   10.27 177.57
## 21    21         DaimlerChrysler 157.13    5.12 195.58
## 75    75              Ford Motor 164.20    0.76 312.56
## 29    29          General Motors 185.52    3.82 450.00
## 2      2        General Electric 134.19   15.59 626.93
```

```
plot(selected$profits~selected$assets,xlab = "assets",ylab = "profits",
     col = "blue",main = "利润与资产散点图")
```



(g)

```
normalize_minmax <- function(x) {
  return(100000*(x - min(x)) / (max(x) - min(x)))
}

# 处理数据
normalized_sales <- normalize_minmax(data$sales)
normalized_assets <- normalize_minmax(data$assets)
normalized_marketvalue <- normalize_minmax(data$marketvalue)
data$sales <- normalized_sales
data$assets <- normalized_assets
data$marketvalue <- normalized_marketvalue

complete = data[!is.na(data$profits),]#找到没有缺失profits的公司

missing = data[is.na(data$profits),]
missing #找到缺失profits的公司
```

```
##      rank                  name        country                    category
## 772   772                   AMP      Australia                   Insurance
## 1085 1085                   HHG United Kingdom                   Insurance
## 1091 1091                   NTL  United States Telecommunications services
## 1425 1425      US Airways Group  United States              Transportation
## 1909 1909 Laidlaw International  United States              Transportation
##         sales profits    assets marketvalue
## 772  2102.840      NA 3376.4322   2033.3617
## 1085 2212.079      NA 4065.6454    624.0107
## 1091 1361.579      NA  816.6107   1802.0212
## 1425 2141.854      NA  657.5616     66.9670
## 1909 1743.914      NA  293.5684    447.4613
```

```
matrix <- matrix( nrow = 5, ncol = 1995)
#下面进行KNN

for (j in 1:5) {
  for (i in 1:1995) {
    matrix[j,i] <- ((complete$sales[i]-missing$sales[j])^2
    +(complete$assets[i]-missing$assets[j])^2
    +(complete$marketvalue[i]-missing$marketvalue[j])^2)
  }
}

result <- rep(0,5)
# 找到最“近”的10个公司的profits并求平均
for (m in 1:5) {
  index <- order(matrix[m,])[1:10]
  #print(index)
 # print(complete[index,c(5,7,8)])
  average <- mean(complete$profits[index])
  result[m] <- average
  print(average)
  #print(missing[m,c(5,7,8)])
  #print(matrix[m,index])
  # 助教哥哥姐姐一定很好奇为啥这个地方这么多代码被注掉了，
  # 因为这个可怜的孩子debug快de了一个点才找到问题，孩子蠢哭了......
}
```

```
## [1] 0.449
## [1] 0.065
## [1] 0.173
## [1] -0.009
## [1] -0.022
```

```
#complete[c(260,807,908,1406,1509),c(5,7,8)]


# 赋值
data$profits[missing$rank] <- result
data$profits[missing$rank]
```

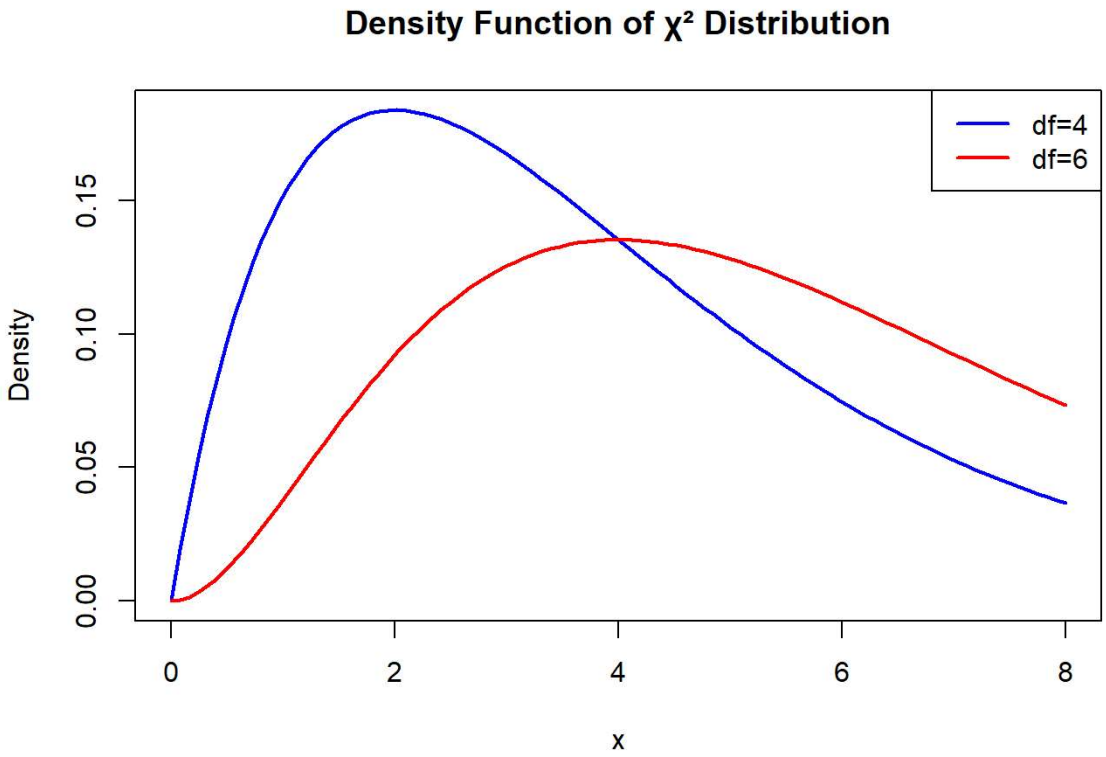
```
## [1]  0.449  0.065  0.173 -0.009 -0.022
```

综上， 0.449 0.065 0.173 -0.009 -0.022 即为赋给残缺profits的值。

Question 3

(a)

```
# Set the range of x-axis
x <- seq(0, 8, length.out = 100)

# Plot the density function for x² distribution with df=4
y_df4 <- dchisq(x, df = 4, ncp = 0)
plot(x, y_df4, type = "l", col = "blue", lwd = 2,
     xlab = "x", ylab = "Density", main = "Density Function of x² Distribution")

# Add the density function for x² distribution with df=6
y_df6 <- dchisq(x, df = 6, ncp = 0)
lines(x, y_df6, col = "red", lwd = 2)

# Add legend
legend("topright", legend = c("df=4", "df=6"), col = c("blue", "red"), lwd = 2)
```




```
# Calculate the area under the density curve for the interval [1, 7]
area_1_7 <- pchisq(7, df = 5) - pchisq(1, df = 5)

# Calculate the area under the density curve for the interval [3,∞)
area_3_inf <- 1 - pchisq(3, df = 5)

area_1_7
```

```
## [1] 0.7419255
```

```
area_3_inf
```

```
## [1] 0.6999858
```

故，$[1,7]$之间的面积为0.7419255，$[3,+\infty]$之间的面积为0.6999858。

(b)

```
# 定义表示方程的函数
equation <- function(lambda) {
  return(exp(-lambda) - 0.8)
}

# 使用uniroot求解lambda
solution <- uniroot(equation, interval = c(0,10))

# 解将会在solution$root中获得
lambda <- solution$root
lambda
```

```
## [1] 0.22315
```

故，λ = 0.22315。

```
#计算P(X = 6)
p = 1 - exp(-6*lambda)
p
```

```
## [1] 0.7378661
```

故，$Pr(X \leq 6) = 0.7378661$。

(c)

因为$X_1, X_2 \overset{\text{i.i.d.}}{\sim} Uniform(0,1)$，则有:

$$f_{(X_1,X_2)}(x_1,x_2) = f_{X_1}(x_1) \cdot f_{X_2}(x_2) = 1 \cdot I_{[0,1]\times[0,1]},$$

其中I_S为集合S的示性函数。

换元令$U = X_1 + 2X_2, V = X_1$，则有$X_1 = V, X_2 = \frac{U-V}{2}$。

代入$X_1 \in [0,1]$且$X_2 \in [0,1]$可得:

$$V \in [0,1], U \in [V, 2+V]$$

雅可比行列式的绝对值$|J|$ =

$$Absolute\ value(\begin{vmatrix} \frac{\partial X_1}{\partial U} & \frac{\partial X_1}{\partial V} \\ \frac{\partial X_2}{\partial U} & \frac{\partial X_2}{\partial V} \end{vmatrix}) = \frac{1}{2}$$

故(U,V)的联合密度为

$$f_{(U,V)}(u,v) = f_{(X_1,X_2)}(x_1,x_2)|J| = \frac{1}{2} \times I_C$$

其中$C = \{(U,V) : V \in [0,1], U \in [V, 2+V]\}$

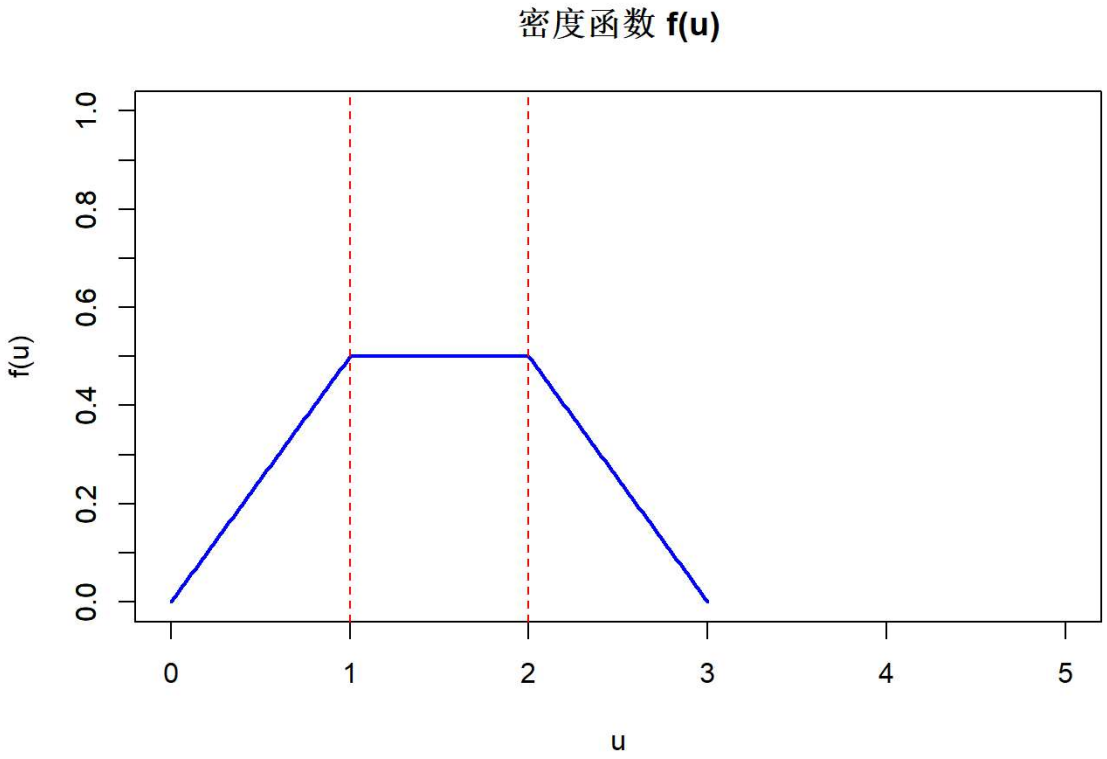
最后我们由联合密度求得$U=X_1+2X_2$的边际密度为 f_U(u) =

$$\begin{cases} \int_0^u \frac{1}{2}\mathrm{d}u, & u\in[0,1] \\ \int_0^1 \frac{1}{2}\mathrm{d}u, & u\in[1,2] \\ \int_{u-2}^1 \frac{1}{2}\mathrm{d}u, & u\in[2,3] \end{cases}$$

即$U=X_1+2X_2$的密度为： f_U(u) =

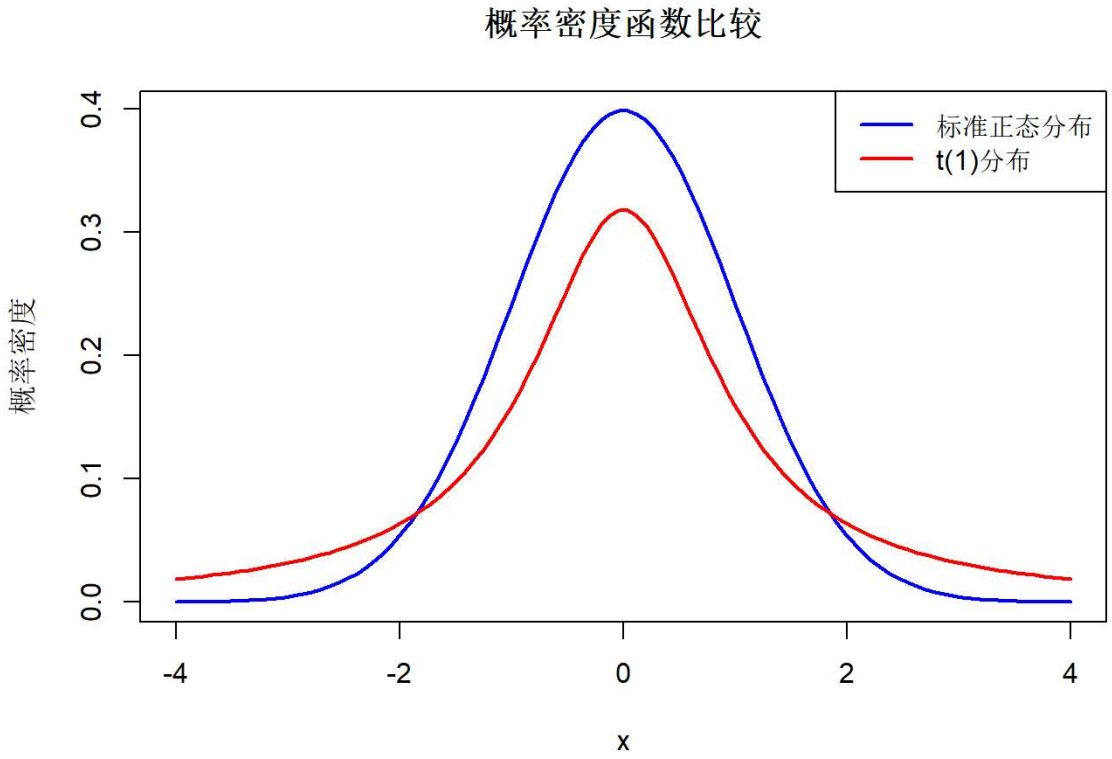
$$\begin{cases} \frac{1}{2}u, & u\in[0,1] \\ \frac{1}{2}, & u\in[1,2] \\ \frac{1}{2}(3-u), & u\in[2,3] \end{cases}$$

```
# 定义密度函数 f_U(u)
f_U <- function(u) {
  if (u >= 0 && u <= 1) {
    return(0.5 * u)
  } else if (u > 1 && u <= 2) {
    return(0.5)
  } else if (u > 2 && u <= 3) {
    return(0.5 * (3 - u))
  } else {
    return(0)
  }
}

# 生成 u 的取值范围
u <- seq(0, 3, by = 0.01)

# 计算对应的密度值
density <- sapply(u, f_U)

# 绘制密度函数图
plot(u, density, type = "l", col = "blue", lwd = 2, xlab = "u", ylab = "f(u)", xlim = c(0,5),yl
im = c(0,1), main = "密度函数 f(u)")

# 添加区间分割线
abline(v = c(1, 2), col = "red", lty = 2)
axis(2, at = seq(0, 1, by = 0.1))
```



(d)

```
# 绘制概率密度函数
x <- seq(-4, 4, length.out = 100)  # x值的范围

# 标准正态分布
dnorm_values <- dnorm(x)  # 计算标准正态分布的概率密度值

# t(1)分布
dt_values <- dt(x, df = 1)  # 计算自由度为1的t(1)分布的概率密度值

# 绘图
plot(x, dnorm_values, type = "l", col = "blue", lwd = 2, xlab = "x", ylab = "概率密度",
     main = "概率密度函数比较")
lines(x, dt_values, col = "red", lwd = 2)
legend("topright", legend = c("标准正态分布", "t(1)分布"), col = c("blue", "red"), lwd = 2)
```




```
## t(1)更厚尾一些
```

由图可得，t(1)更厚尾一些。

(e)

先求超几何分布的期望$E(X)=n\frac{m}{N}$,其中$X\sim H(n,m,N)$

令$l=min\{n,m\}$，则有

$$
\begin{aligned}
E(X) &= \sum_{k=0}^{l} k \cdot P(X=k) \\
&= \sum_{k=0}^{l} k \cdot \frac{C_m^k \cdot C_{N-m}^{n-k}}{C_N^n} \\
&= \sum_{k=1}^{l} k \cdot \frac{C_m^k \cdot C_{N-m}^{n-k}}{C_N^n} \\
&= \sum_{k=1}^{l} m \cdot C_{m-1}^{k-1} \cdot \frac{n}{N} \cdot \frac{C_{n-m}^{n-k}}{C_{N-1}^{n-1}} \\
&= m \cdot \frac{n}{N} \cdot \sum_{k=1}^{l} C_{m-1}^{k-1} \cdot \frac{C_{n-m}^{n-k}}{C_{N-1}^{n-1}} \\
\text{令} i = k-1, & \\
&= m \cdot \frac{n}{N} \cdot \sum_{i=0}^{l-1} C_{m-1}^{i} \cdot \frac{C_{(N-1)-(m-1)}^{(n-1)-i}}{C_{N-1}^{n-1}} \\
&= m \cdot \frac{n}{N} \\
&= n \cdot \frac{m}{N}
\end{aligned}
$$

下面我们推导超几何分布的方差$Var(X) = n \cdot \frac{m}{N} \cdot (1 - \frac{m}{N}) \cdot \frac{N-n}{N-1}$

$$
\begin{aligned}
E(X^2) &= \sum_{k=0}^{l} k^2 \cdot P(X=k) \\
&= \sum_{k=0}^{l} k^2 \cdot \frac{C_m^k \cdot C_{N-m}^{n-k}}{C_N^n} \\
&= \frac{mn}{N} \sum_{k=1}^{n} (k-1+1) \frac{C_{m-1}^{k-1} C_{N-m}^{n-k}}{C_{N-1}^{n-1}} \\
&= \frac{mn}{N} \cdot (\sum_{k=1}^{l} (k-1) \frac{C_{m-1}^{k-1} C_{N-m}^{n-k}}{C_{N-1}^{n-1}} + \sum_{k=1}^{l} \frac{C_{m-1}^{k-1} C_{N-m}^{n-k}}{C_{N-1}^{n-1}}) \\
&= \frac{mn}{N} \cdot (\sum_{k=2}^{l} (k-1) \frac{C_{m-1}^{k-1} C_{N-m}^{n-k}}{C_{N-1}^{n-1}} + 1) \\
&= \frac{mn}{N} \cdot (\frac{(m-1)(n-1)}{(N-1)} \sum_{k=2}^{l} \frac{C_{m-2}^{k-2} C_{N-m}^{n-k}}{C_{N-2}^{n-2}} + 1) \\
&= \frac{mn}{N} \cdot (\frac{(m-1)(n-1)}{(N-1)} + 1)
\end{aligned}
$$

因此，

$$
\begin{aligned}
Var(X) &= E(X^2) - (EX)^2 \\
&= \frac{mn}{N} \cdot (\frac{(m-1)(n-1)}{(N-1)} + 1) - (\frac{nm}{N})^2 \\
&= n \cdot \frac{m}{N} \cdot (1 - \frac{m}{N}) \cdot \frac{N-n}{N-1}
\end{aligned}
$$

```
reps = 10000
result <- rep(0, reps)



for (i in 1:reps) {
  # 45样本不放回取样，设定前28个为统计，其余为科学。
  value <- sample(1:45,8,replace = FALSE)
  for (j in 1:8) {
    if(value[j] <= 28){
      result[i] <- result[i] + 1
    }
  }
}

prob <- table(result)
probability <- prob/reps #得到Pr(X = k), k = 1,2,...,8
probability
```

```
## result
##      0      1      2      3      4      5      6      7      8
## 0.0002 0.0028 0.0207 0.0951 0.2254 0.3109 0.2354 0.0936 0.0159
```

因此我们可以得到各个点的概率为：2^{-4}, 0.0028, 0.0207, 0.0951, 0.2254, 0.3109, 0.2354, 0.0936, 0.0159。

```
print(8*28/45) #期望理论值
```

```
## [1] 4.977778
```

```
mean(result) #样本的期望
```

```
## [1] 4.9804
```

```
print(8*28/45*(1-28/45)*(45-8)/(45-1)) #方差理论值
```

```
## [1] 1.581324
```

```
var(result) #样本的方差
```

```
## [1] 1.594575
```

注意到理论值与试验得出来的结果非常接近，这说明理论推导的结果是正确的。

(f)

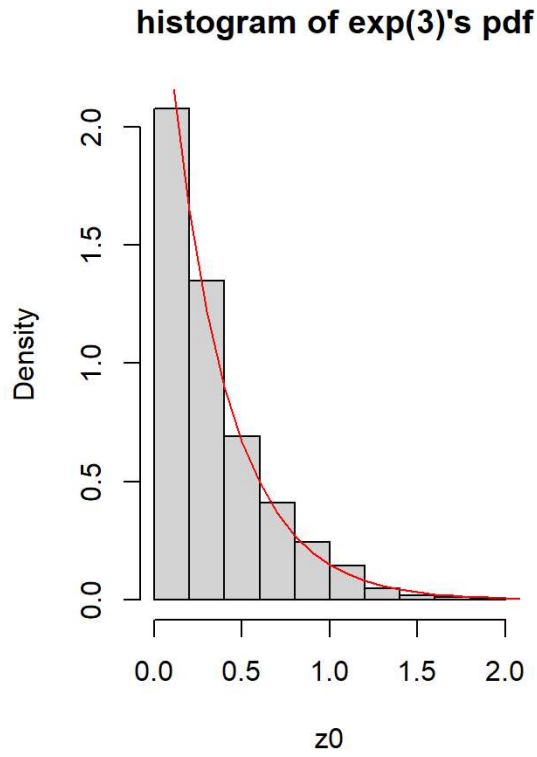
```
# 进行数值搜索
lambda <- 0  # 初始λ值
prev_prob <- 1  # 初始概率
while (prev_prob > 0.6) {
  lambda <- lambda + 0.001
  prob <- sum(dpois(0:8, lambda))
  prev_prob <- prob
}

max_lambda <- lambda - 0.001



print(max_lambda)
```

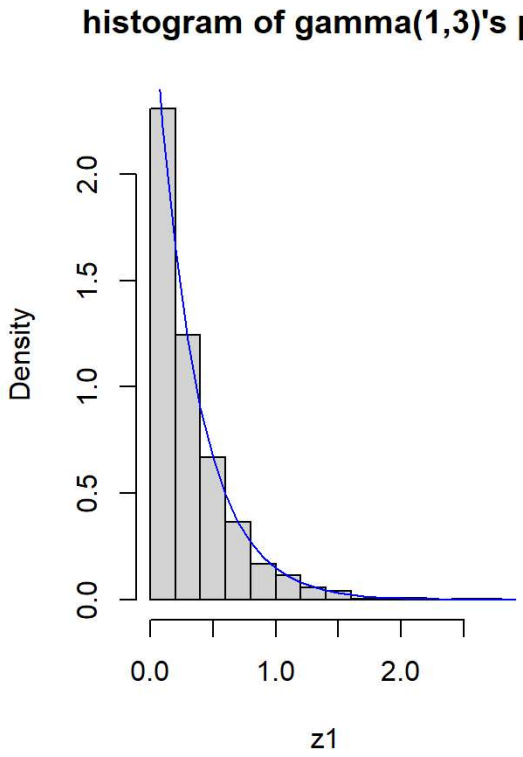
```
## [1] 7.946
```

故，满足条件的最大λ值为7.946。

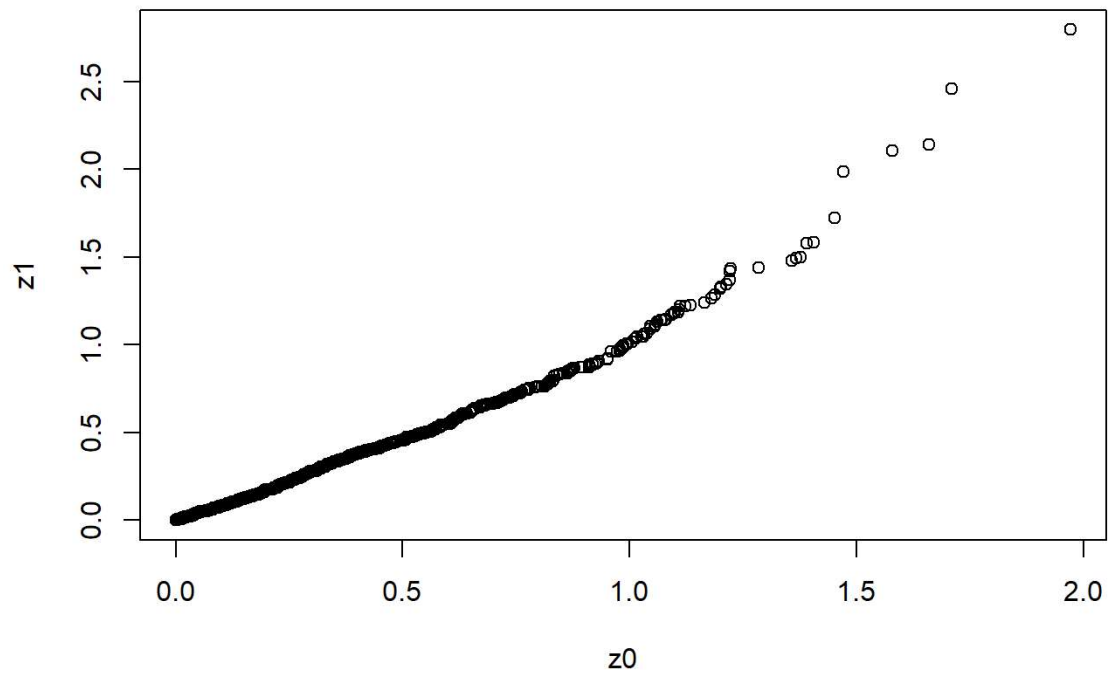
(g)

```
par(mfrow = c(1, 2))

z0 <- rexp(1000,3)

hist(z0,main = " histogram of exp(3)'s pdf",probability = TRUE)
x=pretty(c(0,100),1000);
y=dexp(x,rate = 3)
lines(x,y,col="red")



z1 <- rgamma(1000,1,3)

hist(z1,main = "histogram of gamma(1,3)'s pdf",probability = TRUE)
x=pretty(c(0,100),1000);
y=dgamma(x,1,3)
lines(x,y,col="blue")
```



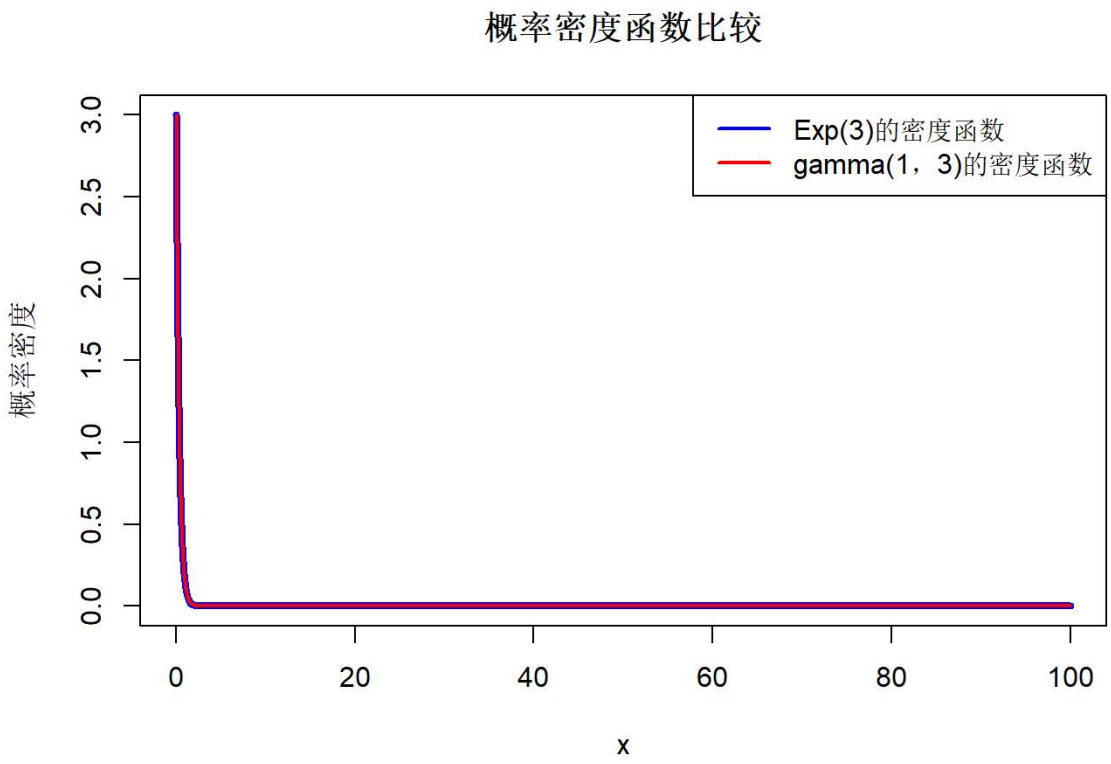



```
qqplot(z0,z1)
```



观察到Q-Q plot 近似为直线$y = x$，因此我们可以说z0 服从Exp(3)和z1服从gamma(1,3)的分布是同一个分布。

```
x <- seq(0, 100, length.out = 1000)  # x值的范围


exp_values <- dexp(x,rate = 3)

gamma_values <- dgamma(x, shape = 1,rate = 3)

# 绘图
plot(x, exp_values, type = "l", col = "blue", lwd = 4, xlab = "x", ylab = "概率密度",
     main = "概率密度函数比较")
lines(x, gamma_values, col = "red", lwd = 2)
legend("topright", legend = c("Exp(3)的密度函数", "gamma(1，3)的密度函数"), col = c("blue", "red"), lwd = 2)
```



(h)

```
reps = 10000
array <- rep(NA,reps)
for (i in 1:reps) {
  U <- sample(1:365,365,replace = TRUE)
  for (j in 2:365) {
    temp <- FALSE
    for (k in 1:(j-1)) {
      if(U[j] == U[k]){
        array[i] = j
        temp <- TRUE
        break
      }

    }
    if(temp == TRUE)
      break
  }
}

result <- array[order(array)[0.75*reps+1]]
result
```

```
## [1] 33
```

故33个人中有75%的几率出现两人同一天生日。

若想计算多少人以至少α概率出现两人同一天生日，则只需将上述代码将参数0.75改成α即可，下面给出50%概率下的结果：

```
reps = 10000
array <- rep(NA,reps)
for (i in 1:reps) {
  U <- sample(1:365,365,replace = TRUE)
  for (j in 2:365) {
    temp <- FALSE
    for (k in 1:(j-1)) {
      if(U[j] == U[k]){
        array[i] = j
        temp <- TRUE
        break
      }

    }
    if(temp == TRUE)
      break
  }
}

result <- array[order(array)[0.5*reps+1]]
result
```

```
## [1] 23
```

即23个人中有50%的几率出现两人同一天生日。

(i)

```
# (i)

n <- 1000 # 给定一个很大的n
p = 0.6 # 给定一个p
a = -1 # 给定a
b = 1 # 给定b
total = 1000
reps = 10
result1 <- rep(0,reps)

for (m in 1:reps) {
  sucess = 0
for(j in 1:total){

  random <- runif(n,0,1)
  r <- rep(NA,n)
  for (i in 1:n) {
    if(random[i] < p){
      r[i] <- 1
    } else{
      r[i]  <- 0
    }
  }
  # 以上我们得到了n个Bernoulli(P)的随机变量

  Z_n <- (sum(r) - n*p)/(sqrt(n*p*(1-p)))

  if((a<Z_n)&(Z_n<=b)){
    sucess <- sucess + 1}

}

result1[m] <- sucess/total


}
f <- function(x){
  return(dnorm(x))
}

result2 <- integrate(f,a,b)


result3 <- mean(result1)
result3
```

```
## [1] 0.6812
```

```
result2
```

```
## 0.6826895 with absolute error < 7.6e-15
```

二者相差不大，故该中心极限定理成立。

(j)

```
p = 0.005 #给定p
n = 1000 #给定n
k = 5
lambda = n*p
total = 1000
reps = 10
result <- rep(0,reps)


for (m in 1:reps) {
  success = 0
  for (j in 1:total) {
    random <- runif(n,0,1)
    r <- rep(NA,n)
    for (i in 1:n) {
      if(random[i] < p){
        r[i] <- 1
      } else{
        r[i]  <- 0
      }
    }
    #至此，已生成若干伯努利随即变量。

    S <- sum(r)
    if(S == k){
      success <- success + 1
    }

  }
  success/total


  result[m] <- success/total
}
  mean(result)
```

```
## [1] 0.1727
```

```
result2 <- exp(-lambda)*lambda^k/factorial(k)
result2
```

```
## [1] 0.1754674
```

二者相差不大，可以得出结论：该极限定理成立。

(k)

```
number = 1000
total = 1000
n = 3
m = 2
result1 <- rep(0,total)
result2 <- rep(0,total)


# 计算概率

for (j in 1:total) {
  r <- rgeom(number,0.6)

  count = 0
  count2 = 0
  count3 = 0
  for (i in 1:number) {
    if(r[i] > n-1) #这里使输出空间由{0,1,2,...}变为{1,2,3,...}
    {count <- count + 1}

    if(r[i] > m){
      count2 <- count2 + 1}

    if(r[i] > m+n){
      count3 <- count3 + 1}

  }

  result1[j] = count/total
  result2[j] = count3/count2
}

mean(result1)
```

```
## [1] 0.064182
```

```
mean(result2)
```

```
## [1] 0.06289777
```

二者近似相等，顾客认为几何分布具有和指数分布一样的无记忆性。

```
number = 1000
total = 1000
n = 3
m = 3
result1 <- rep(0,total)
result2 <- rep(0,total)


# 计算概率

for (j in 1:total) {
  r <- rpois(number,4)

  count = 0
  count2 = 0
  count3 = 0
  for (i in 1:number) {
    if(r[i] > n)
    {count <- count + 1}

    if(r[i] > m){
      count2 <- count2 + 1}

    if(r[i] > m+n){
      count3 <- count3 + 1}

  }

  result1[j] = count/total
  result2[j] = count3/count2
}

mean(result1)
```

```
## [1] 0.56669
```

```
mean(result2)
```

```
## [1] 0.1950088
```

结果相差很大，故泊松分布不具有无记忆性。