HW1

杜金鸿 15338039

目录

习题一

1. 编写 R 函数 $skewness(x)$, 输入向量 x 的值后，按如下公式

$$\psi = \frac{n}{(n-1)(n-2)}\sum_{i=1}^{n}\left(\frac{x_i-\overline{x}}{s}\right)^3$$

计算样本偏度，其中 n 为 x 的元素个数，$\overline{x}$ 和 s 分别为 x 中元素的样本平均值和样本标准差。

```r
skewness <- function(x,                                     # vector x
                     n = length(x),                         # length of x
                     xmean = mean(x),                       # sample mean of x
                     s = sqrt(sum((x - xmean)^2)/(n-1))     # sample standard deviation of x
                     ){
    return(n/(n-1)/(n-2) * sum(((x-xmean)/s)^3))
}

# Example
x <- c(0,2,3)
skewness(x)
```

```
## [1] -0.9352195
```

2. 设 x 是一个长度为 n 的向量, 写一段程序, 计算 x 的长度为 s 的滑动和:

$$S_x(t) = \sum_{i=0}^{s-1} x_{t-i}, \qquad t = s, s+1, \cdots, n$$

(提示：使用 $filter$ 函数)

我们可以用 convolution filter 来计算 $S_x(t)$.

```r
sx <- function(x,                # vector x
               s,                # slide length
               n = length(x)     # length of x
               ){
    return(filter(x, filter = rep(1,s), method = "convolution",
           sides = 1)[s:n])
}
```

```r
# Example
x <- 1:100
sx(x, 5)
```

```
##  [1]  15  20  25  30  35  40  45  50  55  60  65  70  75  80  85  90  95
## [18] 100 105 110 115 120 125 130 135 140 145 150 155 160 165 170 175 180
## [35] 185 190 195 200 205 210 215 220 225 230 235 240 245 250 255 260 265
## [52] 270 275 280 285 290 295 300 305 310 315 320 325 330 335 340 345 350
## [69] 355 360 365 370 375 380 385 390 395 400 405 410 415 420 425 430 435
## [86] 440 445 450 455 460 465 470 475 480 485 490
```

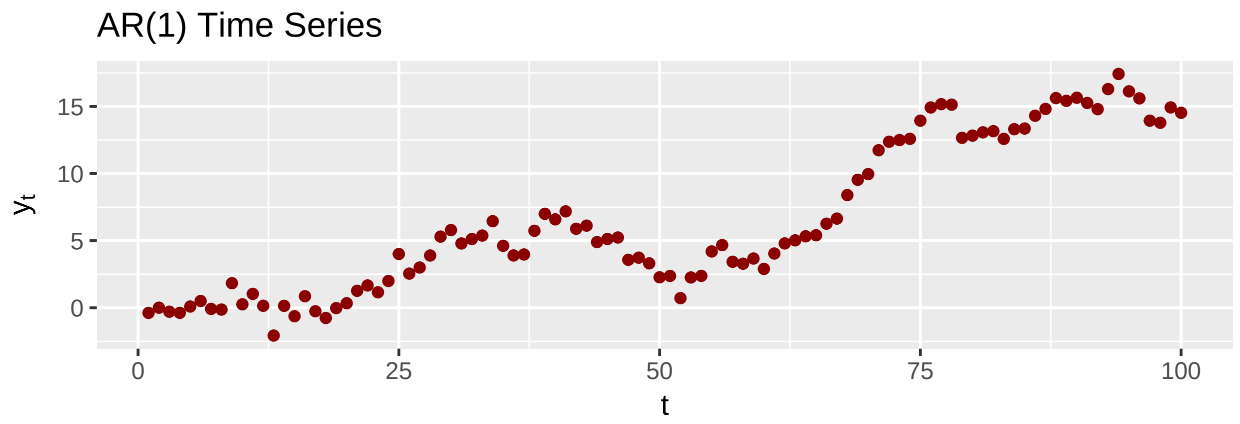
3. 写一个 $AR(1)$ 的模拟函数：

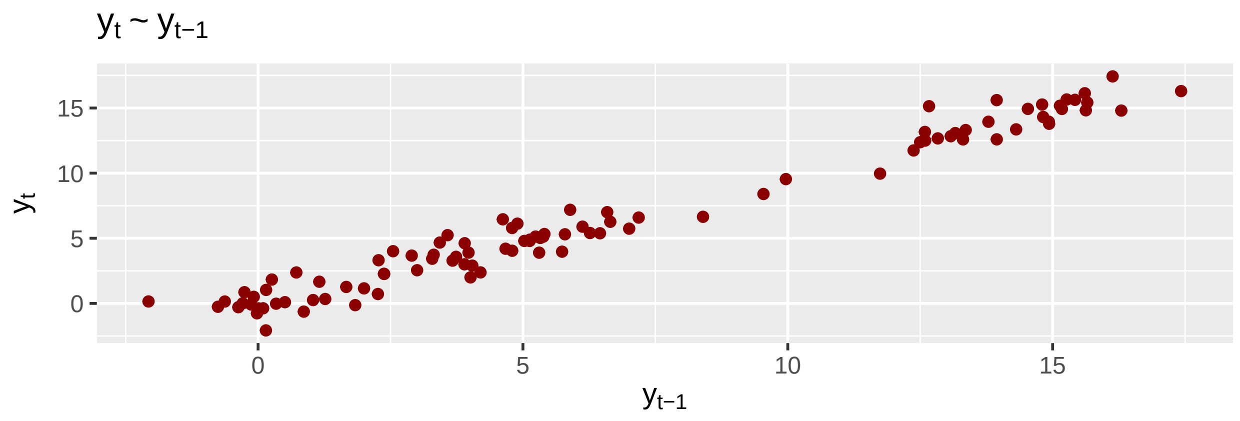
$$x_t = a + bx_{t-1} + \epsilon_t,\ t = 1, 2, \ldots, n,\ Var(\epsilon_t) = \sigma^2$$

函数的参数为 n、a、b、x_0 和 σ，$x_0 = x_{-1} = \cdots = x_{-p+1} = 0$，缺省时 $n = 100$，$a = 0$，$b = 1$，$\sigma = 1$。(提示：使用 $filter$ 函数)

我们可以用 `recursive filter` 来计算 $S_x(t)$.

```r
ar1 <- function(x0,              # initial value
                n = 100,         # total time steps
                a = 0,           # intercept
                b = 1,           # slope
                sigma_ =1        # variance of epsilon_t
                ){
    epsilon <- rnorm(n, mean = 0, sd = sigma_)
    z <- filter(c(a+b*x0,rep(a,n-1))+epsilon,filter = c(b),method = 'recursive')
    return(z)
}

# Example
data <- ar1(0)
library(ggplot2)
library(grid)
library(latex2exp)
# To plot the generated time series
p1 <- ggplot(data= NULL, aes(c(1:100), c(data))) +
    geom_point(color = "darkred") + xlab(label = 't')+
    ylab(label = TeX('$y_{t}$')) +
    ggtitle(label = "AR(1) Time Series")
# To plot y[t]~y[t-1] (high correlation indicates auto-correlation)
p2 <- ggplot(data= NULL, aes(c(data[2:length(data)]), c(data[1:length(data)-1]))) +
    geom_point(color = "darkred") +
    xlab(label = TeX('$y_{t-1}$'))+
    ylab(label = TeX('$y_{t}$'))+
    ggtitle(label = TeX('$y_{t}\\sim y_{t-1}$'))
# To build a subplot
grid.newpage()                                    # new page
pushViewport(viewport(layout = grid.layout(2,1))) # divide the page into 2*1
vplayout <- function(x,y){
  viewport(layout.pos.row = x, layout.pos.col = y)
}
print(p1, vp = vplayout(1,1))
print(p2, vp = vplayout(2,1))
```





4. 编程计算机器 ϵ 的值。

```r
# To estimate the machine epsilon
# within a factor of two (one order of magnitude) of its true value
machine_epsilon <- 1.0
while(1+machine_epsilon/2!=1){
    machine_epsilon <- machine_epsilon/2
}
print(sprintf('Estimated machine epsilon is %g',machine_epsilon))
```

```
## [1] "Estimated machine epsilon is 2.22045e-16"
```

```r
print(sprintf('Machine epsilon is %g',.Machine$double.eps))
```

```
## [1] "Machine epsilon is 2.22045e-16"
```

7. 对函数 e^{-x} ，可以用两种公式计算：

$$(1)e^{-x} = 1 - x + x^2/2 - x^3/3! + \cdots + (-1)^k x^k/k! + \ldots;$$

$$(2)e^{-x} = 1/(1 + x + x^2/2 + x^3/3 + \cdots + x^k/k! + \ldots)$$

对 $x = 1, 2, 3, 4$，计算到 $k = 10$ 的级数并比较两种算法的计算精度。

方法 2 比方法 1 更精确，当 x 增加时，第一个等式右端的尾项计算会发生精度损失，因此方法 1 的近似误差显著增加。

```r
# (1)
x <- c(1:4)
k <- 10
ex_m1 <- matrix(rep(1,4*(k+1)),4,k+1)

for (i in x){
    for (j in c(1:k)){
        ex_m1[i,j+1] <- ex_m1[i,j]*(-1)*i/j
    }
}
ex1 <- apply(ex_m1, 1, sum)
```

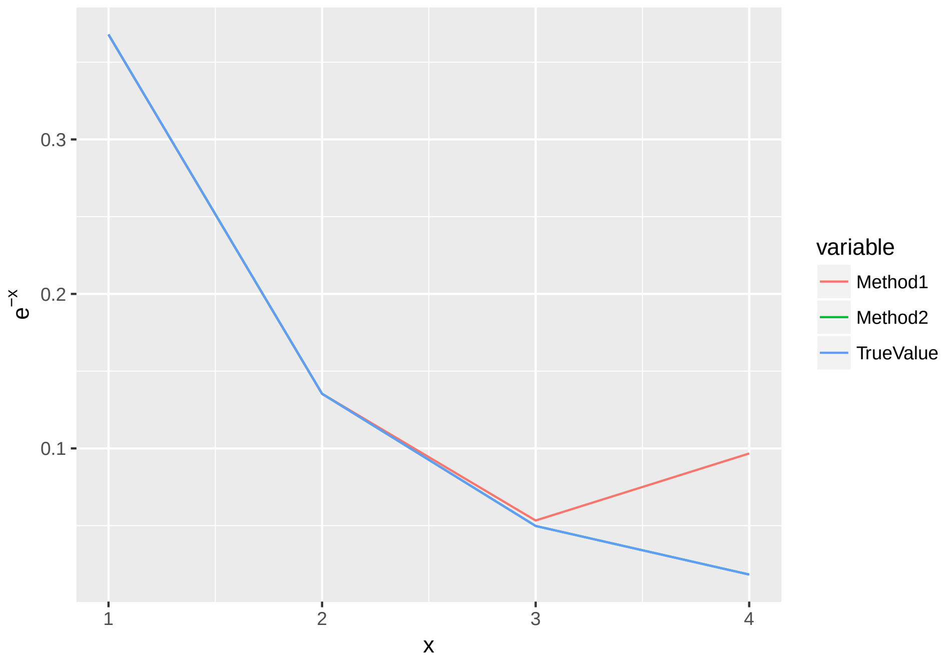
```r
# (2)
ex_m2 <- matrix(rep(1,4*(k+1)),4,k+1)
for (i in x){
    for (j in c(1:k)){
        ex_m2[i,j+1] <- ex_m2[i,j]*i/j
    }
}
ex2 <- 1/apply(ex_m2, 1, sum)
print(ex1)
```

```
## [1] 0.36787946 0.13537919 0.05332589 0.09671958
```

```r
print(ex2)
```

```
## [1] 0.36787944 0.13533641 0.04980163 0.01836780
```

```r
library(reshape2)
ex <- data.frame(Method1 = ex1, Method2 = ex2, TrueValue = exp(-x), x=x)
ggplot(data = melt(ex,id = 'x') , aes(x=x, y=value, colour=variable)) +
    geom_line()+
    xlab(label = 'x')+
    ylab(label = TeX('$e^{-x}$'))
```



8. 有如下一组观测数据：

$$249, 254, 243, 268, 253, 269, 287, 241, 273, 306$$

$$303, 280, 260, 256, 278, 344, 304, 283, 310$$

用两种方法计算样本方差：
(1) 公式

$$S^2 = \frac{1}{n-1}\left(\sum_{i=1}^{n} x_i^2 - n\overline{x}^2\right)$$

(2) 公式

$$S^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

其中每一个中间步骤保留 6 位有效数字 (R 中用 $signif$ 函数把数据四舍五入到指定有效位数)。把结果与使用高精度计算的结果比较。

Method	1	2
signif	733.333	733.433
precise	733.4327485	733.4327485

方法 2 较精确，方法 1 误差较大。

```
data1 <- c(249, 254, 243, 268, 253, 269, 287, 241, 273, 306,
           303, 280, 260, 256, 278, 344, 304, 283, 310)

n <- length(data1)                          # length of data
xbar <- signif(mean(data1), digits = 6)     # mean of data

S2_1 <- 0                                   # signif S^2 by method 1
for (i in c(1:n)){
    S2_1 <- signif(S2_1 + signif(data1[i]^2, digits = 6), digits = 6)
}
xbar2 <- signif(xbar^2, digits = 6)
S2_1 <- signif(S2_1 - signif(n*xbar2, digits = 6), digits = 6)
S2_1 <- signif(S2_1 / (n-1), digits = 6)



S2_2 <- 0                                   # signif S^2 by method 2
for (i in c(1:n)){
    y <- signif(data1[i]-xbar , digits = 6)
    y2 <- signif(y^2 , digits = 6)
    S2_2 <- signif(S2_2 + y2, digits = 6)
}
S2_2 <- signif(S2_2/(n-1), digits = 6)

xbar <- mean(data1)
precise_S2_1 <- (sum(data1^2)-n*xbar^2)/(n-1)# precise S^2 by method 1
precise_S2_2 <- sum((data1 - xbar)^2)/(n-1)  # precise S^2 by method 1
```

10. 对如下 t 的表达式找出发生严重精度损失的 t 值，并变化计算方法以避免精度损失：

$$(1)\frac{1}{t}-\frac{1}{t^2+a}$$

$$(2)e^{-2t^2}-e^{-8t^2};$$

$$(3)\ln(e^{t+s}-e^t);$$

$$(4)[(1+e^{-t})^2-1]t^2e^{-t}.$$

(1) 当 $a\le\frac{1}{4}$、t 接近 $\frac{1-\sqrt{1-4a}}{2}$ 时，$\frac{1}{t}$ 与 $\frac{1}{t^2+a}$ 接近且分母接近 0，严重损失精度。当 $a>\frac{1}{4}$、t 接近 $\pm\infty$ 时，$\left|\frac{1}{t}\right|$ 远大于 $\left|\frac{1}{t^2+a}\right|$，绝对值较小数会被舍去，严重损失精度。改为计算

$$\frac{t^2-t+a}{t(t^2+a)}$$

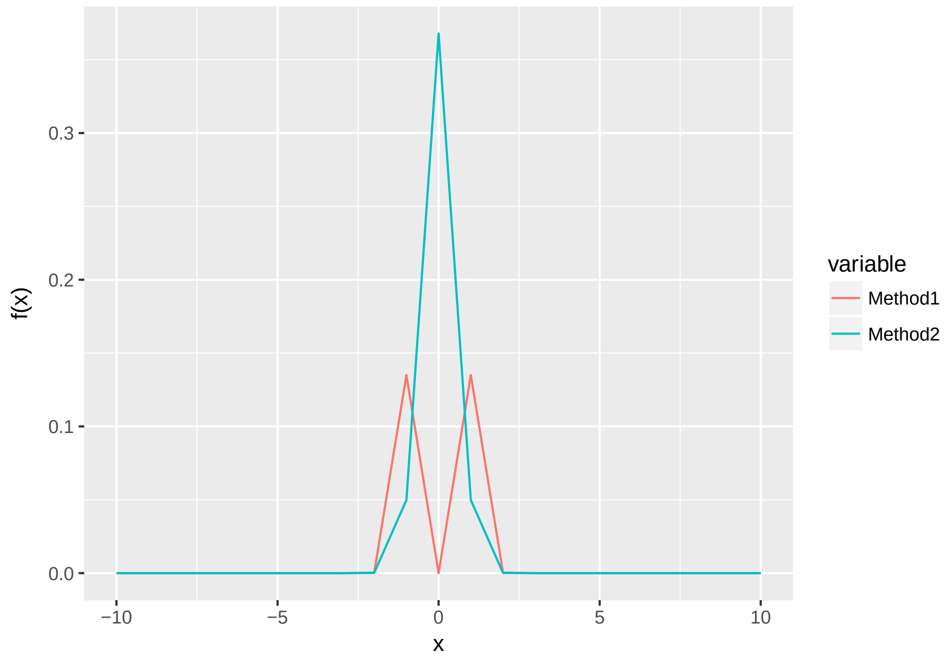
```
a <- 1/16
t <- c((1-sqrt(1-4*a))/2)

f1 <- 1/t-1/(t^2+a)
f2 <- (t^2-t+a)/(t*(t^2+a))
print(f1-f2)
```

```
## [1] -9.200626e-16
```

(2) 当 t 接近 0 时，e^{-2t^2} 与 $e^{-8t^2}=(e^{-2t^2})^4$ 较接近，严重损失精度。改为计算

$$\frac{e^{6t^2}-1}{e^{8t^2}}$$

```r
a <- 2
t <- c(-10:10)
f1 <- exp(-2*t^2)-exp(-8*t^2)
f2 <- (exp(6*t^2-1))/exp(8*t^2)
result <- data.frame(Method1 = f1, Method2 = f2, x=t)
ggplot(data = melt(result,id = 'x') , aes(x=x, y=value, colour=variable)) +
    geom_line()+
    xlab(label = 'x')+
    ylab(label = TeX('$f(x)$'))
```



(3) 当 t 接近 $\pm\infty$ 时，e^{t+s} 与 e^t 较接近，会出现有效位数损失。改为计算

$$\ln[e^t(e^s-1)] = t + \ln(e^s-1)$$

(4) 当 t 接近 $+\infty$ 时，$(1+e^{-t})^2$ 与 1 较接近，会出现有效位数损失。改为计算

$$\frac{(e^t+1)^2-e^t}{e^{3t}}t^2$$

11. 编写用第 26 页描述的 $\tilde{F}_n$ 求逆的方法求样本分位数的程序，并用 R 的 $quantile$ 函数验证。

首先计算

$$\tilde{F}_n(x_{(i)}) = \frac{i-\frac{1}{3}}{n-\frac{1}{3}}$$

然后线性插值 $x_{(i)} \sim \tilde{F}_n(x_{(i)})$。0% 分位数即为 $x_{(1)}$，100% 分位数为 $x_{(n)}$。对其它分位数 $\alpha \in (0,1)$，设

$$x_\alpha = \max\{x_{(i)} : x_{(i)} < \alpha\},$$

$$x_\beta = \min\{x_{(i)} : x_{(i)} \geqslant \alpha\},$$

则 α 估计数为

$$x_\alpha + \frac{x_\beta - x_\alpha}{\tilde{F}_n(x_\beta) - \tilde{F}_n(x_\alpha)} \times [\alpha - \tilde{F}(x_\alpha)]$$

```
F_quantile <- function(
    x,                                 # vector
    n = length(x),                     # length of data
    sorted = sort(x,index.return=TRUE) # sorted data and sorted index
){
    Fn <- (c(1:n)- 1/3)/(n+1/3)
    quantile_x <- rep(0,5)
    quantile_x[1] <- sorted$x[1]
    quantile_x[-1] <- sorted$x[n]
    for (i in c(1:3)){
        quantile_x[i+1] <- sorted$x[which(Fn>=0.25*i)[1]-1] +
        (sorted$x[which(Fn>=0.25*i)[1]] - sorted$x[which(Fn>=0.25*i)[1]-1])/
        (Fn[which(Fn>=0.25*i)[1]] - Fn[which(Fn>=0.25*i)[1]-1]) *
        (0.25*i - Fn[which(Fn>=0.25*i)[1]-1])
    }
    return(quantile_x)
}

# Example
x <- c(5,3,4,2,1,6)
quantile(x,type = 8)
```

```
##       0%      25%      50%      75%     100%
## 1.000000 1.916667 3.500000 5.083333 6.000000
```

```
print(F_quantile(x),digits=7)
```

```
## [1] 1.000000 1.916667 3.500000 5.083333 6.000000
```

13. 把第 28 页的均值和方差的递推算法推广到随机向量的均值和协方差阵计算问题。

令 Γ_n 为前 n 个样本 $X_1,\cdots,X_n\in\mathbb{R}^d$ 的协方差矩阵、$\mu_n\in\mathbb{R}^d$ 为前 n 个样本的均值，则

$$\begin{aligned}
\mu_n &= \frac{n-1}{n}\mu_{n-1}+\frac{1}{n}X_n \\
\Gamma_n &= \frac{1}{n-1}\sum_{i=1}^{n}(X_i-\mu_i)(X_i-\mu_i)^T \\
&= \frac{1}{n-1}\left(\sum_{i=1}^{n}X_iX_i^T-n\mu_n\mu_n^T\right) \\
&= \frac{1}{n-1}\left[\sum_{i=1}^{n-1}X_iX_i^T-(n-1)\mu_{n-1}\mu_{n-1}^T\right]+\frac{1}{n-1}\left[X_nX_n^T+(n-1)\mu_{n-1}\mu_{n-1}^T-n\mu_n\mu_n^T\right] \\
&= \frac{n-2}{n-1}\Gamma_{n-1} \\
&\quad+\frac{1}{n-1}\left[X_nX_n^T+(n-1)\mu_{n-1}\mu_{n-1}^T-n\left(\frac{(n-1)\mu_{n-1}+X_n}{n}\right)\left(\frac{(n-1)\mu_{n-1}+X_n}{n}\right)^T\right] \\
&= \frac{n-2}{n-1}\Gamma_{n-1}-\frac{1}{n}(X_n-\mu_{n-1})(X_n-\mu_{n-1})^T
\end{aligned}$$

```r
mean_and_sd <- function(
    x,                            # m*n matrix, m samples, n dimensions
    ){
    xbar <- x,                    # matrix to store mean
    m <- dim(x)[1],               # the number of samples
    n <- dim(x)[2],               # the dimension of random vectors
    S <- matrix(rep(0,n^2),n,n)   # matrix to store covariance
    for (k in c(2:m)){
        S <- (k-2)/(k-1)*S+outer((xbar[k,]-xbar[k-1,]),(xbar[k,]-xbar[k-1,]))/k
        xbar[k,] <- (k-1)/k* xbar[k-1,] +xbar[k,]/k
    }
    return(list(xbar[dim(x)[1],],S))
}

# Example
x <- matrix(c(1,2,3,2,2,1),3,2)
mean_and_sd(x)
```

```
## [[1]]
## [1] 2.000000 1.666667
##
## [[2]]
##         [,1]       [,2]
## [1,]  1.0 -0.5000000
## [2,] -0.5  0.3333333
```

```r
# To calcaulate the sample mean and
# sample covariance matrix by R function
apply(x,2,mean)
```

```
## [1] 2.000000 1.666667
```

```r
cov(x)
```

```
##      [,1]       [,2]
## [1,]  1.0 -0.5000000
## [2,] -0.5  0.3333333
```

习题二

1. 对线性同余发生器

$$x_n = (ax_{n-1} + c)(\mathbf{mod}\ M),\ n = 1, 2, \ldots$$

证明若从初值 x_0 出发周期等于 M，则以 $0 \sim M-1$ 中任何一个整数作为初值时周期都等于 M；若从某初值 x_0 出发周期小于 M，则以 $0 \sim M-1$ 中任何一个整数作为初值时周期都小于 M。

（1）$\because$ 从 x_0 出发时，$T = M$

$\therefore$

$$\begin{aligned} x_0 &= x_M \\ &= (ax_{M-1} + c)(\text{mod } M) \\ &< M \end{aligned}$$

并且该序列形如 $\{x_0, x_1, \cdots, x_{M-1}, x_0, \cdots, x_{M-1}, \cdots\}$，其中 $x_0, x_1, \cdots, x_{M-1} \in [0, M) \cap \mathbb{Z}$ 且互不相同

$\therefore$ 以 $0 \sim M-1$ 中任何一个整数作为初值时，即某个 x_i $(I \in \{0, 1, \cdots, M-1\})$，周期都等于 M

（2）$\because$ 从 x_0 出发时，$T = m < M$ $(m \in \{1, \cdots, M-2\})$

$\therefore$

$$\begin{aligned} x_0 &= x_M \\ &= (ax_{M-1} + c)(\text{mod } M) \\ &< M \end{aligned}$$

并且该序列形如 $\{x_0, x_1, \cdots, x_{m-1}, x_0, \cdots, x_{m-1}, \cdots\}$，其中 $x_0, x_1, \cdots, x_{m-1} \in [0, M) \cap \mathbb{Z}$ 且互不相同

a. 若以 $\{x_0, x_1, \cdots, x_{m-1}\}$ 中任何一个整数作为初值时，即某个 x_i $(I \in \{0, 1, \cdots, M-1\})$，周期都等于 $m < M$

b. 若以 $\{0, 1, \cdots, M-1\} \setminus \{x_0, x_1, \cdots, x_{m-1}\}$ 中任何一个整数作为初值时：若 $\exists\ j \in \mathbb{N}^+$，s.t. $x'_j \in \{x_0, x_1, \cdots, x_{m-1}\}$，则 $x'_{j+m} = x'_j$，即序列开始发生重复，周期等于 $m < M$。

若 $\forall\ j \in \mathbb{N}^+$，$x_j \notin \{x_0, x_1, \cdots, x_{m-1}\}$，则序列中出现的元素种类不超过 $M - |\{x_0, x_1, \cdots, x_{m-1}\}|$，即周期不超过 M。

综上，若从初值 x_0 出发周期等于 M，则以 $0 \sim M-1$ 中任何一个整数作为初值时周期都等于 M；若从某初值 x_0 出发周期小于 M，则以 $0 \sim M-1$ 中任何一个整数作为初值时周期都小于 M。

2. 写程序生成 n 个如下离散型分布随机数：$P(X=1)=1/3$, $P(X=2)=2/3$。
(1) 生成 $n=100$ 个这样的随机数，计算 X 取 1 的百分比；
(2) 生成 $n=1000$ 个这样的随机数，计算 X 取 1 的百分比；
(3) 生成 $n=10000$ 个这样的随机数，计算 X 取 1 的百分比；
(4) 用中心极限定理推导 n 个这样的随机数取 1 的百分比 f_n 的渐近分布，并用此渐近分布检验上面的三组样本是否与 $P(X=1)=1/3$ 相符。

（1）$n=100$ 时，X 取 1 的百分比为 27%；
（2）$n=1000$ 时，X 取 1 的百分比为 32.4%；
（3）$n=10000$ 时，X 取 1 的百分比为 33.3%；
（4）设 $Y\sim Uniform(0,1)$，设随机变量 $\tilde{X}_i=\begin{cases}1 & ,Y\leqslant\frac{1}{3}\\ 0 & ,Y>\frac{1}{3}\end{cases}$ 表示 X_i 是否取 1 ，则

$$\begin{aligned}\mathbb{E}\tilde{X}&=1\times\mathbb{P}(Y\leqslant\frac{1}{3})+0\times\mathbb{P}(Y>\frac{1}{3})\\&=\frac{1}{3}\\Var\tilde{X}&=\left(1-\frac{1}{3}\right)^2\times\mathbb{P}(Y\leqslant\frac{1}{3})+\left(0-\frac{1}{3}\right)^2\times\mathbb{P}(Y>\frac{1}{3})\\&=\frac{2}{9}\end{aligned}$$

由中心极限定理，$\dfrac{\sum\limits_{i=1}^{n}\tilde{X}_i-\frac{n}{3}}{\sqrt{\frac{2n}{9}}}\xrightarrow{D}N(0,1)$，i.e.，取 1 的比例为 $\frac{1}{n}\sum\limits_{i=1}^{n}\tilde{X}_i\xrightarrow{D}N\left(\frac{1}{3},\frac{2n}{9}\right)$。对上述百分比数值 ans_n 进行假设检验：

$$H_0: ans_n=\frac{1}{3}\qquad\qquad H_1: ans_n\neq\frac{1}{3}$$

$$P-value=\mathbb{P}\left(\left|\frac{\frac{1}{3}-\frac{ans_n}{n}}{\sqrt{\frac{n}{9}}}\right|>Z\right)$$

$P-value$ 分别为 0.0107193, 0.016106, 5.579835×10^{-5}, 均小于 0.05。因此在 $\alpha=0.05$ 水平下，上述三组样本与 $\mathbb{P}(X=1)=\frac{1}{3}$ 相符。

```r
generate <- function(n){
    generated_data <- ifelse(runif(n)>1/3,2,1)
    return(generated_data)
}

set.seed(0)
generated_data <- generate(100)
ans100 <- length(generated_data[generated_data==1])/length(generated_data)*100
```

```
generated_data <- generate(1000)
ans1000 <- length(generated_data[generated_data==1])/length(generated_data)*100
generated_data <- generate(10000)
ans10000 <- length(generated_data[generated_data==1])/length(generated_data)*100

# test
p100 <- (pnorm(abs((1/3-ans100/100)/sqrt(200/9)))-0.5)*2
p1000 <- (pnorm(abs((1/3-ans1000/1000)/sqrt(2000/9)))-0.5)*2
p10000 <- (pnorm(abs((1/3-ans10000)/10000/sqrt(20000/9)))-0.5)*2

print(sprintf('n=100  , P value is %f',p100))
```

```
## [1] "n=100  , P value is 0.010719"
```

```
print(sprintf('n=1000 , P value is %f',p1000))
```

```
## [1] "n=1000 , P value is 0.016106"
```

```
print(sprintf('n=10000, P value is %f',p10000))
```

```
## [1] "n=10000, P value is 0.000056"
```

4. 洗好一副编号分别为 $1, 2, \cdots, 54$ 的纸牌，依次抽取出来，若第 i 次抽取到编号 i 的纸牌则称为成功抽取。编写程序估计成功抽取个数 T 的期望和方差，推导理论公式并与模拟结果进行比较。

考虑 $T = 0$，即全错位排列的情况，$D_1 = 0$，$D_2 = 1$。对 $n \in \mathbb{N}^+$，$n \geqslant 3$，不妨设 n 排在了第 k 位，其中 $k \neq n$，即 $1 \leqslant k \leqslant n-1$ 。那么考虑第 n 位的情况:

(1) 当 k 排在第 n 位时，除了 n 和 k 以外还有 $n-2$ 个数，其错排数为 D_{n-2}。

(2) 当 k 不排在第 n 位时，那么将第 n 位重新考虑成一个新的“第 k 位”，这时的包括 k 在内的剩下 $n-1$ 个数的每一种错排，都等价于只有 $n-1$ 个数时的错排（只是其中的第 k 位会换成第 n 位）。其错排数为 D_{n-1}。

因此

$$D_n = (n-1)(D_{n-2} + D_{n-1})$$

$\therefore \quad \forall\, n \in \mathbb{N}^+$，$n \geqslant 2$,

$$D_n = n! \sum_{i=2}^{n} \frac{(-1)^i}{i!}$$

令

$$D_0 = 1$$

我们有

$$\begin{aligned}\mathbb{P}(T=k) &= \frac{C_{54}^k D_{54-k}}{54!} \qquad (k=1,2,\cdots,53)\\ \mathbb{E}T &= \sum_{k=0}^{54} \frac{kC_{54}^k D_{54-k}}{54!}\\ &= 1\end{aligned}$$

$$\begin{aligned}VarT &= \sum_{k=0}^{54} \frac{(k-\mathbb{E}X)^2 C_{54}^k D_{54-k}}{54!}\\ &= 1\end{aligned}$$

```
# Permutation Generating Function
generate_range <- function(n,   # the length of generated sequences
                           m = 1# the number of generated sequences
                           ){
    generate_data <- matrix(rep(0,m*n),m,n)
    for (i in c(1:m)){
        x <- c(1:n)
        k <- n
        while(k>1){
            u <- runif(1)
            I <- ceiling(k*u)
            y <- x[k]
            x[k] <- x[I]
            x[I] <- y
            k <- k-1
        }
        generate_data[i,] <- x
    }
    return(generate_data)
}

# Generating T data
n <- 10000
Tdata <- apply(t(matrix(rep(c(1:54),n),54,n)) == generate_range(54,n),1,sum)
# Using sample mean to estimate expectation
est_mean <- mean(Tdata)
```

```
# Using sample varaince to estimate variance
est_variance <- sum((Tdata - est_mean)^2)/(length(Tdata)-1)

# Derangement Permutation Function
Dn <- function(n) {
    a <- 0
    b <- 1
    if (n == 1) {
        return(a)
    } else if (n <= 2) { # n = 0 or 2
        return(b)
    } else {
    for (i in 3:n) {
        c <- (i-1) * (a + b)
        a <- b
        b <- c
    }
    return(b)
    }
}

EX = 0
VarX = 0
for (k in c(0:54)){
    EX <- EX + k*choose(54,k)/factorial(54)*Dn(54-k)
}
for (k in c(0:54)){
    VarX <- VarX + (k-EX)^2*choose(54,k)/factorial(54)*Dn(54-k)
}

print(sprintf('The estimated mean is %f',est_mean))
```

```
## [1] "The estimated mean is 1.012300"
```

```
print(sprintf('The estimated variance is %f',est_variance))
```

```
## [1] "The estimated variance is 1.008850"
```

```r
print(sprintf('The true mean is %f',EX))
```

```
## [1] "The true mean is 1.000000"
```

```r
print(sprintf('The true variance is %f',VarX))
```

```
## [1] "The true variance is 1.000000"
```

6. 编写 R 程序，对某个均匀分布随机数发生器产生的序列做均匀性的卡方检验。

将 [0,1] 等分为 k 个不相交区间，落在每个区间的理论概率为 $p_0 = \dfrac{1}{k}$，对应的理论频数为 np_0。

$$H_0 : p_i = p_0, \quad i = 1, 2, \cdots, k \qquad\qquad H_1 : p_{i_0} \neq p_0,\ i_0 \in \{1, 2, \cdots, k\}$$

$$\begin{aligned}\chi^2 &= \sum_{i=1}^{n} \frac{(N_i - np_i)^2}{np_i} \\ &\overset{H_0}{\sim} \chi^2(k-1)\end{aligned}$$

给定 α 水平 0.05，若 $\chi^2 \geqslant \chi^2_\alpha(k-1)$，则拒绝 H_0；反之接受 H_0。

```r
chi2 <- function(R,                 # generated sequence
                 k,                 # the number of groups
                 n = length(R),     # the length of R
                 alpha = 0.05
                 ){
    N <- rep(0,k)
    for (i in c(1:k)){
        N[i] <- sum(R<i/k & R>=(i-1)/k)
    }
    mu <- rep(n/k,k)
    print(N)
    V <- sum((N - mu)^2)/(n/k)
    chisq_V <- qchisq(alpha,df=k-1)
    return(list(V,chisq_V))         # return (chisquare statistic, critical value)
}

set.seed(0)
chi2(runif(10000),20)
```

```
##  [1] 517 523 475 495 458 549 504 506 512 501 497 495 449 484 467 503 492
## [18] 527 516 530
```

```
## [[1]]
## [1] 23.536
##
## [[2]]
## [1] 10.11701
```

10. 设随机变量 X 表示成功概率为 p 的独立重复试验中第 r 次成功所需要的试验次数，称 X 服从负二项分布。
(1) 利用负二项分布与几何分布的关系构造 X 的随机数。
(2) 直接利用负二项分布的概率分布构造 X 的随机数。

设 $X \sim NB(r,p)$, $Y_1,\cdots,Y_r \overset{i.i.d.}{\sim} Geom(p)$, 则 $X = \sum\limits_{i=1}^{r} Y_i$。
$\because$

$$
\begin{aligned}
\mathbb{P}(Y=k) &= pq^{k-1} & k &= 1,2,\cdots \\
p_k &= \mathbb{P}(X=k) \\
&= \binom{k-1}{r-1} p^r q^{k-r} & k &= r, r+1, \cdots \\
p_{k+1} &= \frac{k}{k+1-r} q p_k & k &= r, r+1, \cdots \\
q &= 1-p
\end{aligned}
$$

$\therefore$

$$
\begin{aligned}
F_X(k+1) &= F_X(k) + \frac{k}{k+1-r} q p_k \\
p_r &= p^r
\end{aligned}
$$

(1) 产生 r 个独立同分布、参数为 p 的几何分布随机变量，其和即为 X 随机数。

(2) 产生均匀分布随机数 U，迭代计算 $F_X(k) \quad (k = r, r+1, \cdots)$ 直至 $F_X(k_0 - 1) < U \leqslant F_X(k_0)$，则 k_0 为产生的 X 随机数。

```r
r <- 5
p <- 1/2


# (1)
```

```
set.seed(0)
# The geometry random variable in R
# represents the number of failed trials,
# + 1 to get the number of total trials
X <- sum(rgeom(r,p)+1)

# (2)
generate_geom <- function(
    r,
    p
){
    U <- runif(1)
    k <- r
    pk <- p^r
    FX <- pk
    while (U>FX) {
        pk <- pk * k / (k+1-r)*(1-p)
        FX <- FX + pk
        k <- k+1
    }
    return(k)
}
generate_geom(r,p)
```

```
## [1] 11
```

11. 用变换法生成如下分布的随机数:

(1) $Beta(\frac{1}{n},1)$ **分布，密度为**

$$p(x) = \frac{1}{n}x^{\frac{1}{n}-1},\ \ x \in [0,1]$$

(2) $Beta(n,1)$ **分布，密度为**

$$p(x) = nx^{n-1},\ \ x \in [0,1]$$

(3) 密度为

$$p(x) = \frac{2}{\pi\sqrt{1-x^2}},\ \ x \in [0,1]$$

(4) 柯西分布，密度为

$$p(x) = \frac{1}{\pi(1+x^2)}, \;\; x \in (-\infty, \infty)$$

(5) 密度为

$$p(x) = \cos x, \;\; x \in [0, \frac{\pi}{2}]$$

(6) 威布尔分布，密度为

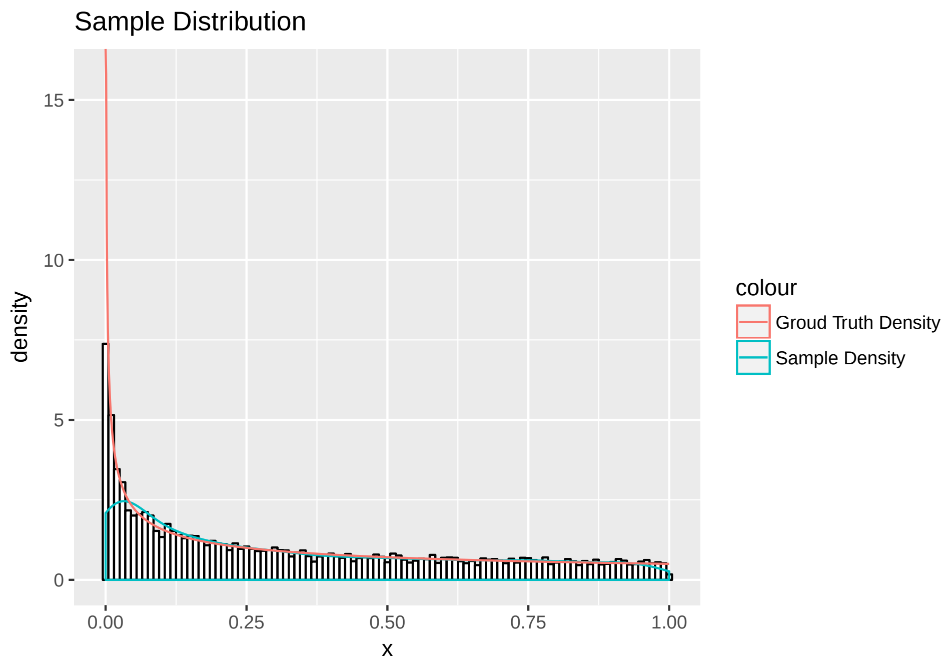
$$p(x) = \frac{\alpha}{\eta} x^{\alpha-1} e^{-\frac{x^\alpha}{\eta}}, \;\; x > 0 \;\; (\alpha > 0, \eta > 0)$$

(1)

$$\begin{aligned} F(x) &= \int_0^x \frac{1}{n} t^{\frac{1}{n}-1} \mathbb{1}_{[0,1]} \mathrm{d}t \\ &= x^{\frac{1}{n}} \\ F^{-1}(y) &= y^n \end{aligned}$$

```
generate_1 <- function(
    n,     # parameter
    m = 1  # length of sequence to be generated
){
    return(runif(m)^n)
}
n <- 2
x1 <- generate_1(n,10000)

# To generated ground truth distribution
x <- seq(0, 1, length = 1000)
y <- dbeta(x, 1/n, 1)

# Visualization
# hist(x1, probability = T,
#      main="The Sample Density Distribution",xlab = 'x')
# y1 <- density(x1,from = 0,to = 1)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",xlim=c(0,1),type='l',
#      xaxs="i", yaxs="i")
# legend("topright",legend=c("Sample Density",TeX("pdf of $Beta(1/n,1)$")),
#        lwd=1,col=c("blue4","red"), cex=0.8)
```

```r
# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.01,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```

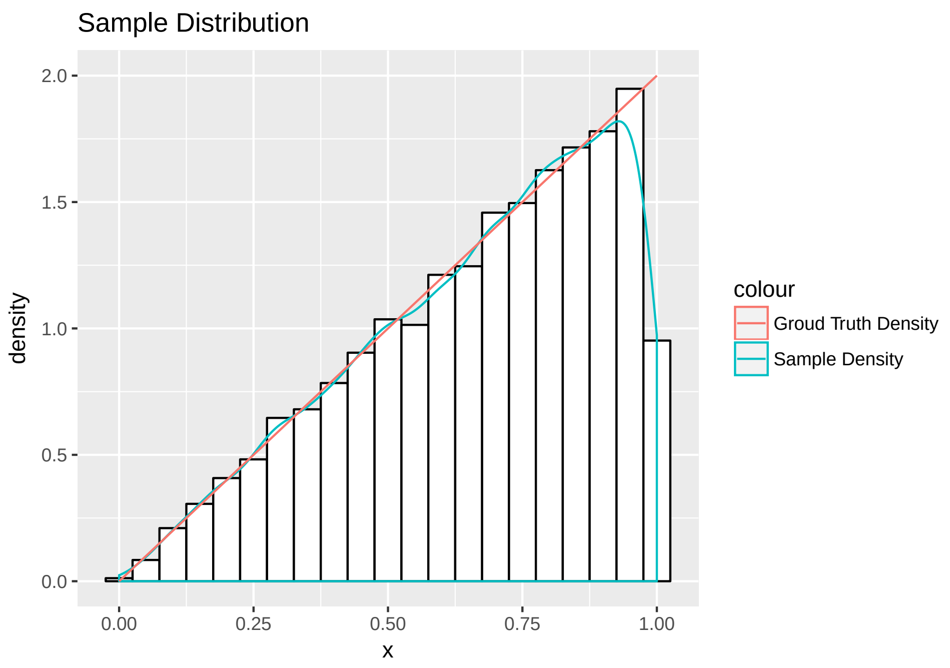


(2)

$$
\begin{aligned}
F(x) &= \int_0^x nt^{n-1}\mathbb{1}_{[0,1]}\mathrm{d}t \\
&= x^n \\
F^{-1}(y) &= y^{\frac{1}{n}}
\end{aligned}
$$

```r
generate_2 <- function(
    n,     # parameter
    m = 1  # length of sequence to be generated
){
    return(runif(m)^(1/n))
}
n <- 2
x1 <- generate_2(n,10000)
```

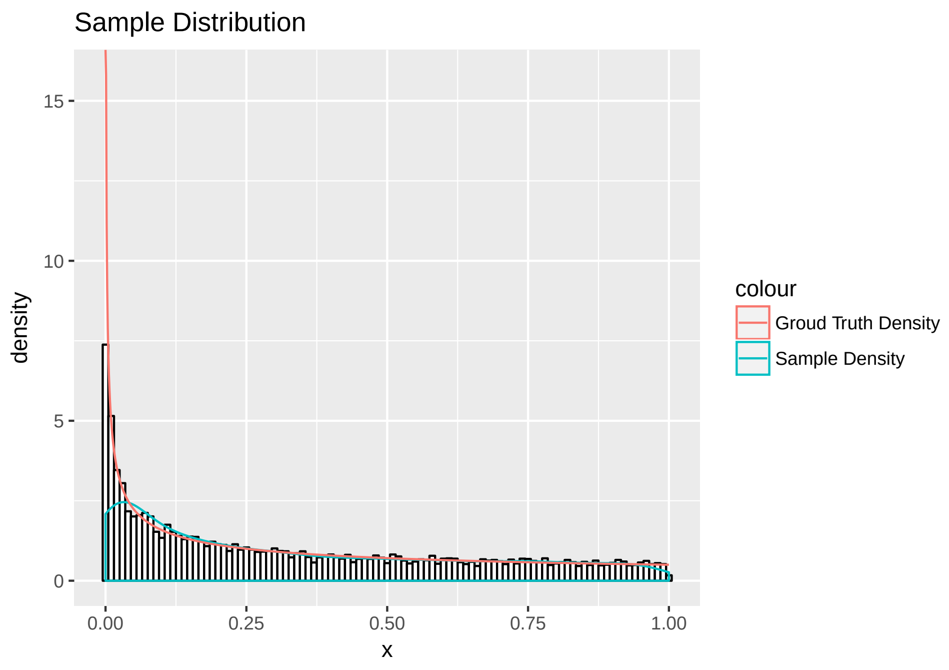
```
# To generated ground truth distribution
x <- seq(0, 1, length = 1000)
y <- dbeta(x, n, 1)

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x')
# y1 <- density(x1,from = 0,to = 1)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",xlim=c(0,1),type='l')
# legend("topleft",legend=c("Sample Density",TeX("pdf of $Beta(n,1)$")),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.05,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



(3)

$$
\begin{aligned}
F(x) &= \int_0^x \frac{2}{\pi\sqrt{1-t^2}} \mathbb{1}_{[0,1]} \mathrm{d}t \\
&= \frac{2}{\pi} \arcsin x \\
F^{-1}(y) &= \sin\left(\frac{\pi}{2} y\right)
\end{aligned}
$$

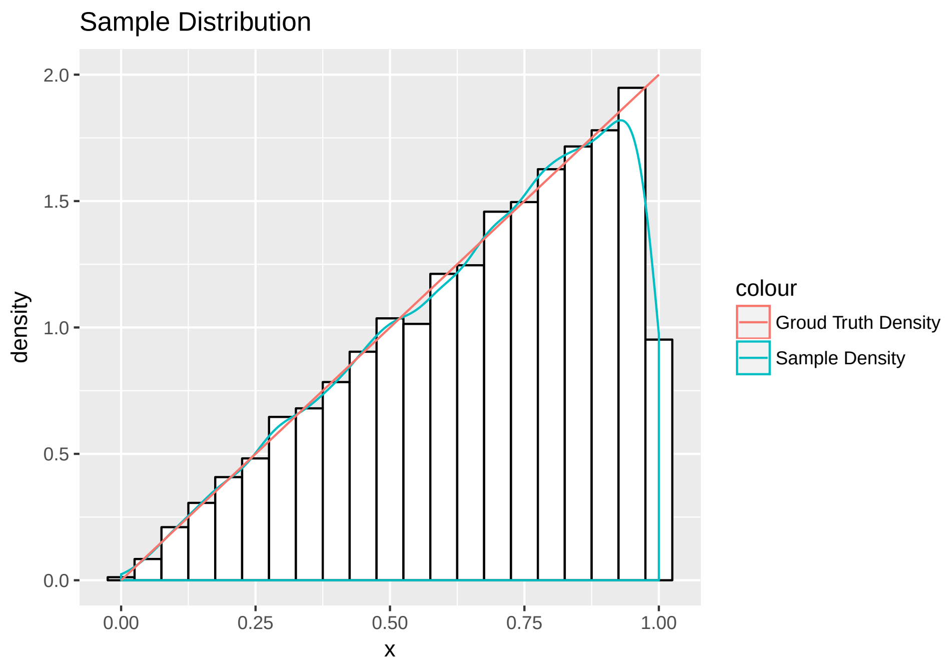
```r
generate_3 <- function(
    n,      # parameter
    m = 1  # length of sequence to be generated
){
    return(sin(runif(m)*pi/2))
}
n <- 2
x1 <- generate_3(n,10000)

# To generated ground truth distribution
x <- seq(0, 1, length = 1000)
y <- 2/pi/sqrt(1-x^2)

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x')
# y1 <- density(x1,from = 0,to = 1)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",xlim=c(0,1),type='l')
# legend("topleft",legend=c("Sample Density",TeX("True pdf")),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.5,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



(4)

$$
\begin{aligned}
F(x) &= \int_{-\infty}^{x} \frac{1}{\pi(1+t^2)}\mathrm{d}t \\
&= \frac{1}{\pi}\arctan x + \frac{1}{2} \\
F^{-1}(y) &= \tan\left(y\pi - \frac{\pi}{2}\right)
\end{aligned}
$$

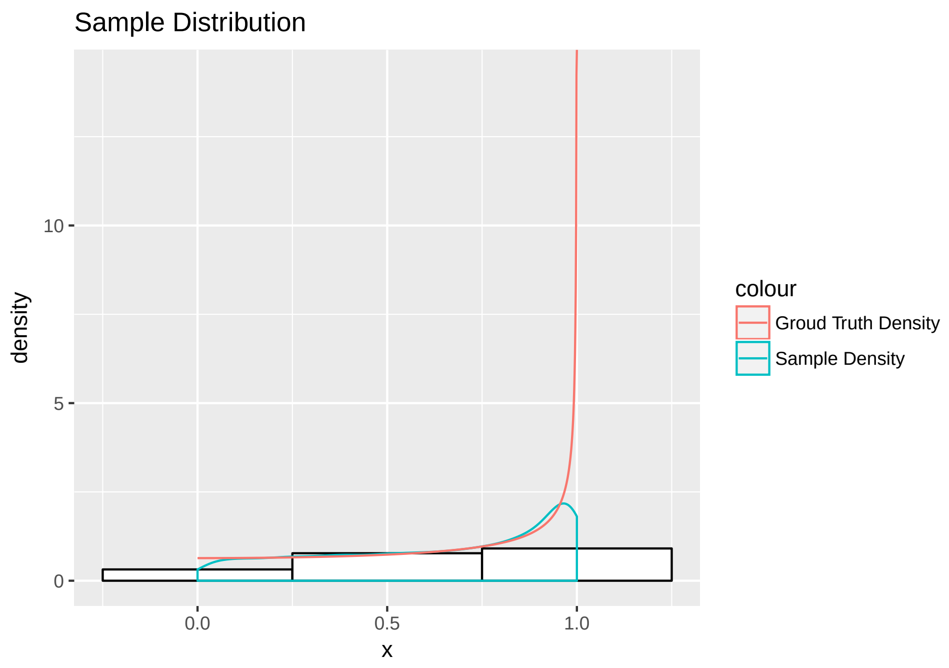
```r
generate_4 <- function(
    n,     # parameter
    m = 1  # length of sequence to be generated
){
    return(tan((runif(m)-1/2)*pi))
}
n <- 2
x1 <- generate_4(n,10000)
x1 <- x1[abs(x1)<10]

# To generated ground truth distribution
x <- seq(-10, 10, length = 1000)
y <- 1/(1+x^2)/pi

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x',
#       breaks = c(-100000,-500,-10,0,10,500,100000),
#       xlim=c(-500,500))
# y1 <- density(x1, from = -500, to = 500)
# lines(y1,col="blue4",lwd=1, xlim=c(-500,500))
# lines(x,y,col="red",xlim=c(-500,500),type='l')
# legend("topleft",legend=c("Sample Density",TeX("True pdf")),
#         lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.5,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



(5)

$$F(x) = \int_0^x \cos t \mathbb{1}_{[0,\frac{\pi}{2}]} \mathrm{d}t$$
$$= \sin x$$
$$F^{-1}(y) = \arcsin y$$

```r
generate_5 <- function(
    n,      # parameter
    m = 1  # length of sequence to be generated
){
    return(asin(runif(m)))
}
n <- 2
x1 <- generate_5(n,10000)

# To generated ground truth distribution
x <- seq(0, pi/2, length = 1000)
y <- cos(x)

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x')
# y1 <- density(x1,from = 0,to = pi/2)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",xlim=c(0,1),type='l')
# legend("topright",legend=c("Sample Density","True pdf"),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.02,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```

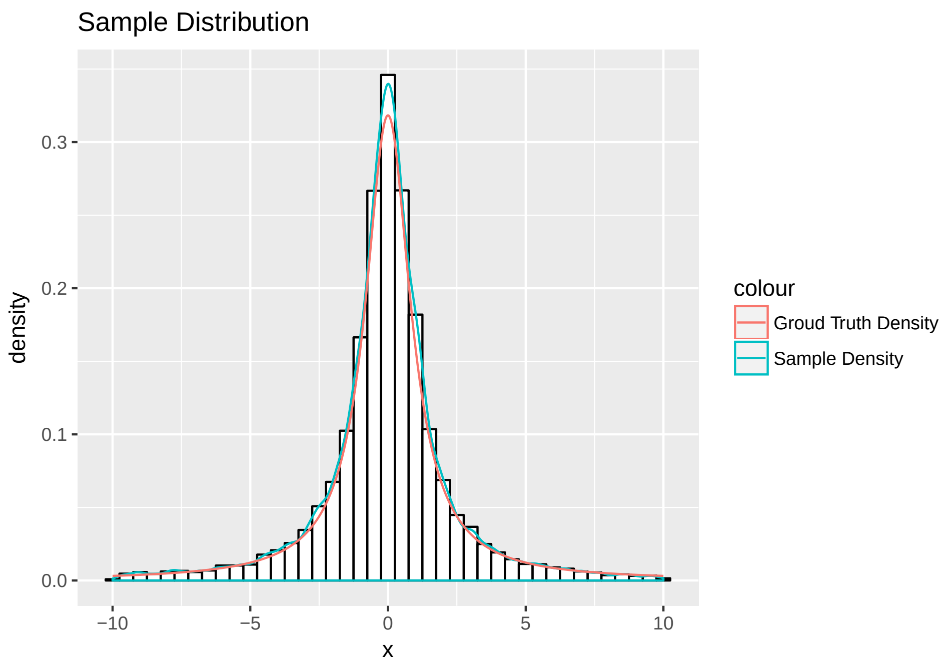


(6)

$$
\begin{aligned}
F(x) &= \int_0^x \frac{\alpha}{\eta} t^{\alpha_1} e^{-\frac{t^\alpha}{\eta}} \mathbb{1}_{[0,\infty)} \mathrm{d}t \\
&= 1 - e^{-\frac{x^\alpha}{\eta}} \\
F^{-1}(y) &= [-\eta \ln(1-y)]^{\frac{1}{\alpha}}
\end{aligned}
$$

```
generate_6 <- function(
    n,      # parameter
    alpha,
    eta,
    m = 1  # length of sequence to be generated
){
    return((- eta * log(1-runif(m)))^(1/alpha))
}
n <- 2
alpha <- 2
eta <- 1
x1 <- generate_6(n,alpha, eta,10000)

# To generated ground truth distribution
x <- seq(0, 5, length = 10000)
y <- alpha/eta*x^(alpha-1)*exp(-x^(alpha)/eta)
```

```
# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x')
# y1 <- density(x1,from = 0)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",type='l')
# legend("topright",legend=c("Sample Density","True pdf"),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.05,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



15. 设 X 为标准指数分布 $Exp(1)$ 随机变量，模拟在 $X < 0.05$ 条件下 X 的分布，密度为

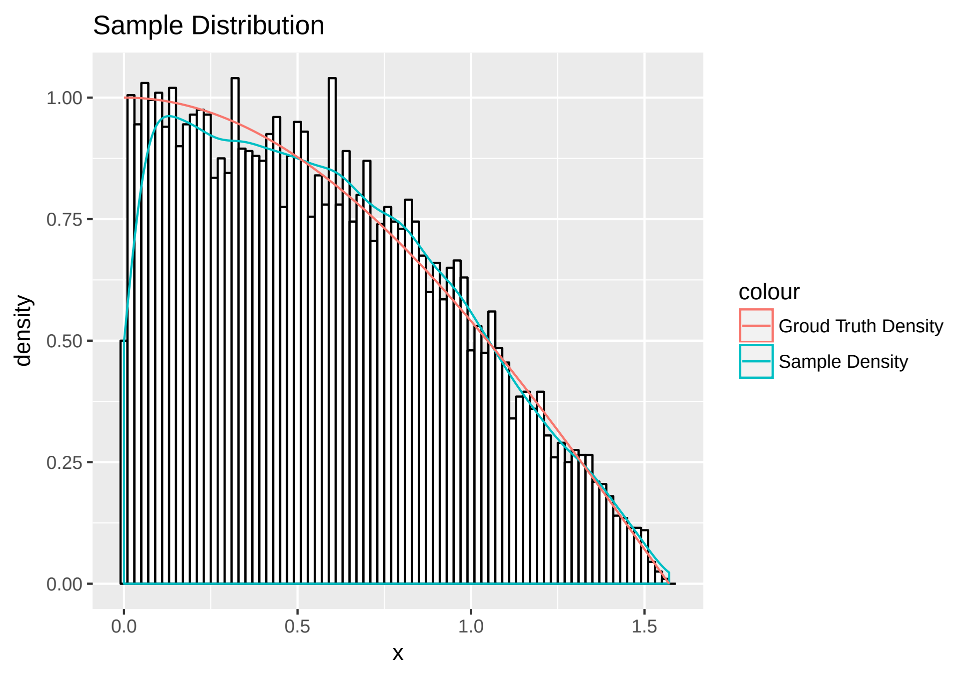
$$p(x) = \frac{e^{-x}}{1-e^{-0.05}},\quad 0 < x < 0.05$$

生成 1000 **个这样的随机数，并用它们估计** $E(X|X<0.05)$**，推导** $E(X|X<0.05)$ **的精确值并与模拟结果比较。**

$$
\begin{aligned}
F(x) &= \int_0^x \frac{e^{-t}}{1-e^{-0.05}} \mathbb{1}_{(0,0.05)} \mathrm{d}t \\
&= \frac{1-e^{-x}}{1-e^{-0.05}} \\
F^{-1}(y) &= -\ln[1-(1-e^{-0.05})y]
\end{aligned}
$$

$$
\begin{aligned}
\mathbb{E}(X|X<0.05) &= \int_0^{0.05} t\frac{e^{-t}}{1-e^{-0.05}} \mathrm{d}t \\
&= \frac{-(t+1)e^{-t}}{1-e^{-0.05}}\Big|_0^{0.05} \\
&= \frac{1-1.05e^{-0.05}}{1-e^{-0.05}} \\
&\approx 0.02479168
\end{aligned}
$$

```r
set.seed(0)
x1 <- -log(1-(1-exp(-0.05))*runif(1000))

# To generate the ground truth distribution
x <- seq(0, 0.05, length = 10000)
y <- exp(-x)/(1-exp(-0.05))

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x')
# y1 <- density(x1,from = 0, to = 0.05)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",type='l')
# legend("topright",legend=c("Sample Density","True pdf"),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=0.001,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



```
# To estimate expectation value of X
Ex1 <- mean(x1)
print(sprintf('The estimated expectation value is %f',Ex1))
```

```
## [1] "The estimated expectation value is 0.024822"
```

```
print(sprintf('The true expectation value is %f',(1-1.05*exp(-0.05))/(1-exp(-0.05))))
```

```
## [1] "The true expectation value is 0.024792"
```

16. 证明定理 2.2.5。设 U_1, U_2 独立且都服从 $U(0,1)$,

$$\begin{cases} X = \sqrt{-2\ln U_1}\cos(2\pi U_2) \\ Y = \sqrt{-2\ln U_1}\sin(2\pi U_2) \end{cases}, \tag{2.6}$$

则 X, Y 独立且都服从标准正态分布。(2.6) 叫做 $BoxMuller$ 变换。

从 (U_1,U_2) 到 (X,Y) 的逆变换为

$$\begin{cases} U_1 = e^{-\frac{1}{2}(X^2+Y^2)} \\ U_2 = \dfrac{1}{2\pi}\arctan\frac{Y}{X} \end{cases}$$

逆变换的 Jacobi 行列式为

$$
\begin{aligned}
J &= \begin{vmatrix} \frac{\partial X}{\partial U_1} & \frac{\partial X}{\partial U_2} \\ \frac{\partial Y}{\partial U_1} & \frac{\partial Y}{\partial U_2} \end{vmatrix} \\
&= -\frac{1}{2\pi} e^{-\frac{1}{2}(X^2+Y^2)}
\end{aligned}
$$

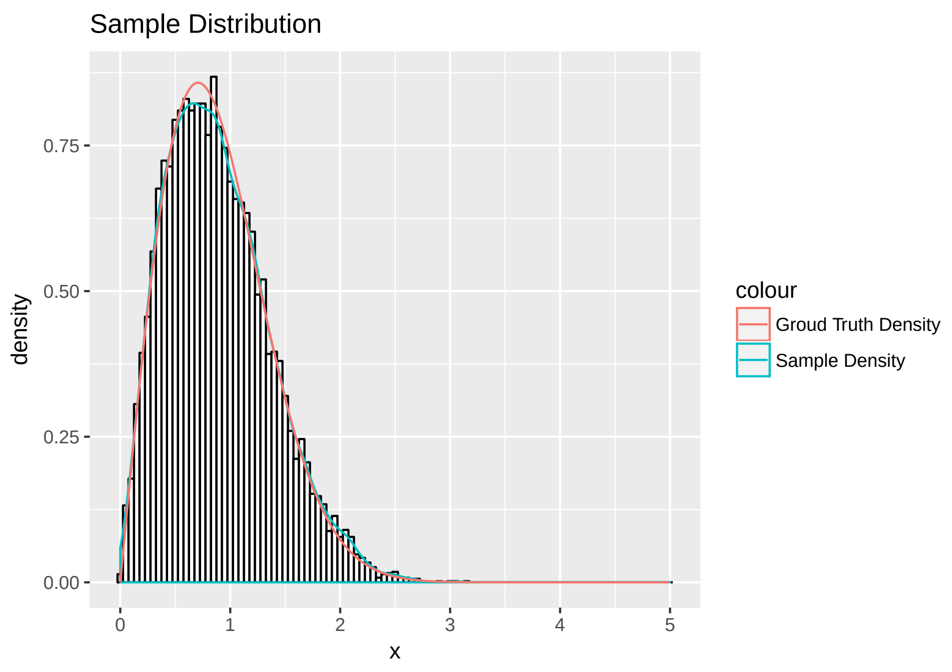
所以 (X, Y) 的联合密度为

$$
\begin{aligned}
f(x,y) &= f_{U_1,U_2}(x,y) \cdot |J| \\
&= \frac{1}{2\pi} e^{-\frac{1}{2}(x^2+y^2)} \\
&= \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}
\end{aligned}
$$

即 X, Y 独立且分别服从标准正态分布。

```r
BoxMuller <- function(n   # length of generated sequence
){
  U1 <- runif(n)
  U2 <- runif(n)
  X <- sqrt(-2*log(U1))*cos(2*pi*U2)
  Y <- sqrt(-2*log(U1))*sin(2*pi*U2)
  return(list(X,Y))
}

generated_data <- BoxMuller(1000)
x1 <- unlist(generated_data[1])
x2 <- unlist(generated_data[2])

# To generate the ground truth distribution
x <- seq(-10, 10, length = 10000)
y <- dnorm(x)

sample_density_x <- ggplot(NULL, aes(x=x1)) +
    geom_histogram(aes(y=..density..),
    binwidth=0.5,
    colour="black", fill="white") +
    geom_density(alpha=.2,  aes(colour = 'Sample Density'))
sample_density_y <- ggplot(NULL, aes(x=x2)) +
    geom_histogram(aes(y=..density..),
    binwidth=0.5,
    colour="black", fill="white") +
    geom_density(alpha=.2,  aes(colour = 'Sample Density'))
```

```r
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
p1 <- sample_density_x + groudtruth_density + labs(title="Sample Distribution of X", x = 'x')
p2 <- sample_density_y + groudtruth_density + labs(title="Sample Distribution of Y", x = 'x')

# To build a subplot
grid.newpage()                                  # new page
pushViewport(viewport(layout = grid.layout(2,1))) # divide the page into 2*1
vplayout <- function(x,y){
  viewport(layout.pos.row = x, layout.pos.col = y)
}
print(p1, vp = vplayout(1,1))
print(p2, vp = vplayout(2,1))
```



19. 设 $X \sim B(n,p)$，k 为满足 $0 \leqslant k \leqslant n$ 的给定的整数，随机变量 Y 的分布函数为

$$P(Y \leqslant y) = P(X \leqslant y|X \geqslant k)$$

记 $\alpha = P(X \geqslant k)$。分别用逆变换法和舍选法生成 Y 的随机数。当 α 取值大的时候还是取值小的时候舍选法不可取?

(1) 逆变换法

$$\begin{aligned}F_Y(y) &= \mathbb{P}(Y \leqslant y)\\&= \mathbb{P}(X \leqslant y|X \geqslant k)\\&= \frac{\mathbb{P}(k \leqslant X \leqslant y)}{\mathbb{P}(X \geqslant k)}\\&= \frac{F_X(y) - 1 + \alpha}{\alpha}\end{aligned}$$

```r
generate_Y <- function(
    n,          # number of trials
    p,          # probability of success
    k,          # parameter
    m = 1       # length of generated sequnce
){
    alpha <- 1-pbinom(k-1,n,p)
    FX <- pbinom(c(k:n),n,p)
    FY <- (FX-1)/alpha+1
    U <- runif(m)
    for (i in c(k:n-1)){
        U[U<FY[i-k+1]] <- i
    }
    U
}
n <- 10
p <- 1/2
k <-5
generate_Y(n,p,5,100)
```

```
##   [1] 5 6 6 5 8 5 5 6 6 7 6 8 6 5 9 6 7 7 5 5 5 5 7 5 7 5 7 6 5 5 6 6 6 5 8
##  [36] 6 5 5 6 7 5 5 5 6 6 5 6 6 5 6 8 5 6 6 7 7 5 7 6 6 5 5 8 7 5 6 8 5 5 8
##  [71] 6 5 5 6 6 7 6 8 5 5 5 5 5 6 5 6 7 6 5 7 6 6 6 5 5 6 7 5 6 6
```

(2) 舍选法

$$\begin{aligned}\frac{f_Y(y)}{f_X(y)} &= \frac{f_X(y)}{f_X(y)\alpha} \qquad\qquad y=k,k+1,\cdots,n\\ &= \frac{1}{\alpha}\end{aligned}$$

$\frac{1}{\alpha}$ 取值越小越好，即 α 取值小时不可取。

```r
generate_Y2 <- function(
    n,          # number of trials
    p,          # probability of success
    k,          # parameter
    m = 1       # length of generated sequnce
){
    Z <- rep(0,1,m)
    alpha <- 1-pbinom(k-1,n,p)
    for (i in c(1:m)){
        Y <- 1
        flag <- 0
        while(Y>flag){
            X <- rbinom(1,n,p)
            Y <- runif(1)
            if (X<k){
                flag <- 0
            } else{
                flag <- 1
            }
        }
        Z[i] <- X
    }
    return(Z)
}
n <- 10
p <- 1/2
k <-5
generate_Y2(n,p,5,100)
```

```
##   [1] 7 6 5 5 6 6 5 6 6 6 7 7 5 5 6 5 5 7 5 5 6 5 6 5 5 7 6 8 6 6 6 7 7 6 6
##  [36] 7 6 5 7 5 6 5 5 6 7 5 6 6 5 7 5 7 6 6 7 5 6 6 5 7 5 5 5 8 6 5 5 5 6 6
##  [71] 5 6 7 7 5 5 7 6 5 5 7 7 5 6 7 5 6 7 5 8 7 6 6 5 6 5 5 5 5 5
```

24. 设随机变量 X 的分布密度为

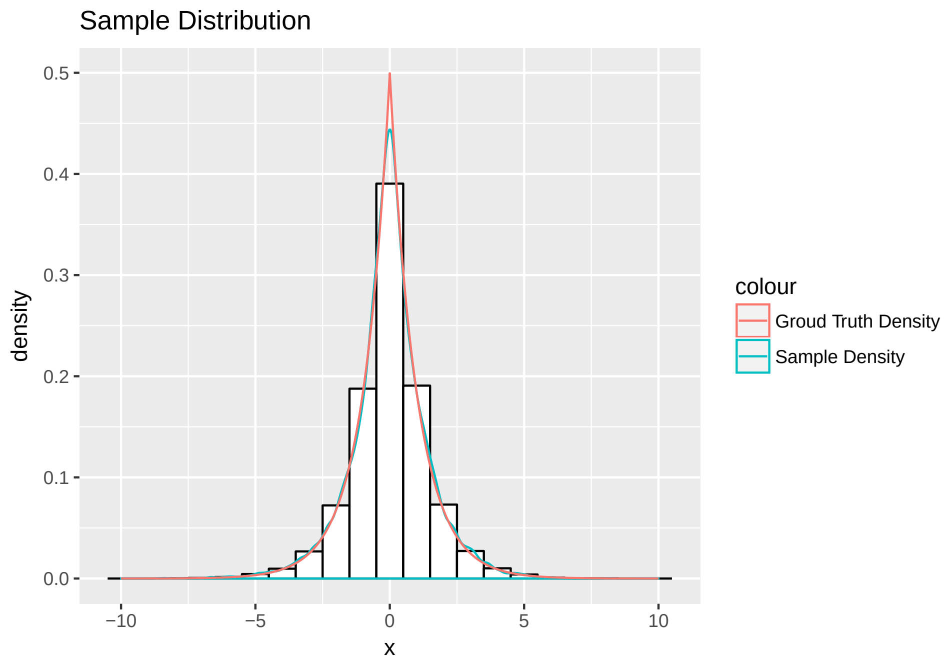
$$p(x) = \frac{1}{2}e^{-|x|},\quad -\infty < x < \infty$$

称 X 服从双指数分布或 $Laplace$ 分布。分别用逆变换法和复合法生成 X 的随机数。

(1) 逆变换法

$$\begin{aligned} F(x) &= \int_{-\infty}^{x} \frac{1}{2}e^{-|t|}\mathrm{d}t \\ &= \begin{cases} \frac{1}{2}e^{x} & ,x \leqslant 0 \\ 1-\frac{1}{2}e^{-x} & ,x > 0 \end{cases} \\ F^{-1}(y) &= \begin{cases} \ln(2y) & ,y \leqslant \frac{1}{2} \\ -\ln(2-2y) & ,y > \frac{1}{2} \end{cases} \end{aligned}$$

```r
generate_Laplace <- function(
    m = 1  # length of sequence to be generated
){
    u <- runif(m)
    z <- ifelse(u<=1/2,log(2*u),-log(2-2*u))
    return(z)
}

x1 <- generate_Laplace(10000)

# To generate the ground truth distribution
x <- seq(-10, 10, length = 10000)
y <- exp(-abs(x))/2

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x',
#      xlim = c(-10,10), ylim = c(0,0.6))
# y1 <- density(x1,from = -10,to = 10)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",type='l')
# legend("topright",legend=c("Sample Density","True pdf"),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=1,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



(2) 复合法

令

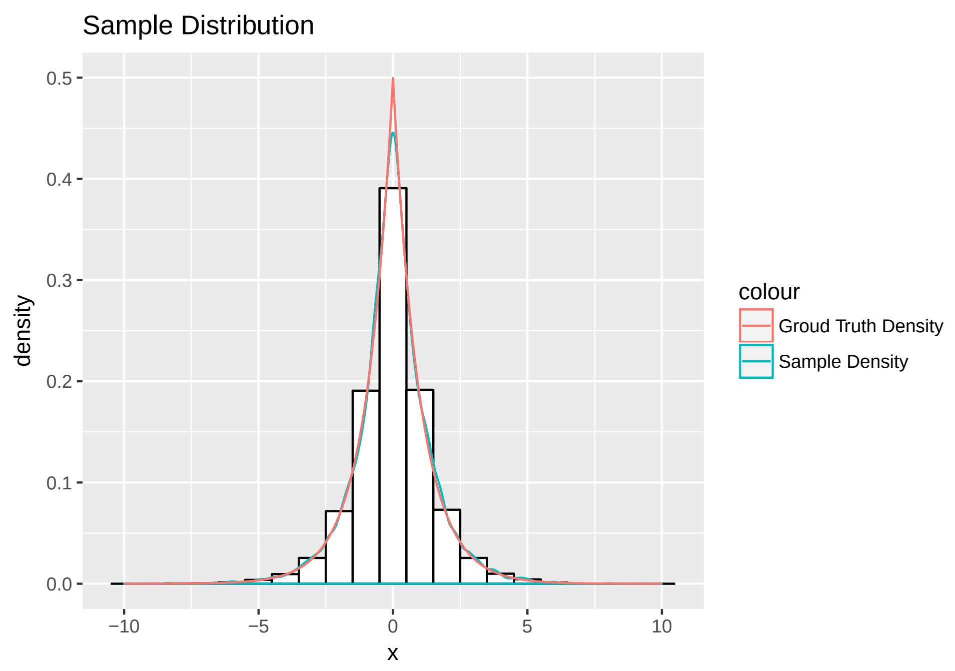
$$\mathbb{P}(I=i)=\frac{1}{2} \qquad\qquad i=1,2$$
$$Z_1\sim Exp(1)$$
$$-Z_2\sim Exp(1)$$

则

$$X=\begin{cases} Z_1 & ,I=1\\ Z_2 & ,I=2\end{cases}$$
$$F(x)=\sum_{i=1}^{2}\mathbb{P}(X\leqslant x|I=i)\mathbb{P}(I=i)$$
$$=\frac{1}{2}\sum_{i=1}^{2}\mathbb{P}(Z_i\leqslant x)$$
$$p(x)=\frac{1}{2}[f_{Z_1}(x)+f_{Z_2}(x)]$$

可以先生成 I，然后根据 I 的值生成 Z_I 作为 X。

```r
generate_Laplace2 <- function(
    m = 1  # length of sequence to be generated
){
    I <- runif(m)
    z <- rep(0,m)
    z <- rexp(m,1)*ifelse(I<=1/2,1,-1)
    return(z)
}

x1 <- generate_Laplace2(10000)

# To generate the ground truth distribution
x <- seq(-10, 10, length = 10000)
y <- exp(-abs(x))/2

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x',
#      xlim = c(-10,10), ylim = c(0,0.6))
# y1 <- density(x1,from = -10,to = 10)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",type='l')
# legend("topright",legend=c("Sample Density","True pdf"),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=1,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```



26. 给出生成密度函数为

$$p(x) = \frac{1}{0.000336} x(1-x)^3,\ 0.8 < x < 1$$

的随机数的有效算法。

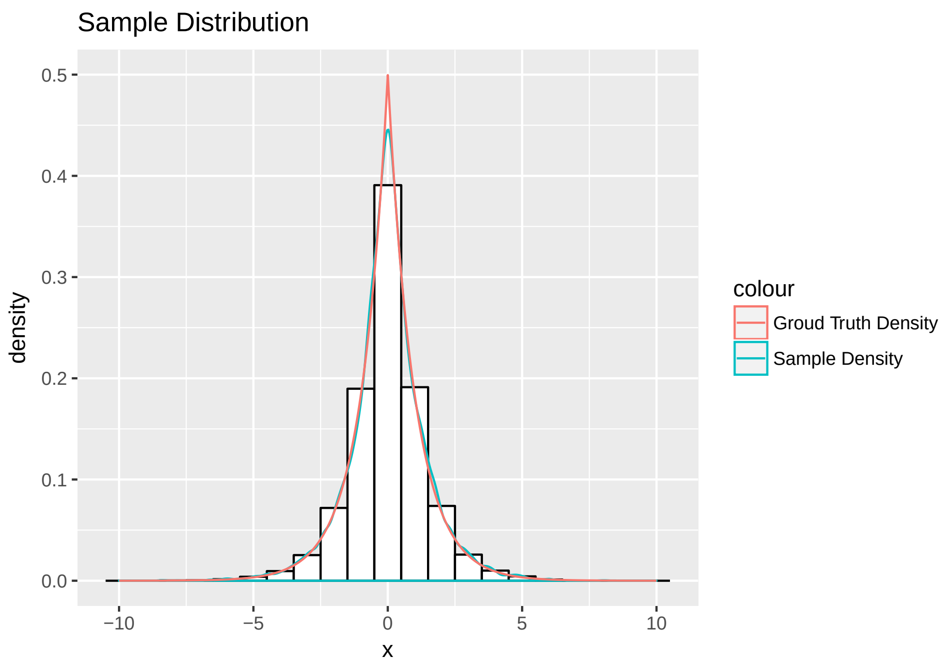
$\because$ 当 $x \in [0.8, 1]$ 时，

$$\begin{aligned} p'(x) &= \frac{1}{0.000336} \frac{\mathrm{d}}{\mathrm{d}x} [(1-x)^3 - (1-x)^4] \\ &= \frac{1}{0.000336} [4(1-x)^3 - 3(1-x)^2] \\ &= \frac{1}{0.000336} (1-4x)(1-x)^2 \leqslant 0 \end{aligned}$$

$\therefore$ $p(x)$ 在 $[0.8, 1]$ 上递减

$$\begin{aligned} p(x) &\leqslant p(0.8) \\ &= \frac{1}{0.000336} \cdot 0.8 \cdot (1-0.8)^3 \\ &= 19.04762 \end{aligned}$$

```r
generate_X <- function(
    m = 1       # length of generated sequnce
){
    M <- 0.8*0.2^3/0.000336
    Z <- rep(0,1,m)
    for (i in c(1:m)){
        Y <- 1
        pX <- 0
        while(Y>pX){
            U <- runif(2)
            X <- 0.8 + 0.2*U[1]
            Y <- M * U[2]
            pX <- X*(1-X)^3/0.000336
        }
        Z[i] <- X
    }
    return(Z)
}

x1 <- generate_X(1000)

# To generate the ground truth distribution
x <- seq(0.8, 1, length = 10000)
y <- x*(1-x)^3/0.000336

# Visualization
# hist(x1, freq = F, main="The Sample Density Distribution",xlab = 'x',
#      xlim = c(0.8,1))
# y1 <- density(x1,from = 0.8,to = 1)
# lines(y1,col="blue4",lwd=1)
# lines(x,y,col="red",type='l')
# legend("topright",legend=c("Sample Density","True pdf"),
#        lwd=1,col=c("blue4","red"), cex=0.8, xjust=1)

# Visualization by ggplot2
sample_density <- ggplot(NULL, aes(x=x1)) +
geom_histogram(aes(y=..density..),
    binwidth=.005,
    colour="black", fill="white") +
geom_density(alpha=.2,  aes(colour = 'Sample Density'))
groudtruth_density <- geom_line( aes(x, y,colour = 'Groud Truth Density'))
sample_density + groudtruth_density + labs(title="Sample Distribution", x = 'x')
```


Sample Distribution
density
0.5
0.4
0.3
0.2
0.1
0.0
−10
−5
0
5
10
x
colour
Groud Truth Density
Sample Density