节选第三章

3.3.$\int_0^1 exp\left\{e^x\right\}dx$

解: 代码如下

```
#法一:随机模拟
r <- runif(10000, min = 0, max = 1)
mean(exp(exp(r)))
#法二:直接用积分函数求解
f <- function(x){exp(exp(x))}
integrate(f, lower = 0, upper = 1)
```

第 3 题运行截图

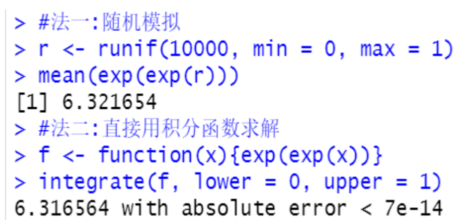


图 1: 第三题 R 代码

3.7.$\int_{-\infty}^{\infty} e^{-x^2}dx$

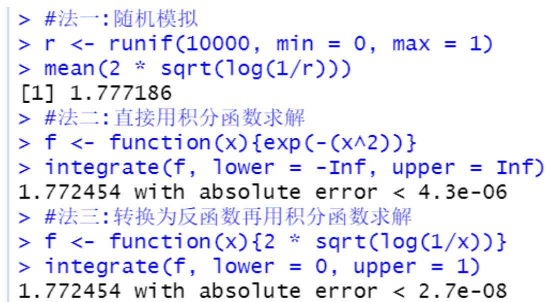
这一题用随机模拟需要注意一点: 由于 R 语言中的 $runif$ 函数不能输入 $-Inf$(负无穷) 和 Inf (正无穷), 因此可以先两边同时取个对数:$\ln y = -x^2$, 再转化为: $\sqrt{\ln\frac{1}{y}} = x^2$, 因此原积分转化为: $2\int_0^1 \sqrt{\ln\frac{1}{y}}dy$

解: 代码如下

```
#法一:随机模拟
r <- runif(10000, min = 0, max = 1)
mean(2 * sqrt(log(1/r)))
#法二:直接用积分函数求解
f <- function(x){exp(-(x^2))}
integrate(f, lower = -Inf, upper = Inf)
```

```
#法三:对数转化再用积分函数求解
f <- function(x){2 * sqrt(log(1/x))}
integrate(f, lower = 0, upper = 1)
```

第 7 题运行截图



```
> #法一:随机模拟
> r <- runif(10000, min = 0, max = 1)
> mean(2 * sqrt(log(1/r)))
[1] 1.777186
> #法二:直接用积分函数求解
> f <- function(x){exp(-(x^2))}
> integrate(f, lower = -Inf, upper = Inf)
1.772454 with absolute error < 4.3e-06
> #法三:转换为反函数再用积分函数求解
> f <- function(x){2 * sqrt(log(1/x))}
> integrate(f, lower = 0, upper = 1)
1.772454 with absolute error < 2.7e-08
```

图 2: 第七题 R 代码

3.9.$\int_0^\infty \int_0^x e^{-(x+y)} dydx$

解: 原式 $= \int_0^\infty e^{-x} \int_0^x e^{-y} dydx = \int_0^\infty e^{-x}(1-e^{-x})dx$

令 $y = e^{-x}$, 则 $\int_1^0 (y-1)dy = \int_0^1 (1-y)dy$

```
#法一:平均值模拟
r <- runif(10000, min = 0, max = 1)
mean(1-r)
#法二:蒙特卡洛模拟
n <- 10000
x<-runif(n, min = 0, max = 1)
y<-runif(n, min = 0, max = 1)
x1<--log(x)
y1<--log(y)
z<-sum((exp(-x1-y1)/(x*y))[x1>y1])/n
z
```

第 9 题运行截图

```
> #法一:平均值模拟
> r <- runif(10000, min = 0, max = 1)
> mean(1-r)
[1] 0.4999782
> #法二:蒙特卡洛模拟
> n <- 10000
> x<-runif(n, min = 0, max = 1)
> y<-runif(n, min = 0, max = 1)
> x1<--log(x)
> y1<--log(y)
> z<-sum((exp(-x1-y1)/(x*y))[x1>y1])/n
> z
[1] 0.509
```

图 3: 第九题 R 代码

3.11.Let U be uniform on (0,1). Use simulation to approximate the following:

(a) $Corr(U, \sqrt{1-U^2})$

(b) $Corr(U^2, \sqrt{1-U^2})$

解: 思路是运用公式 $Corr(x,y) = \frac{Cov(x,y)}{\sigma_x \sigma_y} = \frac{E(xy)-E(x)E(y)}{\sigma_x \sigma_y}$

(a) 题代码:

```
r <- runif(10000, min = 0, max = 1)
EXY <- mean(r * sqrt(1-r^2))
EXEY <- mean(r) * mean(sqrt(1-r^2))
sigma <- sd(r) * sd(sqrt(1-r^2))
corr <- (EXY - EXEY) / sigma
corr
```

(b) 题代码:

```
r <- runif(10000, min = 0, max = 1)
EXY <- mean(r^2 * sqrt(1-r^2))
EXEY <- mean(r^2) * mean(sqrt(1-r^2))
sigma <- sd(r^2) * sd(sqrt(1-r^2))
corr <- (EXY - EXEY) / sigma
corr
```

第 11 题运行截图

```
> r <- runif(10000, min = 0, max = 1)
> EXY <- mean(r * sqrt(1-r^2))
> EXEY <- mean(r) * mean(sqrt(1-r^2))
> sigma <- sd(r) * sd(sqrt(1-r^2))
> corr <- (EXY - EXEY) / sigma
> corr
[1] -0.9228916
> r <- runif(10000, min = 0, max = 1)
> EXY <- mean(r^2 * sqrt(1-r^2))
> EXEY <- mean(r^2) * mean(sqrt(1-r^2))
> sigma <- sd(r^2) * sd(sqrt(1-r^2))
> corr <- (EXY - EXEY) / sigma
> corr
[1] -0.9833632
```

图 4: 第十一题 R 代码

3.12.For uniform(0,1) random variables $U1$, $U2$, ... define

$$N = Minimum\left\{ \ n : \sum_{i=1}^{n} U_i > 1 \ \right\}$$

That is, N is equal to the number of random numbers that must be summed to exceed 1.

(a) Estimate E[N] by generating 100 values of N.

(b) Estimate E[N] by generating 1000 values of N.

(c) Estimate E[N] by generating 10000 values of N.

(d) What do you think is the values of E[N]?

解:(a)-(c) 题代码如下

```
f <- function(n){
    result <- vector(length = n)
    for(i in 1:n) {
        r <- runif(10000, min = 0, max = 1)
        sum = 0
        for(j in 1:length(r)) {
            sum = sum + r[j]
            if(sum > 1) {
                result[i] = j
                break
```

```
            }
        }
    }
    print(mean(result))
}
f(100)
f(1000)
f(10000)
```

第十二题运行截图

```
> f(100)
[1] 2.81
> f(1000)
[1] 2.707
> f(10000)
[1] 2.7185
```

图 5: 第十二题 R 代码

(d) 题: 结合 (a)-(c) 题结果, 可以猜测 E[N] 的值为自然底数 e

3.13. Let U_i, $i \geq 1$, be random numbers.Define N by

$$N = Minimum\left\{\ n : \prod_{i=1}^{0} U_i > 1\ \right\}$$

where $\prod_{i=1}^{0} U_i \equiv 1$.

(a) Find E[N] by simulation

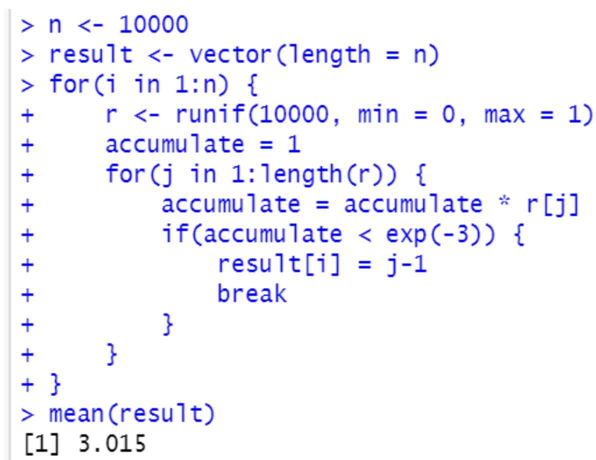
(b) Find $P\{N = i\}$, for i = 0,1,2,3,4,5,6, by simulation.

解:(a) 题代码如下

```
n <- 10000
result <- vector(length = n)
for(i in 1:n) {
    r <- runif(10000, min = 0, max = 1)
    accumulate = 1
    for(j in 1:length(r)) {
        accumulate = accumulate * r[j]
```

```
            if(accumulate < exp(-3)) {
                result[i] = j-1
                break
            }
        }
}
mean(result)
```

第十三题 (a) 运行截图



```
> n <- 10000
> result <- vector(length = n)
> for(i in 1:n) {
+     r <- runif(10000, min = 0, max = 1)
+     accumulate = 1
+     for(j in 1:length(r)) {
+         accumulate = accumulate * r[j]
+         if(accumulate < exp(-3)) {
+             result[i] = j-1
+             break
+         }
+     }
+ }
> mean(result)
[1] 3.015
```

图 6: 第十三题 (a)R 代码

(b) 题代码如下:

```
n <- 10000
count <- vector(length = 7)
for(i in 1:n) {
    r <- runif(10000, min = 0, max = 1)
    accumulate = 1
    for(j in 1:length(r)) {
        accumulate = accumulate * r[j]
        if(accumulate < exp(-3)) {
            if(j-1 <= 6) {
                count[j] = count[j] + 1
```

```
                }
                break
            }
        }
    }
    for(k in 1:length(count)) {
        print(count[k]/n)
    }
```

第十三题 (b) 运行截图

```
> n <- 10000
> count <- vector(length = 7)
> for(i in 1:n) {
+     r <- runif(10000, min = 0, max = 1)
+     accumulate = 1
+     for(j in 1:length(r)) {
+         accumulate = accumulate * r[j]
+         if(accumulate < exp(-3)) {
+             if(j-1 <= 6) {
+                 count[j] = count[j] + 1
+             }
+             break
+         }
+     }
+ }
> for(k in 1:length(count)) {
+     print(count[k]/n)
+ }
[1] 0.0504
[1] 0.1556
[1] 0.2199
[1] 0.2166
[1] 0.1736
[1] 0.0993
[1] 0.0525
```

图 7: 第十三题 (b)R 代码

所以:

$P\{N=0\}=0.0504$	$P\{N=1\}=0.1556$
$P\{N=2\}=0.2199$	$P\{N=3\}=0.2166$
$P\{N=4\}=0.1736$	$P\{N=5\}=0.0993$
$P\{N=6\}=0.0525$	——

3.14. With $x_1 = 23$, $x_2 = 66$, and

$$x_n = 3x_{n-1} + 5x_{n-2} \quad mod(100), \quad n \geq 3$$

We will call the sequence $u_n = x_n/100$, $n \geq 1$, the text's random number sequence. Find its first 14 values.

解: 代码如下

```
result <- vector(length = 14)
x <- vector(length = 14)
result[1] = 23
result[2] = 66
x[1] = 23 %% 100
x[2] = 66 %% 100
for(i in 3:14) {
    x[i] = 3 * result[i-1] + 5 * result[i-2]
    result[i] = x[i] %% 100
}
print(round(result/100,2))
```

第十四题运行截图

```
> result <- vector(length = 14)
> x <- vector(length = 14)
> result[1] = 23
> result[2] = 66
> x[1] = 23 %% 100
> x[2] = 66 %% 100
> for(i in 3:14) {
+     x[i] = 3 * result[i-1] + 5 * result[i-2]
+     result[i] = x[i] %% 100
+ }
> print(round(result/100,2))
 [1] 0.23 0.66 0.13 0.69 0.72 0.61 0.43 0.34 0.17 0.21 0.48 0.49 0.87 0.06
```

图 8: 第十四题 R 代码