**上机作业四**

1. 完成轮船相遇（实验十）的理论分析，用R编程实现仿真模拟分析。

理论分析：

boat <- function(){

  x <- runif(1, min = 0, max = 24)

  y <- runif(1, min = 0, max = 24)

  if((x<=y | x>=y+2) & (y<=x | y>=x+1)) return(T)

  else return(F)

}

N <- 10000;

sum <- 0;

for(i in 1:N){

  if(boat()) sum = sum + 1;

}

p <- sum/N; p

[1] 0.8779

1. 完成最大游程问题（实验十一）的理论分析，用R编程实现仿真模拟分析。

理论分析：

1. 记：在次试验中连续出现次点数为7，：前次都为7，：第一次非7为第次，。则其中，，，，即

由，迭代可得

fun <- function(n, k = 5, p = 1/6, memo){

  if(is.null(memo)) memo <- c(rep(0, k-1), p^k, rep(NA, n-k))

  if(!is.na(memo[n])){

    list(value = memo[n], memory = memo)

  }

  else{

    rt <- p^k

    for(j in 1:k){

      temp <- fun(n = n-j, memo = memo)

      memo <- temp$memory

      rt <- rt + temp$value\*p^(j-1)\*(1-p)

    }

    memo[n] <- rt

    list(value = rt, memory = memo)

  }

}

p <- fun(n = 100, memo = NULL); p$value

[1] 0.01026239

dice1 <- function(){

  t <- 1

  repeat{

    x <- sample(1:6, 2, replace = T)

    if(sum(x) == 7) return(t)

    t <- t + 1

  }

}

N <- 10000;

sum <- 0;

for(i in 1:N){

  sum <- sum + dice1();

}

E <- sum/N; E

[1] 5.9889

dice2 <- function(n, k){

  con <- 0

  for(i in 1:n){

    x <- sample(1:6, 2, replace = T)

    if(sum(x) == 7) con <- con + 1

    else con <- 0

    if(con == k) return(T)

  }

  return(F)

}

N <- 20000;

sum <- 0;

for(i in 1:N){

  if(dice2(100, 5)) sum <- sum + 1;

}

p <- sum/N; p

[1] 0.01105

1. 完成野鸭射击问题（实验十二）的理论分析，用R编程实现仿真模拟分析。

理论分析：

设，则

即。于是

shoot <- function(n, m, p){

  # n: hunter, m: duck

  x <- sample(1:m, n, replace = T)

  flag <- rbinom(n, 1, p) == 1 # must be logical values

  return(m - length(unique(x[flag])))

}

N <- 10000;

sum <- 0;

for(i in 1:N){

  sum <- sum + shoot(10, 10, 0.7);

}

E <- sum/N; E

[1] 4.8311

1. 查找三门问题的相关资料，进行理论分析和模拟分析。

理论分析：

正面考虑，显然不换门赢得汽车的概率是（主持人选择的一定是羊，拉开门前后无影响），从而换门赢得汽车的概率是。反面考虑，特别要注意主持人的选择不是随机的，而是根据第一次选择而定的，即记：换门赢得汽车，：不换门赢得汽车，则，，于是

door <- function(){

  car <- sample(1:3, 1)

  choose <- sample(1:3, 1)

  if(choose == car){

    remain <- T

    change <- F

  }

  else{

    remain <- F

    change <- T

  }

  c(remain, change)

}

N <- 1000;

sum1 <- 0;

sum2 <- 0;

for(i in 1:N){

  rt <- door();

  if(rt[1]) sum1 <- sum1 + 1;

  if(rt[2]) sum2 <- sum2 + 1;

}

p1 <- sum1/N; p1

p2 <- sum2/N; p2

[1] 0.332

[1] 0.668

1. 自主找一道概率统计问题，进行理论分析和模拟分析。

使用逆变换法模拟poisson分布：

则。先产生一个随机数，根据其在poisson分布函数中对应的位置来确定其属于哪一个取值区间，即依次累计的值，直至找到满足

getpoisson <- function(n, lambda){

  rt <- numeric(n)

  for(i in 1:n){

    p <- exp(-lambda)

    x <- runif(1)

    k <- 0

    repeat{

      if(x < p){

        rt[i] <- k

        break

      }

      else{

        k <- k + 1

        p <- p + lambda^k/factorial(k)\*exp(-lambda)

      }

    }

  }

  rt

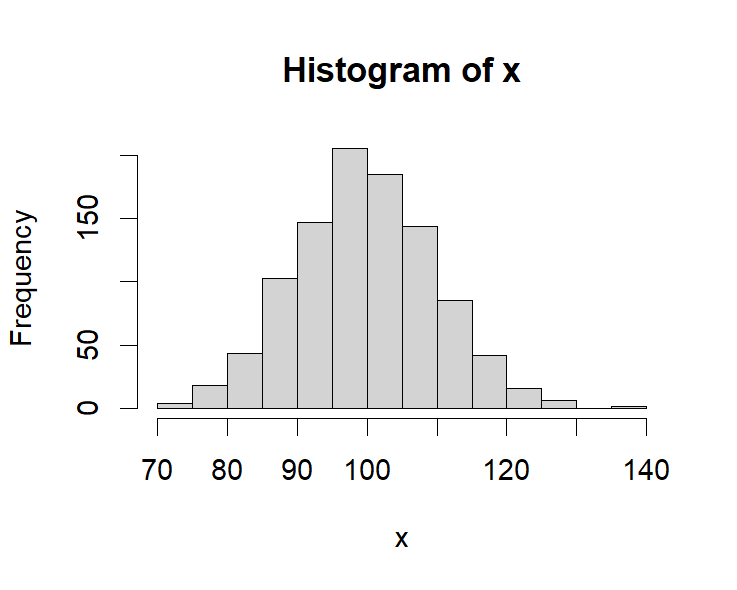
}

lambda <- 100;

n <- 1000;

x <- getpoisson(n,100)

hist(x)



1. 分别用投点法和平均值法估计

(1) (2) 

stochastic <- function(f, a, b, m, M){

  # \int\_a^b f(x)dx, m and M is a low/up bound of integrate field (not f(x) !)

  N = 100000

  x <- runif(N, min = a, max = b)

  y <- runif(N, min = m, max = M)

  (sum(f(x) >= 0 & y <= f(x) & y >= 0) - sum(f(x) < 0 & y >= f(x) & y <= 0))/N\*(b-a)\*(M-m)

}

meanmethod <- function(f, a, b){

  N = 1000

  x <- runif(N, min = a, max = b)

  sum(f(x)\*(b-a))/N

}

> f1 <- function(x){

+ exp(-x)

+ }

> integrate(f1, 2, 4)

0.1170196 with absolute error < 1.3e-15

> stochastic(f1, 2, 4, 0, 1)

[1] 0.11752

> meanmethod(f1, 2, 4)

[1] 0.1167726

>

> f2 <- function(x){

+ sin(x)

+ }

> integrate(f2, 0, pi/3)

0.5 with absolute error < 5.6e-15

> stochastic(f2, 0, pi/3, 0, 1)

[1] 0.4982671

> meanmethod(f2, 0, pi/3)

[1] 0.5050794

7. 用Monte Carlo积分法估计

(1)  (2) 

(3)  (4) 

> f3 <- function(x) exp(exp(x))

> integrate(f3, 0, 1)

6.316564 with absolute error < 7e-14

> stochastic(f3, 0, 1, 0, exp(exp(1)))

[1] 6.324934

> meanmethod(f3, 0, 1)

[1] 6.130032

>

> f4 <- function(x) x\*(1+x^2)^(-2)\*(-1/(x+1)^2)

> integrate(f4, 1, 0)

0.1097061 with absolute error < 3e-15

> -stochastic(f4, 0, 1, -1, 0)

[1] 0.10915

> -meanmethod(f4, 0, 1)

[1] 0.1109257

>

> f5 <- function(x) x\*sin(x)

> integrate(f5, 0, 1)

0.3011687 with absolute error < 3.3e-15

> stochastic(f5, 0, 1, 0, 1)

[1] 0.29967

> meanmethod(f5, 0, 1)

[1] 0.2954073

>

> f6 <- function(x) sin(exp(x))

> integrate(f6, 0, 1)

0.8749572 with absolute error < 9.7e-15

> stochastic(f6, 0, 1, -1, 1)

[1] 0.8702

> meanmethod(f6, 0, 1)

[1] 0.870826

8. 用投点法估计二重积分



stochastic2 <- function(f, a, b, c, d, m, M){

  N = 100000

  x <- runif(N, min = a, max = b)

  y <- runif(N, min = c, max = d)

  z <- runif(N, min = m, max = M)

  (sum(f(x,y) >= 0 & z <= f(x,y) & z >= 0) - sum(f(x,y) < 0 & z >= f(x,y) & z <= 0))/N\*(b-a)\*(d-c)\*(M-m)

}

meanmethod2 <- function(f, a, b, c, d){

  N = 100000

  x <- runif(N, min = a, max = b)

  y <- runif(N, min = c, max = d)

  sum(f(x,y)\*(b-a)\*(d-c))/N

}

> f7 <- function(x,y) exp((x+y)^2)

> stochastic2(f7, 0, 1, 0, 1, 0, exp(4))

[1] 4.859781

> meanmethod2(f7, 0, 1, 0, 1)

[1] 4.930981