REPORT

수강과목	:	전산통계
담당교수	:	노윤환
학 과	:	통계학과
학 번	:	201611531
이 름	:	정호재
제출일자	:	2019.10.07

※ 모의실험을 위한 seed값은 20190926으로 사용할 것.

1. 확률변수 X의 분포함수는 다음과 같다.

$$F(x) = \frac{x-a}{b-a},\ a \le x \le b$$

분포함수의 역함수를 이용하여 난수를 생성하는 알고리즘을 구성하여 모의실험을 하고 코드와 결과를 작성하시오.

(풀이)

분포함수의 역함수는 $F^{-1}(x) = (b-a)x + a$ 이다.

선형합동법을 이용하여 난수생성 알고리즘을 만든다.

임의의 수 a=926, b=2019, m=20190926와 현재 시간으로 초깃값을 설정해준다.

그 후 n을 1,2,…으로 변화시키며 반복 생성을 한다.

$x_n = (b-a) \cdot \ x_{n-1} + a \pmod{m}$

0부터 (m-1)까지의 정수를 m으로 나누어 0부터 1사이의 난수로 변환시킨다.

$u_n = \frac{x_n}{m}$

생성된 수열 중 필요한 개수만큼 생성한다.(ex. 10가지)

```
> set.seed(20190926)
> unif <- function(n) {
+ f <- vector(length = n)
+ m <- 20190926
+ a<-926
+ b<-2019
+ x <- as.numeric(Sys.time())
+ for (i in 1:n) {
+ x <- ((b-a) * x + a) %% m
+ f[i] <- x / m
+ }
+ return(f)
+ }
> unif(10)
 [1] 0.07800848 0.26331268 0.80080148 0.27606032 0.73397064 0.22995622
 [7] 0.34218896 0.01257410 0.74353644 0.68537199
```

2. 다음의 정적분 값을 Hit or Miss 방법, 표본평균법, 주표본기법을 사용하여 추정하고자 한다. 정적분 값을 계산하는 알고리즘을 구성하여 모의실험을 하고 코드와 결과를 작성하시오. 또한 세 방법의 결과를 비교 설명하시오.

(a) $\int_0^2 x^2 dx$

총 10000개의 난수 쌍을 생성하여 100번의 반복을 통하여 추정을 하기위해 함수생성 및 값이 입력될 저장 공간을 만들어준다.

```
> set.seed(20190926)
> g=function(x) x^2
> n=10000
> out1=numeric(100)
> out2=numeric(100)
> out3=numeric(100)
```

g(x)의 구간은 (0,2)이고 피적분함수의 최댓값이 4이므로 a=0 b=2 c=4으로 둔다.

Hit or Miss 방법

```
> set.seed(20190926)
> system.time(
+ for(i in 1:100){
+ const=4
+ x=runif(n, 0, 2)
+ y=runif(n, 0, const)
+ out1[i]=8*mean(y<=g(x))
+ }
+ )
 사용자  시스템 elapsed
   0.06    0.00    0.06
```

표본평균법

```
> set.seed(20190926)
> system.time(
+ for(i in 1:100){
+ x=runif(n,0,2)
+ out2[i]=2*mean(g(x))
+ }
+ )
 사용자  시스템 elapsed 
   0.04    0.00    0.05 
```

주표본기법

피적분함수와 형태가 비슷한 확률밀도함수로 $f(x)=\dfrac{3}{8}x^2$을 선택한다.

```
> x=runif(n)
> f<-function(x)3/8*x^2
> set.seed(20190926)
> system.time(
+ for(i in 1:100){
+ x=rep(NA,n)
+ C <- 3/8
+ h <- 1
+ k <- 1
+ while(k<=n){
+ U <- runif(1)
+ Y <- runif(1)
+ if(U <= f(Y)*(C*h)){
+ x[k] <- Y;
+ k <- k+1
+ }
+ }
+ out3[i] <- mean(g(x)/f(x))
+ }
+ )
 사용자  시스템 elapsed 
 113.66    0.24  119.29 
```

계산되는 시간은 표본평균법 < Hit or miss < 주표본기법 순으로 시간이 걸렸다.

#결과

```
> apply(cbind(out1,out2,out3), 2, summary)
            out1     out2     out3
Min.     2.59040 2.590924 2.666667
1st Qu.  2.64960 2.650194 2.666667
Median   2.67400 2.669194 2.666667
Mean     2.67392 2.667274 2.666667
3rd Qu.  2.69760 2.686902 2.666667
Max.     2.77440 2.719034 2.666667
> integrate(g,0,2)
2.666667 with absolute error < 3e-14
```

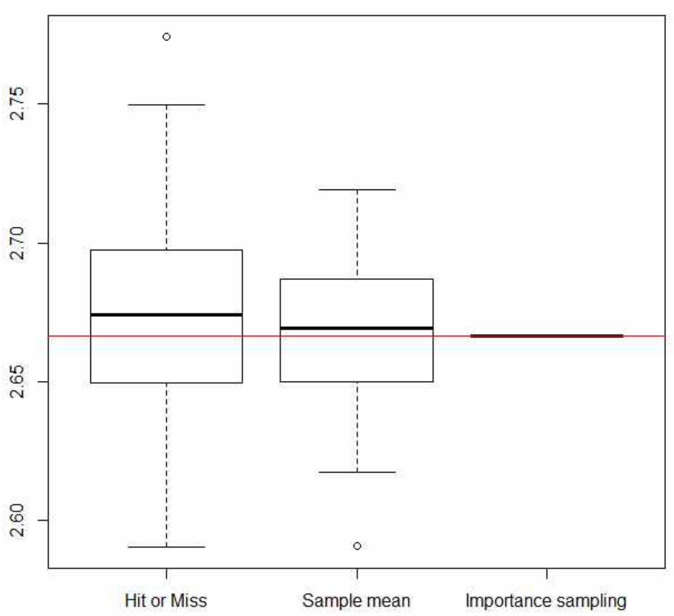
실제 적분 값과 계산 값들의 평균과 비교했을 때 실제 값과의 차이는
주표본기법 < 표본평균법 < Hit or miss 순으로 Hit or miss의 차이가 가장 컸다.

```
> apply(cbind(out1,out2,out3), 2, var)
        out1         out2         out3
0.0012338424 0.0005926059 0.0000000000
```

분산은 주표본기법 < 표본평균법 < Hit or miss 순으로 Hit or miss의 분산이 가장 컸다.

주표본기법에서 추정량 $g(x)/f(x)$는 상수 $\frac{8}{3}$가 되고 분산은 0이 되었다.

```
> boxplot(cbind("Hit or Miss"=out1,"Sample mean"=out2,"Importance sampling"=out3))
> abline(h=8/3,col="red")
```



(b) $\int_0^1 e^x dx$

총 10000개의 난수 쌍을 생성하여 100번의 반복을 통하여 추정을 하기위해 함수생성 및 값이 입력될 저장 공간을 만들어준다.

```
> set.seed(20190926)
> g=function(x) exp(x)
> n=10000
> out1=numeric(100)
> out2=numeric(100)
> out3=numeric(100)
```

g(x)의 구간은 (0,1)이고 피적분함수의 최댓값이 e이므로 a=0 b=1 c=e으로 둔다.

```
# Hit or Miss 방법
> set.seed(20190926)
> system.time(
+ for(i in 1:100){
+ const=exp(1)
+ x=runif(n, 0, 1)
+ y=runif(n, 0, const)
+ out1[i]=exp(1)*mean(y<=g(x))
+ }
+ )
 사용자  시스템 elapsed 
   0.14    0.00    0.14 
```

```
# 표본평균법
> set.seed(20190926)
> system.time(
+ for(i in 1:100){
+ x=runif(n,0,1)
+ out2[i]=mean(g(x))
+ }
+ )
 사용자  시스템 elapsed 
   0.10    0.00    0.11 
```

주표본기법
피적분함수와 형태가 비슷한 확률밀도함수를 구하기 위해 테일러급수를 사용하면

$e^x = \sum_{n=0}^{\infty} \frac{x^n}{n!} = 1 + x + \frac{x^2}{2!} + \cdots$ 이므로 $f(x) = \frac{3}{5}(1 + x + \frac{x^2}{2!})$로 잡아주면

여기서 $\frac{3}{5}$는 $\int_0^1 f(x)dx = 1$이 되도록 잡아준 상수이다.

```
> x=runif(n)
> f<-function(x)3/5*(1+x+x^2/2)
> set.seed(20190926)
> system.time(
+ for(i in 1:100){
+ x=rep(NA,n)
+ C <- 3/5
+ h <- 1
+ k <- 1
+ while(k<=n){
+ U <- runif(1)
+ Y <- runif(1)
+ if(U <= f(Y)*(C*h)){
+ x[k] <- Y;
+ k <- k+1
+ }
+ }
+ out3[i] <- mean(g(x)/f(x))
+ }
+ )
 사용자  시스템 elapsed 
    9.42    0.00    9.47 
```

계산되는 시간은 표본평균법 < Hit or miss < 주표본기법 순으로 시간이 걸렸다.

#결과

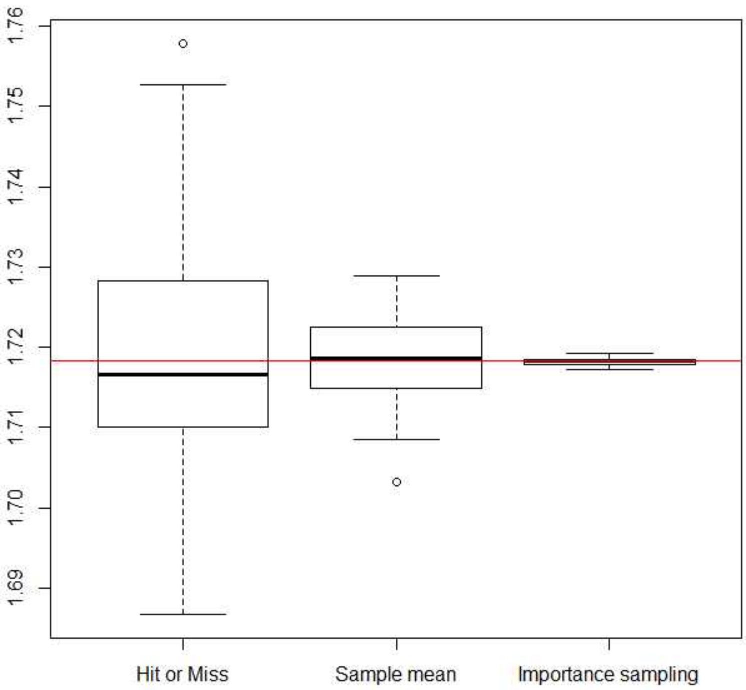
```
> apply(cbind(out1,out2,out3), 2, summary)
              out1     out2     out3
Min.      1.686694 1.703282 1.717206
1st Qu.   1.710071 1.714972 1.717884
Median    1.716595 1.718630 1.718286
Mean      1.718498 1.718463 1.718244
3rd Qu.   1.728148 1.722449 1.718501
Max.      1.757913 1.728842 1.719325
> integrate(g,0,1)
1.718282 with absolute error < 1.9e-14
```

실제 적분 값과 계산 값들의 평균과 비교했을 때 실제 값과의 차이는
주표본기법 < 표본평균법 < Hit or miss 순으로 Hit or miss의 차이가 가장 컸다.

```
> apply(cbind(out1,out2,out3), 2, var)
        out1         out2         out3
1.766149e-04 2.405042e-05 2.022335e-07
```

분산은 주표본기법 < 표본평균법 < Hit or miss 순으로 Hit or miss의 분산이 가장 컸다.

```
> boxplot(cbind("Hit or Miss"=out1,"Sample mean"=out2,"Importance sampling"=out3))
> abline(h=1.718282,col="red")
```



따라서 위의 결과들로 보았을 때 표본평균법 추정치의 분포가 Hit or Miss 추정치 분포보다 속도가 빠르고 및 더 작은 분산이 가지며, 좋은 추정량을 제공한다.

주표본기법은 표본평균법보다 더 작은 분산을 가지며, 좋은 추정량을 제공한다는 장점이 있지만, 속도가 느리고 중요함수 선택이 힘들다는 단점이 있다.