Homework 1 - Monte Carlo Methods

Mengyu Zhang / mz2777

Problem 1

The standard Laplace distribution has density $f(x) = 0.5e^{-|x|}, x \in (-\infty, \infty)$. Please provide an algorithm that uses the inverse transformation method to generate a random sample from this distribution. Use the $U(0,1)$ random number generator in **R**, *write a* **R**-function to implement the algorithm. Use visualization tools to validate your algorithm (i.e., illustrate whether the random numbers generated from your function truely follows the standard Laplace distribution.)

Answer:

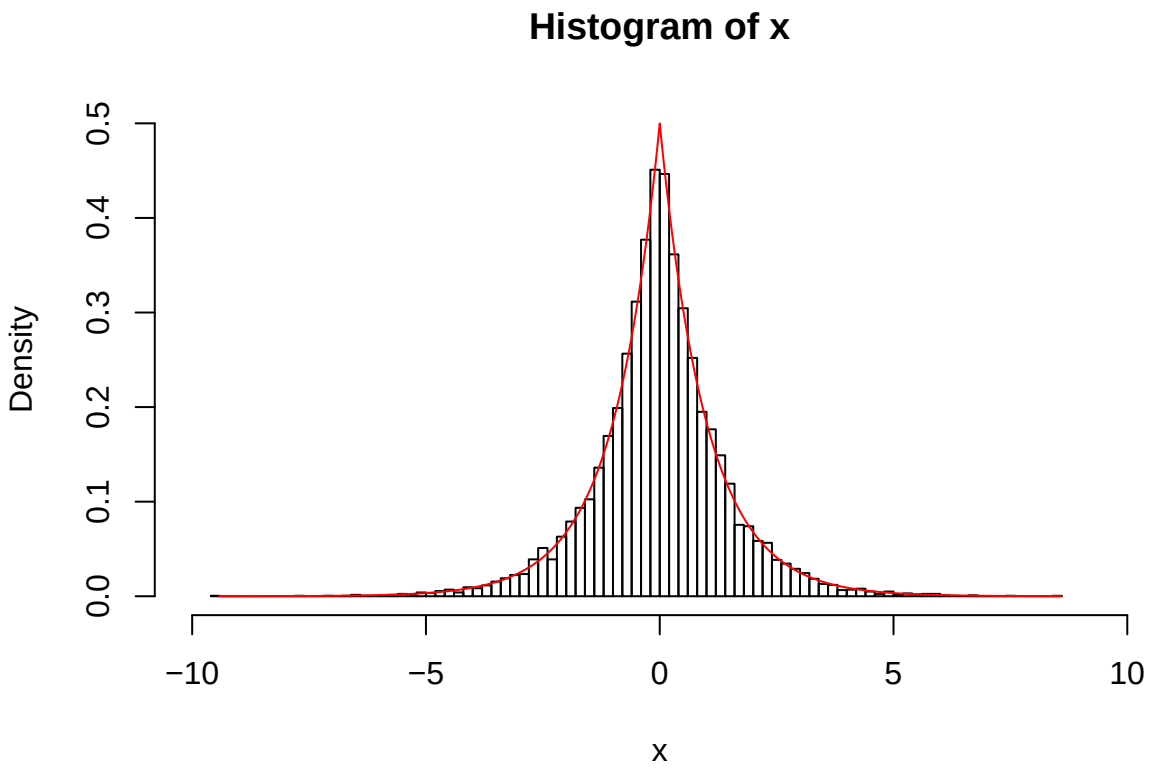
First we get the cdf of this distribution

$$F(x) = \begin{cases} \frac{1}{2}e^{x} & x \in (-\infty, 0] \\ 1 - \frac{1}{2}e^{-x} & x \in (0, \infty) \end{cases}$$

The inverse function would be

$$X = F^{-1}(U) = \begin{cases} \log 2U & U \in [0, \frac{1}{2}] \\ -\log(2 - 2U) & U \in (\frac{1}{2}, 1] \end{cases}$$

```
# Random number generating function
rannum1 <- function(n){
  U <- runif(n)
  X <- rep(0, n)
  for (i in 1:n){
    if(U[i] <= 0.5)
      X[i] <- log(2*U[i])
    else
      X[i] <- -log(2-2*U[i])
  }
  return(X)
}

# Generate the random number
set.seed(1000)
x = rannum1(10000)

#Visualization
xx = seq(min(x), max(x), length = 10000)
yy = 0.5*exp(-abs(xx))
hist(x, prob = T, breaks = 100, xlim = c(-10,10), ylim = c(0,0.5))
lines(xx,yy,col = "red")
```



Comments: Red line stands for the density function of X $f(x) = 0.5e^{-|x|}, x \in (-\infty, \infty)$, and we can see from the plot that the random numbers generated from the function above truely follows the standard Laplace distribution.

Problem 2

Use the inverse transformation method to derive an algorithm for generating a Pareto random number with $U \sim U(0, 1)$, where the Pareto random number has a probability density function

$$f(x; \alpha, \gamma) = \frac{\gamma \alpha^{\gamma}}{x^{\gamma+1}} I\{x \geq \alpha\}$$

with two parameters $\alpha > 0$ and $\gamma > 0$. Use visualization tools to validate your algorithm (i.e., illustrate whether the random numbers generated from your function truely follows the target distribution.)

Answer:

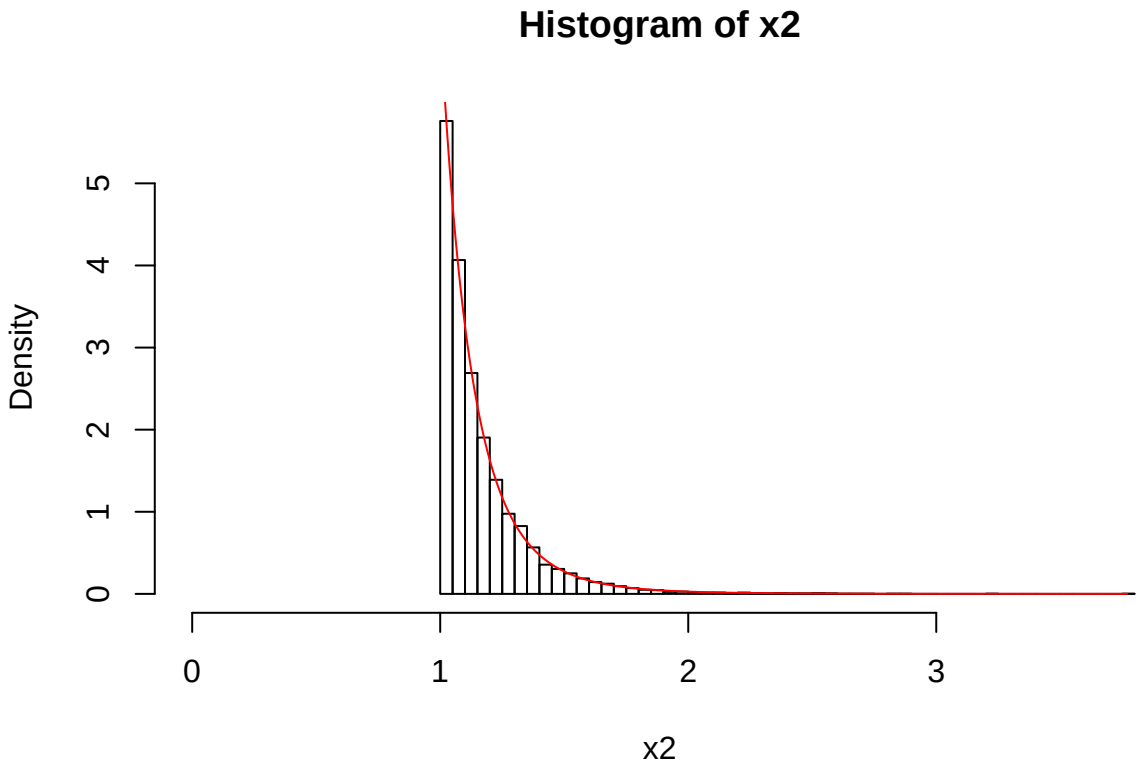
First we get the cdf of this distribution

$$F(x) = 1 - \left(\frac{\alpha}{x}\right)^{\gamma}$$

The inverse function would be

$$X = F^{-1}(U) = \frac{\alpha}{(1-U)^{1/\gamma}}$$

```r
# Random number generating function
rannum2 <- function(n, alpha, gamma){
  U <- runif(n)
  X <- rep(0, n)
  for (i in 1:n){
    X[i] = alpha/(1-U[i])^(1/gamma)
  }
  return(X)
}

#Set the parameter alpha and gamma
alpha = 1
gamma = 7

# Generate the random number
set.seed(1000)
x2 = rannum2(10000, alpha, gamma)

#Visualization
xx2 = seq(min(x2), max(x2), length = 10000)
yy2 = gamma*alpha^gamma/xx2^(gamma+1)
hist(x2, prob = T, breaks = 80, xlim = c(0, max(x2)))
lines(xx2,yy2,col = "red")
```



Comments: Red line stands for the density function of X

$$f(x;\alpha,\gamma) = \frac{7 \times 1^7}{x^{7+1}} I\{x \geq 1\}$$

and we can see from the plot that the random numbers generated from the function above truely follows the Pareto distribution.

Problem 3

Construct an algorithm for using the acceptance/rejection method to generate 100 pseudorandom variable from the pdf

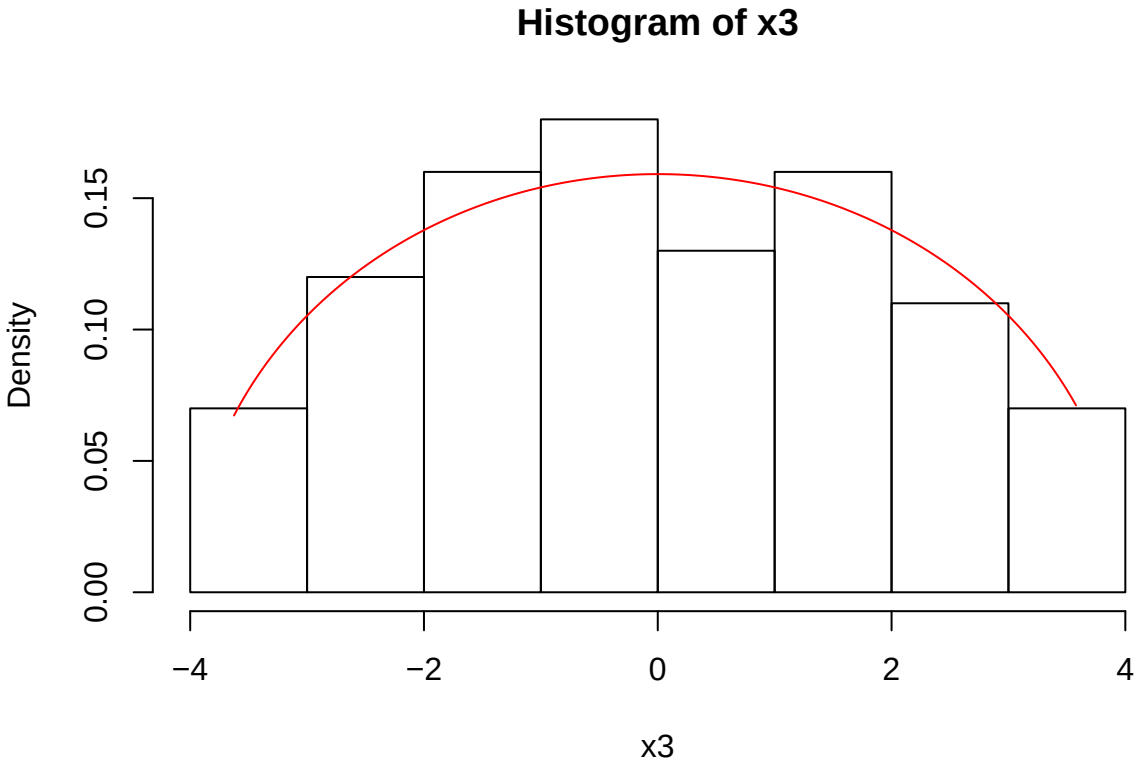
$$f(x) = \frac{2}{\pi\beta^2}\sqrt{\beta^2 - x^2},\ -\beta \le x \le \beta.$$

The simplest choice for $g(x)$ is the $U(-\beta, \beta)$ distribution but other choices are possible as well. Use visualization tools to validate your algorithm (i.e., illustrate whether the random numbers generated from your function truely follows the target distribution.)

Answer:

Using uniform distribution

```
# Functions

accrej <- function(fdens, gdens, M, x)
  return(x[runif(length(x)) <= fdens(x) / (M * gdens(x))])

unifbetadens = function(x)
  return(1/(2*beta) * (x >= -beta & x <= beta))

targetdens = function(x)
  return(2/(pi*beta^2)*sqrt(beta^2-x^2))

# Set the beta
beta = 4

# Generate 100 random number
set.seed(141)
x = runif(1000, -beta, beta)
x3 = accrej(targetdens, unifbetadens, 4/pi, x)
x3 = x3[1:100]

#Visualization
xx3 = seq(min(x3), max(x3), length = 1000)
yy3 = 2/(pi*beta^2)*sqrt(beta^2-xx3^2)
hist(x3, prob = T)
lines(xx3,yy3,col = "red")
```



Comments: Red line stands for the density function of X, and we can see from the plot that the random numbers generated from the function above truely follows the targeted distrbution.

Problem 4

Develop two Monte Carlo methods for the estimation of $\theta = \int_0^1 e^{x^2} dx$ and implement in **R** .

Answer:

1. First, uniform distribution was chosen to estimate the integral.

```
# Using uniform distribution
set.seed(12321)
N = 10000
u41 = runif(N)
y41 = sum(exp(u41^2))/N
y41
```

```
## [1] 1.462283
```

2. Second, $p(x) = \frac{e^x}{e-1}$ was used to estimate the integral.

The cumulative density function is $F(x) = \frac{e^x - 1}{e-1}$*, and its inverse funciton is* $u = \log(ue - u + 1)$*. Then we generate the random number following importance distribution and use it to estimate the integral.*

```r
set.seed(1231)
N = 10000
u42 = runif(N)
v = log(u42*exp(1)-u42+1)
y42 = sum(exp(v^2)*(exp(1)-1)/exp(v))/N
y42
```

```
## [1] 1.462242
```

Problem 5

Show that in estimating $\theta = E\sqrt{1-U^2}$ it is better to use $-U^2$ rather than $-U$ as the control variate, where $U \sim U(0,1)$. To do this, use simulation to approximate the necessary covariances. In addition, implement your algorithms in **R**.

Answer:

First

By using the following formula, we can get variances

$$m_1(x) = U^2, m_2(x) = U$$
$$c_2 = \frac{\text{Cov}(\sqrt{1-U^2}, U)}{\text{Var}(U)}$$
$$c_1 = \frac{\text{Cov}\left(\sqrt{1-U^2}, U^2\right)}{\text{Var}\left(U^2\right)}$$

```r
#Your R codes/functions

u = runif(10000)
c_u2 = (cov(sqrt(1 - u^2), u^2) / var(u^2)) * (-1)
c_u = (cov(sqrt(1 - u^2), u) / var(u)) * (-1)
func_u2 = (sqrt(1 - u^2) + c_u2*(u^2 - 1/3)) # m1(x) = u^2
func_u = sqrt(1 - u^2) + c_u * (u - 0.5) # # m2(x) = u
var(func_u2)
```

```
## [1] 0.001761039
```

```r
var(func_u)
```

```
## [1] 0.007846296
```

The variances are 0.0017 and 0.0077 for U^2 and U respectively, which means that in estimating $\theta = E\sqrt{1-U^2}$ it is better to use $-U^2$ rather than $-U$ as the control variate, where $U \sim U(0,1)$.

In General

Using$-U^2$

$\theta = \int_0^1 \sqrt{1-U^2}dU$. $g(U) = \sqrt{1-U^2}$ $m_1(U) = -U^2$.

```
gfun<-function(x) sqrt(1-x^2)
mfun<-function(x) -x^2
set.seed(123)
uran<-runif(10000)
ga<-gfun(uran)
ma<-mfun(uran)

theta1<- mean(ga)
hha <- -1/3 + (ga-ma)
theta2 <- mean(hha)

# Variance when not using the control variate
var(ga)
```

```
## [1] 0.04848323
```

```
# Variance when using -U^2 as the control variate
var(hha)
```

```
## [1] 0.007761115
```

```
# Variance reduction
(var(ga)-var(hha))/var(ga)
```

```
## [1] 0.8399217
```

The final result of integral is 0.788531 (0.7849381 with control variate).

Using $-U$

$m_2(x) = -U$

```
gfun<-function(x) sqrt(1-x^2)
mfun<-function(x) -x
set.seed(123)
uran<-runif(10000)
ga<-gfun(uran)
ma<-mfun(uran)

theta3<- mean(ga)
hha <- -0.5 + (ga-ma)
theta4 <- mean(hha)

# Variance when not using the control variate
var(ga)
```

```
## [1] 0.04848323
```

```r
# Variance when using -U as the control variate
var(hha)
```

```
## [1] 0.01441725
```

```r
# Variance reduction
(var(ga)-var(hha))/var(ga)
```

```
## [1] 0.7026343
```

The final result of integral is 0.788531 (0.7860804 with control variate).

Comment: The varience are 0.007 and 0.014 respectively when using $m_1(U) = -U^2$ and $m_2(x) = -U$, and the variace reduction when using $m_1(U) = -U^2$ is larger than that when using $m_2(x) = -U$. It is obvious that the varience for control variable $-U^2$ is smaller than that for $-U$.

Problem 6

Obtain a Monte Carlo estimate of

$$\int_1^\infty \frac{x^2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx$$

by importance sampling and evaluate its variance. Write a **R** *function to implement your procedure.*

Answer:

When x tends to infinity, actually when x take the number that larger than 5, $\frac{x^2}{\sqrt{2\pi}} e^{-\frac{x^2}{2}}$ tends to 0, so I decided to using uniform distribution $U(0, M)$, where M is very large. In the following code, we set $M = 15$.

```r
gfun<-function(x) x^2/sqrt(2*pi)*exp(-x^2/2)
mfun<-function(x) x*exp(-x)
set.seed(123)
uran<-runif(10000, 1, 15)
ga<-gfun(uran)
ma<-mfun(uran)


theta7<- mean(ga*14)
hha <- 2/exp(1) + (ga-ma)*14
theta8 <- mean(hha)

# # Variance without and with using the control variate
c(var(ga*14), var(hha))
```

```
## [1] 1.109362 0.237631
```

```r
# Variance reduction
(var(ga*14)-var(hha))/var(ga*14)
```

```
## [1] 0.785795
```

The final result of integral is 0.4001074 (0.4006881 with control variate).