Homework 1 - Monte Carlo Methods

Renjie Wei UNI: rw2844

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 truly follows the standard Laplace distribution.)

Answer:

Since $f(x) = 0.5e^{-|x|}, x \in (-\infty, \infty)$.

Therefore,

$$F(x) = \int_{-\infty}^{x} 0.5e^{-|s|}ds = \int_{-\infty}^{x} 0.5e^{s}ds(\text{if}x < 0)\int_{-\infty}^{0} 0.5e^{s}ds + \int_{0}^{x} 0.5e^{-s}ds(\text{if}x \geq 0)$$

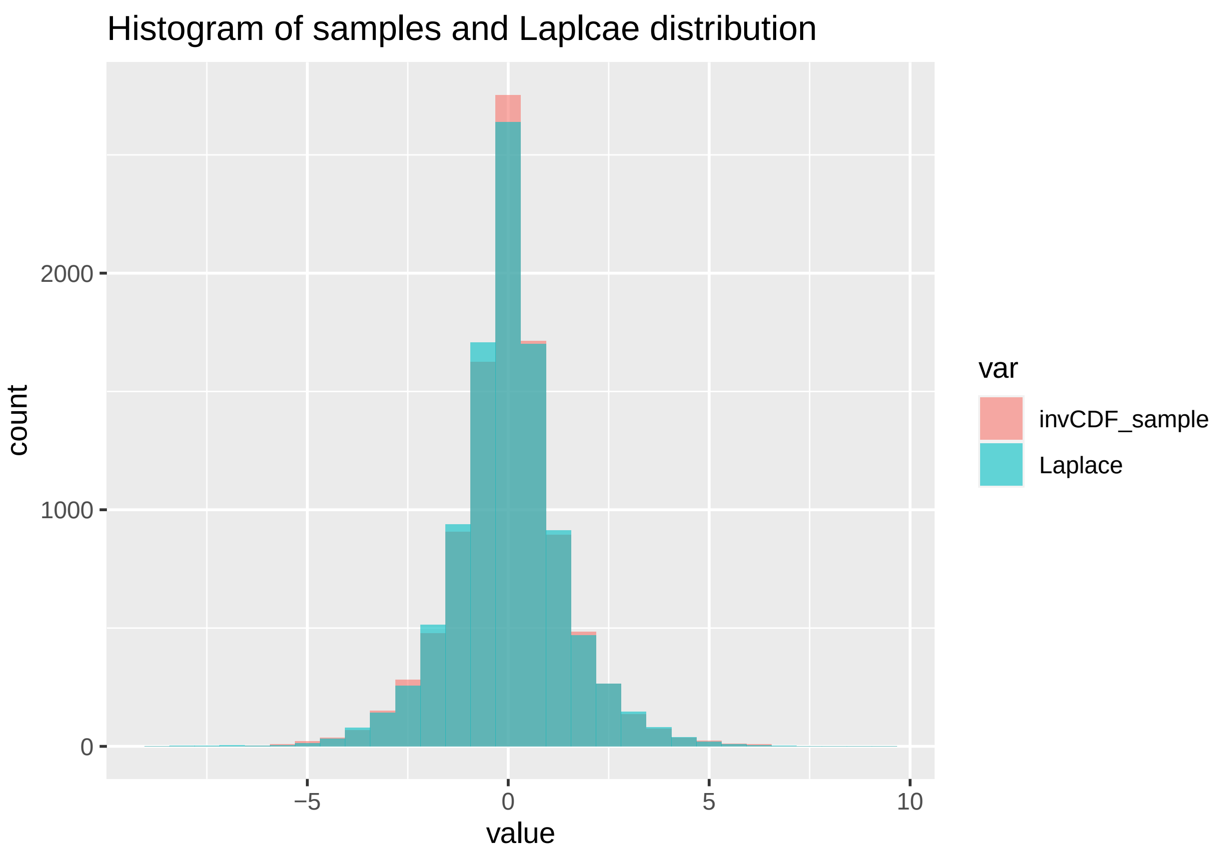
Hence,

$$F(x) = 0.5e^{x} \cdot \mathbb{I}(X < 0) + (1 - 0.5e^{-x}) \cdot \mathbb{I}(X \geq 0) = 0.5[1 + (-1)^{\mathbb{I}(X<0)}(1 - e^{-|x|})]$$

To get the inverse function $F^{-1}(U)$, solve $u = F(x)$:

$u = 0.5e^{x}$ $(\text{when}x < 0 \to 0 \leq u < 0.5)x = \ln(2u) or u = 1 - 0.5e^{-x}$ $(\text{when}x \geq 0 \to 0.5 \leq u \leq 1)x = -\ln(2-2u)$

```r
#Your R codes/functions
#install.packages("ExtDist")
library(ExtDist)
library(ggplot2)
set.seed(2022)
n=1e4
U<-runif(n)
L = rLaplace(n)
# inverse CDF
X = (U<0.5)*log(2*U)-(U>=0.5)*log(2-2*U)
dfL<-data.frame(
  var = c(rep("Laplace",n),rep("invCDF_sample",n)),
  value = c(L,X)
)
# the histogram of inverse CDF samples and samples from Laplace distribution
ggplot(dfL, aes(x=value, fill=var)) +
  geom_histogram(alpha=0.6, position='identity') +
  ggtitle("Histogram of samples and Laplcae 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 truly follows the target distribution.)

Answer:

Since $f(x;\alpha,\gamma) = \frac{\gamma\alpha^{\gamma}}{x^{\gamma+1}} I\{x \geq \alpha\}, \alpha > 0,\ \gamma > 0.$

Therefore,

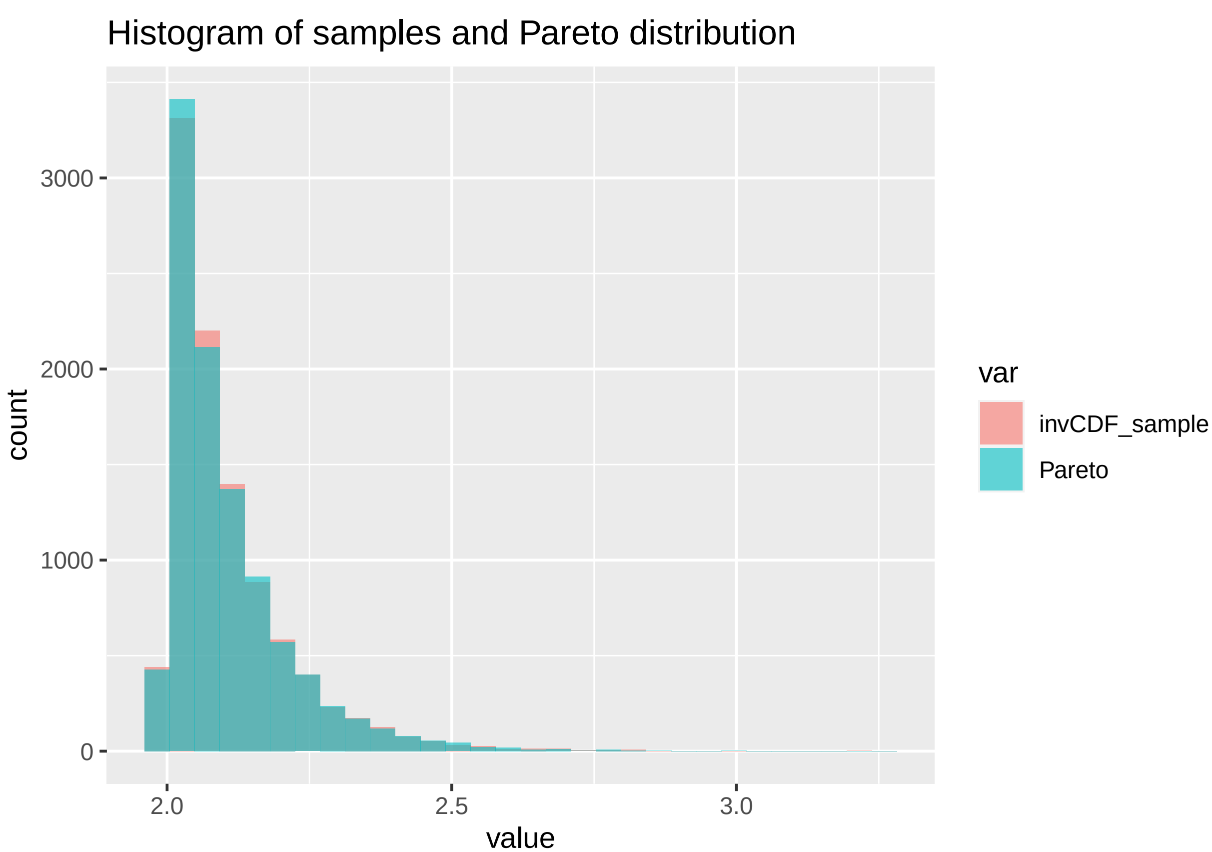
$$F(x) = \int_{\alpha}^{x} \frac{\gamma\alpha^{\gamma}}{s^{\gamma+1}} ds = \alpha^{\gamma} s^{-\gamma}\Big|_{\alpha}^{x} = 1 - \alpha^{\gamma} x^{-\gamma}$$

To get the inverse function $F^{-1}(U)$, solve $u = F(x)$:

$$u = 1 - \alpha^{\gamma} x^{-\gamma} (0 < u \leq 1) x = \frac{\alpha}{\sqrt[\gamma]{1-u}}$$

```r
#Your R codes/functions
#install.packages("EnvStats")
library("EnvStats")
set.seed(2022)
n=1e4
Up<-runif(n)
# generating N random samples directly from Pareto(2,20) (parameters are tuned in order to make the sta
alphaP <- 2
gammaP <- 20
P = rpareto(n,alphaP,gammaP)
# now using the inverse CDF method
Xp = 2/(1-Up)^(1/gammaP)

dfP<-data.frame(
  var = c(rep("Pareto",n),rep("invCDF_sample",n)),
  value = c(P,Xp)
)
# the histogram of inverse CDF samples and samples from Laplace distribution
ggplot(dfP, aes(x=value, fill=var)) +
  geom_histogram(alpha=0.6, position='identity') +
  ggtitle("Histogram of samples and 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 truly follows the target distribution.)

Answer:

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

We can show that

$$\max f(x) = \frac{2}{\pi}$$

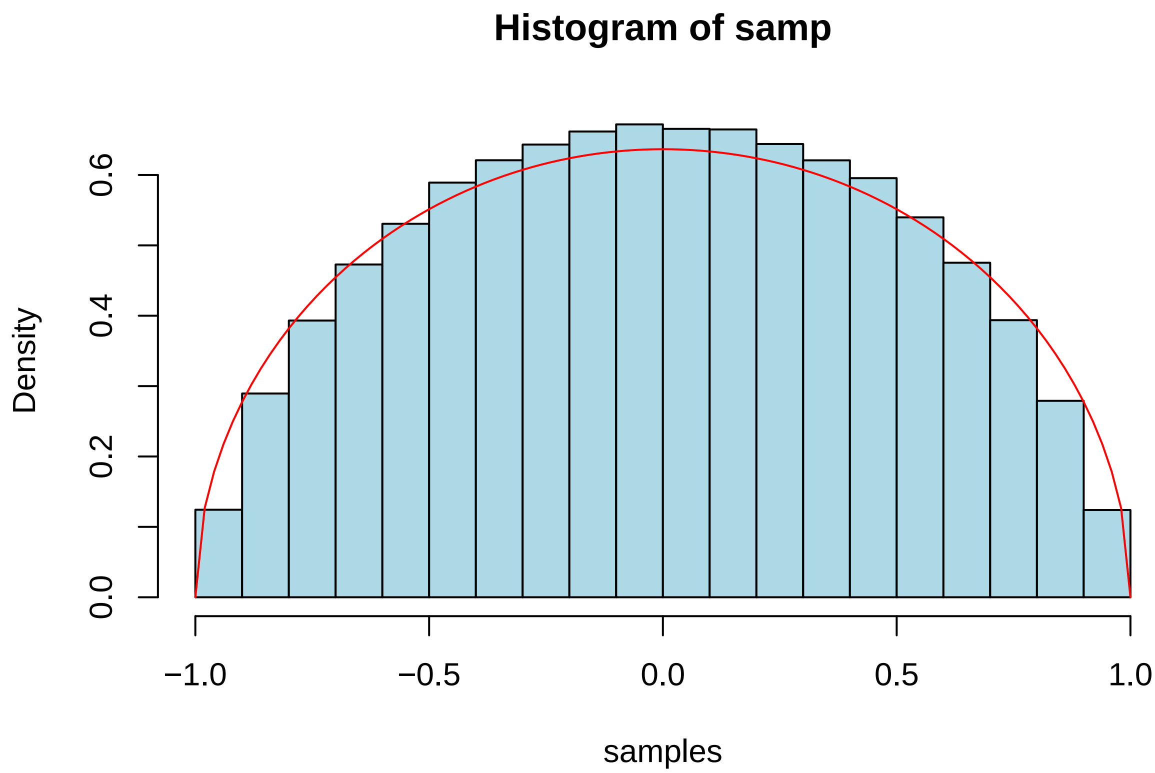
When we decided to use the $U(-\beta, \beta)$ as the convenient distribution $g(x)$, then we should choose M base on $M = \sup_{x\in[-\beta,\beta]} f(x)/g(x) = 4\beta/\pi$.

```
#Your R codes/functions
# First, create a function to calculate the pdf of x with input parameter beta and Uniform samples xs
targetdens <- function(beta,x){
  return((2/pi*(beta^2))*((beta^2-x^2)^1/2))
}

convdens <- function(beta,x){
  return(runif(length(x), min = -beta, max = beta))
}
# And the accept-rejection algorithm
myAccrej <- function(M, x, beta){
  U = runif(length(x))
  selected = x[U <= (targetdens(beta,x) / (M * convdens(beta,x)))]
  return(selected)
}

# Start sampling
set.seed(2022)
Beta = 1
M = (4*Beta)/pi
x = runif(1e6,-Beta,Beta)
samp = myAccrej(M=M,x,Beta)

# plot the sample histogram and the density curve, the plot shows that the shape of these two distribut
hist(samp, prob=T, col="lightblue",xlab = "samples" )
curve((2/pi)*sqrt(1-x^2), -1,1, add=T, col="red")
```


Histogram of samp
Density
0.0
0.2
0.4
0.6
−1.0
−0.5
0.0
0.5
1.0
samples