Computational Statistics Computer Lab 3 (Group 7)

Qinyuan Qi(qinqi464) Satya Sai Naga Jaya Koushik Pilla (satpi345)

2024-01-25

Question 1: Sampling algorithms for a triangle distribution (Solved by Qinqi Qi)

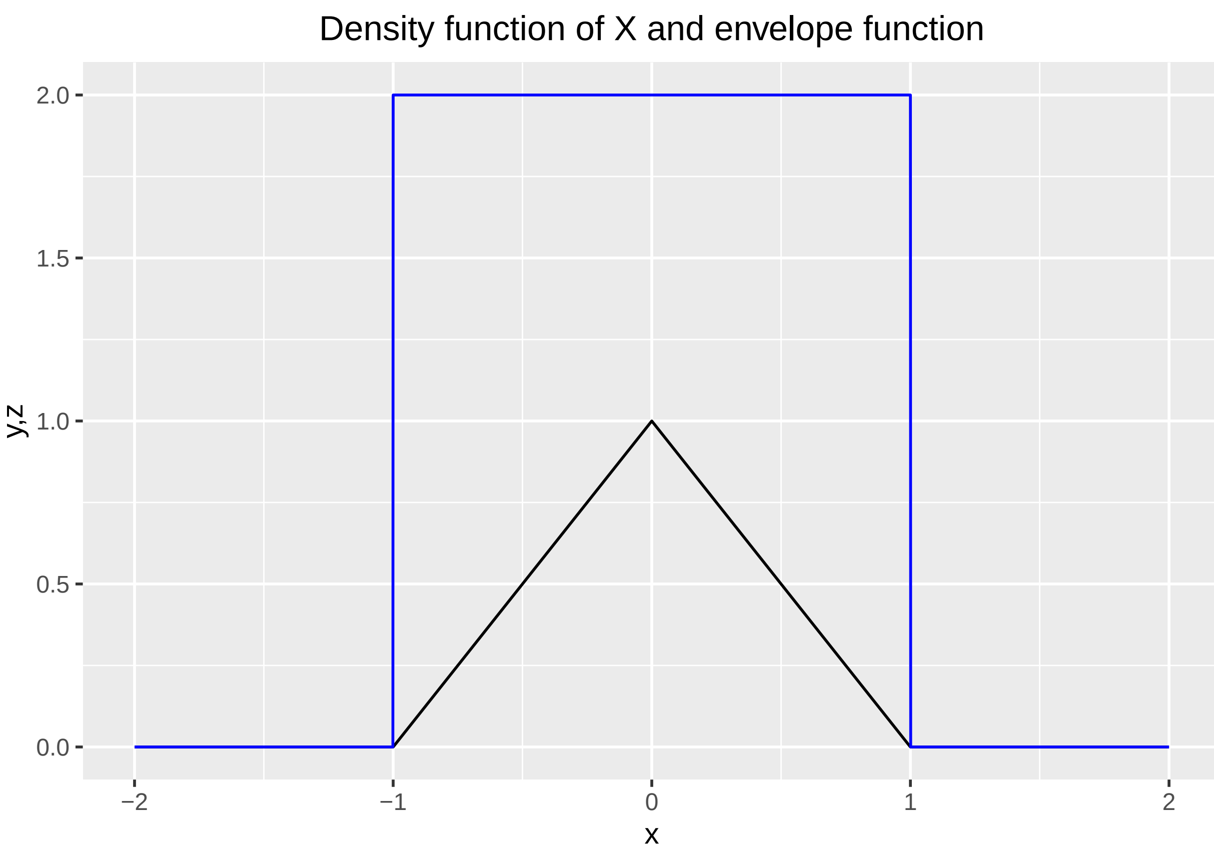
Answer:

(A)

We define an envelope function to cover the density function.

$$g(x) = \begin{cases} 1, & \text{if } -1 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

where $a = 0.5$, $e(x) = g/a = 2$

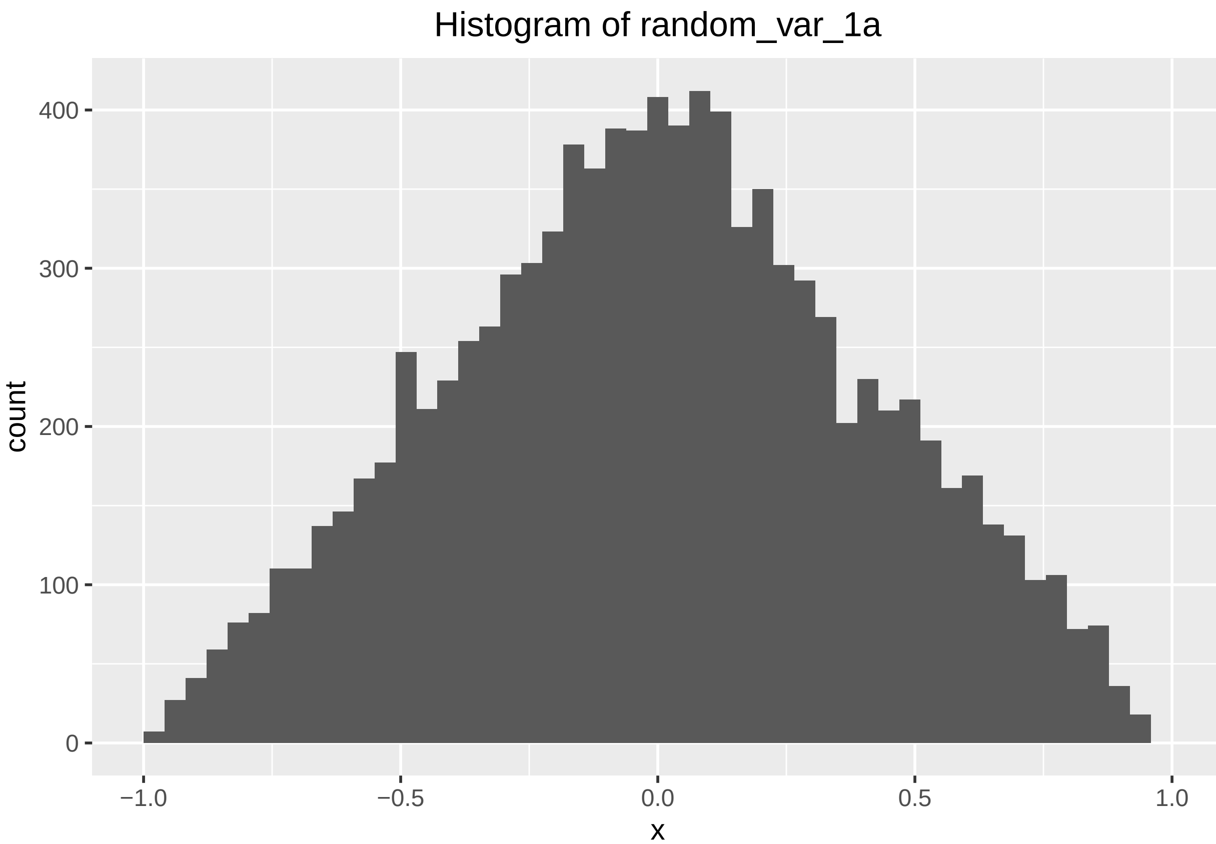


The following code is the implementation of rejection sampling algorithm.

our g function is $g(x) = 1$ in the range $[-1, 1]$, otherwise $g(x) = 0$ our e function is $e(x) = 1/a$ in the range $[-1, 1]$, otherwise $e(x) = 0$ and we will use $a = 0.5$ here.

```r
########################## [ 1 a2 ] ##############################################

generate_random_var_1a <- function(n, a) {
  samples <- numeric(n)
  # Generate a sample from the proposal distribution in [-1,1]
  count <- 1
  e <- 1 / a

  while (count <= n) {

    x <- runif(1, -1, 1)
    u <- runif(1)

    if (x > 0 && u <= (1 - x) / e) {
      samples[count] <- x
      count <- count + 1
    }

    if (x <= 0 && u <= (x + 1) / e) {
      samples[count] <- x
      count <- count + 1
    }
  }

  return(samples)
}

a <- 0.5
set.seed(12345)
random_var_1a <- generate_random_var_1a(10000, a)
ggplot2::ggplot(data.frame(x = random_var_1a), aes(x)) +
  ggplot2::geom_histogram(bins = 50) +
  ggplot2::ggtitle("Histogram of random_var_1a") +
  ggplot2::xlim(-1,1)+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```




```
var_of_random_var_1a <- var(random_var_1a)
cat("variance of 1st method is",var_of_random_var_1a)
```

```
## variance of 1st method is 0.1647185
```

(B)

According to lecture 3 slides 9-10, we let

$$f(x) = \begin{cases} 2-2x, & \text{if } 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

and have

$$F(x) = \int_{-\infty}^{x} f(t)dt = \begin{cases} 0, & \text{if } x < 0 \\ 2x - x^2, & \text{if } 0 \leq x \leq 1 \\ 1, & \text{if } x > 1 \end{cases}$$

then

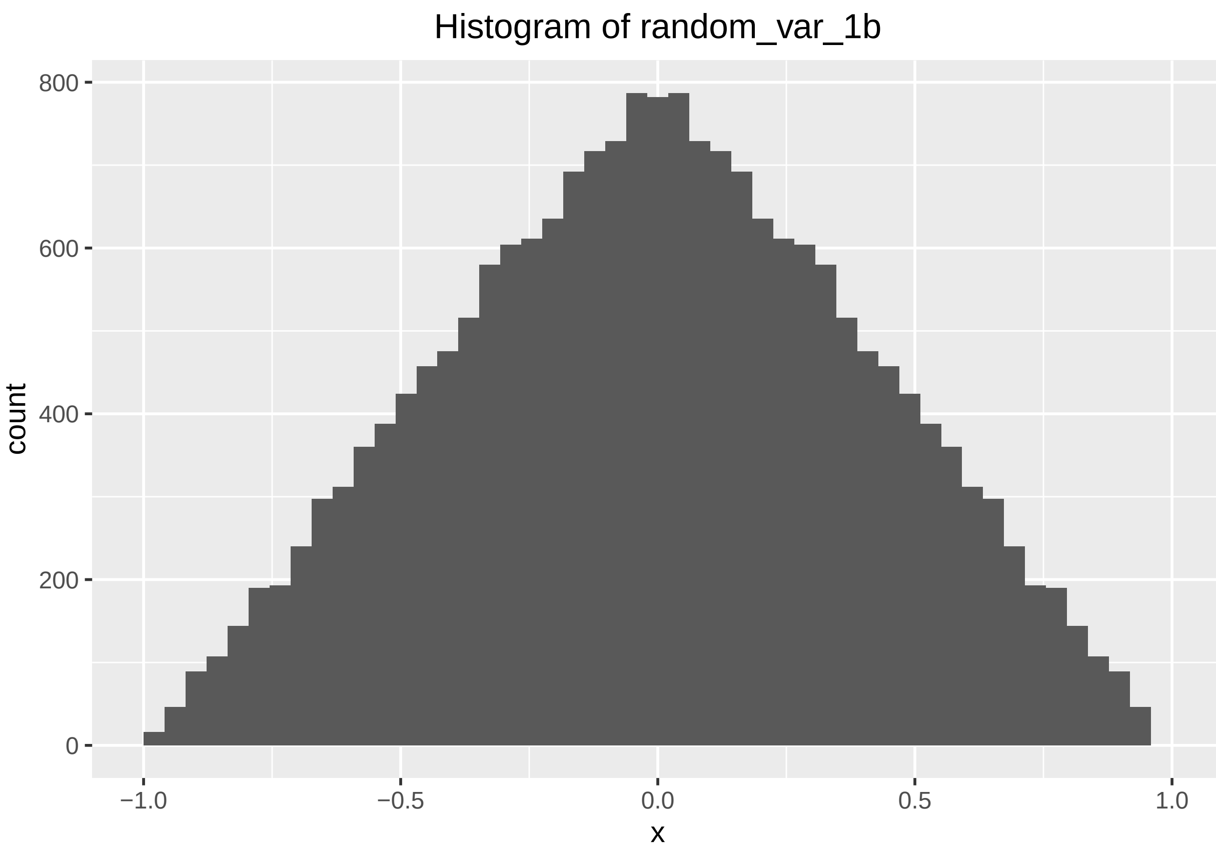
$$F^{-1}(y) = 1 - \sqrt{1-y}$$

For -Y, do the same and we have $x = -1 + \sqrt{1-y}$

```
######################### [ 1 b ] #############################################
generate_random_var_1b <- function(n){
  u <- runif(n)
  x1 <-  - 1 + sqrt(1 - u)
  x2 <-  1 - sqrt(1 - u)
  return(c(x1, x2))
}
```

```r
set.seed(12345)
random_var_1b <- generate_random_var_1b(10000)
ggplot2::ggplot(data.frame(x = random_var_1b), aes(x)) +
  ggplot2::geom_histogram(bins = 50) +
  ggplot2::ggtitle("Histogram of random_var_1b") +
  ggplot2::xlim(-1,1)+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```




```r
var_of_random_var_1b <- var(random_var_1b)
cat("variance of 2nd method is",var_of_random_var_1b)
```

```
## variance of 2nd method is 0.1658306
```

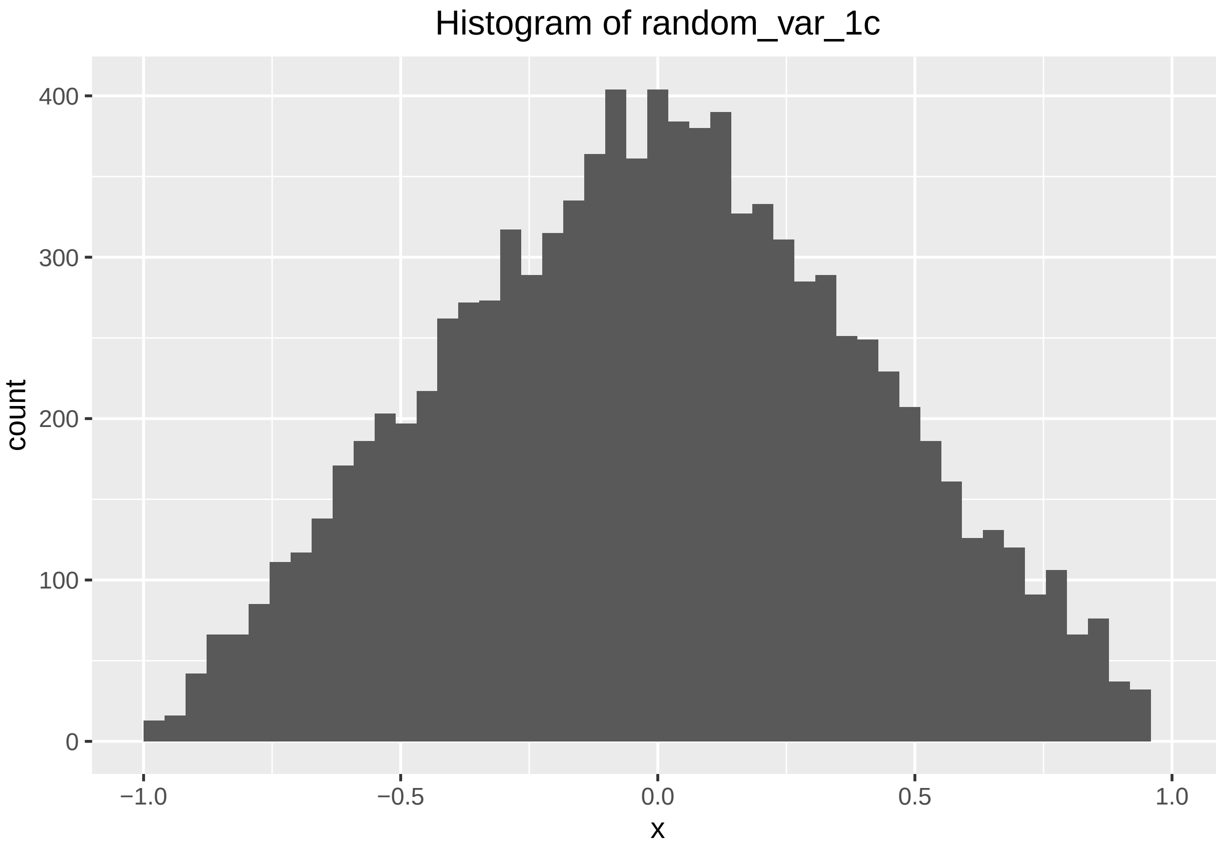
(C)

The code to generate a random variable following a triangle distribution as follows.

```r
########################## [ 1 c ] ##############################################
generate_random_var_1c <- function(n){
  u1_1c <- runif(n)
  u2_1c <- runif(n)
  return(u1_1c - u2_1c)
}

random_var_1c <- generate_random_var_1c(10000)
ggplot2::ggplot(data.frame(x = random_var_1c), aes(x)) +
```

```r
  ggplot2::geom_histogram(bins = 50) +
  ggplot2::ggtitle("Histogram of random_var_1c") +
  ggplot2::xlim(-1,1)+
  theme(plot.title = element_text(hjust = 0.5))
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```




```r
var_of_random_var_1c <- var(random_var_1c)
cat("variance of 3rd method is",var_of_random_var_1c)
```

```
## variance of 3rd method is 0.1656517
```

(D)

Since we already plotted the data, we will not plot them here.

The variance of the three methods is as follows

	variance
1a	0.1647185
1b	0.1658306
1c	0.1656517

According to the result, the 1a method, which is rejection sampling has the smallest variance.

1c is the simplest way to generate the random variable, but it can not adapt to some specific distribution.

For method 1b, since we need to calculate CDF and inverse function, in some cases, it's hard to do that.

Based on the statement above, we will use 1a to generate the random variable(also because it has the smallest variance).

Question 2: Laplace distribution (Solved by Satya Sai Naga Jaya Koushik Pilla)

Answer:

(a) We have

$$DE(\mu, \lambda) = \frac{\lambda}{2} e^{-\lambda|x-\mu|}$$

The related CDF function is expressed as follows.

$$F_{DE}(x) = \begin{cases} \frac{1}{2} e^{\lambda(x-\mu)} & x < \mu \\ 1 - \frac{1}{2} e^{-\lambda(x-\mu)} & x \geq \mu \end{cases}$$

Combine them using the sign function, we have

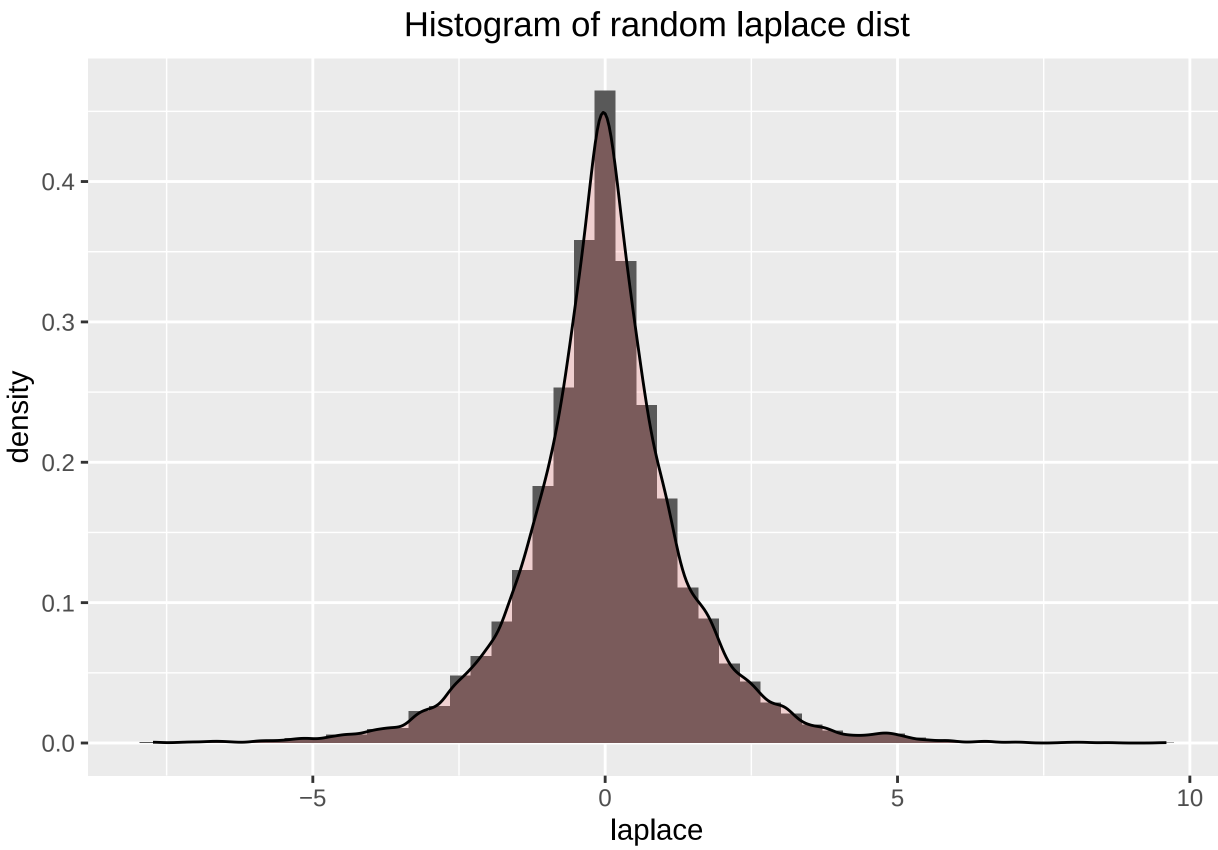
$$F_{DE}(x) = \frac{1}{2} + \frac{1}{2} sign(x-\mu)(1 - e^{-\lambda|x-\mu|})$$

Then we calculate the inverse function of CDF using the given $\mu = 1$ and $\lambda = 0$.

$$F_{DE}^{-1}(x) = \mu_{DE} - \frac{1}{\lambda} sign(\mu - 0.5) ln(1 - 2|\mu - 0.5|)$$

According to the formula above, we can generate random variables following a Laplace distribution.

```r
########################## [ 2 a ] ######################################

laplace_dist <- function(mu, lambda, p){
  return(mu - (1 / lambda) * sign(p - 0.5) * log(1 - 2 * abs(p - 0.5)))
}

n = 10000

x_rand <- runif(n,0,1)
data1 <- data.frame(x_unif = x_rand)

data1$laplace <- laplace_dist(0,1,x_rand)

ggplot(data1, aes(laplace)) +
  geom_histogram(aes(y=..density..), bins = 50) +
  geom_density(alpha=0.2,fill="#FF6666") +
  ggtitle("Histogram of random laplace dist") +
  theme(plot.title = element_text(hjust = 0.5))
```



According to the plot, the plot we generated follows the Laplace distribution.

(b) Since $\mu = 0$ and $\lambda = 1$, we have PDF of laplace distribution as follows.

$$f_l(x) = \frac{1}{2}e^{-|x|} = \begin{cases} \frac{1}{2}e^{-x} & x >= 0 \\ \frac{1}{2}e^{x} & x < 0 \end{cases}$$

Meanwhile, we have PDF of the normal distribution as follows.

$$f_n(x) = \frac{1}{\sqrt{2\pi}}e^{-\frac{x^2}{2}}$$

Since

$$\frac{f_n(x)}{f_l(x)} = \sqrt{\frac{2}{\pi}}e^{(-\frac{x^2}{2}+|x|)}$$

When $x = \pm 1$, this expression will get its maximum value.

$$a = max\frac{f_n(x)}{f_l(x)} = \sqrt{\frac{2}{\pi}}e^{\frac{1}{2}}$$

Using this a value, we can generate the random variables that are needed.

The histogram of the random variable is generated as below, the real rejection rate R is 0.2503748 and the expected rejection rate E(R) is 0.2398265 which is very close to each other.

The two histogram plots below are almost the same.

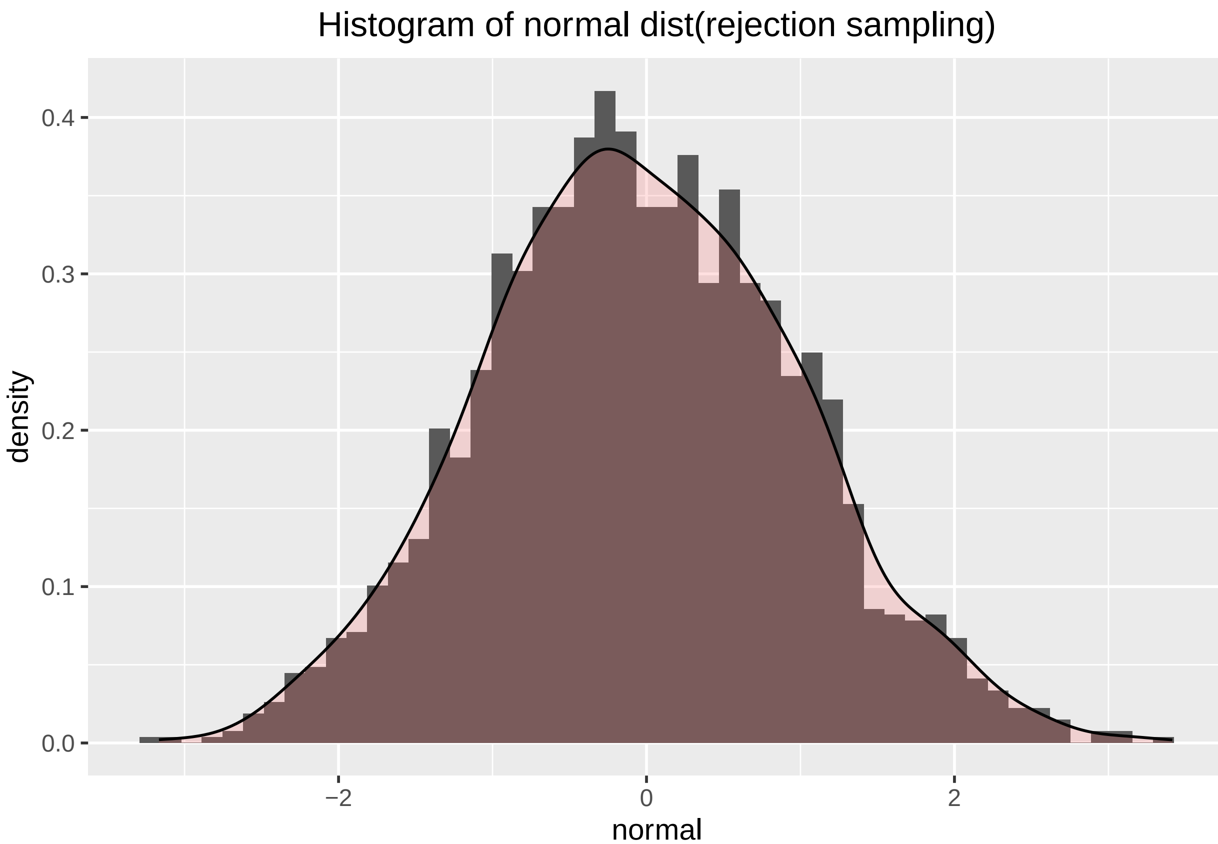
```r
########################## [ 2 b ] ##############################################
# calc a value using the formula above
a <- sqrt(2/pi)*exp(0.5)

# pdf of normal distribution
pdf_normal <- function(x){
  return(exp(-x^2 / 2) / sqrt(2 * pi))
}

# pdf of laplace distribution
pdf_laplace <- function(x){
  return(0.5 * exp(-abs(x)))
}

n2 <- 2000

# save generated samples in this vector
samples <- c()

for(i in 1:n){
  if (length(samples) >= n2) {
    break
  }

  Y = data1$laplace[i]
  u = runif(1)
  g = pdf_laplace(Y)
  f = pdf_normal(Y)

  if (u <=f/(a * g)) {
    samples <- c(samples, Y)
  }
}

cat("Rejection rate R: ", 1 - (n2 / i), "\n")
```

```
## Rejection rate R:  0.2503748
```

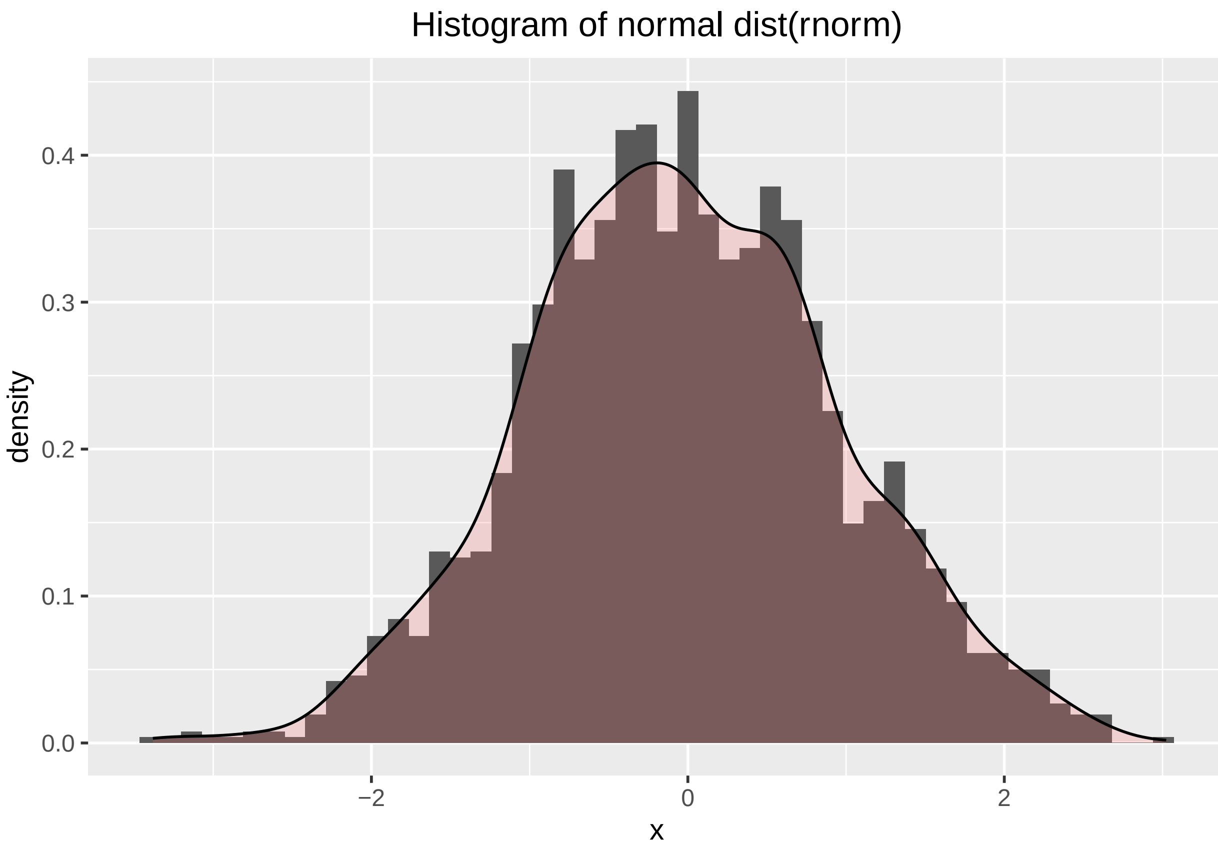
```r
cat("rejection rate E(R): ", 1 - (1 / a), "\n")
```

```
## rejection rate E(R):  0.2398265
```

```r
data2 <- data.frame(normal = samples)
ggplot(data2, aes(normal)) +
  geom_histogram(aes(y=..density..), bins = 50) +
  geom_density(alpha=0.2,fill="#FF6666") +
  ggtitle("Histogram of normal dist(rejection sampling)") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
# Generate 2000 random variables from the distribution using rnorm
random_var_2b2 <- rnorm(n2, 0, 1)

ggplot2::ggplot(data.frame(x = random_var_2b2), aes(x)) +
   geom_histogram(aes(y=..density..), bins = 50) +
   geom_density(alpha=0.2,fill="#FF6666") +
   ggplot2::ggtitle("Histogram of normal dist(rnorm)") +
   theme(plot.title = element_text(hjust = 0.5))
```


Histogram of normal dist(rnorm)
density
0.4
0.3
0.2
0.1
0.0
−2
0
2
x

Appendix: Code for this report

```
########################## Init code for question 1 ############################
rm(list = ls())
library(ggplot2)
########################## [ 1 a1 ] #############################################

x <- seq(-2, 2, by = 0.001)
y <- rep(0, length(x))
z <- rep(0, length(x))

x1_index <- which(x > 1 | x < -1)
x2_index <- which(x >= -1 & x <= 0)
x3_index <- which(x > 0 & x <= 1)

y[x1_index] <- 0
y[x2_index] <- x[x2_index] + 1
y[x3_index] <- 1 - x[x3_index]

z0_index <- which(x > 1 | x < -1)
z1_index <- which(x >= -1 & x <= 1)

z[z1_index] <- 2
z[-z1_index] <- 0

data <- data.frame(x, y, z)

graph <- ggplot2::ggplot(data) +
  ggplot2::geom_line(mapping=ggplot2::aes(x=x,y=y)) +
  ggplot2::geom_line(mapping=ggplot2::aes(x=x,y=z),color="blue") +
  ggplot2::ggtitle("Density function of X and envelope function") +
  ggplot2::xlab("x") +
  ggplot2::ylab("y,z")+
  theme(plot.title = element_text(hjust = 0.5))

graph
########################## [ 1 a2 ] #############################################

generate_random_var_1a <- function(n, a) {
  samples <- numeric(n)
  # Generate a sample from the proposal distribution in [-1,1]
  count <- 1
  e <- 1 / a

  while (count <= n) {

    x <- runif(1, -1, 1)
    u <- runif(1)

    if (x > 0 && u <= (1 - x) / e) {
      samples[count] <- x
      count <- count + 1
    }
```

```r
    if (x <= 0 && u <= (x + 1) / e) {
      samples[count] <- x
      count <- count + 1
    }
  }

  return(samples)
}

a <- 0.5
set.seed(12345)
random_var_1a <- generate_random_var_1a(10000, a)
ggplot2::ggplot(data.frame(x = random_var_1a), aes(x)) +
  ggplot2::geom_histogram(bins = 50) +
  ggplot2::ggtitle("Histogram of random_var_1a") +
  ggplot2::xlim(-1,1)+
  theme(plot.title = element_text(hjust = 0.5))
var_of_random_var_1a <- var(random_var_1a)
cat("variance of 1st method is",var_of_random_var_1a)
########################## [ 1 b ] #############################################
generate_random_var_1b <- function(n){
  u <- runif(n)
  x1 <-  - 1 + sqrt(1 - u)
  x2 <-  1 - sqrt(1 - u)
  return(c(x1, x2))
}

set.seed(12345)
random_var_1b <- generate_random_var_1b(10000)
ggplot2::ggplot(data.frame(x = random_var_1b), aes(x)) +
  ggplot2::geom_histogram(bins = 50) +
  ggplot2::ggtitle("Histogram of random_var_1b") +
  ggplot2::xlim(-1,1)+
  theme(plot.title = element_text(hjust = 0.5))


var_of_random_var_1b <- var(random_var_1b)
cat("variance of 2nd method is",var_of_random_var_1b)
########################## [ 1 c ] #############################################
generate_random_var_1c <- function(n){
  u1_1c <- runif(n)
  u2_1c <- runif(n)
  return(u1_1c - u2_1c)
}

random_var_1c <- generate_random_var_1c(10000)
ggplot2::ggplot(data.frame(x = random_var_1c), aes(x)) +
  ggplot2::geom_histogram(bins = 50) +
  ggplot2::ggtitle("Histogram of random_var_1c") +
  ggplot2::xlim(-1,1)+
  theme(plot.title = element_text(hjust = 0.5))

var_of_random_var_1c <- var(random_var_1c)
```

```r
cat("variance of 3rd method is",var_of_random_var_1c)
########################## Init code for question 2#############################
knitr::opts_chunk$set(echo = TRUE)
rm(list = ls())
library(ggplot2)
########################## [ 2 a ] ######################################

laplace_dist <- function(mu, lambda, p){
  return(mu - (1 / lambda) * sign(p - 0.5) * log(1 - 2 * abs(p - 0.5)))
}

n = 10000

x_rand <- runif(n,0,1)
data1 <- data.frame(x_unif = x_rand)

data1$laplace <- laplace_dist(0,1,x_rand)

ggplot(data1, aes(laplace)) +
  geom_histogram(aes(y=..density..), bins = 50) +
  geom_density(alpha=0.2,fill="#FF6666") +
  ggtitle("Histogram of random laplace dist") +
  theme(plot.title = element_text(hjust = 0.5))
########################## [ 2 b ] ##############################################
# calc a value using the formula above
a <- sqrt(2/pi)*exp(0.5)

# pdf of normal distribution
pdf_normal <- function(x){
  return(exp(-x^2 / 2) / sqrt(2 * pi))
}

# pdf of laplace distribution
pdf_laplace <- function(x){
  return(0.5 * exp(-abs(x)))
}

n2 <- 2000

# save generated samples in this vector
samples <- c()

for(i in 1:n){
  if (length(samples) >= n2) {
    break
  }

  Y = data1$laplace[i]
  u = runif(1)
  g = pdf_laplace(Y)
  f = pdf_normal(Y)

  if (u <=f/(a * g)) {
```

```r
    samples <- c(samples, Y)
  }
}

cat("Rejection rate R: ", 1 - (n2 / i), "\n")
cat("rejection rate E(R): ", 1 - (1 / a), "\n")

data2 <- data.frame(normal = samples)
ggplot(data2, aes(normal)) +
  geom_histogram(aes(y=..density..), bins = 50) +
  geom_density(alpha=0.2,fill="#FF6666") +
  ggtitle("Histogram of normal dist(rejection sampling)") +
  theme(plot.title = element_text(hjust = 0.5))


# Generate 2000 random variables from the distribution using rnorm
random_var_2b2 <- rnorm(n2, 0, 1)

ggplot2::ggplot(data.frame(x = random_var_2b2), aes(x)) +
   geom_histogram(aes(y=..density..), bins = 50) +
   geom_density(alpha=0.2,fill="#FF6666") +
   ggplot2::ggtitle("Histogram of normal dist(rnorm)") +
   theme(plot.title = element_text(hjust = 0.5))
```