Group 12 lab 3 report

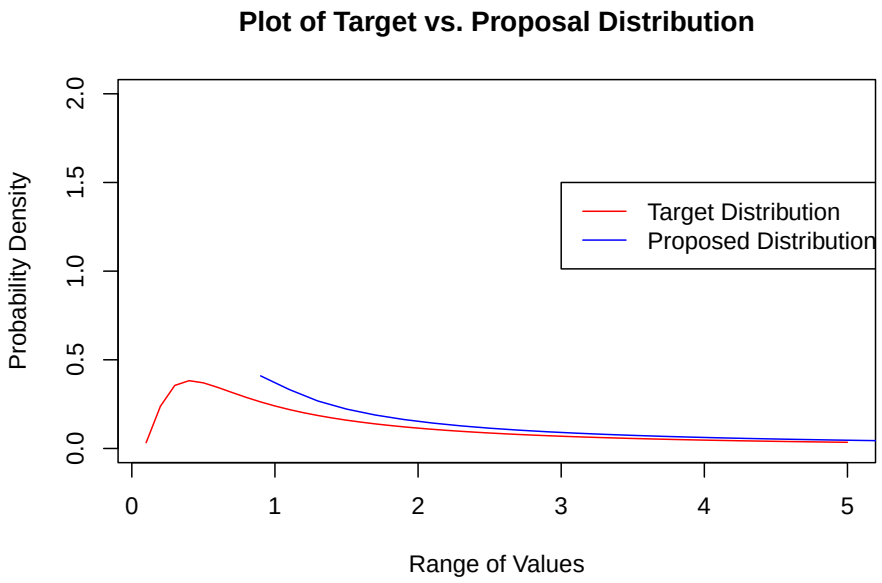
varsi146, aswma317

2022-12-01

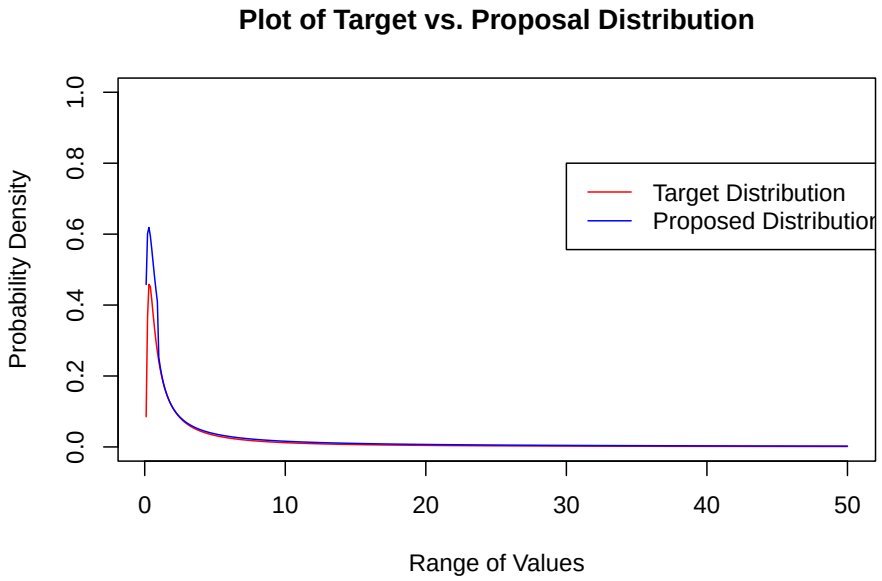
Question 1

Sub Question 1

Below we can see the plot of $f(x)$(in red) and $f_p(x)$(in blue):



As seen above, power law distribution cannot be used just by itself as its support lies on $[T_{min}, \infty]$ and for values of x less than T_{min} we cannot use only the power law distribution and hence we employ the log-normal distribution to envelop the target distribution from values ranging from $[0, T_{min}]$(as seen below)



The sensible values of parameters for the power law distribution seem to be $\alpha = 1.2$ & $T_{min} = 0.9$

Derivation of Majorizing Density

In our case, our majorizing density $g(x)$ consists of:

A log-normal distribution for values of x in between $[0, T_{min}]$ which has a density function as shown below:

$$f_1(x) = \frac{1}{x\sigma\sqrt{2\pi}} exp(-\frac{(ln(x) - \mu)^2}{2\sigma^2})$$

$$where, \sigma = 1.3, \mu = 0.4$$

We select these values of σ and μ so as to make sure that the log-normal distribution properly envelops the target distribution for values in between $[0, T_{min}]$.

And for values of x in between $[T_{min}, \infty]$, we use the power-law distribution with density:

$$f_2(x) = \frac{\alpha - 1}{T_{min}} \cdot (\frac{x}{T_{min}})^{-\alpha} \mathbf{1}_{(T_{min}, \infty)}(x)$$

Therefore, our majorizing density is as follows:

$$g(x) = \alpha_1 f_1(x) + \alpha_2 f_2(x)$$

where, α_1 and α_2 are weights assigned to the distributions in the mixture density. They can be derived in the following manner:

$$\alpha_1 = \int_0^{T_{min}} f(x)dx = \int_0^{T_{min}} c(\sqrt{2\pi})^{-1} e^{-\frac{c^2}{2x}} x^{-\frac{3}{2}} dx$$

$$c = 1, T_{min} = 0.9$$

On solving for the mentioned values of c and T_{min} we get a value of:

$$\alpha_1 = 0.291$$

.

Similarly, we solve for α_2, for the same values of c and T_{min} as above:

$$\alpha_2 = \int_{T_{min}}^{\infty} f(x)dx = \int_{T_{min}}^{\infty} c(\sqrt{2\pi})^{-1} e^{-\frac{c^2}{2x}} x^{-\frac{3}{2}} dx$$

And, we get value of

$$\alpha_2 = 0.708$$

therefore,

$$g(x) = (0.291) \cdot \frac{1}{x\sigma\sqrt{2\pi}} exp(-\frac{(ln(x)-\mu)^2}{2\sigma^2}) + (0.708) \cdot \frac{\alpha - 1}{T_{min}} \cdot (\frac{x}{T_{min}})^{-\alpha} \mathbf{1}_{(T_{min},\infty)}(x)$$

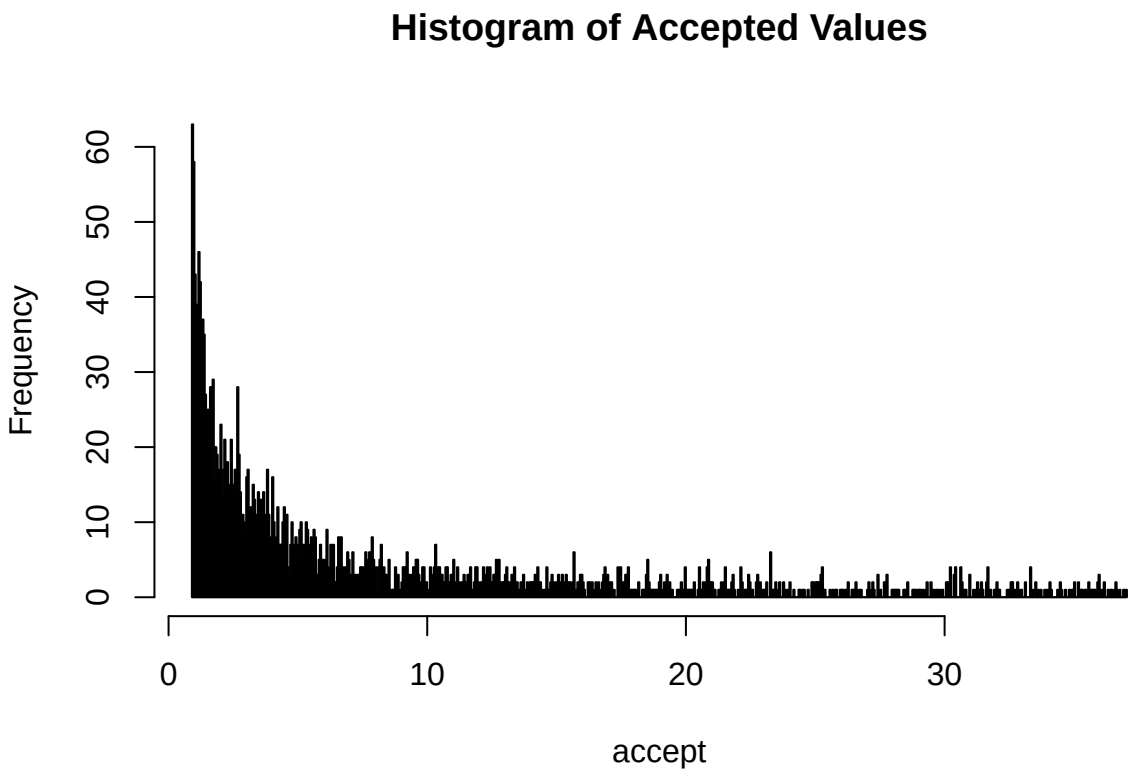
Sub Question 2

We implement an acceptance/rejection algorithm, where a certain number sampled is of the target distribution if it satisfies the following condition:

$$U \leq f_X(Y)/(cf_Y(Y))$$

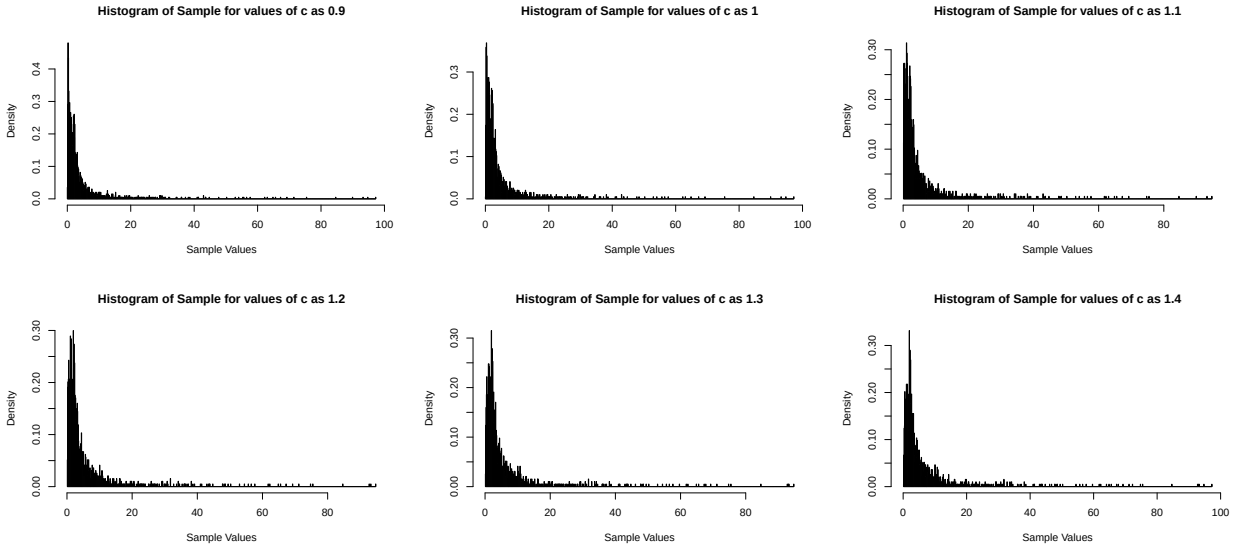
where, U is numbers generated from Uniform distribution, f_X is the target distribution and f_Y is the proposal distribution and c is the majorizing constant.

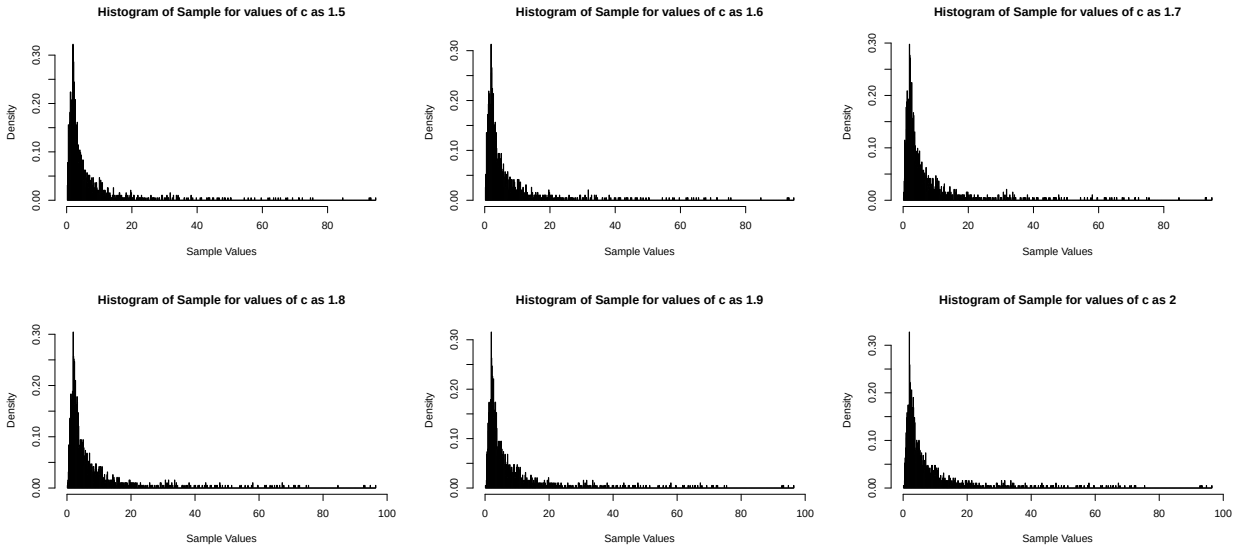
Below is a histogram generated from the implemented sampler:



Sub Question 3

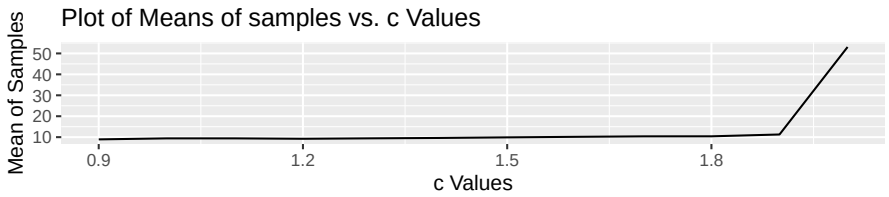
Upon implementing the above sampler for different values of c, we can see below a graphical representation of each such sample:

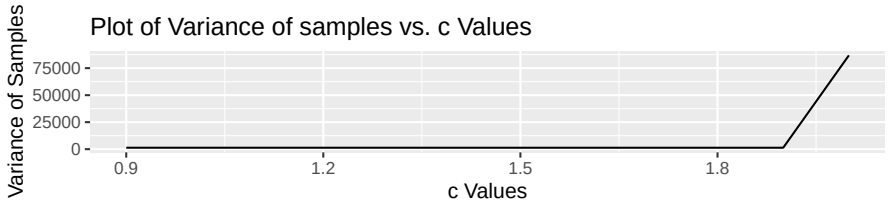


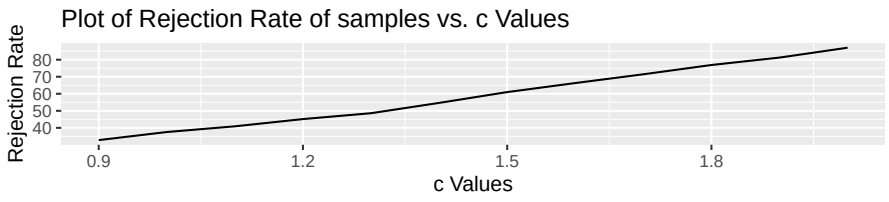


The mean and variance of each sample, along with their rejection rates can be seen in the data frame below:

```
##    cVal Mean_sample Variance_sample Rejection_Rate
## 1   0.9    8.935605        1339.617          32.80
## 2   1.0    9.421635        1337.873          37.55
## 3   1.1    9.421635        1337.873          40.90
## 4   1.2    9.231959        1336.602          45.15
## 5   1.3    9.443570        1334.143          48.60
## 6   1.4    9.617134        1331.458          54.65
## 7   1.5    9.898229        1327.826          61.00
## 8   1.6   10.156771        1324.946          66.30
## 9   1.7   10.387841        1322.312          71.45
## 10  1.8   10.407302        1321.981          76.90
## 11  1.9   11.281420        1317.547          81.25
## 12  2.0   53.074688       86861.958          87.05
```







From the graphs above:

We can make the following inferences about the mean and variances of each sample:

1. The mean increases slightly with an increase in c values, and then spikes for values of c after 1.6.
2. The variance gradually increases with respect to c values, and spikes by a huge magnitude for values of c after 1.6.
3. The rejection rate on the other hand has a linear relationship with the c values, that is, it increases steadily with an increase in c values. With an increase in c values, the majorizing constant also increases which results in an increase in the rejection rate.

Question 2

Sub Question 1

Obtain $DE(0,1)$ from $Unif(0,1)$ using inverse CDF method

Answer: From the question, it is given that:

$DE(\mu, \alpha) = \alpha/2 * exp(-\alpha|x-\mu|)$

This is the density function. For $DE(0,1)$, the above becomes:

$DE(0,1) = exp(-|x|)/2$

Case 1: when $x <= \mu$, the above formula becomes $exp(x)/2$

Case 2: when $x > \mu$, the above formula becomes $exp(-x)/2$

Note: $\mu = 0$ above

Now that the density is available, we can use the inverse-CDF method to generate $DE(0,1)$ from $Unif(0,1)$.

Inverse-CDF:

Step 1: Find the CDF:

Case 1: For $x <= 0$:

$F(X) = \int_{-\infty}^{x} exp(x)/2 = 1/2[exp(x) - exp(-\infty)] = exp(x)/2$

Case 2: For $x > 0$:

$F(X) = \int_{-\infty}^{0} exp(x)/2 + \int_{0}^{x} exp(-x)/2 = 1 - exp(-x)/2$

Step 2: Assign the CDF to U and find the RV X in terms of U:

Case 1: for $x <= \mu$:

$U = exp(X)/2$

$2U = exp(X)$

Taking log on both sides:

$X = ln(2U)$, for $U <= \mu$ Or $U <= 0.5$ as U is $Uniform(0,1)$ and the mean is 1/2(0.5).

Case 1: for $x > \mu$:

$U = 1 - exp(-X)/2$

$2U = 2 - exp(-X)$

$exp(-X) = 2 - 2U$

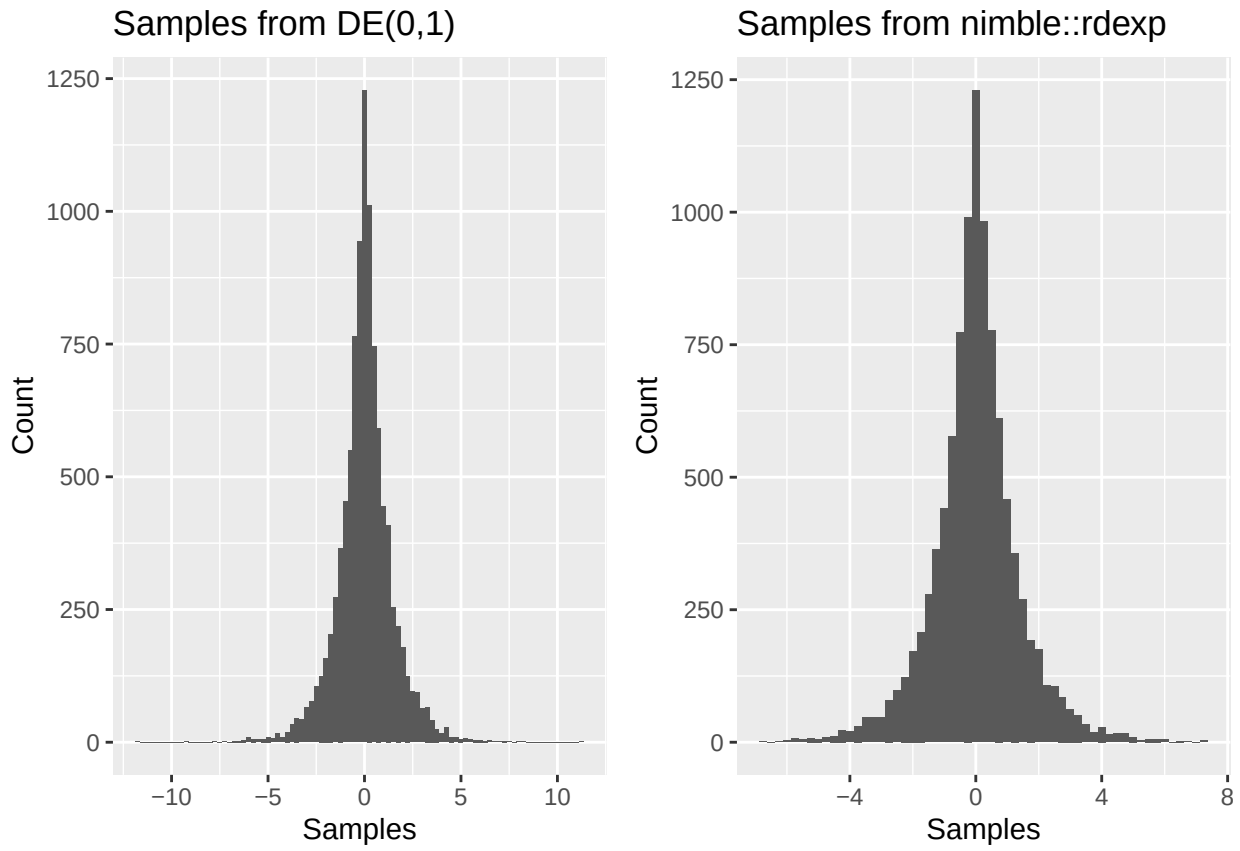
$exp(X) = 1/(2-2U)$

Taking log on both sides:

$X = ln(1/(2-2U))$, for $U > 0.5$

Sub Question 2

Generate 10000 random numbers and plot the histogram. Comment whether the result looks reasonable.



As seen, the histogram is similar to the density plot of Double Exponential Laplace Distribution. Also, the mean of the histogram almost matches with that of the given distribution - $DE(0, 1)$.

```
mean(laplace_samples)
```

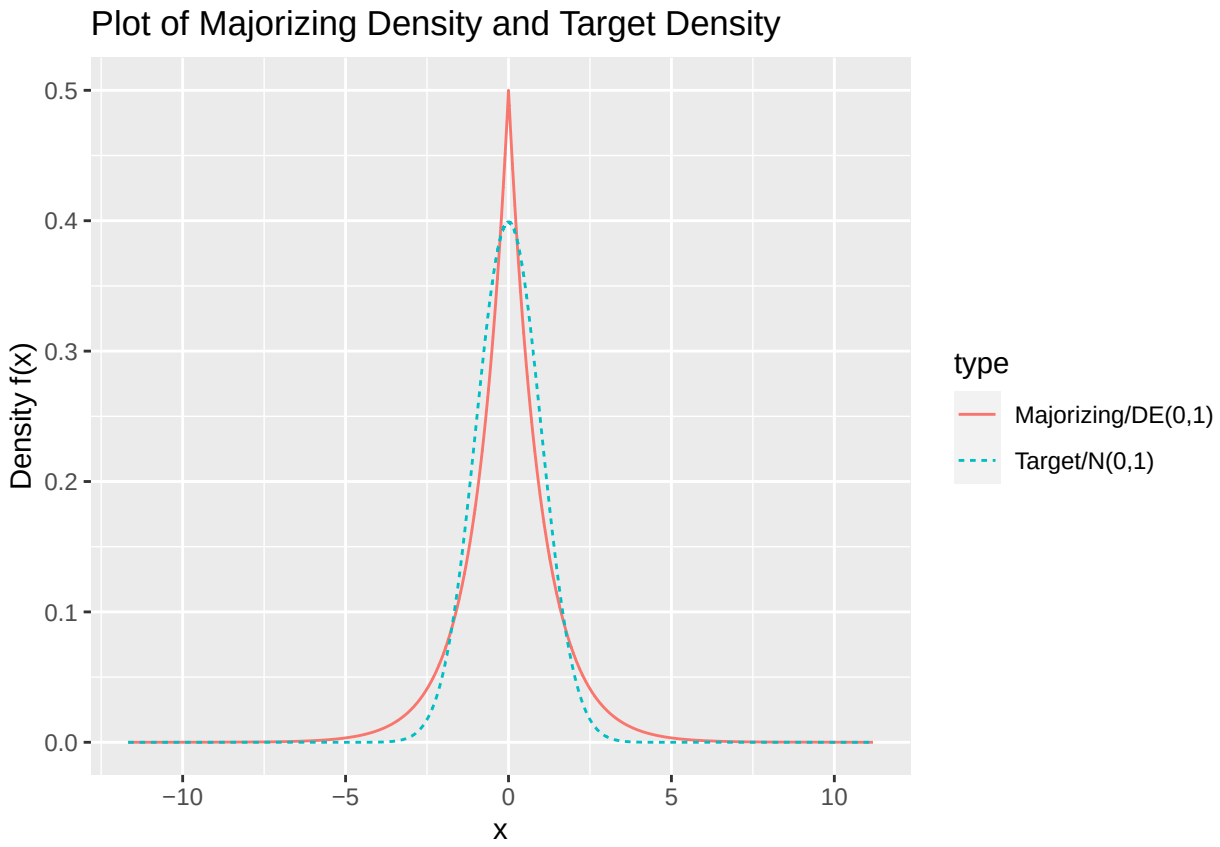
```
## [1] 0.01929563
```

Question 2

Sub Question 1

Use the Acceptance/rejection method with DE(0; 1) as a majorizing density to generate N(0, 1) variables

Step1: Get the probability densities for both majorizing and target functions using the samples generated(referred to as laplace samples going forward) from the Uniform distribution using the inverse CDF

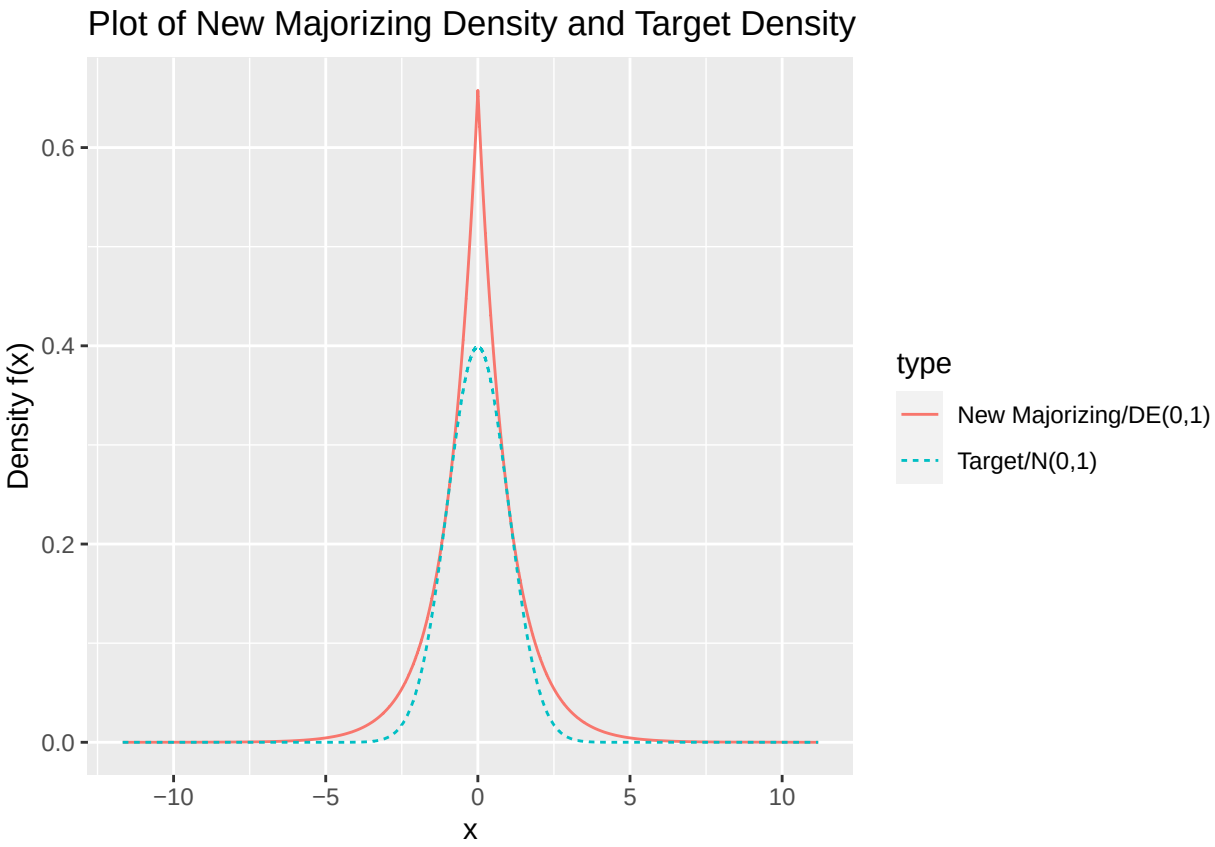


method.

As seen, there are some places where the majorizing curve doesn't cover the target curve.

Step2: Get the majorizing const. The majorizing constant is given by $max(f(x)/g(x))$, where $f(x)$ is the target density and $g(x)$ is the proposed density.

```
## Majorizing Const: 1.315489
```



Upon applying the majorizing constant, it is evident that the new proposed curve now entirely covers the target distribution.

Step3: Use accept/reject algorithm to get 2000 samples: We sample from the laplace samples and based on the probability$(f(x)/c * g(x))$, we randomly either select or reject the sample. $f(x)$ is the target density, $g(x)$ is the proposed density and c is the majorizing constant.

```
## Number of accepted samples: 2000
```

```
## Number of rejected samples: 673
```

Sub Question 2

Compute the average rejection rate R in the acceptance rejection procedure. What is the expected rejection rate ER and how close is it to R?

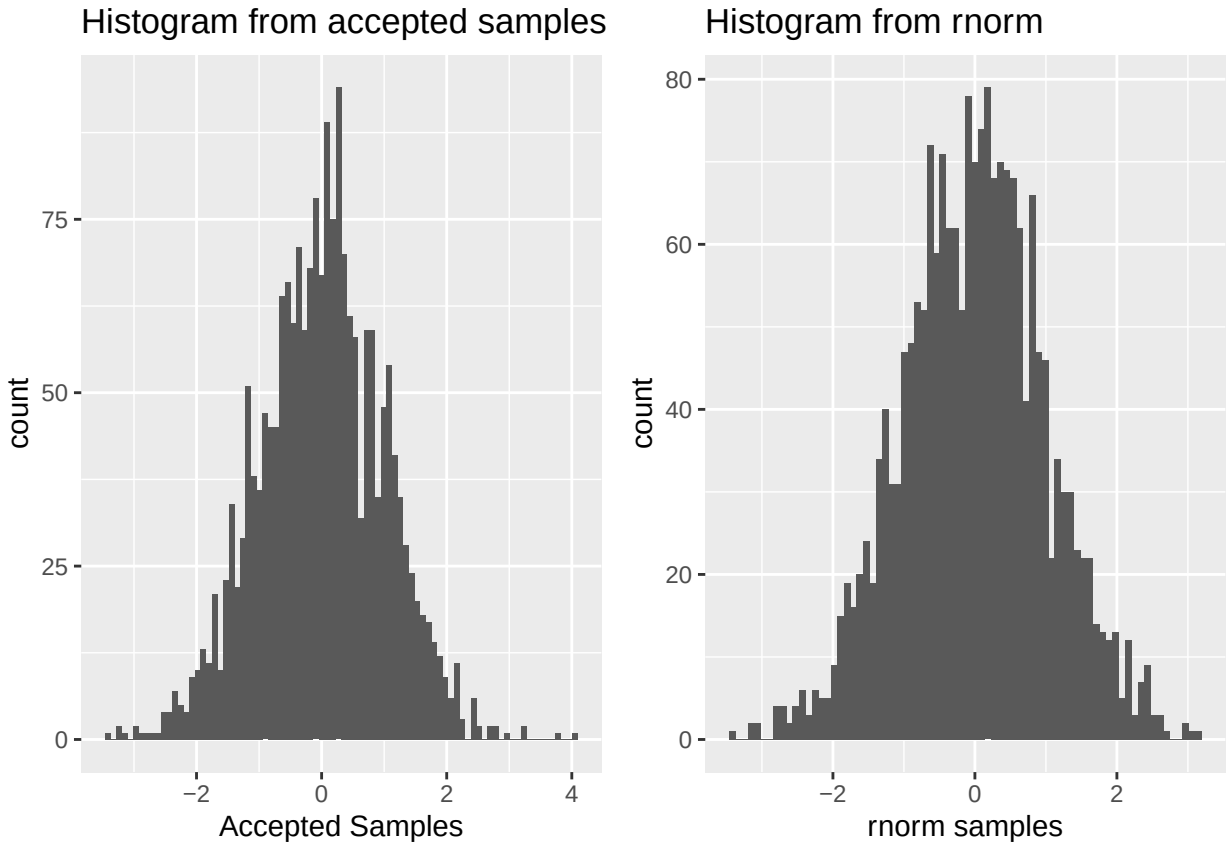
```
## Expected rejection rate: 0.2398265
```

```
## Average rejection rate: 0.251777
```

As seen above, the rejection rate is very close to the expected rejection rate.

Sub Question 3

Generate 2000 numbers from N(0, 1) using standard rnorm() procedure, plot the histogram and compare the obtained two histograms.



Both looks similar. Also, the means are comparable:

```
## Accepted sample mean: 0.002990647 Rnorm mean: -0.01091422
```

Appendix : All code for this report.

```
knitr::opts_chunk$set(echo = TRUE)
library(ggplot2)
library(nimble)
library(gridExtra)
library(dplyr)
library(poweRlaw)
############################Question 1############################
# Target Distribution
f_x <- function(x, c){
  func <- (c/sqrt(2*pi))*(exp(-(c^2)/(2*x)))*(x^(-3/2))
  return(func)
}
```

```r
# Proposal Distribution
f_px <- function(x, t_min, alpha){

  sapply(x, function(y){
    res <- NA
    if (y >= 0  & y <= t_min) {
      res <- (exp(-((log(y)-0.4)^2)/(2*1.3^2)))/(y*sqrt(2*pi)*1.3)
    }else{
      res <- ((alpha - 1)/t_min)*((y/t_min)^(-alpha))
    }
    res
  }, simplify = TRUE)

}

range_c <- seq(0.1,50, 0.1)

# Calculation of majorizing constant by obtaining the max of ratio of target distribution over proposed

c_ratio <- max(f_x(range_c, c = 1)/f_px(range_c, t_min = 0.9, alpha = 1.2))

range <- seq(0.1,5, 0.1)
range_new <- seq(0.9, 10.85, 0.2)
c_ratio_1 <- max(f_x(range_new, c = 1)/f_px(range_new, t_min = 0.9, alpha = 1.2))

plot(x = range, f_x(range, c = 1.1), type = 'l',ylim = c(0, 2), col = 'red',xlab = 'Range of Values', yl
lines(range_new, c_ratio_1*f_px(range_new, t_min = 0.9, alpha = 1.3), type = 'l', col = 'blue')
legend(3, 1.5, legend=c("Target Distribution",
                        "Proposed Distribution"),
       col=c("red", "blue"), lty = rep(1,2))

plot(x = range_c, f_x(range_c, c = 1), type = 'l',ylim = c(0, 1), col = 'red', xlab = 'Range of Values'
lines(range_c, c_ratio*f_px(range_c, t_min = 0.9, alpha = 1.2), type = 'l', col = 'blue')
legend(30, .8, legend=c("Target Distribution",
                        "Proposed Distribution"),
       col=c("red", "blue"), lty = rep(1,2))

count <- 1
accept <- c()
#
set.seed(12345)
U <- runif(10000, 0, 1)
set.seed(12345)
X <- rplcon(10000, xmin = 0.9, alpha = 1.2)

while (count <= 10000 & length(accept) < 2500) {
  test_u <- U[count]
  test_x <- f_x(X[count], c = 1)/(c_ratio*f_px(X[count], t_min = 0.9, alpha = 1.2))
  if (test_u <= test_x) {
    accept <- rbind(accept, X[count])
    count <- count+1
  }
  count <- count+1
```

```
}
hist(accept, breaks = 1000, main = 'Histogram of Accepted Values')


sample_list <- list()
rejection_rate <- c()

rmajorizing<-function(n){
  sapply(1:n,function(i){
    res<-NA
    component<-sample(1:2,1,prob=c(2/3,1/3))
    if(component==1){res<-rlnorm(1, meanlog = 0.4, sdlog = 1.3)}
    if(component==2){res<-rplcon(1,xmin = 0.9, alpha = 1.2)+1}
    res
  })
}

sampler_func <- function(prop_func, tr_func, cVal){
  U <- runif(10000, 0, 1)
  X <- rmajorizing(10000)
  for (i in 1:length(cVal)) {
    accepted_sample <- c()
    rejected_sample <- c()
    counter <- 1
    c_ratio <- max(f_x(range_c, cVal[i])/f_px(range_c, t_min = 0.9, alpha = 1.2))
    while ((counter <= 10000 & length(accepted_sample) < 2000)) {
      if (U[counter] <= tr_func(X[counter], cVal[i])/(c_ratio*prop_func(X[counter],t_min = 0.9, alpha =
        accepted_sample <- rbind(accepted_sample, X[counter])
        counter <- counter + 1
      }else{
        rejected_sample <- rbind(rejected_sample, X[counter])
      }

      counter <- counter + 1
    }
    rejection_rate <<- rbind(rejection_rate,
                             (length(rejected_sample)/length(accepted_sample))*100)
    sample_list[[i]] <<- as.data.frame(accepted_sample)
  }

  # print(counter)
  mean_var_df <- data.frame(matrix(ncol = 4, nrow = 0))
  colnames(mean_var_df) <- c('cVal',
                             'Mean_sample',
                             'Variance_sample',
                             'Rejection_Rate')
  mean_vals <- lapply(1:length(sample_list),
                      function(x)mean(unlist(sample_list[[x]][1:50,])))
  var_vals <- lapply(1:length(sample_list),
                     function(x)var(unlist(sample_list[[x]][1:50,])))
  for (i in 1:length(sample_list)) {
    mean_var_df <- rbind(mean_var_df,
                         data.frame('cVal' = cVal[i],
```

```
                                    'Mean_sample' = unlist(mean_vals[i]),
                                    'Variance_sample' = unlist(var_vals[i]),
                                    'Rejection_Rate' = rejection_rate[i]))
  }
  return(mean_var_df)
}
vals_df <- sampler_func(prop_func = f_px,
                        tr_func = f_x,
                        cVal = seq(0.9, 2, .1))

for (i in 1:length(sample_list)) {
  hist(sample_list[[i]][[1]][which(sample_list[[i]][[1]]<=100)],breaks = 1000, freq = F, main = paste('
}
vals_df
plot_mean <- ggplot(data = vals_df)+
  geom_line(aes(x = cVal,
                y = Mean_sample))+
  ggtitle(label = 'Plot of Means of samples vs. c Values')+
  xlab('c Values')+
  ylab('Mean of Samples')

plot_var <- ggplot(data = vals_df)+
  geom_line(aes(x = cVal,
                y = Variance_sample))+
  ggtitle(label = 'Plot of Variance of samples vs. c Values')+
  xlab('c Values')+
  ylab('Variance of Samples')

plot_rej <- ggplot(data = vals_df)+
  geom_line(aes(x = cVal,
                y = Rejection_Rate))+
  ggtitle(label = 'Plot of Rejection Rate of samples vs. c Values')+
  xlab('c Values')+
  ylab('Rejection Rate')

gridExtra::grid.arrange(plot_mean,plot_var,plot_rej)

##########################Question 2##########################
unif_samples <- runif(n = 10000, min = 0, max = 1)
laplace_samples <- sapply(unif_samples, function(u){
  if(u >= 0.5)return(log(1/(2 - 2*u) ))
  else return(log(2*u))
})
plot1 <- ggplot(
  data.frame(f_x = laplace_samples), aes(f_x)) +
  geom_histogram(binwidth = 0.25) +
  xlab('Samples') + ylab('Count') +
  ggtitle('Samples from DE(0,1)')
plot2 <- ggplot(
  data.frame(f_x = nimble::rdexp(10000, location = 0, scale = 1)), aes(f_x)) +
  geom_histogram(binwidth = 0.25) +
  xlab('Samples') + ylab('Count') +
  ggtitle('Samples from nimble::rdexp')
```

```r
grid.arrange(plot1, plot2, ncol = 2)
mean(laplace_samples)
#Function to get the majorizing density
majorizing_den_fn <- function(x){
  f_x <- sapply(x, function(x){
    if(x<=0) return(exp(x)/2)
    else return(exp(-x)/2)
  })
  return(f_x)
}

y_de <- majorizing_den_fn(laplace_samples)
y_norm <- sapply(laplace_samples, function(i) dnorm(x = i, mean = 0, sd = 1, log = FALSE))
# plot(x = x, y = y, type = 'l', main = 'Majorizing Density - DE(0,1)')
major_den_df <- data.frame(x = laplace_samples,
                           type = rep('Majorizing/DE(0,1)',length(laplace_samples)),
                           y = y_de)
target_den_df <- data.frame(x = laplace_samples,
                            type = rep('Target/N(0,1)', length(laplace_samples)),
                            y = y_norm)
p_density_df <- rbind(major_den_df,target_den_df)
plot3<- ggplot(p_density_df, aes(x = x, y = y)) +
  geom_line(aes(color = type, linetype = type)) +
  ylab('Density f(x)') +
  ggtitle('Plot of Majorizing Density and Target Density')
plot3
majorizing_const <- max(y_norm/y_de)
cat('Majorizing Const:', majorizing_const)
new_y_de <- y_de * majorizing_const
new_major_den_df <- data.frame(x = laplace_samples,
                               type = rep('New Majorizing/DE(0,1)',length(laplace_samples)),
                               y = new_y_de)
p_density_df <- rbind(new_major_den_df,target_den_df)
plot4 <- ggplot(p_density_df %>% filter(type != 'Majorizing/DE(0,1)'),
                aes(x = x, y = y)) +
  geom_line(aes(color = type, linetype = type)) +
  ylab('Density f(x)') +
  ggtitle('Plot of New Majorizing Density and Target Density')
plot4
accept_v <- c()
reject_v <- c()
while(length(accept_v)<2000){
  sample_index <- sample(1:length(laplace_samples), 1)#Get a random index for picking sample
  sample <- laplace_samples[sample_index]
  u <- runif(1)
  target_den <- dnorm(x = sample, mean = 0, sd = 1, log = FALSE)
  majorizin_den <- majorizing_den_fn(sample)
  # cat('\nSample:', sample,'Target den:',target_den, 'Majorizing den:', majorizin_den)
  if(u <= target_den/(majorizing_const * majorizin_den)){
    accept_v <- c(accept_v, sample)
  }else{
    reject_v <- c(reject_v, sample)
  }
```

```
}
cat('Number of accepted samples:', length(accept_v))
cat('Number of rejected samples:', length(reject_v))
avg_reject_rate <- length(reject_v) / (length(reject_v) + length(accept_v))
expt_reject_rate <- 1 - (1/majorizing_const)
cat('Expected rejection rate:', expt_reject_rate)
cat('Average rejection rate:', avg_reject_rate)
plot5 <- ggplot(data.frame(Accept_Samples = accept_v), aes(x = Accept_Samples)) +
  geom_histogram(binwidth = 0.09) +
  ggtitle('Histogram from accepted samples') +
  xlab('Accepted Samples')
rand_samples <- rnorm(2000, mean = 0, sd = 1)
plot6 <- ggplot(data.frame(Accept_Samples = rand_samples),
       aes(x = Accept_Samples)) +
  geom_histogram(binwidth = 0.09) +
  ggtitle('Histogram from rnorm') +
  xlab('rnorm samples')
grid.arrange(plot5, plot6, nrow = 1)
cat('Accepted sample mean:', mean(accept_v), 'Rnorm mean:', mean(rand_samples))
```