# Different distribution

## Generate double exponential distribution DE(0,1) from Unif(0,1) by using the inverse CDF method

* There are three steps to generate samples from a distribution using inverse CDF method:

1. Generate random probabilities. This can be done using uniform random generator between 0 and 1.
2. From the pdf of a distribution, find the inverse CDF of the distribution.
3. Substitute the probabilities in the inverse CDF of the distribution to get the samples.

Repeat above procedure many times to generate samples so that it represent the distribution

We have Laplace distribution as :

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

CDF of Laplace distribution, **When x >=** $\mu$, the sign of the mod operator does not change and we get CDF as :

$$F(X) = 1 - \frac{1}{2} * (1 - e^{-\alpha * (x-\mu)})$$

**When x <** $\mu$, the sign of(x-$\mu$) flips and we get CDF as:

$$F(x) = \frac{1}{2} * (e^{\alpha * (x-\mu)})$$

The inverse of laplace distribution,

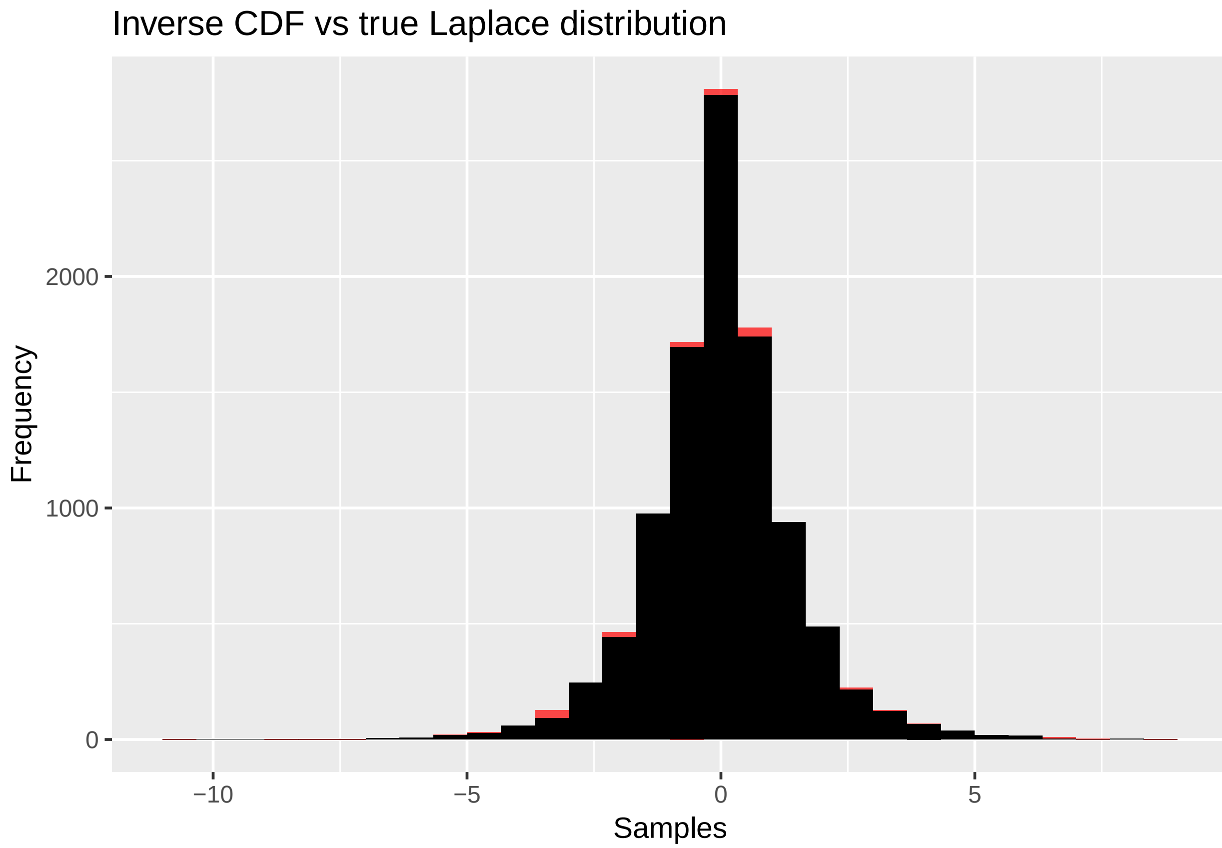
**When P<0.5** and this happens when x < $\mu$ :

$$F^{-1}(P) = \frac{1}{\alpha} * log(2P) + \mu$$

When **P >=0.5* and this happens when x >= $\mu$ :

$$F^{-1}(P) = \mu - \frac{1}{\alpha} * log(2 - log2P)$$

```r
library(ggplot2)

set.seed(123456)
p <- data.frame(runif(10000,0,1))  #Generate probabilities between 0 and 1
colnames(p) <- "Uniform_num"

#Generate samples from inverse of CDF
c <- ifelse(p$Uniform_num<0.5, log(2*p$Uniform_num), -log(2-(2*p$Uniform_num)))

p$X <- c

#Estimating distribution of true laplace
p$Laplace <- rmutil::rlaplace(10000,0,1)

ggplot(p) + geom_histogram(aes(X),fill="red",alpha =0.7) +
    geom_histogram(aes(Laplace),fill="black") + xlab("Samples") +
    ylab("Frequency") + ggtitle("Inverse CDF vs true Laplace distribution")
```




The graph shows how similar is the our lapalace histogram generated(red) with inverse CDF method with true laplace(black) for 10000 random numbers. Moreover, the generated distribution looks similar to Laplace distribution which resembles 2 exponential distribution back to back.

### Acceptance/Rejection method

This is used when it is not possible to find inverse CDF of a distribution so that we can get RV using inverse but we have a pdf for the distribution.

The idea is to find a probabiliy distribution, g(x), from which we can generate a RV and able to tell whether this RV can be accepted for our target distribution f(x).

$$g(x|\mu = 0, \alpha = 1) = \frac{1}{2} * e^{-|x|} - LapalceDistribution$$

$$f(x|\mu = 0, \alpha = 1) = \frac{1}{\sqrt{2\pi}} * e^{\frac{-x^2}{2}} - NormalDistribution$$

We assume that the ratio $\frac{f(x)}{g(x)}$ is bounded by a constant C. C(g(x)) acts as a envelope for the target function f(x). This fraction inherently implies that how many fraction of RV for f(x) is included in C*g(x). We need to maximize this fraction so that we can cover most of the points in g(x) for f(x). For this we need to take differential and equate it to zero.

$$C \geq \frac{f(x)}{g(x)}$$

$$C \geq \frac{\sqrt{2} * e^{-\frac{x^2}{2}+|x|}}{\sqrt{\pi}}$$

Then we differentiate the above ratio and equate to 0 to get the value of x which will give the value of C.

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

Setting the above differential to zero we get the maximum value of above equation at x=1, value of C is obtained.

$$C = \sqrt{\frac{2e}{\pi}}$$

The condition to accept RV generated from g(x) as RV for f(x) is :

$$U \leq \frac{f(x)}{C * g(x)}$$

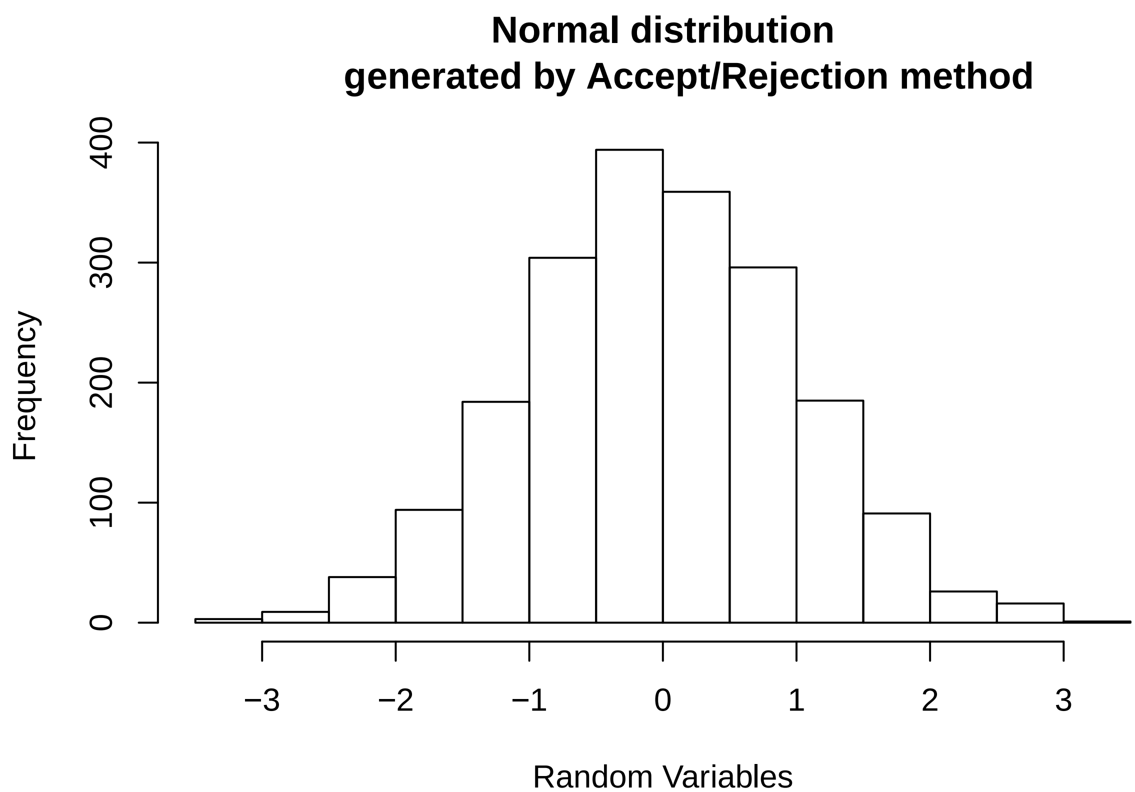
$$U \leq 0.5 * e^{\frac{x^2}{2}+|x|}$$

* **Following is the Acceptance Rejection algorithm :**

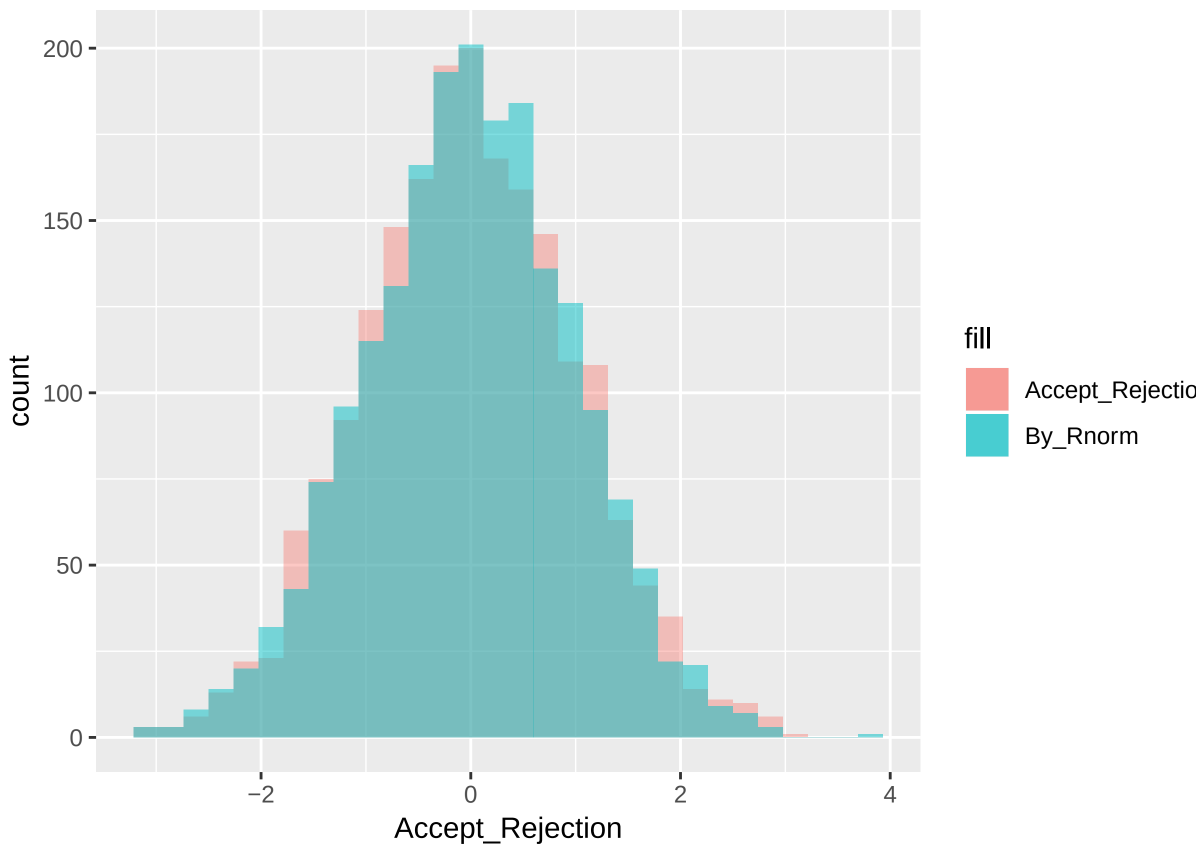
1. Sample $X \sim g(x)$.
2. Sample $U \sim Unif(0,1)$.
3. Reject X if U > $\frac{f(x)}{C.g(x)}$. Go to step 1.
4. Else accept X for f(x).
5. Keep repeating the above step for desired number of samples.

```
accept_reject <- function(sam) {
    f_x <- c()
    cnt <- 1
    while(cnt<=sam){
        U <- runif(1,0,1)
#generate a random variable from laplace distribution
        r_x <- ifelse(U<0.5, log(2*U), -log(2-(2*U)))
        uni <- runif(1)
        frac <- exp(-(r_x^2)/2 + abs(r_x) - 0.5)  #value of f(X)/c(g(X))

        if(uni<=frac){
            f_x[cnt]<- r_x
            cnt <- cnt+1
        }
        total <<- total + 1
    }

return(f_x)
}

n = 2000
total <-0
```

```r
values <- accept_reject(n)
normal_dist <- rnorm(2000,0,1)

compare_data <- cbind(values,normal_dist)
compare_data <- as.data.frame(compare_data)

colnames(compare_data) <- c("Accept_Rejection","By_Rnorm")

hist(compare_data$Accept_Rejection,main = "Normal distribution
     generated by Accept/Rejection method", xlab = "Random Variables")
```




We can see that the normal distribution generated by acceptance rejection method is nearly same as distribution




generated by rnorm.

The expected rejection rate is equal to:

$$1 - \frac{1}{c} = 0.2398264$$

The average rejection rate is :

$$1 - \frac{2000}{total}$$

Where total is the total number of iterations required to generate 2000 samples. Our average rejection rate is nearly equal to expected rejection rate.

```
The average rejection rate is : 0.2595335
```