# Exercise 1 : Negative weighted mixture

Matthieu Galtier and Hannah Gloaguen

2024-10-27

## Definition

### Question 1

For $f(x)$ to be a probability density function, it must satisfy two conditions :

1) $f(x) \geq 0, \forall x \in \mathbb{R}$
2) $\int_{\mathbb{R}} f(x)dx = 1$

We need $f(x)$ to be positive $\forall x \in \mathbb{R}$:

$f(x) \geq 0 \Leftrightarrow \lambda(f_1(x) - af_2(x)) \geq 0, \forall \lambda \geq 0 \Leftrightarrow \; f_1(x) - af_2(x) \geq 0 \Leftrightarrow \frac{1}{\sqrt{2\pi}\sigma_1} \exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\} - \frac{a}{\sqrt{2\pi}\sigma_2} \exp\left\{\frac{-(x-\mu_2)^2}{2\sigma_2^2}\right\} \geq 0$
$\Leftrightarrow \frac{1}{\sigma_1} \exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\} \geq \frac{a}{\sigma_2} \exp\left\{\frac{-(x-\mu_2)^2}{2\sigma_2^2}\right\}$

When $x \to \pm\infty$, we have: $\frac{1}{\sigma_1} \exp\left\{-\frac{(x-\mu_1)^2}{2\sigma_1^2}\right\} \approx \frac{1}{\sigma_1} \exp\left\{-\frac{x^2}{2\sigma_1^2}\right\} \approx \exp\left\{\frac{-x^2}{2\sigma_1^2}\right\}$

and $\frac{1}{\sigma_2} \exp\left\{-\frac{(x-\mu_2)^2}{2\sigma_2^2}\right\} \approx \frac{1}{\sigma_2} \exp\left\{-\frac{x^2}{2\sigma_2^2}\right\} \approx \exp\left\{\frac{-x^2}{2\sigma_2^2}\right\}$

So we need to compare $\exp\left\{\frac{-x^2}{2\sigma_1^2}\right\}$ and $\exp\left\{\frac{-x^2}{2\sigma_2^2}\right\}$. Which is the same as comparing $-\frac{x^2}{2\sigma_1^2}$ and $-\frac{x^2}{2\sigma_2^2}$ because the exponential function is non decreasing. $\frac{-x^2}{2\sigma_1^2} \geq \frac{-x^2}{2\sigma_2^2} \Leftrightarrow \frac{2\sigma_1^2}{x^2} \geq \frac{2\sigma_2^2}{x^2} \Leftrightarrow \quad \sigma_1^2 \geq \sigma_2^2$

So, for $f(x)$ to be a density, we need $\quad \sigma_1^2 \geq \sigma_2^2$.

### Question 2

$f_1(x) - af_2(x) \geq 0 \Leftrightarrow f_1(x) \geq af_2(x) \Leftrightarrow \quad \frac{f_1}{f_2}(x) \geq a^* \geq a \Rightarrow a^* = \min_{x \in \mathbb{R}} \frac{f_1}{f_2}(x)$

Let $g(x) := \frac{f_1(x)}{f_2(x)}$

$g(x) = \frac{\sigma_2}{\sigma_1} \exp\left\{\frac{1}{2}\left(\frac{(x-\mu_2)^2}{{\sigma_2}^2} - \frac{(x-\mu_1)^2}{{\sigma_1}^2}\right)\right\}$

$\min _{x \in \mathbb{R}} g(x) \Leftrightarrow \min _{x \in \mathbb{R}} h(x) $ where $h(x) := \frac{1}{\sigma_2^2}(x-\mu_2)^2 - \frac{1}{\sigma_1^2}(x-\mu_1)^2, \forall x \in \mathbb{R}$

By developing $h(x)$, we get :

$$\begin{aligned} h(x) &= x^2\left(\frac{1}{\sigma_2^2} - \frac{1}{\sigma_1^2}\right) + x\left(\frac{2\mu_1}{\sigma_1^2} - \frac{2\mu_2}{\sigma_2^2}\right) + \frac{\mu_2^2}{\sigma_2^2} - \frac{\mu_1^2}{\sigma_1^2} \\ &= x^2\frac{\sigma_1^2 - \sigma_2^2}{\sigma_1^2\sigma_2^2} + 2x\frac{\mu_1\sigma_2^2 - \mu_2\sigma_1^2}{\sigma_1^2\sigma_2^2} + \frac{\mu_2^2}{\sigma_2^2} - \frac{\mu_1^2}{\sigma_1^2} \end{aligned}$$

$h(x)$ is a $2^{\text{nd}}$ degree polynomial function and $\sigma_1^2 - \sigma_2^2 \geq 0$ by assumption. So $\max_{x \in \mathbb{R}} h(x) = h(\bar{x})$ where $h'(\bar{x}) = 0$

$$h'(\bar{x}) = \frac{2\bar{x}\left(\sigma_1^2 - \sigma_2^2\right)}{\sigma_1^2\sigma_2^2} + \frac{2\left(\mu_1\sigma_2^2 - \mu_2\sigma_1^2\right)}{\sigma_1^2\sigma_2^2} = 0$$

$$\Leftrightarrow \bar{x}\left(\sigma_1^2 - \sigma_2^2\right) = \mu_2\sigma_1^2 - \mu_1\sigma_2^2 \Leftrightarrow \bar{x} = \frac{\mu_2\sigma_1^2 - \mu_1\sigma_2^2}{\sigma_1^2 - \sigma_2^2}$$

$$\begin{aligned}
h(\bar{x}) &= \frac{1}{\sigma_2^2}\left(\left(\frac{\mu_2\sigma_1^2 - \mu_1\sigma_2^2}{\sigma_1^2 - \sigma_2^2}\right) - \mu_2\right)^2 - \frac{1}{\sigma_1^2}\left(\left(\frac{\mu_2\sigma_1^2 - \mu_1\sigma_2^2}{\sigma_1^2 - \sigma_2^2} - \mu_1\right)^2\right. \\
&= \frac{1}{\sigma_2^2}\left(\frac{\mu_2\sigma_1^2 - \mu_1\sigma_2^2 - \mu_2\sigma_1^2 + \mu_2\sigma_2^2}{\sigma_1^2 - \sigma_2^2}\right)^2 - \frac{1}{\sigma_1^2}\left(\frac{\mu_2\sigma_1^2 - \mu_1\sigma_2^2 - \mu_1\sigma_1^2 + \mu_1\sigma_2^2}{\sigma_1^2 - \sigma_2^2}\right)^2 \\
&= \frac{1}{\left(\sigma_1^2 - \sigma_2^2\right)^2}\left(\frac{1}{\sigma_2^2}\left(\mu_2\sigma_2^2 - \mu_1\sigma_2^2\right)^2 - \frac{1}{\sigma_1^2}\left(\mu_2\sigma_1^2 - \mu_1\sigma_1^2\right)^2\right) \\
&= \frac{1}{\left(\sigma_1^2 - \sigma_2^2\right)^2}\left(\sigma_2^2\left(\mu_2 - \mu_1\right)^2 - \sigma_1^2\left(\mu_2 - \mu_1\right)^2\right) \\
&= \frac{1}{\left(\sigma_1^2 - \sigma_2^2\right)^2}\left(\left(\mu_2 - \mu_1\right)^2\left(\sigma_2^2 - \sigma_1^2\right)\right) \\
&= \frac{-\left(\mu_2 - \mu_1\right)^2}{\sigma_1^2 - \sigma_2^2}
\end{aligned}$$

So, $a^* = g(\bar{x}) = \frac{\sigma_2}{\sigma_1}\exp\left\{-\frac{1}{2}\frac{\left(\mu_2 - \mu_1\right)^2}{\sigma_1^2 - \sigma_2^2}\right\}$

For $f(x)$ to be a probability density function, we must have $\int_{\mathbb{R}} f(x)dx = 1$ :

$\int_{\mathbb{R}} f(x)\,dx = \lambda\int_{\mathbb{R}} f_1(x)\,dx + \lambda a\int_{\mathbb{R}} f_2(x)\,dx = 1$

$$\Leftrightarrow \lambda - \lambda a = 1 \Leftrightarrow \lambda(1 - a) = 1$$

$$\Leftrightarrow \lambda = \frac{1}{1 - a}$$

## Question 3

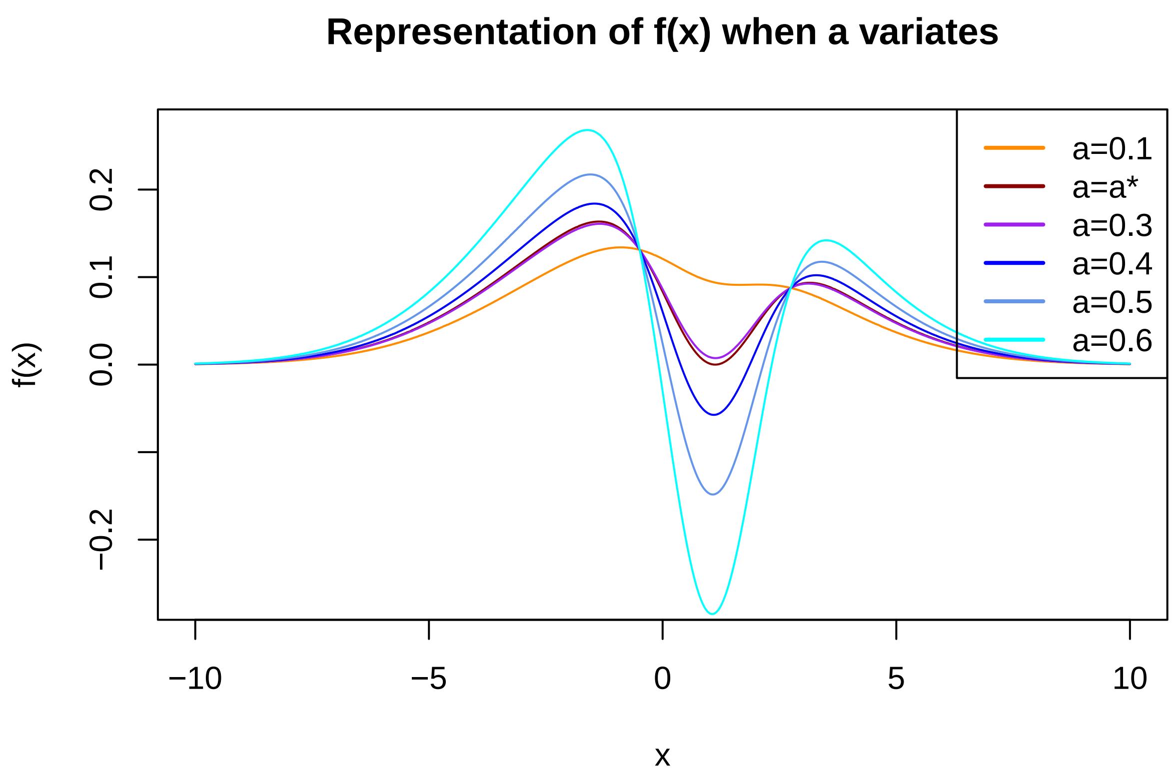
### Creation of the function f

```
f<-function(a, m_1, m_2, s_1, s_2, x){
  return((1/(1-a))*(dnorm(x, m_1, s_1)-a*dnorm(x, m_2, s_2)))
}
```

### Plot the pdf for different values of a

```
a_star<-function(m_1, m_2, s_1, s_2){
  return((s_2/s_1)*(exp((-1/2)*((m_2-m_1)^2)/(s_1^2-s_2^2))))}

X<-seq(-10,10,0.001)

plot(X, f(0.1, 0, 1, 3, 1, X), type='l', ylim=c(-0.27, 0.27), xlim=c(-10, 10),
     ylab="f(x)", xlab="x", col='#FF8C00', main='Representation of f(x) when a variates')
lines(X, f(a_star(0, 1, 3, 1), 0, 1, 3, 1, X), type='l', col='#8B0000')
```

```r
lines(X, f(0.3, 0, 1, 3, 1, X), type='l', col='purple')
lines(X, f(0.4, 0, 1, 3, 1, X), type='l', col='blue')
lines(X, f(0.5, 0, 1, 3, 1, X), type='l', col='#6495ED')
lines(X, f(0.6, 0, 1, 3, 1, X), type='l', col='#00FFFF')

legend("topright", legend = c("a=0.1", "a=a*", "a=0.3", "a=0.4", "a=0.5", "a=0.6"),
       col = c("#FF8C00", "#8B0000", "purple", "blue", "#6495ED", "#00FFFF"), lwd = 2)
```

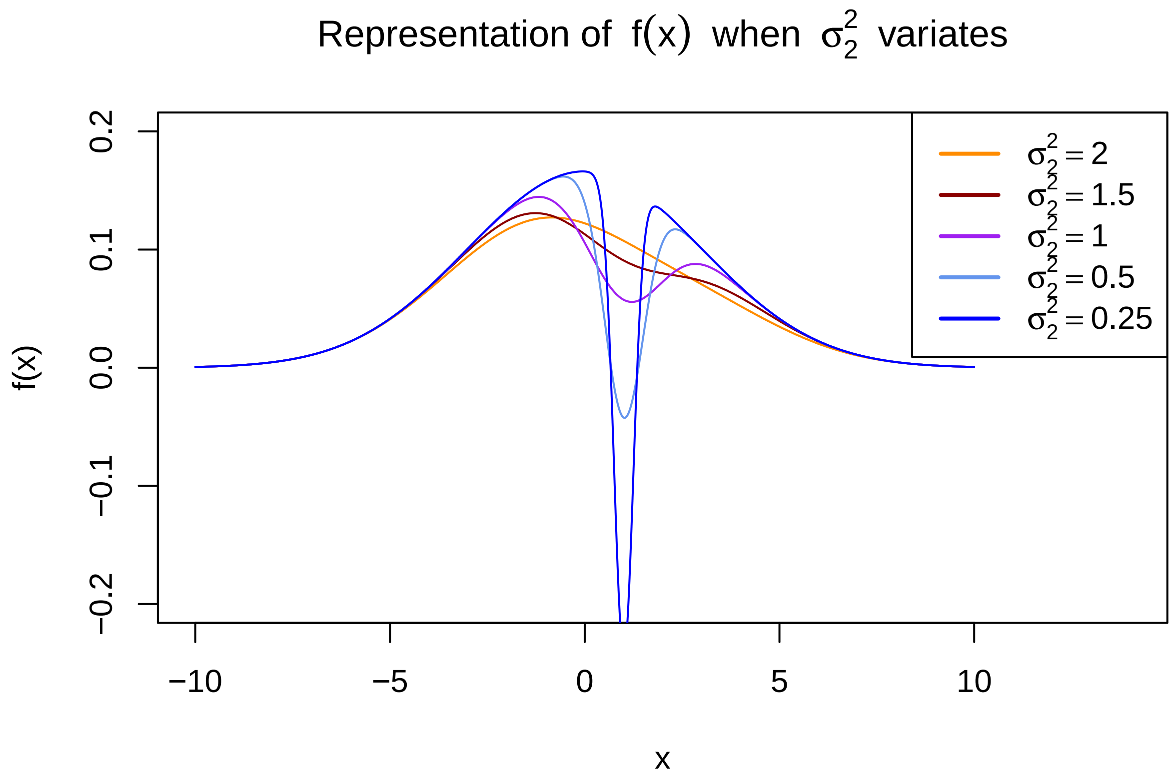



$f(x) \geq 0 \Leftrightarrow \frac{f_1(x)}{f_2(x)} \geq a^* \geq a > 0$. So, for $f$ to be non negative, a must belong to $]0, a^*]$.

**Plot the pdf for different values of $\sigma[2]^2$**

```r
plot(X, f(0.2, 0, 1, 3, 2, X), type='l', ylim=c(-0.2, 0.2), xlim=c(-10, 14),
     ylab="f(x)", xlab="x", col='#FF8C00', main = bquote("Representation of " ~ f(x) ~ " when " ~ sigma
lines(X, f(0.2, 0, 1, 3, 1.5, X), type='l', col='#8B0000')
lines(X, f(0.2, 0, 1, 3, 1, X), type='l', col='purple')
lines(X, f(0.2, 0, 1, 3, 0.5, X), type='l', col='#6495ED')
lines(X, f(0.2, 0, 1, 3, 0.25, X), type='l', col='blue')

legend("topright", legend=c(expression(sigma[2]^2 == 2),
                            expression(sigma[2]^2 == 1.5),
                            expression(sigma[2]^2 == 1),
                            expression(sigma[2]^2 == 0.5),
```

```r
                        expression(sigma[2]^2 == 0.25)),
       col = c("#FF8C00", "#8B0000", "purple", "#6495ED", "blue"), lwd=2)
```




We see that we must have $\sigma_2^2 \geq 1$ for $f$ to be non negative.

From now on, we set the following numerical values: $\mu_1 = 0, \mu_2 = 1, \sigma_1 = 3, \sigma_2 = 1$ and $a = 0.2$.

```r
cat("a* = ", a_star(0, 1, 3, 1),"\n" )
```

```
## a* =  0.3131377
```

We have $\sigma_1^2 = 9 \geq \sigma_2^2 = 1 \geq 1$ and $a = 0.2 < a^* = 0.3131$. So, these numerical values satisfy the constraints.

# Inverse c.d.f Random Variable simulation

## Question 4

### Show that the cdf is available in closed form

We know that $F(x)$ can be written such that: $F(x) = \frac{1}{1-a}F_1(x) - \frac{a}{1-a}F_2(x), \forall x \in \mathbb{R}$, where we know and can compute $F_1(x)$ and $F_2(x)$. In particular, we can compute them with R functions.

**Creation of the function F**

```
F<-function(a, m_1, m_2, s_1, s_2, x){return((1/(1-a))*(pnorm(x, m_1, s_1)-a*pnorm(x, m_2, s_2)))}
```

### Construction of the algorithm for the inverse method

F is continuous because it is a linear transformation of two continuous functions and for every 0<u<1, $F^{-1}(u)$ exists because :

```
cat("F(-113)=", F(0.2, 0, 1, 3, 1, -113),"\n" )
```

```
## F(-113)= 0
```

```
cat("F(17)=", F(0.2, 0, 1, 3, 1, 17),"\n" )
```

```
## F(17)= 1
```

So we can construct a dichotomy algorithm :

```
dicho<-function(a, m_1, m_2, s_1, s_2, u, e){
  x_min<--113
  x_max<-17
  while ((x_max-x_min)>e) {
    if (F(0.2, 0, 1, 3, 1, (x_min+x_max)/2)>u){
      x_max<-(x_min+x_max)/2
    }
    else if (F(0.2, 0, 1, 3, 1, (x_min+x_max)/2)<u){
      x_min<-(x_min+x_max)/2
    }
    else {
      return((x_min+x_max)/2)
    }
  }
  return((x_min+x_max)/2)
}
```

### Generate a random variable that follows the law of F

```
U<-runif(1)
Y<-dicho(0.2, 0, 1, 3, 1, U, 0.001)
print(Y)
```

```
## [1] 2.611153
```

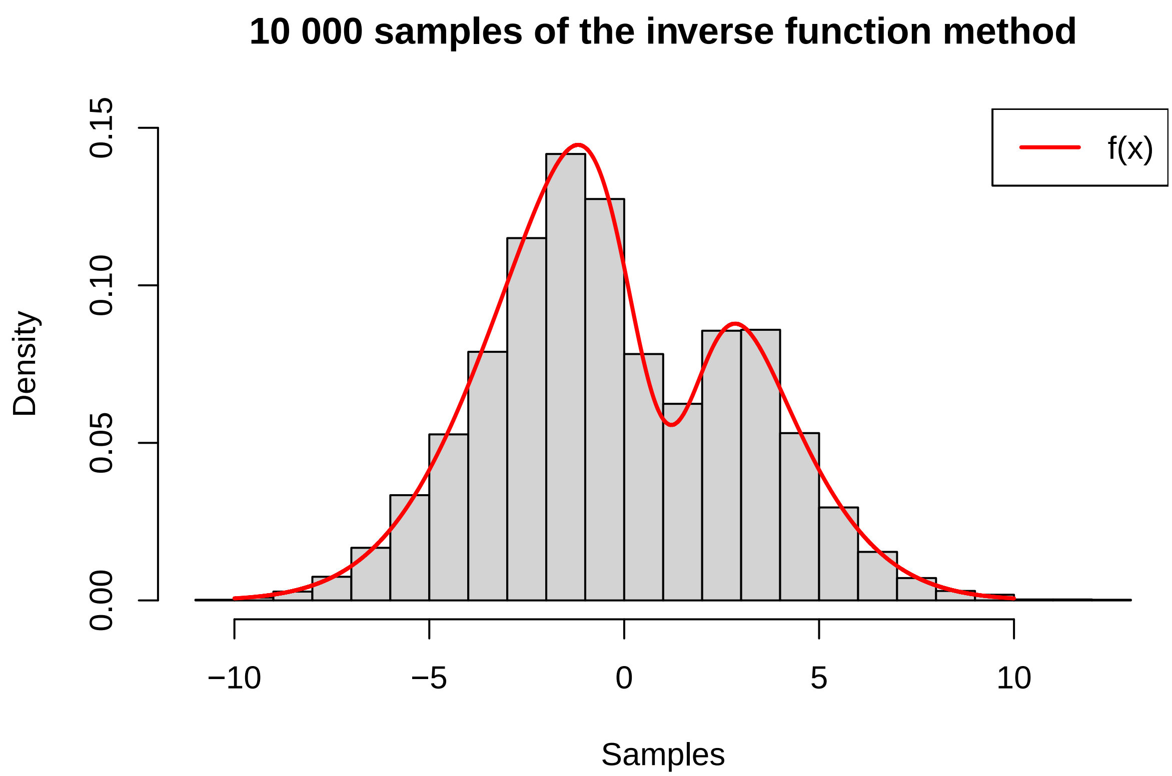
## Question 5

### Creation of a function that generates n samples from f using the inverse function method

```
inv_cdf<-function(n){
  U<-runif(n)
  L<-c()
  for (i in 1:n){
    Y<-dicho(0.2, 0, 1, 3, 1, U[i], 0.001)
    L[i]=Y
  }
  return(L)
}
```

**Generation of 10 000 samples to prove that the method is correct**

```
L<-inv_cdf(10000)
X<-seq(-10,10,0.001)
hist(L, breaks=20, probability=TRUE, main="10 000 samples of the inverse function method",
     ylim=c(0, 0.15), xlab="Samples")
lines(X,f(0.2,0,1,3,1,X), type='l', lwd=2, col='red')
legend("topright",legend = c("f(x)"), col=c('red'), lwd=2)
```




We graphically observe that the algorithm is correct.

## Accept-Reject Random Variable simulation

### Question 6

First, we have to find a constant $M$ and a function $g$ such that : $f(x) \leq \ Mg(x), \forall x \in \mathbb{R}$

We know that :

$f(x) = \frac{1}{1-a} f_1(x) - \frac{a}{1-a} f_2(x) \leq \frac{1}{1-a} f_1(x)$ , because $\frac{a}{1-a} f_2(x) \geq 0$.

So we choose $g(x) = f_2(x)$ and $M = \frac{1}{1-a}$.

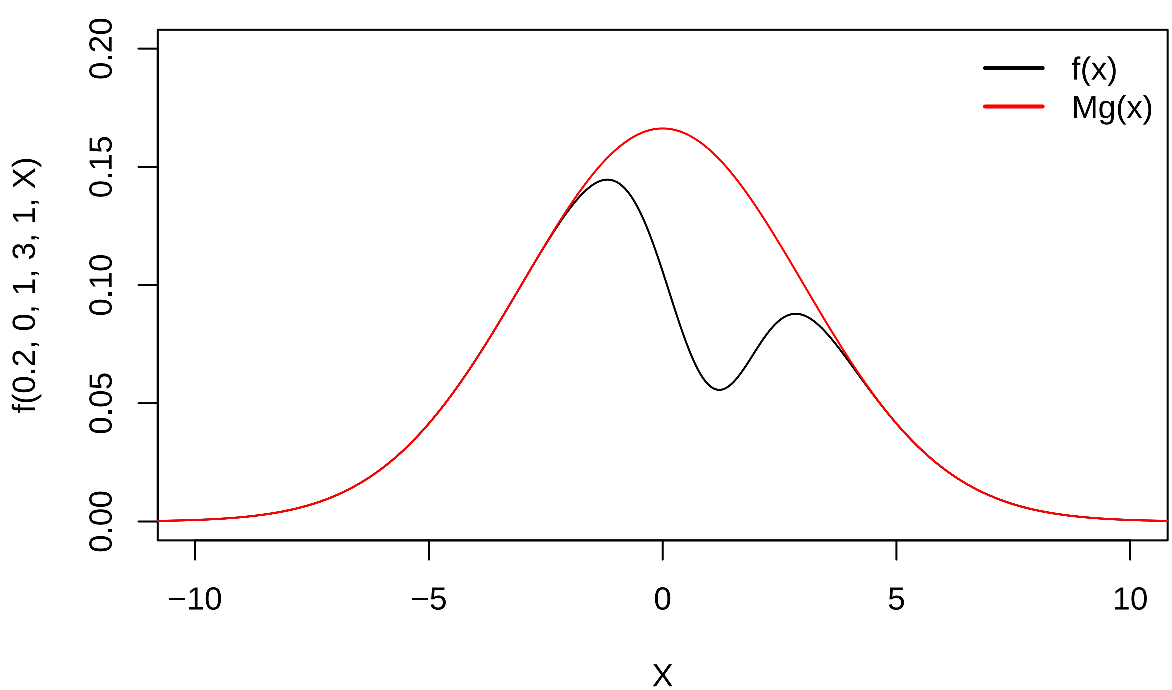
With the $a$ given in the exercise, we get :

```
M<-1/(1-0.2)
cat("M=", M,"\n" )
```

```
## M= 1.25
```

Here is a graphical proof that $f(x) \leq \ Mg(x)$ :

```
X<-seq(-40, 20, 0.001)
plot(X,xlim = c(-10,10) ,ylim=c(0,0.2), f(0.2, 0, 1, 3, 1, X), type='l')
lines(X, (1/0.8)*dnorm(X, 0, 3), col='red')
legend("topright", legend = c("f(x)","Mg(x)"), col=c('black', 'red'),
       lwd=2, bty="n")
```



Then we generate one sample $X$ from the distribution $g(x)$ and another following the uniform distribution $U([0,1])$. If $U \leq \frac{Mg(x)}{f(x)}$, we accept the sample X and it is then retained as a sample from $f(x)$, but if it is rejected, we generate new samples. We keep doing this until we have enough samples accepted.

The acceptance rate $A$ of the accept-reject algorithm is the following : $A = \frac{1}{M}$

With $M = \frac{1}{1-a}$, we get that $A = 1 - a$. If we compute $A$ with the $a$ given in the exercise, we get :

```r
A<-1-0.2
cat("A=", A,"\n" )
```

```
## A= 0.8
```

## Question 7

**Creation of the Accept Reject algorithm and computation of the acceptance rate**

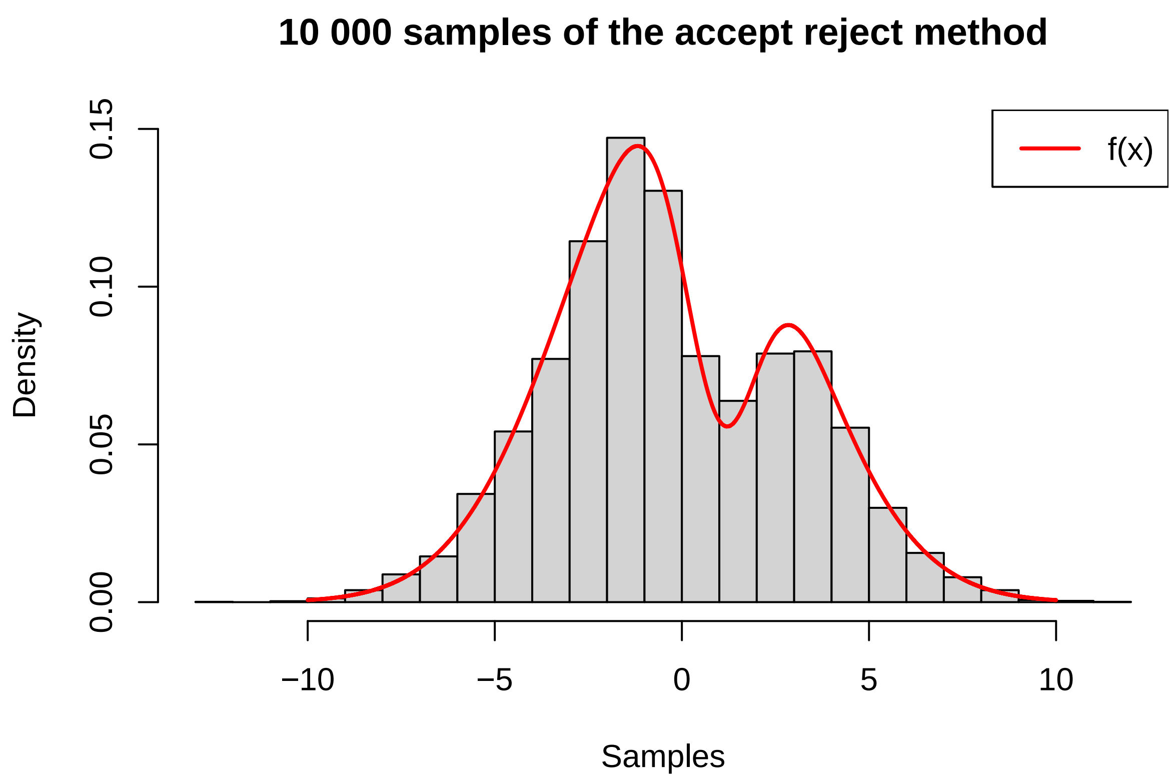
```r
accept_reject<-function(n){
  Y1<-c()
  M<-1/(1-0.2)
  nb<-0
  for (i in 1:n){
    X<-rnorm(1, 0, 3)
    U<-runif(1, min=0, max=1)
    nb<-nb+1

    while(f(0.2, 0, 1, 3, 1, X)/(M*dnorm(X, 0, 3))<U){
      nb<-nb+1
      X<-rnorm(1, 0, 3)
      U<-runif(1, 0, 1)
    }
    Y1[i]<-X
  }
  return(list(Y = Y1, total_attempts = nb))
}
```

**Generation of 10 000 samples**

```r
n<-10000
result<-accept_reject(n)
Y<-result$Y
```

**Graphically checking that the algorithm is correct**

```r
X<-seq(-10,10,0.001)
hist(Y, breaks=20, probability=TRUE, main="10 000 samples of the accept reject method",
     ylim=c(0, 0.15), xlab="Samples")
lines(X,f(0.2,0,1,3,1,X), type='l', lwd=2, col='red')
legend("topright",legend = c("f(x)"), col=c('red'), lwd=2)
```




We graphically observe that the algorithm is correct.

**Checking if the empirical acceptance rate is coherent with the theoretical one**

```
total_attempts<-result$total_attempts
cat("empirical acceptance rate:", n/total_attempts, '\n')
```

```
## empirical acceptance rate: 0.8007046
```

```
cat("theorical acceptance rate:", 1/M)
```

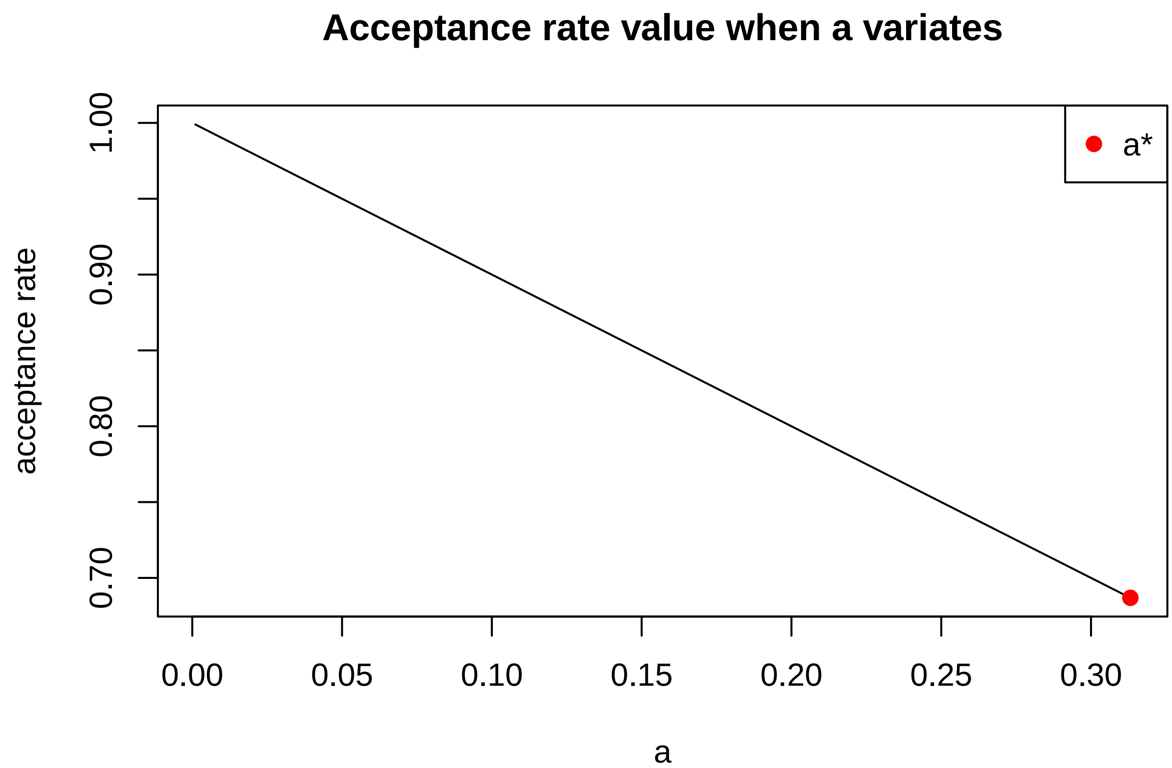
```
## theorical acceptance rate: 0.8
```

We observe that the empirical acceptance rate is very close to the theoretical acceptance rate.

## Question 8

**Plot of the acceptance rate for different values of a**

```
X<-seq(0.001, a_star(0, 1, 3, 1), 0.001)
plot(X, 1-X, type='l', xlab='a', ylab='acceptance rate', main='Acceptance rate value when a variates')
points(a_star(0, 1, 3, 1), 1-a_star(0, 1, 3, 1), col='red', pch=19)
legend("topright", legend=c('a*'), col=c('red'), pch=19)
```




So for $f$ to be non negative, a must belong to $]0, a^*]$ and when $a \to a^*$, the acceptance rate is minimized.

## Random Variable simulation with stratification

We consider a partition $P = (D_0, D_1, \dots, D_k),\ k \in \mathbb{N}$ of $\mathbb{R}$ such that $D_0$ covers the tails of $f_1$ and $f_1$ is upper bounded and $f_2$ is lower bounded in $D_1, \dots, D_k$.

We consider the following dominating function $g$ conditioned on the partition:

$$g(x) = \begin{cases} \frac{1}{Z} f_1(x) \text{ if } x \in D_0 \\ \frac{1}{Z} \left(\sup_{D_i} f_1(x) - a \inf_{D_i} f_2(x)\right) \text{ if } x \in D_i,\ i\ = 1, ..., k \end{cases}$$

where $Z$ is the normalization constant of $f$, so $Z = 1 - a$.

### Question 9

We are going to do k+2 accept-rejects, one on each $D_i,\ i = 1, \dots, k$ and 2 on $D_0$ (one on each tail).

If $D_0 = ]-\infty, \alpha] \cup [\beta, +\infty[$, we have :

$$f(x \mid x \in ]-\infty, \alpha]) = \frac{f(x)}{\int_{-\infty}^{\alpha} f(x)dx} \mathbb{1}_{]-\infty,\alpha]}(x) \leq \frac{1}{1-a} f_1(x \mid x \in ]-\infty, \alpha]) = \frac{1}{1-a} \frac{f_1(x)}{\int_{-\infty}^{\alpha} f_1(x)dx} \mathbb{1}_{]-\infty,\alpha]}(x)$$

Where:

- $f(x \mid x \in ]-\infty, \alpha])$ is the density of $f$ given $]-\infty, \alpha]$,
- $f_1(x \mid x \in ]-\infty, \alpha])$ is the density of $f_1$ given $]-\infty, \alpha]$.

We have the same on $[\beta, +\infty[$:

$$f(x \mid x \in [\beta, +\infty[) = \frac{f(x)}{\int_\beta^{+\infty} f(x)dx} \mathbb{1}_{[\beta,+\infty[}(x) \leq \frac{1}{1-a} f_1(x \mid x \in [\beta, +\infty[) = \frac{1}{1-a} \frac{f_1(x)}{\int_\beta^{+\infty} f_1(x)dx} \mathbb{1}_{[\beta,+\infty[}(x)$$

Where:

- $f(x \mid x \in [\beta, +\infty[)$ is the density of $f$ given $[\beta, +\infty[$,
- $f_1(x \mid x \in [\beta, +\infty[)$ is the density of $f_1$ given $[\beta, +\infty[$.

For each $i = 1, \dots, k$, we know that:

$$\begin{aligned} & f(x) \leq g(x) \quad \forall x \in \mathbb{R} \\ \Leftrightarrow & f\left(x \mid x \in D_i\right) \leq \frac{g\left(x \mid x \in D_i\right)}{\int_{D_i} f(x)dx} \mathbb{1}_{D_i}(x) = \frac{\sup_{D_i} f_1(x) - a \inf_{D_i} f_2(x)}{(1-a)\int_{D_i} f(x)dx} \mathbb{1}_{D_i}(x) \end{aligned}$$

We define $h_i(x)$ the density of a Uniform on $D_i$, $(\mathcal{U}(D_i))$. If $D_i = [\alpha, \beta]$, $h_i(x) = \frac{1}{\beta-\alpha}\mathbb{1}_{[\alpha,\beta]}(x)$.

And we take $M_i = \frac{g(x)}{\int_{D_i} f(x)dx} h_i^{-1}(x) = \frac{g(x)(\beta-\alpha)}{\int_\alpha^\beta f(x)dx} \in \mathbb{R}_+^*$, $x \in D_i$ , witch is constant on $D_i$.

So we still have, $f\left(x \mid x \in D_i\right) \leq M_i h_i(x) = \frac{g(x)}{\int_{D_i} f(x)dx} h_i^{-1}(x) h_i(x) = \frac{g(x)}{\int_{D_i} f(x)dx}$.

So for this accept-reject, the acceptance rate is $\frac{1}{M_i} = \frac{\int_{D_i} f(x)dx}{g(x)} h_i(x) = \frac{(1-a)\int_{D_i} f(x)dx}{(\beta-\alpha)(\sup_{D_i} f_1(x) - a \inf_{D_i} f_2(x))}$.

So the general acceptance rate is:

$$\sum_{i=1}^{k} P(X \in D_i) \frac{1}{M_i} + P(X \in D_0)(1-a)$$

We first decide what are the tails:

```
tail_inf<-dicho(0.2, 0, 1, 3, 1, 0.0001, 0.01)
tail_sup<-dicho(0.2, 0, 1, 3, 1, 0.9999, 0.01)
cat("P(X=", tail_inf, ") = 0.0001\n")
```

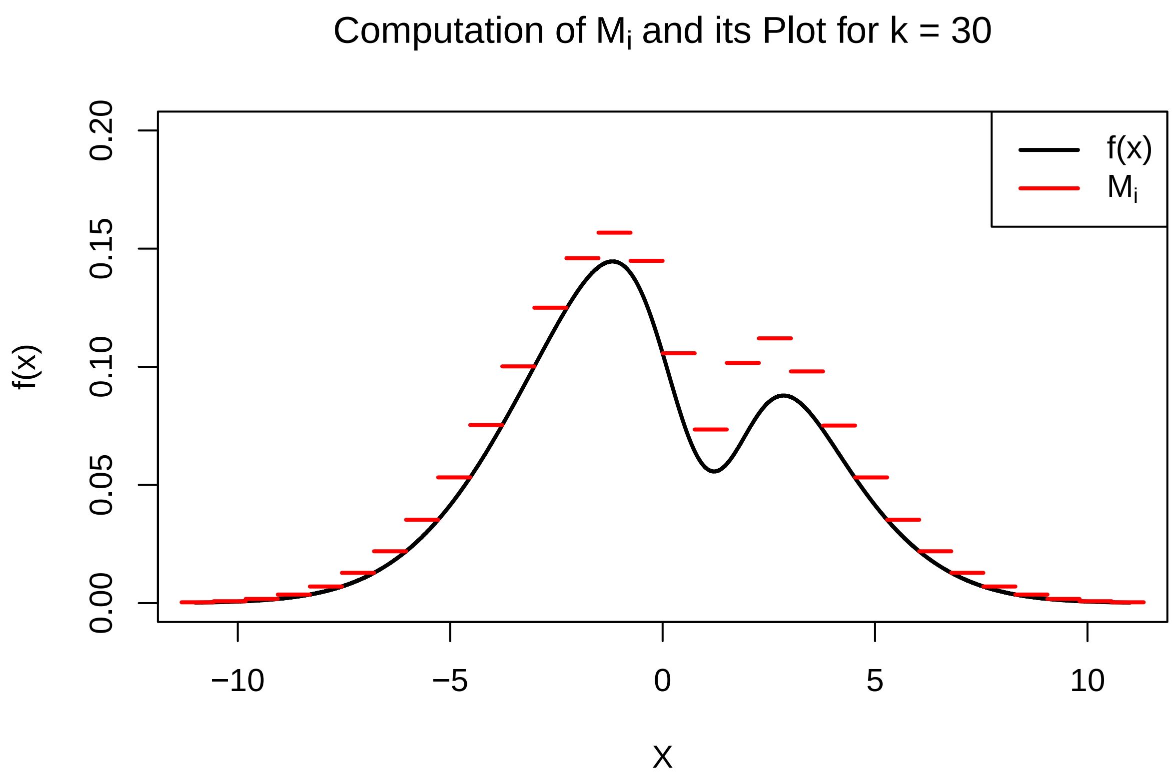
```
## P(X= -11.32245 ) = 0.0001
```

```
cat("P(X=", tail_sup, ") = 0.9999")
```

```
## P(X= 11.32281 ) = 0.9999
```

```r
f_1<-function(x){dnorm(x, 0, 3)}
f_2<-function(x){dnorm(x, 1, 1)}
```

```r
k<-30

X<-seq(-11,11,0.001)
intervals <- seq(tail_inf, tail_sup, length.out = k + 1)
plot(X, f(0.2,0,1,3,1,X), type='l', lwd=2, ylim = c(0, 0.20), main = bquote("Computation of" ~ M[i] ~ "

for (i in 1:(length(intervals) - 1)) {
  # Calculer M_i pour l'intervalle [intervals[i], intervals[i+1]]
  max_f_1 <- optimize(f_1, interval = c(intervals[i], intervals[i + 1]), maximum = TRUE)$objective
  min_f_2 <- optimize(f_2, interval = c(intervals[i], intervals[i + 1]), maximum = FALSE)$objective
  M_i <- ((max_f_1 - 0.2 * min_f_2) / (1 - 0.2))

  # Tracer une ligne horizontale pour M_i
  segments(x0 = intervals[i], y0 = M_i, x1 = intervals[i + 1], y1 = M_i, col = "red", lwd = 2)
}

legend("topright", legend = c("f(x)", expression(M[i])), col = c("black", "red"), lwd = 2)
```




## Question 10

The biggest is $n_\delta$ the biggest will the acceptance rate be.

If we show that $R(n)$, the function that compute $n_\delta$ is increasing and tends to 1 when $n \longrightarrow +\infty$, the we prove that for all $\delta \in [0, 1]$ the exists a $n$ such that $R(n) > \delta$.

## Question 11

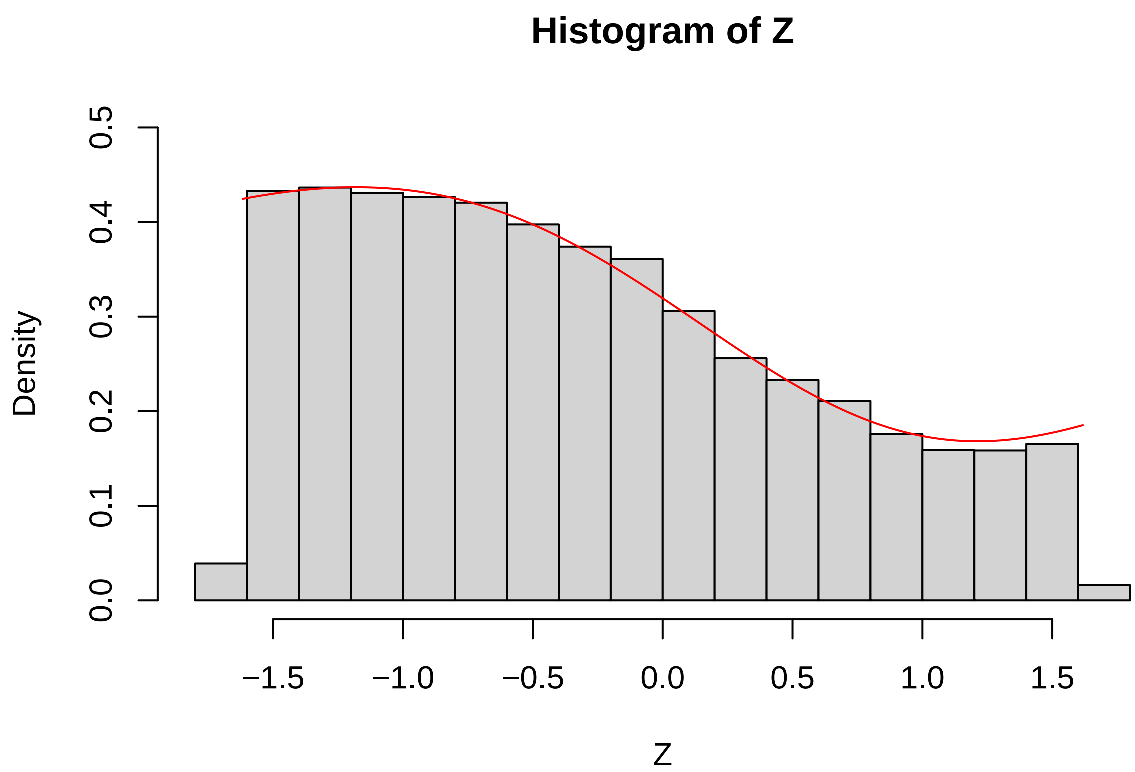
**Creation of a function that will do accept-reject on each $D_i$**

```
f_now<-function(x){f(0.2, 0, 1, 3, 1, x)}

accept_reject_stratified<-function(n, i, intervals){
  # We want to generate observation on f normalized on D_i
  f_normalized<-function(x){f(0.2,0,1,3,1,x)/integrate(f_now, intervals[i], intervals[i+1])$value}

  Y1<-c()

  # We take M = M_i/(beta - alpha) to simplify
  max_f_1 <- optimize(f_1, interval = c(intervals[i], intervals[i + 1]), maximum = TRUE)$objective
  min_f_2 <- optimize(f_2, interval = c(intervals[i], intervals[i + 1]), maximum = FALSE)$objective
  M<-(max_f_1 - 0.2 * min_f_2) / ((1 - 0.2)*integrate(f_now, intervals[i], intervals[i+1])$value)

  # We count the number of try
  nb<-0

  for (j in 1:n){
    # X follows a Uniform on D_i
    X<-runif(1, intervals[i], intervals[i+1])

    U<-runif(1, min=0, max=1)
    nb<-nb+1

    # We test F(X)/M<U because M = M_i*(beta - alpha)
    while(f_normalized(X)/M<U){
      nb<-nb+1
      X<-runif(1, intervals[i], intervals[i+1])
      U<-runif(1, 0, 1)
    }
    Y1[j]<-X
  }
  return(list(Y = Y1, M_i = M*(intervals[i+1]-intervals[i]), total_attempts = nb))
}
```

We test our accept_reject_stratified:

```
k<-7
i<-4

intervals <- seq(tail_inf, tail_sup, length.out = k + 1)
f_normalized_example<-function(x){f(0.2,0,1,3,1,x)/integrate(f_now, intervals[i], intervals[i+1])$value

Z<-accept_reject_stratified(10000, i, intervals)$Y
hist(Z, probability = TRUE, ylim = c(0, 0.5), breaks = 20)
```

```
X<-seq(intervals[i],intervals[i+1],0.001)
lines(X, f_normalized_example(X), type = "l", col= 'red')
```




We visually see that it works so we use it to do an accept-reject on $\mathbb{R}$:

```
k<-10
stratified<-function(n){
  Y<-c()
  # Counter of the number of attempts
  nb<-0
  # List of all the M_i
  M<-c()

  # Theoretical acceptance rate
  N<-c()

  # Generation of a random partition:
  #cut_points <- sort(runif(k-1, min = tail_inf, max = tail_sup))
  #intervals <- c(tail_inf, cut_points, tail_sup)

  # Generation of a partition with equals parts:
  intervals <- seq(tail_inf, tail_sup, length.out = k + 1)

  for (i in 1:k){
    P<-F(0.2, 0, 1, 3, 1, intervals[i+1])-F(0.2, 0, 1, 3, 1, intervals[i])
    n_i <- round(n*P)
```

```r
  result<-accept_reject_stratified(n_i,i, intervals)

  Y<-c(Y, result$Y)
  M <- c(M, result$M_i)
  nb <- nb + result$total_attempts
  N <- c(N, P)
}

# Generation of the normalized function on the tails
f_normalized_D0_1<-function(x){f(0.2,0,1,3,1,x)/(integrate(f_now, -Inf, intervals[1])$value)}
f_normalized_D0_2<-function(x){f(0.2,0,1,3,1,x)/(integrate(f_now, intervals[k+1], Inf)$value)}

M_0 <- 1/(1-0.2)
M <- c(M, M_0, M_0)

# number of observation we are going to generate on each tail
P0_1 <- F(0.2, 0, 1, 3, 1, intervals[1])
P0_2 <- 1-F(0.2, 0, 1, 3, 1, intervals[k+1])
N <- c(N, P0_1, P0_2)

n0_1 <- round(P0_1*n) # Usually 1
n0_2 <- round(P0_2*n) # Usually 1

if (length(Y)%%2 != 0){ # If the number of observation is odd ...
  if (F(0.2, 0, 1, 3, 1, intervals[1])<(1-F(0.2, 0, 1, 3, 1, intervals[k+1]))){
    n0_2 <- n0_2 + 1 # ... we add an observation on the one that has the biggest probability
  } else {
    n0_1 <- n0_1 + 1
  }
}

# The we do an accept reject on the left tail
for (j in 1:n0_1){

  # Generation of an X on the left tail
  repeat {
  X <- rnorm(1, 0, 3)
  if (X <= intervals[1]) {
    break
    }
  }

  U <- runif(1, 0, 1)
  nb <- nb + 1

  while (f_normalized_D0_1(X)/M_0*(dnorm(X, 0, 3)/pnorm(intervals[1], 0, 3))<U) {
    nb <- nb + 1

    repeat {
    X <- rnorm(1, 0, 3)
    if (X <= intervals[1]) {
      break
      }
```

```
      }

      U <- runif(1, 0, 1)
    }
    Y<-c(Y, X)
  }

  # Eccept reject on the right tail
  for (j in 1:n0_2){

    # Generation of an X on the raight tail
    repeat {
    X <- rnorm(1, 0, 3)
    if (X >= intervals[k+1]) {
      break
      }
    }

    U <- runif(1, 0, 1)
    nb <- nb + 1

    while (f_normalized_D0_1(X)/M_0*(dnorm(X, 0, 3)/(1-pnorm(intervals[k+1], 0, 3)))<U) {
      nb <- nb + 1

      repeat {
      X <- rnorm(1, 0, 3)
      if (X >= intervals[k+1]) {
        break
        }
      }

      U <- runif(1, 0, 1)
    }
    Y<-c(Y, X)
  }

  return(list(Y=Y, M=M, total_attempts=nb, N = N))
}
```

```
n<-10000
result<-stratified(n)
Y_strat<-result$Y
M<-result$M
total_attempts<-result$total_attempt
N <- result$N
```

### Computation of the acceptance rates

```
empirical_acceptance_rate<-length(Y)/total_attempts
cat("Empirical acceptance rate: ", empirical_acceptance_rate, "\n")
```

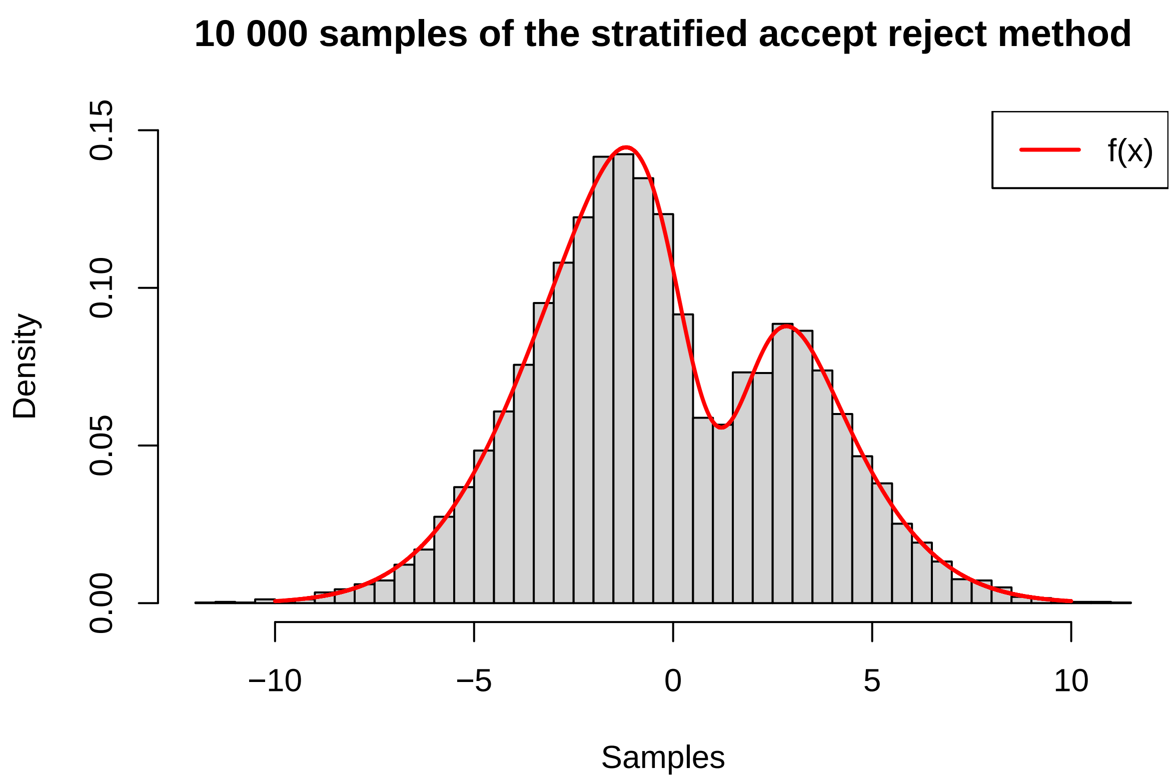
```
## Empirical acceptance rate:  0.6598482
```

```r
theoretical_acceptance_rate<-sum((1/M)*N)
cat("Theoritecal acceptance rate: ", theoretical_acceptance_rate, "\n")
```

```
## Theoritecal acceptance rate:  0.6736327
```

**Graphically checking that it's correct**

```r
X<-seq(-10,10,0.001)
hist(Y_strat, breaks=40, probability=TRUE, main="10 000 samples of the stratified accept reject method"
     ylim=c(0, 0.15), xlab="Samples")
lines(X,f(0.2,0,1,3,1,X), type='l', lwd=2, col='red')
legend("topright",legend = c("f(x)"), col=c('red'), lwd=2)
```




### Question 12

**Computation of $n_\delta$**

```r
a <- 0.2

# We compute the acceptance rate
compute_acceptance_rate <- function(intervals) {
```

```
  R <- 0

  for (i in 1:(length(intervals) - 1)) {
    alpha <- intervals[i]
    beta <- intervals[i + 1]

    max_f_1 <- optimize(f_1, interval = c(intervals[i], intervals[i + 1]), maximum = TRUE)$objective
  min_f_2 <- optimize(f_2, interval = c(intervals[i], intervals[i + 1]), maximum = FALSE)$objective
    M_i <- (max_f_1 - 0.2 * min_f_2)*(beta - alpha) / ((1 - 0.2)*integrate(f_now, alpha, beta)$value)

    P_X_in_D_i <- F(0.2, 0, 1, 3, 1, beta)-F(0.2, 0, 1, 3, 1, alpha)
    R <- R + P_X_in_D_i / M_i
  }

  # We add the contribution of D_0
  P_X_in_D_0 <- F(0.2, 0, 1, 3, 1, intervals[1] + (1 - F(0.2, 0, 1, 3, 1, intervals[length(intervals)])
  R <- R + P_X_in_D_0 * (1 - a)

  return(R)
}


# We test for each k if the acceptance rate is better than delta
compute_n_delta <- function(delta) {
  k <- 1
  while (TRUE) {
    intervals <- seq(tail_inf, tail_sup, length.out = k + 1)
    R <- compute_acceptance_rate(intervals)
    if (R > delta) {
      break
    }
    k <- k + 1
  }
  return(k)
}
```

```
n_delta <- compute_n_delta(0.6)
cat("The number of intervals n_delta is:", n_delta)
```

```
## The number of intervals n_delta is: 8
```

**Creation of stratified(n, $\delta$)**

```
stradified_delta<-function(n, delta){
  k <- compute_n_delta(delta)
  result <- stratified(n)
  Y <- result$Y
  total_attempts <- result$total_attempts
  return(list(Y=Y, total_attempts = total_attempts, partition = seq(tail_inf, tail_sup, length.out = k+
}
```

```r
result_delta <- stradified_delta(10000, 0.6)
Y_delta <- result_delta$Y
total_attempts <- result_delta$total_attempts
partition <- result_delta$partition
```

**Check that the code is correct**

```r
cat("Empirical acceptance rate: ", length(Y_delta)/total_attempts, "\n")
```

```
## Empirical acceptance rate:  0.6544931
```

```r
cat("Partition used: ", partition)
```

```
## Partition used:  -11.32245 -8.491791 -5.661133 -2.830475 0.0001831055 2.830841 5.661499 8.492157 11.3
```

# Cumulative density function

## Question 13

The cumulative density function $F_X(x)$ is the following :

$$F_X(x) = \mathbb{P}(X \leqslant x) = \int_{-\infty}^{x} f(t)dt$$
$$= \int_{-\infty}^{x} \frac{1}{1-a}\left(f_1(t) - af_2(t)\right)dt$$
$$= \frac{1}{1-a}\left(F_1(x) - aF_2(x)\right)$$

The Monte Carlo estimator of $F_X(x)$ using $n$ random variables $(X_i)_{i=1}^{n}$ $i.i.d$ following the law of X is the following :

$$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} 1_{X_i \leq x}$$

## Question 14

### Strong consistency

An estimator $F_n$ of $F_X$ is said to be strongly consistent if, for a given $x$, $F_n(x) \xrightarrow{a.s} F_X(x)$ when $n \longrightarrow +\infty$where $a.s$ means almost surely convergence.

So, we want $\lim_{n\to+\infty} F_n(x) = F_X(x)$ almost surely.

We know that the $1_{X_i \leq x}$ are $i.i.d$ because the $X_i$ are $i.i.d$ so,

- $\mathbb{E}\left[F_n(x)\right] = \mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{\{X_i \leqslant x\}}\right] = \mathbb{E}\left[\mathbb{1}_{\{X_i \leqslant x\}}\right] = \mathbb{P}\left(X_i \leqslant x\right) = F_X(x)$

So $F_n(x)$ is an unbiased estimator of $F_X(x)$.

- $\mathrm{Var}\left(\mathbb{1}_{\mathrm{X_i} \leqslant x}\right) = F_X(x)\left(1 - F_X(x)\right)$ with $\mathrm{Var}\left(\mathbb{1}_{\mathrm{X_i} \leqslant x}\right) = \mathrm{Var}\left(\mathbb{1}_{\mathrm{X} \leqslant x}\right)$ because the $1_{X_i \leq x}$ are $i.i.d.$

$\Rightarrow \text{Var}\left(F_n(x)\right) = \frac{\text{Var}\left(\mathbb{1}_{X \leqslant x}\right)}{n} = \frac{F_X(x)(1-F_X(x))}{n}$.

We observe that $\text{Var}\left(F_n(x)\right) \longrightarrow 0$ when $n \longrightarrow +\infty$.

Then, because $F_X(x)$ is a known cumulative density function, we know that the function $1_{X_i \leq x}$ is integrable. So, by the Strong Law of Large Numbers ( SLLN ), we have :

$F_n(x) = \frac{1}{n}\sum_{i=1}^{n} 1_{X_i \leqslant x} \xrightarrow{a \cdot s} E\left[1_{X_i \leqslant x}\right] = F_X(x)$

So, $F_n(x) \xrightarrow{a \cdot s} F_X(x)$ and this proves the strong consistency of $F_n(x)$.

#### Link with Glivenko-Cantelli theorem

This theorem extends this consistency to all x simultaneously :

$\sup_{x \in \mathbb{R}} |F_n(x) - F_X(x)| \xrightarrow{\text{as.}} 0.$

So $F_n(x)$ is a good approximation of $F_X(x)$.

## Question 15

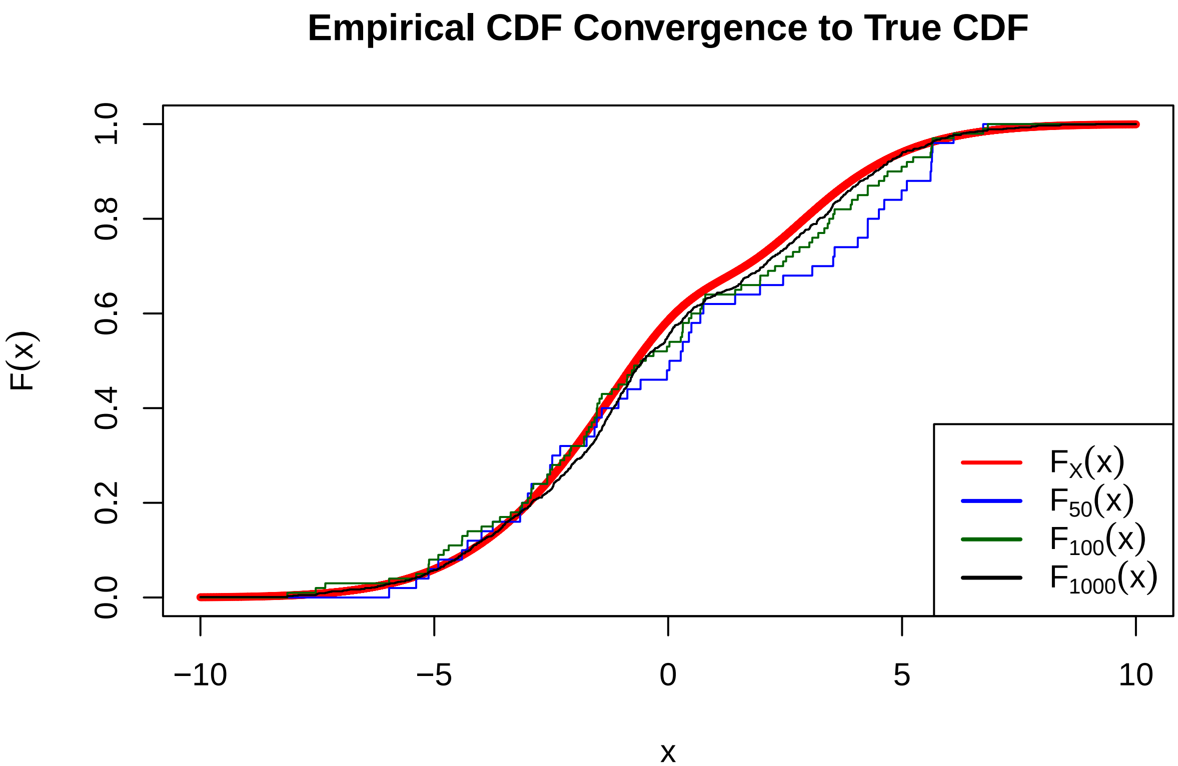
#### Estimation of the empirical cdf

```
empirical_cdf<-function(x,Xn){
  return((1/length(Xn))*sum(1*(Xn<=x)))
}
```

#### Graphical illustration of strong consistency

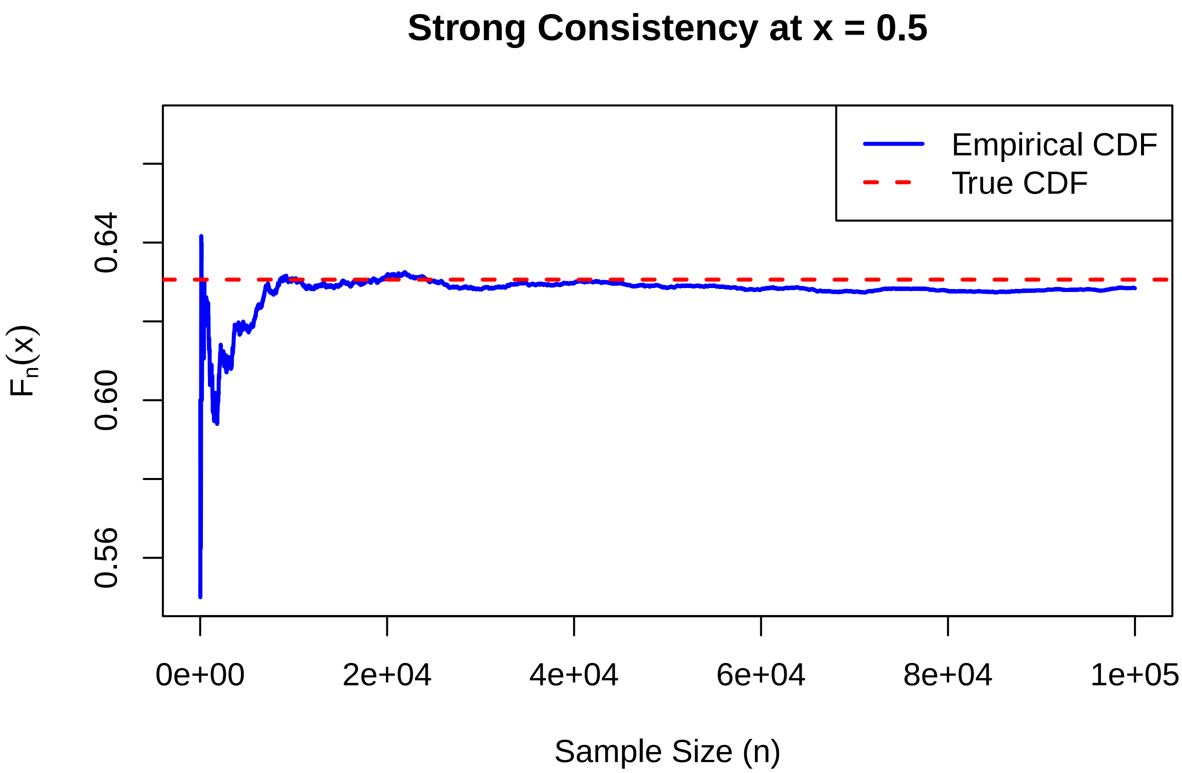
We will use to estimate $F_X(x)$ the 10 000 observations of law $F_X(x)$ we estimated before.

```
X<-seq(-10,10,0.001)
plot(X, F(0.2, 0, 1, 3, 1, X), type = 'l', col = "red", lwd = 4,
     main = "Empirical CDF Convergence to True CDF",
     xlab = "x", ylab = expression(F(x)))
lines(X,sapply(X, empirical_cdf, Xn = Y[1:50]), type='l', lwd=1, col="blue")
lines(X,sapply(X, empirical_cdf, Xn = Y[1:100]), type='l', lwd=1, col="darkgreen")
lines(X,sapply(X, empirical_cdf, Xn = Y[1:1000]), type='l', lwd=1)
legend("bottomright",legend = c(expression(F[X](x)),
                               expression(F[50](x)),
                               expression(F[100](x)),
                               expression(F[1000](x))),
       col=c('red', 'blue', 'darkgreen', 'black'), lwd=2)
```




For instance, for $x = 0.5$:

```
n_values <- seq(10,100000,10)
Y <- accept_reject(100000)$Y

x<-0.5
true_cdf <- F(0.2, 0, 1, 3, 1, x)

plot(n_values,sapply(n_values, function(n) empirical_cdf(x, Y[1:n])),type="l", col="blue", lwd=2,
     xlab="Sample Size (n)", ylab=expression(F[n](x)),
     main=paste("Strong Consistency at x =", x),
     ylim = c(0.55, 0.67))
abline(h=true_cdf, col="red", lwd=2, lty=2)
legend("topright",legend = c("Empirical CDF", "True CDF"),lty=c(1, 2), col=c('blue','red'), lwd=2)
```




## Question 16

### Central limit theorem reminder

Let $(X_i)_{i\in\mathbf{N}}$ sequence of $i.i.d$ random variables defined on $(\Omega, \mathcal{A}, \mathbb{P})$ such that $\mathbf{E}\left[\left|X_1^2\right|\right] < \infty$,

we have : $\sqrt{n}\frac{(\overline{X_n} - \mathbf{E}[X_1])}{\sqrt{\text{var}(X_1)}} \xrightarrow[n\to\infty]{\mathcal{L}} \mathcal{N}(0,1)$.

### 95% confidence interval

$\forall x \in \mathbb{R}$, the CLT tells us that :

$$\sqrt{n}\left(F_n(x) - \mathbb{E}\left[F(x)\right]\right) \overset{\mathscr{L}}{\sim} \mathcal{N}\left(0, \text{Var}\left(F(x)\right)\right.$$

$$\Leftrightarrow \sqrt{n}\left(\frac{F_n(x) - F_X(x)}{\sqrt{F_X(x)\left(1 - F_X(x)\right)}}\right) \overset{\mathscr{L}}{\sim} \mathcal{N}(0,1)$$

A 95% confidence interval is equivalent to say that $\alpha = 0.05$ and :

$$\mathbb{P}\left(q_{\frac{\alpha}{2}} \leqslant \sqrt{n}\left(\frac{F_n(x) - F_X(x)}{\sqrt{F_X(x)\left(1 - F_X(x)\right)}}\right) \leqslant q_{1-\frac{\alpha}{2}}\right) \geqslant 1 - \alpha = 0,95.$$

$$\Leftrightarrow q_{\frac{\alpha}{2}}\sqrt{\frac{F_X(x)\left(1 - F_X(x)\right)}{n}} \leqslant F_n(x) - F_X(x) \leqslant q_{1-\frac{\alpha}{2}}\sqrt{\frac{F_X(x)\left(1 - F_X(x)\right)}{n}}$$

$$\Leftrightarrow F_n(x) - q_{1-\frac{\alpha}{2}}\sqrt{\frac{F_X(x)\left(1 - F_X(x)\right)}{n}} \leqslant F_X(x) \leqslant F_n(x) - q_{\frac{\alpha}{2}}\sqrt{\frac{F_X(x)\left(1 - F_X(x)\right)}{n}}$$

with $q$ being the quantile distribution of the Normal Law $\mathcal{N}(0,1)$.

We can't keep this confidence interval because we don't know $F_X(x)$ and also because we can't make an confidence interval for $F_X(x)$ depending on itself.

We saw that $F_n(x) \xrightarrow{a \cdot s} F_X(x)$ in the previous question. So with n large enough, we can take $F_n(x)$ ( the empirical estimator of $F_X(x)$ ) instead of $F_X(x)$ because of its strong consistency and the fact that $F_n(x)$ converges asymptotically towards $F_X(x)$.

With $q_{1-\frac{\alpha}{2}} = -q_{\frac{\alpha}{2}} = 1.96$, we finally get the 95% confidence interval for $F_X(x)$:

$F_X(x) \in \left[F_n(x) - 1.96\sqrt{\frac{F_n(x)(1-F_n(x))}{n}}; F_n(x) + 1.96\sqrt{\frac{F_n(x)(1-F_n(x))}{n}}\right].$

## Question 17

Using a margin error $M = 0.01$, we know that M is the margin of error is the width of the confidence interval. So, we now have the following equality :

$M = 1.96\sqrt{\frac{F_n(x)(1-F_n(x))}{n}} \Leftrightarrow n = \frac{1.96^2\sqrt{F_n(x)(1-F_n(x))}}{M^2}.$

We can deduce the number of simulation needed to have a 95% confidence interval of $F(x)$ for $x = 1$ and $x = -15$ with these codes :

```
n<-1000000
result<-accept_reject(n)
Y<-result$Y
```

### Simulations needed for x=1

```
x <- 1
tol <- 0.01 #tolerence for the confidence interval
q_alpha <- 1.96

n <- 1000
while (TRUE) {
  F_empirical <- empirical_cdf(x, Y[1:n])
  var_empirical <- F_empirical * (1 - F_empirical)
  n_required <- trunc((q_alpha^2 * var_empirical) / tol^2)
  if (n >= n_required) break
  n <- n + 1
}
cat("number of simulation needed for x =", x, ":", n, "\n")
```

```
## number of simulation needed for x = 1 : 8527
```

### Simulations needed for x=-15

For $x = -15$, the $n_{required}$ will be very large because of the few data we have around -15.

```r
x <- -15
tol <- 0.01 #tolerence for the confidence interval
q_alpha <- 1.96

n <- 1000
while (TRUE) {
  F_empirical <- empirical_cdf(x, Y[1:n])
  var_empirical <- F_empirical * (1 - F_empirical)
  n_required <- trunc((q_alpha^2 * var_empirical) / tol^2)
  if (n >= n_required) break
  n <- n + 1
}
cat("number of simulation needed for x =", x, ":", n, "\n")
```

```
## number of simulation needed for x = -15 : 1000
```

## Empirical quantile function

### Question 18

Let $(X_i)_{i=1}^n$ be a sequence of i.i.d random variables following the same law as $X$. For any $u \in [0,1]$, we define the empirical quantile function as follow:

$$\begin{aligned} Q_n(u) :&= \inf \{x \in \mathbb{R} : u \leq F_n(x)\} \\ &= \inf \left\{ x \in \mathbb{R} : u \leq \frac{1}{n} \sum_{k=1}^{n} \mathbb{1}_{\{X_k \leq x\}} \right\} \\ &= \inf \left\{ x \in \mathbb{R} : u \times n \leq \sum_{k=1}^{n} \mathbb{1}_{\{X_k \leq x\}} \right\} \end{aligned}$$

Let us sort the $X_i$ in an ascending order, so $X_1$ will be the minimal value of $(X_i)_{i=1}^n$ and $X_n$ will be the maximal value of $(X_i)_{i=1}^n$.

This mean that $Q_n(u)$ gives us the minimal $x$ such that the $u \times n$ first value of $(X_i)_{i=1}^n$ are greater or equal to $x$. For $x$ to be minimal we must take $x = X_{\lceil u.n \rceil}$ because :

$n \cdot u \leq \lceil n \cdot u \rceil \leq \sum_{k=1}^{n} \mathbb{1}_{\{X_k \leq X_{\lceil n.u \rceil}\}} = \lceil n.u \rceil + \sum_{k=\lceil n.u \rceil + 1}^{n} \mathbb{1}_{\{X_k = X_{\lceil n.u \rceil}\}}$

Also, if we have take $x = X_{\lceil u.n \rceil - 1}$, we would either have:

- $\sum_{k=1}^{n} \mathbb{1}_{\{X_k \leq X_{\lceil n.u \rceil - 1}\}} = \lceil n.u \rceil - 1 < \lceil n.u \rceil$ if $X_{\lceil u.n \rceil - 1} < X_{\lceil u.n \rceil}$, so $x \neq X_{\lceil u.n \rceil - 1}$
- or $\sum_{k=1}^{n} \mathbb{1}_{\{X_k \leq X_{\lceil n.u \rceil - 1}\}} = \lceil n.u \rceil + \sum_{k=\lceil n.u \rceil + 1}^{n} \mathbb{1}_{\{X_k = X_{\lceil n.u \rceil}\}}$ if $X_{\lceil u.n \rceil - 1} = X_{\lceil u.n \rceil}$, so $x = X_{\lceil u.n \rceil - 1} = X_{\lceil u.n \rceil}$

### Question 19

$Y_{j,n} = \mathbb{1}_{X_j < Q(u) + \frac{t}{\sqrt{n}} \frac{\sqrt{u(1-u)}}{f(Q(u))}}$

$\sqrt{n}\,(Q_n(u) - Q(u)) \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \frac{u(u-u)}{f(Q(u))^2}\right)$

## Question 20

When $u \longrightarrow 0$ and $u \longrightarrow 1$, the function $u \mapsto f(Q(u))^2$ converges faster then $u \mapsto u(1-u)$. So $\frac{u(1 \cdot u)}{f(Q(u))^2} \to +\infty$ when $u \to 0$ and $u \to 1$.

It means that the empirical estimation function becomes extremely imprecise at the extreme ($u \longrightarrow 0$ and $u \longrightarrow 1$).

## Question 21

### Function that returns the empirical quantile

```
empirical_quantile<-function(u, Xn){
  if (u>0 & u<1){
   Xn_sorted<-sort(Xn)
   return(Xn_sorted[ceiling(u*length(Xn))])
  }
  if (u==0){return(min(Xn))}
  if (u==1){return(max(Xn))}
}
```

### Checking our intuition when $u$ is close to 0 and 1

```
result<-accept_reject(100000)
Y<-result$Y

U<-c(0.0000001, 0.5, 0.75, 0.9999999)
n1<-100
n2<-1000
n3<-10000

for (u in U){
  print("Real value:")
  cat(paste0("$Q(", u, ") = ", dicho(0.2, 0, 1, 3, 1, u, 0.00001), "$ \n \n"))

  print("Value estimaded:")
  cat(paste0("for n = ", n1, ": Q_n(", u, ") = ", empirical_quantile(u, Y[1:n1]), "\n"))
  cat(paste0("for n = ", n2, ": Q_n(", u, ") = ", empirical_quantile(u, Y[1:n2]), " \n"))
  cat(paste0("for n = ", n3, ": Q_n(", u, ") = ", empirical_quantile(u, Y[1:n3]), "\n"))
  cat(paste0("for n = ", length(Y), ": Q_n(", u, ") = ", empirical_quantile(u, Y), " \n\n\n"))
}
```

```
## [1] "Real value:"
## $Q(1e-07) = -15.721993625164$
##
## [1] "Value estimaded:"
## for n = 100: Q_n(1e-07) = -7.63413191948367
## for n = 1000: Q_n(1e-07) = -9.23419555688131
## for n = 10000: Q_n(1e-07) = -11.4610780531903
## for n = 100000: Q_n(1e-07) = -13.7234383095457
```

```
##
##
## [1] "Real value:"
## $Q(0.5) = -0.689485251903534$
##
## [1] "Value estimaded:"
## for n = 100: Q_n(0.5) = -0.491773762311945
## for n = 1000: Q_n(0.5) = -0.826367620703573
## for n = 10000: Q_n(0.5) = -0.653305349809663
## for n = 100000: Q_n(0.5) = -0.693899455891235
##
##
## [1] "Real value:"
## $Q(0.75) = 2.33498078584671$
##
## [1] "Value estimaded:"
## for n = 100: Q_n(0.75) = 2.44198559792549
## for n = 1000: Q_n(0.75) = 2.16330582344195
## for n = 10000: Q_n(0.75) = 2.34520197370203
## for n = 100000: Q_n(0.75) = 2.33081517510161
##
##
## [1] "Real value:"
## $Q(0.9999999) = 15.7219879031181$
##
## [1] "Value estimaded:"
## for n = 100: Q_n(0.9999999) = 7.4434068875251
## for n = 1000: Q_n(0.9999999) = 11.3551006341291
## for n = 10000: Q_n(0.9999999) = 11.3551006341291
## for n = 100000: Q_n(0.9999999) = 14.1946221050258
```

We observe that the empirical quantile function struggles to estimate values near 0 and 1 when it easily estimates other values.

## Question 22

By question 19, we know that :

$\sqrt{n}\left(Q_n(u) - Q(u)\right) \xrightarrow{\mathcal{L}} \mathcal{N}\left(0, \dfrac{u(1-u)}{f(Q(u))^2}\right),$

So, for a given quantile $Q(u)$, the Central Limit Theorem gives us the following 95% confidence interval :

$Q(u) \in \left[Q_n(u) - 1.96\frac{\sqrt{u(1-u)}}{\sqrt{n}f(Q(u))}; Q_n(u) + 1.96\frac{\sqrt{u(1-u)}}{\sqrt{n}f(Q(u))}\right].$

But we deduced by the Glivenko-Cantelli theorem the almost surely convergence of $Q_n(u) \xrightarrow{\text{a.s.}} Q(u)$. So with n large enough, we can take $Q_n(u)$ ( the empirical estimator of $Q(u)$ ) instead of $Q(u)$ because of its strong consistency and the fact that $Q_n(u)$ converges asymptotically towards $Q(u)$.

We finally get the 95% confidence interval for $Q(u)$:

$Q(u) \in \left[Q_n(u) - 1.96\frac{\sqrt{u(1-u)}}{\sqrt{n}f(Q_n(u))}; Q_n(u) + 1.96\frac{\sqrt{u(1-u)}}{\sqrt{n}f(Q_n(u))}\right].$

Using a margin error $M = 0.01$, we know that M is he margin of error is the width of the confidence interval. So, we now have the following equality :

$M = 1.96\frac{\sqrt{u(1-u)}}{\sqrt{n}f(Q_n(u))} \Leftrightarrow n = \frac{1.96^2 u(1-u)}{M^2[f(Q_n(u))]^2}$.

We can deduce the number of simulation needed to have a 95% confidence interval of $Q(u)$ for $u \in \{0.5, 0.9, 0.99, 0.999, 0.9999\}$ with this code :

```
required_length <- function(u_values, pdf, q_function, M, a, m_1, m_2, s_1, s_2, Y) {
  n_values <- numeric(length(u_values))

  for (i in 1:5) {
    u <- u_values[i]
    q_u <- q_function(u, Y)
    f_q_u <- pdf(a, m_1, m_2, s_1, s_2, q_u)
    required_n <- (1.96^2 * u * (1 - u)) / (M^2 * (f_q_u)^2)
    n_values[i] <- ceiling(required_n) # ceiling is to have a n integer if it is decimal
  }
  return(data.frame(u = u_values, n_required = n_values))
}
u_values <- c(0.5, 0.9, 0.99, 0.999, 0.9999)
M <- 0.01

results <- required_length(u_values, f, empirical_quantile, M, 0.2, 0, 1, 3, 1, Y)
print(results)
```

```
##        u n_required
## 1 0.5000     503712
## 2 0.9000     916649
## 3 0.9900    4450287
## 4 0.9990   23191771
## 5 0.9999  231729759
```

# Quantile estimation Naïve Reject algorithm

## Question 23

We want to stimulate a random variable $X$ conditional to the event $\{X \in A\},\ A \subset \mathbb{R}$.

Instead of doing an accept-reject on $f(x \mid x \in A)$, we will do an accept-reject on $f(x),\ x \in \mathbb{R}$ and the chose only the observations $X$ such that $X \geq q$.

On the conditions we chose we know that:

$$f(x) = \frac{1}{1-a}\left(f_1(x) - a\ f_2(x)\right) \leq \frac{1}{1-a}f_1(x),\ \forall x \in \mathbb{R}$$

Then we take $M = \frac{1}{1-a}$ and $g(x) = f_1(x)$.

We will first generate a random variable $Y \sim f_1$ and $U \sim \mathcal{U}(A)$. Then if $\frac{f(Y)}{Mf_1(Y)} \geq U$, we test if $Y \geq q$. If this two inequality are satisfied, we chose $Y$ to simulate an observation of $f(x)$. Otherwise we generate others $Y$ and $U$ until the inequality are satisfied.

## Question 24

We want to stimulate $\forall q \in \mathbb{R},\ \delta = \mathbb{P}(X \geq q)$, ie, $\delta = \mathbb{P}(X \in [q, +\infty[) = \mathbb{E}[\mathbb{1}_{X \in [q,+\infty[}]$.

So a Monte Carlo estimator of $\delta$ is $\hat{\delta}_n^{Reject} = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{X_i \in [q,+\infty[}$.

We will use the previous question algorithm with $A = [q, +\infty[$ . Then $\delta$ correspond to the empirical acceptance rate of this algorithm.

```r
accept_reject_quantile <- function(q,n){
  Y<-c()
  M<-1/(1-0.2)
  nb <- 0

  for (i in 1:n){
    X<-rnorm(1, 0, 3)
    U<-runif(1, min=0, max=1)

      # We check if the first solution is satisfied
      while(f(0.2, 0, 1, 3, 1, X)/(M*dnorm(X, 0, 3))<=U){
        X<-rnorm(1, 0, 3)
        U<-runif(1, 0, 1)
      }

    # Then if the second solution is satified we add 1 to 1 to our counting variable
    if (X >= q){
          nb <- nb+1
    }

    Y[i]<-X
    }


  return(nb/n)
}
```

Then we test this with different values of q:

```r
u <- 0.9
n <- 10000
q <- dicho(0.2,0,1,3,1,u,0.001) # <- Q(0.5)

delta_estimated <- accept_reject_quantile(q,n)

cat("For Q(", u, "), an estimation of delta is :", delta_estimated)
```

```
## For Q( 0.9 ), an estimation of delta is : 0.1018
```

## Question 25

Let $\hat{\delta}_n = \frac{1}{n} \sum_{i=1}^{n} \mathbb{1}_{X_i \geq q}$. Since $(X_i)_{1 \leq i \leq n}$ are i.i.d we have:

$$\mathbb{E}[\mathbb{1}_{X_1 \geq q}] = \mathbb{P}(X_1 \geq q) = \delta$$

$$\mathbb{E}[\not{\mathbb{1}}^2_{X_1\geq q}] = \mathbb{P}(X_1 \geq q) = \delta \leq 1 < +\infty$$

$$\mathbb{V}(\not{\mathbb{1}}_{X_1\geq q}) = \mathbb{E}[\not{\mathbb{1}}^2_{X_1\geq q}] - \mathbb{E}[\not{\mathbb{1}}_{X_1\geq q}]^2 = \delta - \delta^2 = \delta(1-\delta) < +\infty$$

So we can apply the Central Limit Theorem:

$$\sqrt{n}\frac{\hat{\delta}_n - \delta}{\sqrt{\delta(1-\delta)}} \xrightarrow[n\to\infty]{\mathcal{L}} \mathcal{N}(0,1)$$

When replacing $\delta$ by $\hat{\delta}_n$ in variance to compute an asymptotic confidence interval (because n is large):

$$\sqrt{n}\frac{\hat{\delta}_n - \delta}{\sqrt{\hat{\delta}_n(1-\hat{\delta}_n)}} \xrightarrow[n\to\infty]{\mathcal{L}} \mathcal{N}(0,1)$$

For $\alpha = 0.05$, we have:

$\mathbb{P}(q_{\frac{\alpha}{2}} \leq \sqrt{n}\frac{\hat{\delta}_n-\delta}{\sqrt{\hat{\delta}_n(1-\hat{\delta}_n)}} \leq q_{1-\frac{\alpha}{2}}) = 1-\alpha = 0.95 \;\Leftrightarrow\; \mathbb{P}(q_{\frac{\alpha}{2}}\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}} \leq \hat{\delta}_n - \delta \leq q_{1-\frac{\alpha}{2}}\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}}) = 1-\alpha$

$\Leftrightarrow\; \mathbb{P}(\hat{\delta}_n - q_{1-\frac{\alpha}{2}}\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}} \leq \delta \leq \hat{\delta}_n - q_{\frac{\alpha}{2}}\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}}) = 1-\alpha$

We know $q_{1-\frac{\alpha}{2}} = -q_{\frac{\alpha}{2}} = 1.96$, then:

$\Leftrightarrow\; \mathbb{P}(\hat{\delta}_n - 1.96\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}} \leq \delta \leq \hat{\delta}_n + 1.96\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}}) = 1-\alpha$

So the 95% confidence interval is:

$$\delta \in \left[\hat{\delta}_n - 1.96\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}}\ ;\ \hat{\delta}_n + 1.96\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}}\right]$$

To guarantee a precision $\epsilon$, we impose the following condition:

$$2\left(1.96\sqrt{\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{n}}\right) \leq 2\epsilon \;\Leftrightarrow\; \sqrt{\frac{n}{\hat{\delta}_n(1-\hat{\delta}_n)}} \geq \frac{1.96}{\epsilon} \;\Leftrightarrow\; \frac{n}{\hat{\delta}_n(1-\hat{\delta}_n)} \geq \left(\frac{1.96}{\epsilon}\right)^2 \;\Leftrightarrow\; n \geq \frac{1.96^2\hat{\delta}_n(1-\hat{\delta}_n)}{\epsilon^2}$$

```
nb_simulation_needed <- function(epsilon, u){
  q <- dicho(0.2,0,1,3,1,u,0.001)
  n <- 100000
  delta <- accept_reject_quantile(q, n)
  var <- delta*(1-delta)
  return(ceiling((1.96)^2*var/epsilon^2))
}

IT <- function(epsilon, u){
 q <- dicho(0.2,0,1,3,1,u,0.001)
 n <- nb_simulation_needed(epsilon, u)
 #n <- 100000
 cat("The number of simulation needed to estimate P(X>= Q(", u, ") with precision", epsilon," is :", n,
```

```r
 delta <- accept_reject_quantile(q, n)
 cat("P(X>= Q(",u,") is estimated by: ",delta, "\n")
 var <- sqrt(delta*(1-delta)/n)

 borne_inf <- delta - 1.96*var
 borne_sup <- delta + 1.96*var

 return(list(borne_inf = borne_inf, borne_sup = borne_sup))
}
```

We test it for $u = 0.95$ and $\epsilon = 0.001$:

```r
IT_test <- IT(0.001, 0.95)
```

```
## The number of simulation needed to estimate P(X>= Q( 0.95 ) with precision 0.001  is : 181992
## P(X>= Q( 0.95 ) is estimated by:  0.04962856
```

```r
borne_inf <- IT_test$borne_inf
borne_sup <- IT_test$borne_sup
cat("A confidence intervale is: [", borne_inf, ";", borne_sup, "] \n")
```

```
## A confidence intervale is: [ 0.04863076 ; 0.05062635 ]
```

# Importance sampling

## Question 26

### Sampling distribution $g(x)$

We want to select an appropriate sampling distribution $g(x)$ for importance sampling and explain why it is better than a classical Monte Carlo estimator. Importance sampling is a variance reduction technique used to improve the efficiency of Monte Carlo estimates when sampling directly from the target distribution $f(x)$ is doesn't work.

To propose $g(x)$, we must check that:

- $g(x) > 0$ wherever $f(x) > 0$, and
- $g(x)$ is easy to sample from.

So, we can choose a normal distribution $\boldsymbol{N}(\mu_0, {\sigma_0}^2)$ because of its relevance for $f(x)$ and the fact that by adjusting $\mu_0$ and ${\sigma_0}^2$, the Normal distribution can approximate $f(x)$ effectively in the regions we want.

And so we get : $g(x) = \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp\left(-\frac{(x-\mu_0)^2}{2\sigma_0^2}\right)$.

We could take $\mu_0$ as a weighted average of $\mu_1$ and $-\mu_2$. So, $\mu_0$ could be: $\mu_0 = \frac{\mu_1 - a\ \mu_2}{1-a}$

For $\sigma_0^2$, the good choice is to take a large variance to cover the spread of $f(x)$. So,

$\sigma_0^2 = \max\left(\sigma_1^2, \sigma_2^2\right)$

### Importance sampling estimator

The classical Monte Carlo estimator for $\delta = \mathbb{E}_f[h(X)]$ is : $\hat{\delta}^n = \frac{1}{n}\sum_{i=1}^n h\left(X_i\right), \quad X_i \sim f(x)$.

But Importance Sampling rewrites $\delta$ while using a sampling distribution $g(x)$ like this :

$\delta = \int h(x)f(x)dx = \int h(x)\frac{f(x)}{g(x)}g(x)dx = \mathbb{E}_g\left[h(X)\frac{f(X)}{g(X)}\right]$.

With $h$ a measurable function such as $h \in \mathbb{L}^{\nvDash}$ with respect to the measure $\succapprox$. Here $h(x) = \nVdash_{x \geq q}, \ \forall x \in \mathbb{R}$.

And the Importance Sampling estimator is :

$\hat{\delta}_{IS}^n = \frac{1}{n}\sum_{i=1}^n h\left(X_i\right)\frac{f(X_i)}{g(X_i)} = \frac{1}{n}\sum_{i=1}^n \frac{\frac{1}{1-a}(f_1(x)-af_2(x))}{\frac{1}{\sqrt{2\pi\sigma_0^2}}exp\left(-\frac{(x-\mu_0)^2}{2\sigma_0^2}\right)}\nVdash_{X_i \geq q}, \quad X_i \sim g(x)$.

### Preferable choice of estimator

We saw that the classical Monte Carlo estimator presents an error of order $\frac{\sigma}{\sqrt{n}}$, so we can reduce the error by reducing $\sigma^2$ or by raising the number of estimations $n$.

So, the Importance Sampling estimator is preened to the classical Monte Carlo estimator if the variance of the first one is smaller than the one of the second one. $i.e$ :

$\text{Var}\left[\hat{\delta}_n^{IS}\right] < \text{Var}\left[\hat{\delta}_n\right]$.

Or else, with a lot of estimations, we will have a smaller error for the Importance Sampling estimator and we will prefer it to the Monte Carlo one.

## Question 27

The probability density function of a Cauchy distribution is: $g(x) = \frac{1}{\pi\gamma\left[1+\left(\frac{x-\mu_0}{\gamma}\right)^2\right]}$,

with $\mu_0 = \frac{\mu_1 - a\mu_2}{1-a}$ as defined before in order to be centered as $f(x)$.

For $\gamma$, which is the scale parameter, we want it to capture the full spread of $f(x)$. So, we take :

$\gamma = \max\left(\sigma_1, \sigma_2\right)$.

## Question 28

### 95% Confidence Interval for $\delta$ with precision $\varepsilon$

We know that $\hat{\delta}_{IS}^n$ is an unbiased estimator since :

$x_i$ are id, so :

$$\begin{aligned}
\mathbb{E}_g\left[\hat{\delta}_n^{IS}\right] &= \mathbb{E}_g\left[\frac{1}{n}\sum_{i=1}^n \frac{f\left(X_i\right)}{g\left(X_i\right)}\nVdash_A\left(X_i\right)\right] \\
&= \mathbb{E}_g\left[\frac{f\left(X_i\right)}{g\left(X_i\right)}\nVdash_A\left(X_i\right)\right] \\
&= \int_{\mathbb{R}} \frac{f(x)}{g(x)}\nVdash_A(x)g(x)dx \\
&= \int_A f(x)dx
\end{aligned}$$

And $\int_A f(x)dx = \delta$, so :

$\mathbb{E}_g\left[\hat{\delta}_n^{IS}\right] = \delta$ .

With this result, we get that :

$\mathrm{Var}\left(\hat{\delta}_n^{IS}\right) = \frac{1}{n}\left(\mathbb{E}_g\left[\left(\frac{f(X)}{g(X)}\mathbb{1}_A(X)\right)^2\right] - \delta^2\right)$

The Central Limit Theorem tells us that :

$\sqrt{n}\left(\hat{\delta}_n^{IS} - \mathbb{E}\left[\hat{\delta}_n^{IS}\right]\right) \overset{\mathcal{L}}{\sim} \mathcal{N}\left(0, \mathrm{Var}\left(\hat{\delta}_n^{IS}\right)\right)$

So we can deduce a 95% confidence interval based on this CLT :

$\Leftrightarrow \sqrt{n}\left(\frac{\hat{\delta}_n^{IS} - \delta}{\sqrt{\mathrm{Var}(\hat{\delta}_n^{IS})}}\right) \overset{\mathcal{L}}{\sim} \mathcal{N}(0,1)$

$$\Rightarrow \mathbb{P}\left(q\frac{\alpha}{2} \leqslant \sqrt{n}\left(\frac{\hat{\delta}_n^{\mathrm{IS}} - \delta}{\sqrt{\mathrm{Var}\left(\hat{\delta}_n^{\mathrm{IS}}\right)}}\right) \leqslant q^{1-\frac{\alpha}{2}}\right) \geqslant 0,95$$

With $q_{1-\frac{\alpha}{2}} = -q_{\frac{\alpha}{2}} = 1.96$, we get that : $-1.96\sqrt{\frac{\mathrm{Var}(\hat{\delta}_n^{\mathrm{IS}})}{n}} \leqslant \hat{\delta}_n^{\mathrm{IS}} - \delta \leqslant 1.96\sqrt{\frac{\mathrm{Var}(\hat{\delta}_n^{\mathrm{IS}})}{n}}$

$\Rightarrow \delta \in \left[\hat{\delta}_n^{IS} - 1.96\sqrt{\frac{\mathrm{Var}(\hat{\delta}_n^{IS})}{n}}\ ,\ \hat{\delta}_n^{IS} + 1.96\sqrt{\frac{\mathrm{Var}(\hat{\delta}_n^{IS})}{n}}\right]$.

### Ceiling n

Now, we want to know the smaller samplings $n$ in order to have the 95% Confidence Interval. So, we want to have an $\varepsilon \geq 0$ such that :

Now, we want to have an $\varepsilon$ such that:

$$\begin{aligned}
&(\text{borne_sup} - \text{borne_inf}) \leqslant 2\varepsilon \\
&\Rightarrow \hat{\delta}_n^{\mathrm{IS}} + 1.96\sqrt{\frac{\mathrm{Var}\left(\hat{\delta}_n^{\mathrm{IS}}\right)}{n}} - \left(\hat{\delta}_n^{\mathrm{IS}} - 1.96\sqrt{\frac{\mathrm{Var}\left(\hat{\delta}_n^{\mathrm{IS}}\right)}{n}}\right) \leqslant 2\varepsilon \\
&\Leftrightarrow 2\left(1.96\sqrt{\frac{\mathrm{Var}\left(\hat{\delta}_n^{\mathrm{IS}}\right)}{n}}\right) \leqslant 2\varepsilon \\
&\Leftrightarrow 1.96^2 \times \frac{\mathrm{Var}\left(\hat{\delta}_n^{\mathrm{IS}}\right)}{n} \leqslant \varepsilon^2 \\
&\Leftrightarrow n \geqslant 1.96^2 \times \frac{\mathrm{Var}\left(\hat{\delta}_n^{\mathrm{IS}}\right)}{\varepsilon^2}
\end{aligned}$$

We will use these previous results for the confidence interval and the ceiling in the following code.

```
mu0 <- (0-0.2*1)/(1-0.2)
gamma <- max(3,1)

IS_quantile <- function(n,u){
```

```
  # Quantile of our fonction
  q <- dicho(0.2,0,1,3,1,u,0.001)

  # Generation of cauchy observations
  X<-rcauchy(n,location=mu0,scale=gamma)
  weight<-(f(0.2, 0, 1, 3, 1, X)/dcauchy(X,location=mu0,scale=gamma))*(X>=q)

  delta <- mean(weight)

  var <- var(weight)

  return(list(d =delta, var = var))
}

nb_simulation_needed_IS <- function(epsilon, u){
  q <- dicho(0.2,0,1,3,1,u,0.001)
  n <- 100000

  result <- IS_quantile(n,u)
  delta <- result$d
  var <- result$var

  return(ceiling((1.96)^2*var/epsilon^2))
}

IT_IS <- function(epsilon, u){
  n <- nb_simulation_needed_IS(epsilon, u)
  #n <- 100000
  cat("The number of simulation needed to estimate P(X>= Q(", u, ") with precision", epsilon," is :", n

  result <- IS_quantile(n,u)
  delta <- result$d
  cat("P(X>= Q(",u,") is estimated by: ",delta, "\n")

  var <- result$var

  borne_inf <- delta - 1.96*sqrt(var/n)
  borne_sup <- delta + 1.96*sqrt(var/n)

   return(list(borne_inf = borne_inf, borne_sup = borne_sup, var = var))
}
```

We test it for $u = 0.95$ and $\epsilon = 0.001$:

```
IT_IS_test <- IT_IS(0.001, 0.95)
```

```
## The number of simulation needed to estimate P(X>= Q( 0.95 ) with precision 0.001  is : 174383
## P(X>= Q( 0.95 ) is estimated by:  0.0499782
```

```
borne_inf <- IT_IS_test$borne_inf
borne_sup <- IT_IS_test$borne_sup
cat("A confidence intervale is: [", borne_inf, ";", borne_sup, "] \n")
```

```
## A confidence intervale is: [ 0.04896748 ; 0.05098892 ]
```

# Control Variate

## Question 29

### Score definition

The score is the gradient of the log-likelihood function with respect to the parameters that we are interest in.

### Partial derivative of the log-likelihood of $f(x|\theta_1, \theta_2)$

With $\theta_1 = \left(\mu_1, \sigma_1^2\right)$ and $\theta_2 = \left(\mu_2, \sigma_2^2\right)$, the log-likelihood for $f\left(x \mid \theta_1, \theta_2\right)$ is:

$\log\left(f\left(x \mid \theta_1, \theta_2\right)\right) = \log\left(\frac{1}{1-a}\left(f_1\left(x \mid \theta_1\right) - af_2\left(x \mid \theta_2\right)\right)\right)$

Also, we know that the score function is: $s_{\mu_1}(x \mid \theta) = \frac{\partial \log(f(x|\theta_1,\theta_1))}{\partial \mu_1}$, which is exactly what we want to compute and in this case, we will be interested in $\mu_1$. So, by adding the real function $f$, we get :

$s_{\mu_1}(x \mid \theta) = \frac{\partial}{\partial \mu_1}\log\left(\frac{1}{1-a}\right) + \frac{\partial}{\partial \mu_1}\log\left(f_1\left(x \mid \theta_1\right) - af_2\left(x \mid \theta_2\right)\right)$

But, $\frac{1}{1-a}$ doesn't depend on $\mu_1$, so $\frac{\partial}{\partial \mu_1}\log\left(\frac{1}{1-a}\right) = 0$. And,

$s_{\mu_1}(x \mid \theta) = \frac{\partial}{\partial \mu_1}\log\left(f_1\left(x \mid \theta_1\right) - af_2\left(x \mid \theta_2\right)\right)$

Then, we know that :

$$f_1\left(x \mid \theta_1\right) = \frac{1}{\sqrt{2\pi}\sigma_1}\exp\left\{\frac{-\left(x-\mu_1\right)^2}{2\sigma_1^2}\right\} \text{ and } f_2\left(x \mid \theta_2\right) = \frac{1}{\sqrt{2\pi}\sigma_2}\exp\left\{\frac{-\left(x-\mu_2\right)^2}{2\sigma_2^2}\right\}$$

$\Rightarrow s_{\mu_1}(x \mid \theta) = \frac{\partial}{\partial \mu_1}\log\left(\frac{1}{\sqrt{2\pi}\sigma_1}\exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\} - \frac{a}{\sqrt{2\pi}\sigma_2}\exp\left\{\frac{-(x-\mu_2)^2}{2\sigma_2^2}\right\}\right)$.

Let $g\left(x \mid \theta\right) = \frac{1}{\sqrt{2\pi}\sigma_1}\exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\} - \frac{a}{\sqrt{2\pi}\sigma_2}\exp\left\{\frac{-(x-\mu_2)^2}{2\sigma_2^2}\right\}$, and $h\left(x \mid \theta_1\right) = \frac{1}{\sqrt{2\pi}\sigma_1}\exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\}$.

$\Rightarrow s_{\mu_1}(x \mid \theta) = \frac{\partial}{\partial \mu_1}\log\left(g\left(x \mid \theta\right)\right)$. So, with the formula of the derivative of the *log*,

$\Rightarrow s_{\mu_1}(x, \theta) = \frac{\frac{\partial}{\partial \mu_1}g(x|\theta)}{g(x|\theta)}$

But, $-\frac{a}{\sqrt{2\pi}\sigma_2}\exp\left\{\frac{-(x-\mu_2)^2}{2\sigma_2^2}\right\}$ doesn't depend on $\mu_1$, so $\frac{\partial}{\partial \mu_1}g(x \mid \theta) = \frac{\partial}{\partial \mu_1}h(x \mid \theta_1)$.

$\Rightarrow s_{\mu_1}(x, \theta) = \frac{\frac{\partial}{\partial \mu_1}h(x|\theta_1)}{g(x|\theta)}$

By computation, we have : $\frac{\partial}{\partial \mu_1}h\left(x, \theta_1\right) = \frac{x-\mu_1}{\sqrt{2\pi}\sigma_1^3}\exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\}$

$$\Rightarrow s_{\mu_1}(x \mid \theta) = \frac{\frac{x-\mu_1}{\sqrt{2\pi}\sigma_1^3}\exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\}}{\frac{1}{\sqrt{2\pi}\sigma_1}\exp\left\{\frac{-(x-\mu_1)^2}{2\sigma_1^2}\right\} - \frac{a}{\sqrt{2\pi}\sigma_2}\exp\left\{\frac{-(x-\mu_2)^2}{2\sigma_2^2}\right\}} = \frac{\frac{x-\mu_1}{\sigma_1^2}f_1(x)}{(1-a)f(x)} = \frac{x-\mu_1}{(1-a)\sigma_1^2}\frac{f_1(x)}{f(x)}.$$

## Question 30

Let $h_0 : \mathbb{R}^d \to \mathbb{R}$ such that $\mathbb{E}\left[h_0(X)\right] = m$ and $\mathrm{Var}\left(h_0(X)\right) < +\infty$, $\forall b \in \mathbb{R}$, we have:

$$\hat{\delta} n^{CV} = \frac{1}{n} \sum_{i=1}^{n} \left(h\left(X_i\right) - b\left(h_0\left(X_i\right) - m\right)\right)$$

But, the score function comes from the likelihood; and, because of the property of the maximum likelihood estimation, we know that it must be equal to zero to have an estimator of the maximum likelihood ML.

This is why the score function $s_{\mu 1}\left(x \mid \theta\right)$ satisfies the property: $\mathbb{E}[s_{\mu 1}(x \mid \theta)] = 0$.

So we can replace $h_0\left(X_i\right) - m$ by $s_{\mu 1}\left(x \mid \theta\right)$ because they both have a zero expectation, and we finally get the following Control Variate estimator :

$\hat{\delta}_n^{CV} = \frac{1}{n} \sum_{i=1}^{n} \left(h\left(X_i\right) - b\left(s_{\mu_1}(X_i \mid \theta)\right)\right) = \frac{1}{n} \sum_{i=1}^{n} \left( \nVdash_{X_i \geq q} - \frac{b(X_i - \mu_1)}{(1-a)\sigma_1^2} \frac{f_1(X_i)}{f(X_i)} \right), \; X_i \sim f.$

## Question 31

Let $Y_i = \nVdash_{X_i \geq q} - b(s_{\mu_1}(X_i|\theta))$

By the TCL, we have:

$$\sqrt{n}\left(\overline{Y}_n - \mathbb{E}[Y_1]\right) \xrightarrow[n \to \infty]{\mathcal{L}} \mathcal{N}(0, Var(Y_1))$$

By strong law of large number, we have:

$$\hat{\sigma}_n^2 = \frac{1}{n-1} \sum_{k=1}^{n} (Y_k - \overline{Y}_n)^2 \xrightarrow{\mathbb{P}} Var(Y_1)$$

And $x \longmapsto \frac{1}{x}$ is continuous on $\mathbb{R}_+^*$:

$$\sqrt{\frac{n}{\hat{\sigma}_n^2}}(\overline{Y}_n - \mathbb{E}[Y_1]) \xrightarrow[n \to \infty]{\mathcal{L}} \mathcal{N}(0,1) \;\Leftrightarrow\; \sqrt{\frac{n}{\hat{\sigma}_n^2}}(\hat{\delta}_n^{CV}(b) - \delta) \xrightarrow[n \to \infty]{\mathcal{L}} \mathcal{N}(0,1)$$

So a confidence interval of level $1 - \alpha = 0.95$ is:

$$\delta \in \left[\hat{\delta}_n^{CV}(b) - 1.96\sqrt{\frac{\hat{\sigma}_n^2}{n}} \; ; \; \hat{\delta}_n^{CV}(b) + 1.96\sqrt{\frac{\hat{\sigma}_n^2}{n}}\right]$$

Then, we have:

$$1.96\sqrt{\frac{\hat{\sigma}_n^2}{n}} \leq \epsilon \Leftrightarrow \frac{\hat{\sigma}_n^2}{n} \leq \frac{\epsilon^2}{1.96^2} \Leftrightarrow n \geq \frac{1.96^2 \hat{\sigma}_n^2}{\epsilon^2}$$

To estimate the $b^*$ that minimizes the variance we are going to use the burn-in-period: we will estimate $b^*$ on the 50 first observations and then estimate $\hat{\delta}_n$ on the other observations.

$$\hat{b}_{50}^* = \frac{\sum_{k=1}^{50}(h_0(X_k) - m)(h(X_k) - \overline{h}_{50})}{\sum_{k=1}^{50}(h_0(X_k) - m)^2} = \frac{\sum_{k=1}^{50} s_{\mu_1}(X_k|\theta)(\nVdash_{X_k \geq q} - \frac{1}{50}\sum_{i=1}^{50} \nVdash_{X_i \geq q})}{\sum_{k=1}^{50} s_{\mu_1}(X_k|\theta)^2}$$

```r
s_mu1 <- function(x){(x*dnorm(x,0, 3))/((1-a)*9*f(0.2, 0, 1, 3, 1, x))}

b_star <- function(X){
  y <- (X>=q)
  s_mu1 <- s_mu1(X)
  return(sum(s_mu1*(y - mean(y))) / sum(s_mu1^2))
}

CV_quantile <- function(u,n){
  # Quantile of our fonction
  q <- dicho(0.2,0,1,3,1,u,0.001)

  Y<-accept_reject(n)$Y

  b <- b_star(Y[1:50])

  X <- Y[51:length(Y)]

  s_mu1 <- s_mu1(X)
  weight<-(X>=q)-b*s_mu1

  delta <- mean(weight)

  var <- var(weight)

  return(list(d =delta, var = var, b_star = b))
}

nb_simulation_needed_CV <- function(epsilon, u){
  n <- 1000000

  result <- CV_quantile(u, n)
  delta <- result$d
  var <- result$var

  return(ceiling((1.96)^2*var/epsilon^2))
}


IT_CV <- function(epsilon, u){
  n <- nb_simulation_needed_CV(epsilon, u)
  cat("The number of simulation needed to estimate P(X>= Q(", u, ") with precision", epsilon," is :", n

  result <- CV_quantile(u, n)
  delta <- result$d
  cat("P(X>= Q(",u,") is estimated by: ",delta, "\n")

  var <- result$var

  borne_inf <- delta - 1.96*sqrt(var/n)
  borne_sup <- delta + 1.96*sqrt(var/n)

   return(list(borne_inf = borne_inf, borne_sup = borne_sup, var = var))
```

```
}
```

We test it for $u = 0.95$ and $\epsilon = 0.001$:

```
IT_CV_test <- IT_CV(0.001, 0.95)
```

```
## The number of simulation needed to estimate P(X>= Q( 0.95 ) with precision 0.001  is : 153004
## P(X>= Q( 0.95 ) is estimated by:  0.05027768
```

```
borne_inf <- IT_CV_test$borne_inf
borne_sup <- IT_CV_test$borne_sup
cat("A confidence intervale is: [", borne_inf, ";", borne_sup, "] \n")
```

```
## A confidence intervale is: [ 0.04925437 ; 0.05130099 ]
```

## Question 32

### Computation of the variances

For $\hat{\delta}_n^{Reject}$, since the $(X_i)_{1\leq i\leq n}$ we have that:

$$\mathbb{V}\left(\hat{\delta}_n^{Reject}\right) = \mathbb{V}\left(\frac{1}{n}\sum_{i=1}^{n} \mathbb{1}_{X_i \geq q}\right) = \frac{1}{n^2}\sum_{i=1}^{n} \mathbb{V}(\mathbb{1}_{X_i\geq q}) = \frac{1}{n}\mathbb{V}(\mathbb{1}_{X_1\geq q}) = \frac{1}{n}\left(\mathbb{E}[\mathbb{1}^2_{X_1\geq q}] - \mathbb{E}[\mathbb{1}_{X_i\geq q}]^2\right) = \frac{\delta(1-\delta)}{n}$$

So we estimate it by: $\hat{\sigma}^2 = \frac{\hat{\delta}_n^{Reject}(1-\hat{\delta}_n^{Reject})}{n}$.

For $\hat{\delta}_n^{IS}$ and $\hat{\delta}_n^{CV}$ we use the variance computed in the algorithm.

### Algorithm complexity

For the computation of $\hat{\delta}_n^{Reject}$, each iteration costs $O(1)$ and the probability of the condition of an iteration $((\mathbb{P}(\frac{f(x)}{Mg(x)} \geq u))$ is $\frac{1}{M} = 1 - a = 0.8$.Then the total complexity is $O(\frac{n}{1-a})$.

For the computation $\hat{\delta}_n^{IS}$, the complexity is $O(n)$.

For the computation of $\hat{\delta}_n^{CV}$, the complexity of sampling the $Y_i$ via an accept-reject method is $O(n)$, the complexity of computing $b^*$ is $O(1)$ since the number $l = 50$ is fixed and the complexity of computing $\delta$ is $O(n)$. Then the final complexity is $O(n)$.

### Computation cost for a required precision

For the naïve estimator, to achive a required precision $\epsilon$ we must have: $\Leftrightarrow \quad n \geq \frac{1.96^2\hat{\delta}_n(1-\hat{\delta}_n)}{\epsilon^2}$. So the computation cost is $O(\frac{\hat{\delta}_n(1-\hat{\delta}_n)}{(1-a)\epsilon^2})$.

For the importance sampling estimator, to achive a required precision $\epsilon$ we also must have: $\Leftrightarrow \quad n \geq \frac{1.96^2\hat{\delta}_n(1-\hat{\delta}_n)}{\epsilon^2}$. So the computation cost is $O(\frac{\mathbb{V}(\hat{\delta}_n)}{\epsilon^2})$.

Then for the control variate estimator, the computation cost is $O(\frac{\mathbb{V}(\hat{\delta}_n)}{\epsilon^2})$.

So we have:

```r
n<-10000
q<-dicho(0.2,0,1,3,1,0.5,0.001)
epsilon <- 0.001

delta1 <- accept_reject_quantile(q, n)
var1 <- round(delta1*(1-delta1)/n, 7)
var2 <- round(IT_IS(0.001, 0.95)$var, 3)
```

```
## The number of simulation needed to estimate P(X>= Q( 0.95 ) with precision 0.001  is : 174253
## P(X>= Q( 0.95 ) is estimated by:  0.05064764
```

```r
var3 <- round(IT_CV(0.001, 0.95)$var, 3)
```

```
## The number of simulation needed to estimate P(X>= Q( 0.95 ) with precision 0.001  is : 459102
## P(X>= Q( 0.95 ) is estimated by:  0.04885747
```

```r
n1 <- round(delta1*(1-delta1)/(1-0.2)*epsilon^2, 7)
n2 <- round(var2/epsilon^2, 3)
n3 <- round(var3/epsilon^2, 3)
```

```r
library(knitr)

comparison_table <- data.frame(
  Aspect = c("Complexity", "Variance", "Cost for precision", "Ease of implementation"),
  `Naïve` = c("O(10 000/1-a)", var1, paste("O(",n1,")"), "Simple"),
  `Importance sampling` = c("O(10 000)", var2, paste("O(",n2,")"), "Moderate because of g(x)"),
  `Control Variate` = c("O(10 000)", var3, paste("O(",n3,")"), "complexe because of b_star")
)

# Create the table using kable
kable(comparison_table, caption = "Comparison of Efficiency between the 3 methodss")
```

Table 1: Comparison of Efficiency between the 3 methodss

| Aspect | Naïve | Importance.sampling | Control.Variate |
|---|---|---|---|
| Complexity | O(10 000/1-a) | O(10 000) | O(10 000) |
| Variance | 2.5e-05 | 0.047 | 0.114 |
| Cost for precision | O( 3e-07 ) | O( 47000 ) | O( 114000 ) |
| Ease of implementation | Simple | Moderate because of g(x) | complexe because of b_star |