TP 3 : Estimation bayésienne.

```
suppressMessages(library(tidyverse))
```

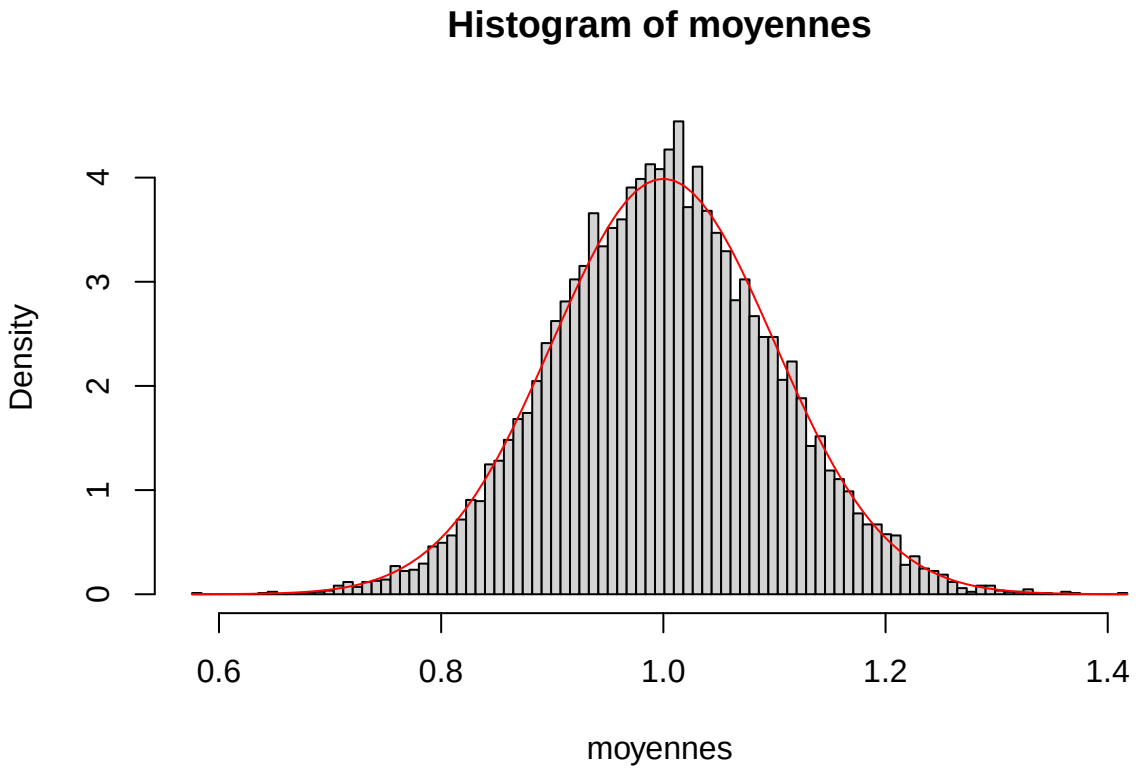
1 Échantillons gaussiens de variance connue

1.1 Statistique classique

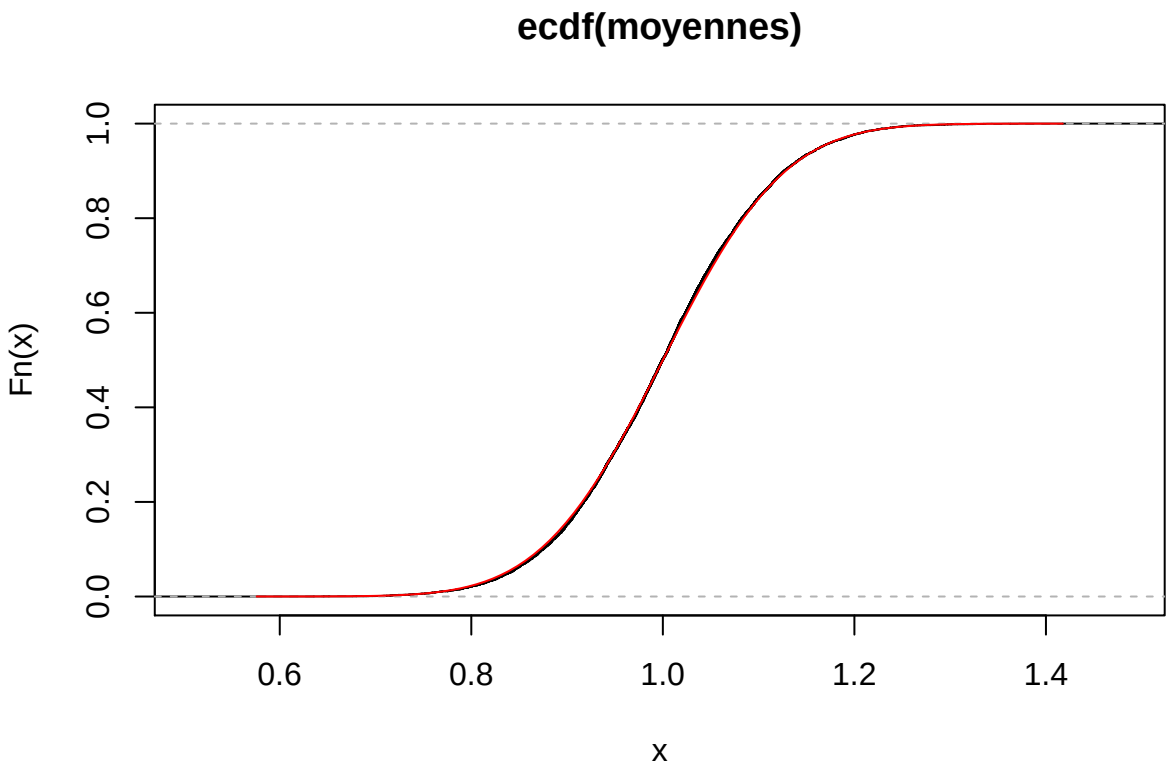
On s'intéresse ici à un echantillon $X = X_{1:n} \sim \mathcal{N}(\mu, \sigma^2)^{\otimes n}$, où l'on suppose la variance σ^2 connue. Le paramètre d'intérêt est donc $\theta = \mu$. Sous les hypothèses qui précèdent, la moyenne $\bar{X}_n$ de l'échantillon suit la loi gaussienne $\mathcal{N}(\mu, \sigma^2/n)$.

1. En utilisant une boucle `for`, simuler `Nsim` jeux de données suivant le modèle d'échantillon gaussien, et calculer les `Nsim` moyennes de chacun de ces échantillons. On enregistrera ces moyennes dans le vecteur `moyennes`. Notez qu'il est inutile d'enregistrer les échantillons.
2. Tracer un histogramme de la distribution des moyennes des `Nsim` jeux de données simulés. Superposer à cet histogramme la densité de la loi de l'estimateur, rappelée ci-dessus. Expliquez ce que vous observez.
3. Tracer la fonction de répartition empirique des moyennes des `Nsim` jeux de données. Superposer à cette fonction la fonction de répartition de la loi $\mathcal{N}(\mu, \sigma^2/n)$. Expliquez ce que vous observez.
4. Tracer la fonction quantile empirique des moyennes des `Nsim` jeux de données, en fonction des quantiles de la loi $\mathcal{N}(0, 1)$. Expliquez ce que vous observez.

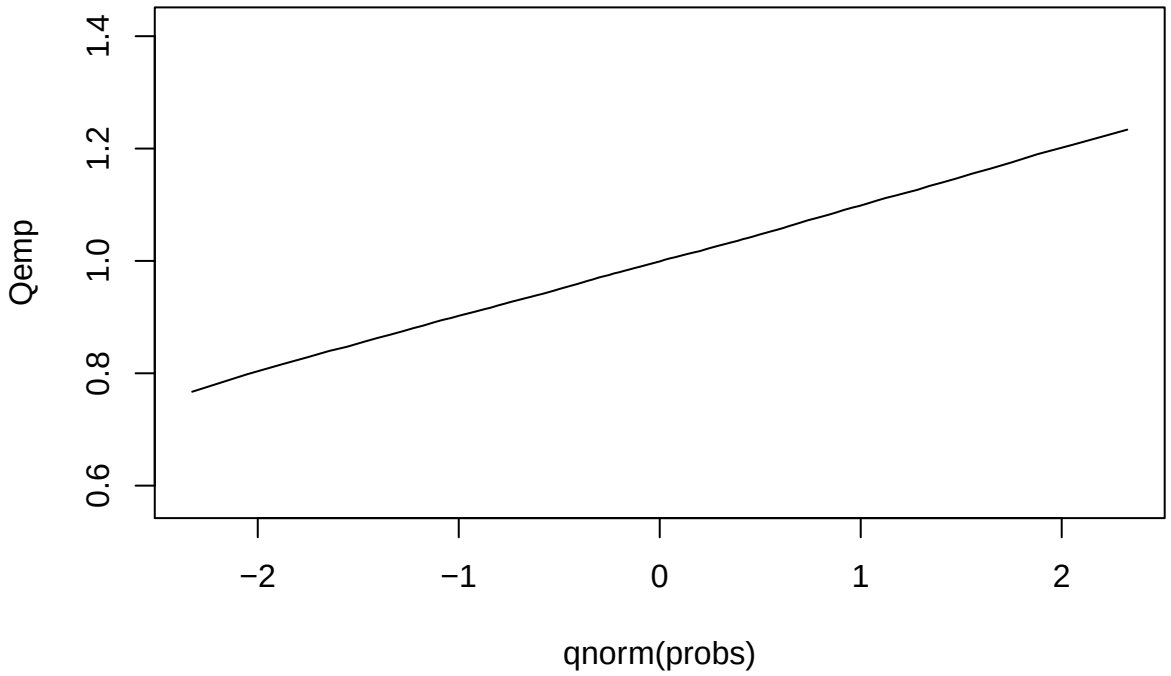
```
# 1.
n0 <- 100
mu0 <- 1
sigma0 <- 1
Nsim <- 10000
moyennes <- rep(0, times = Nsim)
for (i in 1:Nsim) {
  y <- rnorm(n0,mu0,sigma0)
  moyennes[i] <- mean(y)
}
# 2.
x.grid <- seq(from = min(moyennes), to = max(moyennes), length.out = 100)
hist(moyennes, x.grid, probability = TRUE)
lines(x.grid, dnorm(x.grid,mu0,sigma0/sqrt(n0)), col = 'red')
```




```r
# 3.
Femp <- ecdf(moyennes)
plot(Femp, do.points = FALSE)
lines(x.grid,pnorm(x.grid,mu0,sigma0/sqrt(n0)), col = 'red')
```



```r
# 4.
probs <- seq(0,1,0.01)
Qemp <- quantile(moyennes, probs)
plot(qnorm(probs),Qemp,type='l')
```



5. On rappelle que l'erreur quadratique moyenne d'un estimateur $\hat{\theta}$ d'un paramètre θ est $EQM(\theta) = \mathbb{E}_\theta(\|\hat{\theta} - \theta\|^2)$. Montrer que cette erreur vaut σ^2/n quelque soit θ. Proposer une méthode de Monte Carlo qui permet d'estimer cette erreur quadratique moyenne en $\mu = 1$ en utilisant les simulations réalisées plus haut.

Par la loi des grands nombres,

$$EQM(\mu) = \mathbb{E}_\theta(\|\bar{X}_n - \mu\|^2) \sim \frac{1}{Nsim} \sum_{i=1}^{Nsim} (\bar{X}_n^{(i)} - \mu)^2$$

où $\bar{X}_n^{(i)}$ désigne la moyenne empirique de l'échantillon numéro i.

```r
SE <- mean((moyennes - mu0)^2)
print(c('Estimation du risque :', as.character(SE)))
```

```
## [1] "Estimation du risque :" "0.00978093150407023"
```

```r
print(c('Risque :', as.character(sigma0^2/n0)))
```

```
## [1] "Risque :" "0.01"
```

```r
print(c('erreur relative :',
        as.character(abs(SE - sigma0^2/n0)/(sigma0^2/n0))))
```

```
## [1] "erreur relative :"  "0.0219068495929769"
```

1.2 Statistique bayésienne

On suppose maintenant que la loi a priori, et le modèle statistique sont donnés par

$$\mu \sim \mathcal{N}(m, \tau^2), \quad X_{1:n} \mid \mu \sim \mathcal{N}(\mu, \sigma_0^2)^{\otimes n}.$$

1. Montrer que la loi a posteriori est donnée par $\mu \mid X_{1:n} = x_{1:n} \sim \mathcal{N}(\hat{\mu}(x_{1:n}), v^2)$, où

$$\frac{1}{v^2} = \frac{1}{\tau^2} + \frac{n}{\sigma_0^2}, \quad \hat{\mu}(x_{1:n}) = \frac{\frac{m}{\tau^2} + \frac{n\bar{x}_n}{\sigma_0^2}}{\frac{1}{\tau^2} + \frac{n}{\sigma_0^2}}$$

On s'intéresse à un seul jeu de données dans cette section.

2. Simuler un jeu de données de taille $n_0 = 30$ et de moyenne μ de votre choix, et l'enregistrer dans la variable `jdd_gauss`.

```
n0 <- 30
mu0 <- 5
sigma0 <- 1
jdd_gauss <- rnorm(n0,mu0,sigma0)
```

3. On suppose que l'on sait a priori que la valeur de μ est autour de 0, et on fixe donc m à 0. On suppose aussi que l'on sait a priori qu'il y a 95% de chance que la vraie valeur soit entre -10 et 10. En déduire une valeur de $\tau = \sqrt{\tau^2}$, la calculer et enregistrer le résultat dans la variable `tau`.
4. Calculer les valeurs numériques de $v = \sqrt{v^2}$ et $\hat{\mu}(x_{1:n})$ et enregistrer les résultats dans les variables `v` et `hat_mu`.

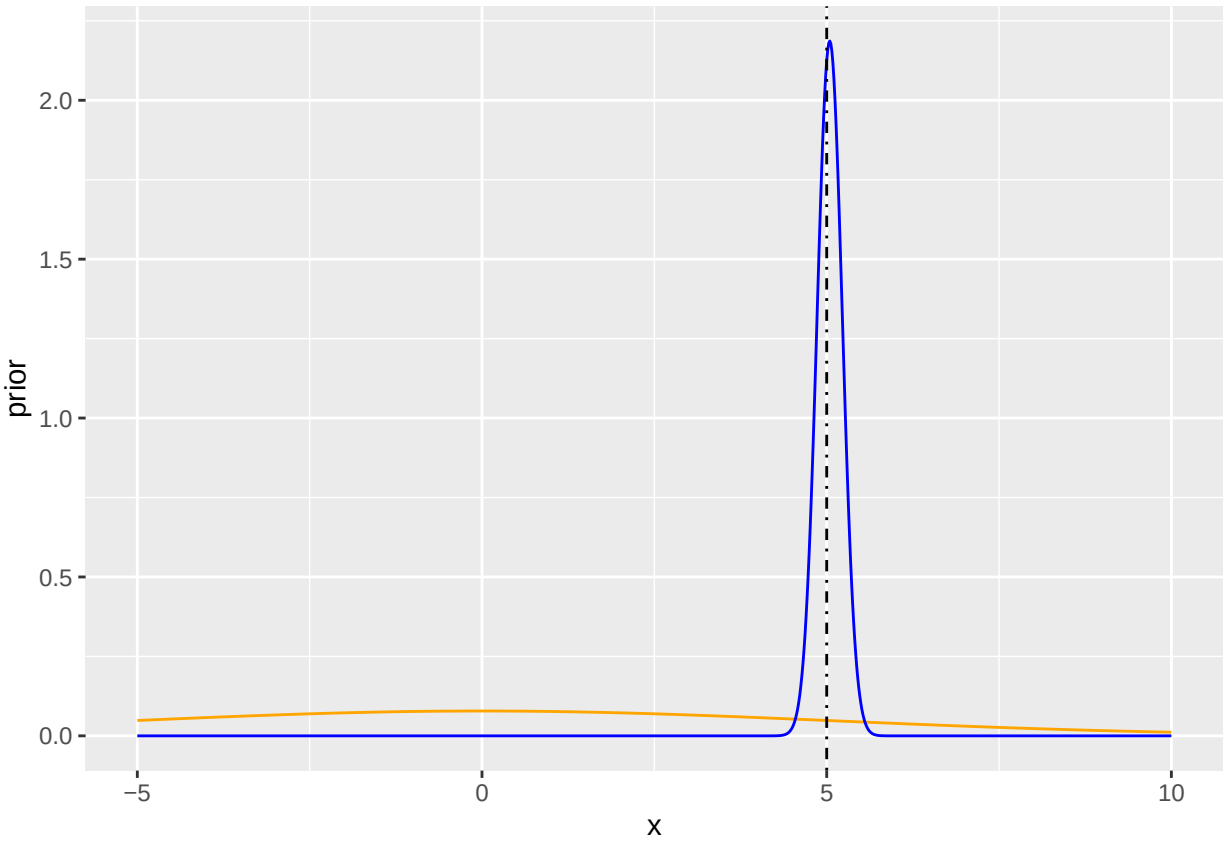
```
# 3.
tau <- 10/qnorm(0.975)
m <- 0

# 4.
v <- 1/tau^2 + n0/sigma0^2
v <- sqrt(1/v)
hat_mu <- ((sum(jdd_gauss)/sigma0^2) + (m/tau^2)) * v^2
```

5. Sur le même graphique, représenter :

- la densité de la loi a priori de μ en orange,
- la densité de la loi a posteriori en bleu,
- la vraie valeur de μ sous forme d'un trait vertical noir, en pointillé.

On pourra commencer par créer un tableau de données, dont la première colonne est une grille des valeurs de x entre -5 et $\mu + 5$.

```
x.grid <- seq(from = -5, to = mu0+5, length.out = 1e3)
comp_loi_gauss <- tibble(
  x = x.grid,
  prior = dnorm(x.grid,m,tau),
  post = dnorm(x.grid,hat_mu,v)
)
ggplot(comp_loi_gauss,aes(x = x)) +
  geom_line(aes(y = prior) , color = 'orange') +
  geom_line(aes(y = post), color = 'blue') +
  geom_vline(xintercept = mu0, linetype = 'dotdash' , color = 'black')
```



6. Pourquoi les valeurs de la densité a priori ont-elles l'air d'être si faibles ? Commenter le graphique obtenu.

La loi a priori est la loi $\mathcal{N}(0, \tau^2)$ avec un écart type τ de l'ordre de 5. Comme cet écart-type est assez élevé, la densité de la loi a priori est assez plate, et quand on zoome sur $[-5; 10]$, on ne voit que le haut de la "bosse". La loi a posteriori a elle un écart-type de l'ordre de 0.2. La densité de la loi a posteriori est donc beaucoup plus piquée autour de sa moyenne. De plus cette moyenne est proche de la vraie valeur.

7. Calculer un intervalle de crédibilité de probabilité 95% sur cet exemple. Calculer aussi un intervalle de confiance de niveau 95%. Comparer les résultats obtenus.

Un intervalle de crédibilité est un intervalle qui contient 95% de la masse de la loi a posterori.

```
Icred <- hat_mu + qnorm(0.975) * v * c(-1,1)
Iconf <- mean(jdd_gauss) + qnorm(0.975) * sigma0/sqrt(n0) * c(-1,1)
cat('Intervalle de crédibilité à 0.95:', Icred, fill = TRUE)
```

```
## Intervalle de crédibilité à 0.95: 4.68615 5.40137
```

```
lcred = Icred[2]  - Icred[1]
cat('Longueur de l intervalle de credibilite: ',lcred, fill = TRUE)
```

```
## Longueur de l intervalle de credibilite:  0.7152199
```

```
cat('Intervalle de confiance à 0.95:' , Iconf, fill = TRUE)
```

```
## Intervalle de confiance à 0.95: 4.692379 5.408057
```

```
lconf = Iconf[2]  - Iconf[1]
cat('Longueur de l intervalle de confiance: ',  lconf, fill = TRUE)
```

```
## Longueur de l intervalle de confiance:  0.7156777
```

Les deux intervalles contiennent la vraie valeur de μ. L'intervalle de crédibilité est un peu plus précis que l'intervalle de confiance.

2 Échantillon binomial

Dans cette partie, on s'intéresse à un échantillon $X_{1:n} \sim \mathcal{B}(p)^{\otimes n}$, dont le paramètre d'intérêt est $\theta = p$.

2.1 Statistique classique

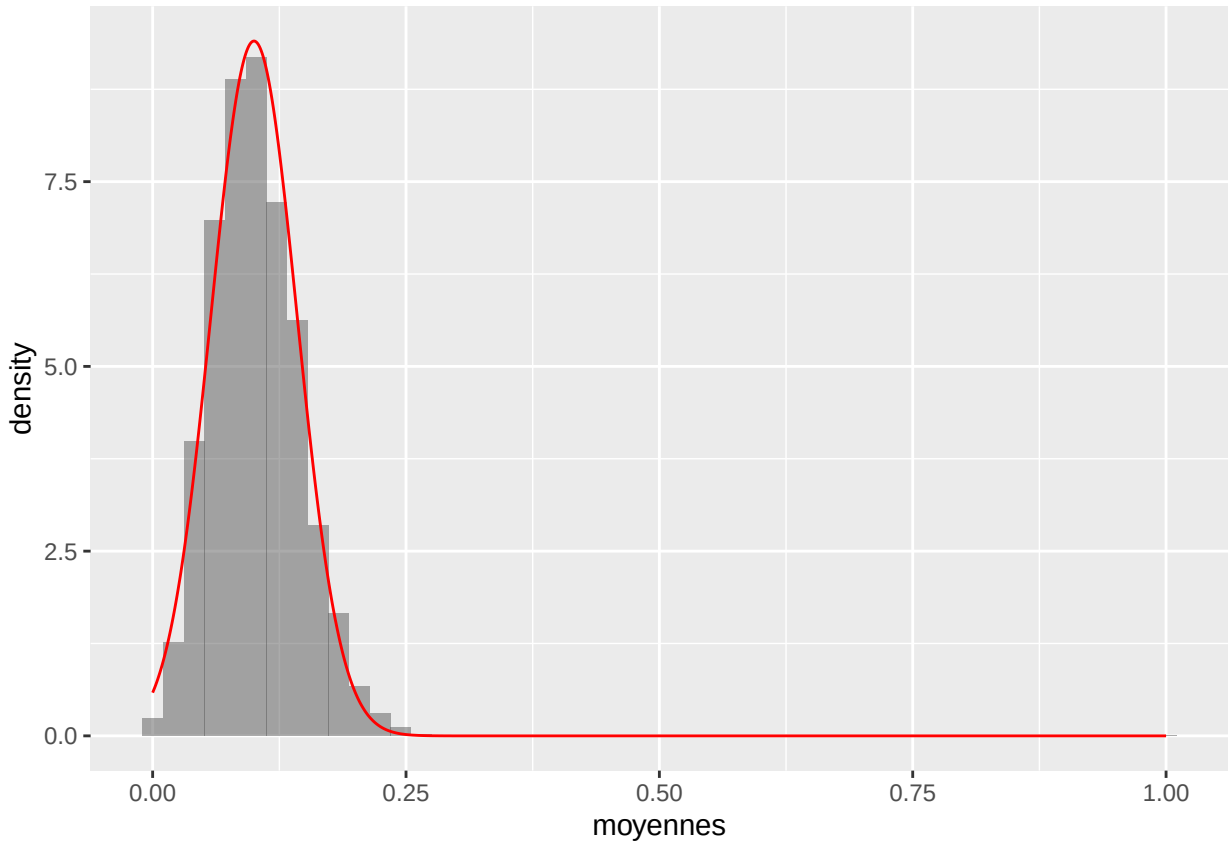
Reproduire l'étude de la partie 1.1 sur le modèle binomial, avec des échantillons de taille $n = 50$, une vraie valeur $p = 0.1$. On s'intéresse à l'estimateur $\bar{X}_n$ de p.

1. En utilisant une boucle `for`, simuler `Nsim` jeux de données suivant le modèle et calculer les moyennes de ces échantillons.

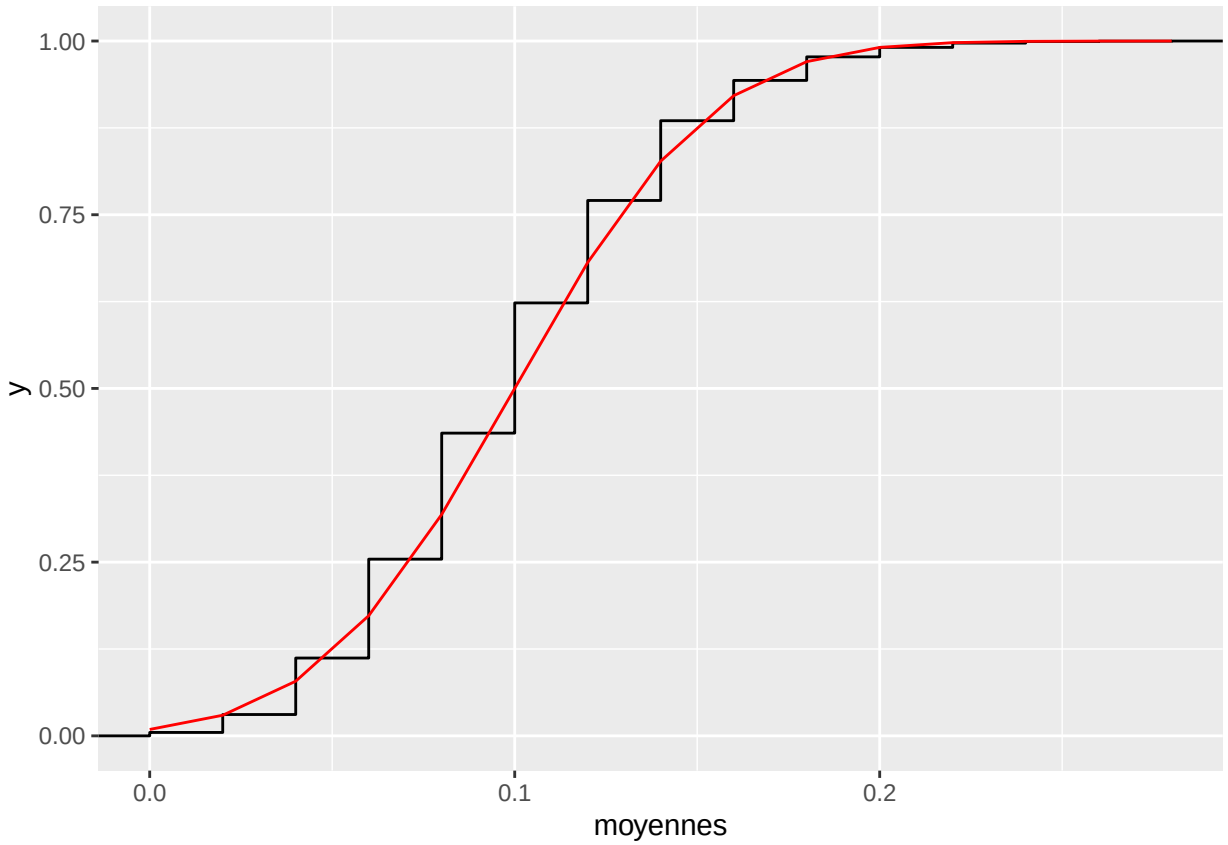
```
# 1.
n1 <- 50
p1 <- 0.1
Nsim <- 1e4
moyennes <- rep(0, times = Nsim)
for (i in 1:Nsim) {
  moyennes[i] <- rbinom(1,n1,p1)/n1
}
```

2. Tracer un histogramme de la distribution des moyennes simulées. Superposer à cet histogramme l'approximation par une loi normale de sa distribution, c'est-à-dire une loi gaussienne centrée en p et de variance $p(1-p)/n$. Pour tracer cette densité, on utilisera une grille de longueur 1000 sur les nombres entre 0 et 1.
3. Tracer la fonction de répartition empirique des moyennes des `Nsim` jeux de données. Superposer à cette fonction la fonction de répartition de la loi $\mathcal{N}(p, p(1-p)/n)$.
4. Tracer la fonction quantile empirique des moyennes des `Nsim` jeux de données, en fonction des quantiles de la loi $\mathcal{N}(0,1)$. Expliquez ce que vous observez.

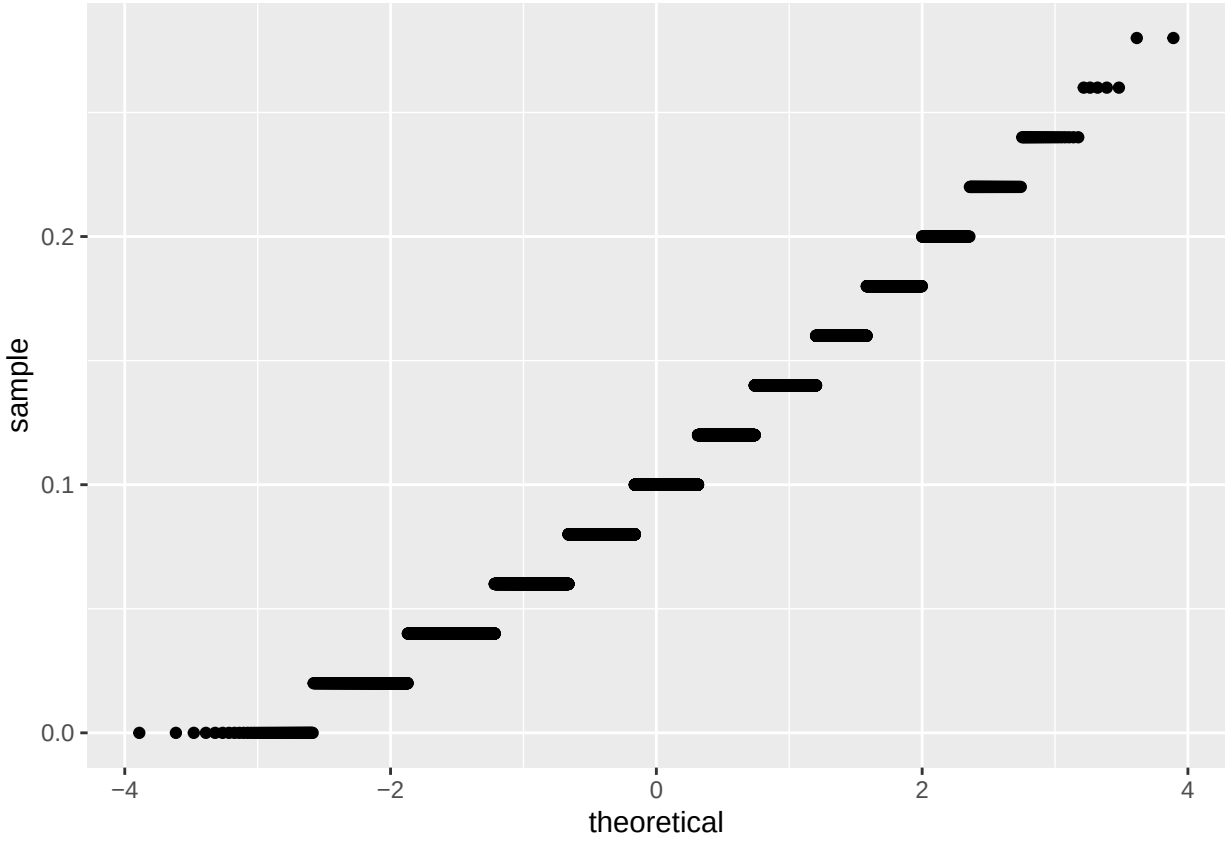
```
# 2.
p.grid <- seq(from = 0, to = 1, length.out = 1e3)
ggplot(data = NULL, mapping = aes(x = moyennes)) +
  geom_histogram(aes(y = ..density..), bins = 50, alpha = 0.5) +
  geom_line(aes(x = p.grid,
                y = dnorm(p.grid, mean = p1, sd = sqrt(p1*(1 - p1)/n1))),
            color = 'red')
```




```
#3
ggplot(data = NULL, aes(x = moyennes)) +
  stat_ecdf() +
  geom_line(aes(y = pnorm(moyennes, mean = p1, sd = sqrt(p1*(1 - p1)/n1))), color = 'red')
```




```
#4
ggplot(data = NULL, aes(sample = moyennes)) +
  stat_qq(distribution = qnorm)
```



5. Avec une méthode de Monte Carlo, approcher l'erreur quadratique moyenne de l'estimation de $p = 0.1$ par la moyenne de l'échantillon.

```
EQM <- mean((moyennes - p1)^2)
cat('Le risque quadratique est estimé à ', EQM, '. \n')
```

```
## Le risque quadratique est estimé à  0.00176508 .
```

```
cat('Sa vraie valeur est ', p1*(1 - p1)/n1, '. \n')
```

```
## Sa vraie valeur est  0.0018 .
```

2.2 Statistique bayésienne

Reproduire l'étude de la partie 1.2 sur un jeu de données de taille $n = 50$ simulé avec $p = 0.1$. On supposera que la loi a priori est la loi uniforme sur $[0; 1]$ (aucune information précise sur la postition de p...)

On rappelle que, dans ce cas, la loi a posteriori est la loi beta donnée par

$$P \mid X_{1:n} = x_{1:n} \sim \text{Beta}\left(1 + s_n, 1 + n - s_n\right), \quad \text{avec } s_n = \sum_{i=1}^{n} x_i.$$

1. Montrer que la loi a posteriori est donnée par la formule ci-dessus.
2. Simuler un jeu de données de taille $n = 50$ et l'enregistrer dans la variable `jdd_bin`.

```
n1 <- 50
p1 <- 0.1
jdd_bin <- rbinom(n1,1,p1)
```

3. Calculer les valeurs numériques de $\alpha = 1 + s_n$ et $\beta = 1 + n - s_n$.

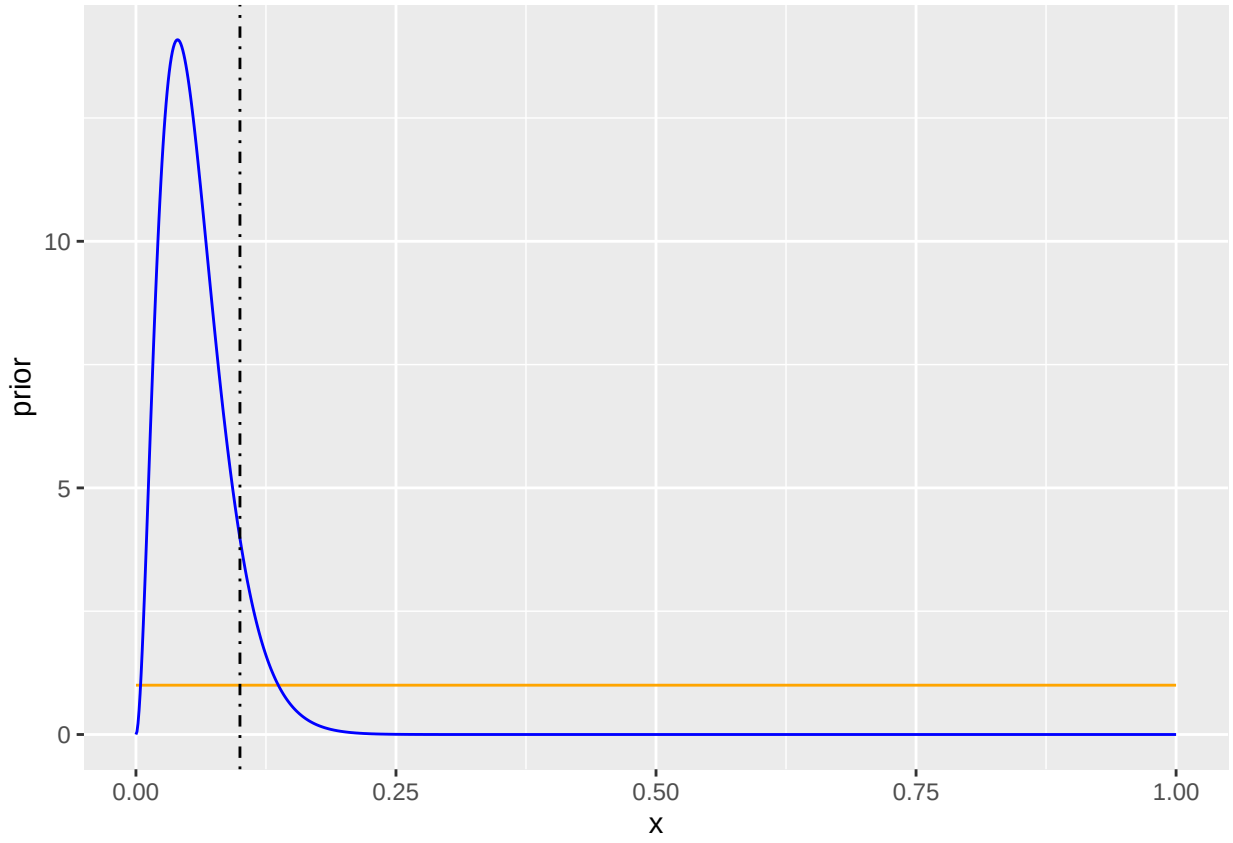
```r
alpha <- 1 + sum(jdd_bin)
beta <- n1 + 2 - alpha
```

4. Sur le même graphique, représenter :

- la densité de la loi a priori en orange,
- la densité de la loi a posteriori en bleu,
- la vraie valeur de p sous forme d'un trait vertical noir, en pointillé.

On pourra commencer par créer un tableau de données, dont la première colonnne est une grille de valeurs possibles de p entre 0 et 1.

```r
p.grid <- seq(from = 0, to = 1, length.out = 1e3)
comp_loi_bin <- tibble(
  x = p.grid,
  prior = 1,
  post = dbeta(p.grid,alpha,beta)
)
ggplot(comp_loi_bin,aes(x = x)) +
  geom_line(aes(y = prior) , color = 'orange') +
  geom_line(aes(y = post), color = 'blue') +
  geom_vline(xintercept = p1, linetype = 'dotdash' , color = 'black')
```



5. La densité a posteriori peut-elle être strictement positive pour des valeurs de p strictement négatives ou strictement plus grandes que 1 ? Commenter le graphique obtenu.

La densité a posteriori est une loi Beta, dont le support est dans [0,1]. Elle ne peut donc pas prendre des valeurs négatives, ou supérieures à 1. La densité a posteriori est beaucoup plus piquée que la loi a priori, autour d'une valeur qui n'est pas très éloignée de la vraie valeur.

6. Calculer un intervalle de crédibilité de probabilité 95 % sur cet exemple. Calculer aussi un intervalle de confiance, et comparer les résultats obtenus.

```
Icred <- c(qbeta(0.025,alpha,beta),qbeta(0.975, alpha, beta))
lcred <- Icred[2] - Icred[1]
phat <- mean(jdd_bin)
Iconf_asymp <- phat + qnorm(0.975) * sqrt(phat*(1 - phat)/n1) * c(-1,1)
lconf <- Iconf_asymp[2] - Iconf_asymp[1]
cat('Intervalle de crédibilité à 0.95: [', Icred[1], ';' , Icred[2] ,']. \n')
```

```
## Intervalle de crédibilité à 0.95: [ 0.01229909 ; 0.1345865 ].
```

```
cat('Longueur de l intervalle de crédibilité :', lcred, '. \n')
```

```
## Longueur de l intervalle de crédibilité : 0.1222874 .
```

```
cat('Intervalle de confiance à 0.95: [', Iconf_asymp[1], ';' , Iconf_asymp[2] ,']. \n')
```

```
## Intervalle de confiance à 0.95: [ -0.01431612 ; 0.09431612 ].
```

```
cat('Longueur de l intervalle de confiance :', lconf, '. \n')
```

```
## Longueur de l intervalle de confiance : 0.1086322 .
```

Les deux intervalles contiennent la vraie valeur. L'intervalle de crédibilité est inclus dans [0;1], alors que l'intervalle de confiance contient des valeurs négatives. La largeur de l'intervalle de crédibilité est plus grande que celle de l'intervalle de confiance.

3 Estimateur de James-Stein

On s'intéresse à l'estimateur du vecteur $\theta = (\theta_1, \ldots, \theta_d)$ de dimension d à partir d'une seule observation vectorielle y de dimension d. Le modèle statistique est : $Y \sim \mathcal{N}_d(\theta, \sigma^2 I_d)$. On supposera σ^2 connu égal à 1, et $d \geq 4$.

L'objectif est de comparer le risque quadratique moyen de deux estimateurs. Le premier estimateur est le maximum de vraisemblance, $\hat{\theta}_{ML} = Y$, et le second est l'estimateur de James-Stein donné par

$$\hat{\theta}_{JS} = \left(1 - \frac{(d-3)\sigma^2}{\|Y\|^2}\right)^+ Y.$$

On peut montrer que l'erreur quadratique moyenne ne dépend de la vraie valeur de θ qu'à travers $\|\theta\|$.

Proposer, et mettre en œuvre, une méthode de Monte Carlo qui permet d'approcher l'erreur quadratique moyenne en : $\theta = (u, \ldots, u)$ pour $u = 0.1$ ou 0.5, ou 1, ou 2, et $d = 10$.

```
Nsim <- 1e3
d <- 10
Y <- matrix(rnorm(n = d * Nsim), nrow = d, ncol = Nsim)

EQM <- function(u = 0.1, Y){
# Y représente une matrice d x Nsim contenant des N(0,1)
# On change la moyenne de Y.
Y <- Y + u
d <- nrow(Y)

# On calcule Nsim estimations de theta par la méthode de James-Stein
test <- (1 - (d - 3)/colSums(Y^2))
```

```r
test <- diag(test > 0)
htJS <- Y %*% test

# On estime les risques quadratiques des estimateurs James-Stein et du max de vraisemblance par Monte-C
EQMJS <- mean((htJS - u)^2)*d
EQMML <- mean((Y - u)^2)*d
return(c(mean = u, EQMJS = EQMJS, EQMML = EQMML))
}

# Comparaison graphique des deux risques.
U <- c(seq(from = 0,  to = 1, by = 0.1 ), 1.5, c(2:10))
nU <- length(U)
ResEQM <- tibble( mean = rep(NA,2*nU), EQM = rep(NA,2*nU),
                  Method = rep(c('JS','ML'),nU))
for (i in (1:nU)) {
  ResEQM$mean[c(2*i - 1,2*i)] <- c(U[i],U[i])
  ResEQM$EQM[c(2*i - 1,2*i)] <- EQM(u = U[i], Y )[2:3]
}
 ggplot(data = ResEQM, aes(x = mean, y = EQM, color = Method)) +
   geom_line() +
   labs(title = 'Comparaison des risque quadratiques', x = 'Mean', y = 'EQM')
```