TP5

02/05/2022

#Question1

On simule un échantillon de 10 lois normales de moyenne 2 et d'écart-type 1

```
##  [1] 0.5182262 2.8872795 1.0751511 3.6607793 2.9069363 3.1280586 0.3123855
##  [8] 4.5613361 2.3405595 3.3196414
```

Soit un échantillon iid de gaussiennes, on a :

$$lnL_n(x_1, ..., x_n, \theta) = -nln(\sigma) - n\frac{ln(2\pi)}{2} - \frac{1}{2\sigma^2}\sum(xi - m)^2$$

On trouve alors résolvant les équations de ln L (dérivée nulle):

$$\hat{\mu} = x\bar{n}$$

$$\hat{\sigma^2} = \frac{1}{n-1}\sum(xi - x\bar{n})^2$$

```
## [1] 2.471035
```

```
## [1] 1.970964
```

On a

$$\frac{\sqrt{n}(\hat{\mu} - \mu)}{\sqrt{\hat{\sigma^2}}} \sim Student(n-1)$$

D'où, avec

$$\alpha = 0.05$$

:

$$\mathbb{P}(t_{\frac{\alpha}{2}}^{n-1} \leq \frac{\sqrt{n}(\hat{\mu} - \mu)}{\sqrt{\hat{\sigma^2}}} \leq t_{1-\frac{\alpha}{2}}^{n-1}) = 1 - \alpha$$

Ainsi l'intervalle de confiance à 95% est:

$$I(\mu, n) = [\hat{\mu} - \sqrt{\frac{\hat{\sigma^2}}{n}}t_{1-\frac{\alpha}{2}}^{n-1}, \hat{\mu} - \sqrt{\frac{\hat{\sigma^2}}{n}}t_{\frac{\alpha}{2}}^{n-1}]$$

```
##     97.5%
## 0.5359621
```

```
##     2.5%
## 2.311789
```

Or

$$\hat{\theta} \sim \mathbb{N}(\theta, I_n(\theta)^{-1}) et I_n(\theta)^{-1} = -E(H_n(ln(L_n)))$$

En calculant le hessien de la log vraisemblance plus haut:

$$\begin{pmatrix} -\frac{n}{\sigma^2} & \frac{2}{\sigma^3}(n\mu - \sum X_i) \\ \frac{2}{\sigma^3}(n\mu - \sum X_i) & \frac{n}{\sigma^2} - 3\frac{\sum X_i - \mu^2}{\sigma^4}) \end{pmatrix}$$

D'où

$$I_n(\theta)^{-1} = \begin{pmatrix} \frac{2n}{\sigma^2} & 0 \\ 0 & \frac{n}{\sigma^2} \end{pmatrix}$$

Ainsi, via la régularité du modèle:

$$I_n(\theta)^{-1}(\hat{\theta} - \theta) \sim \mathbb{N}(0, I_n)$$

Grâce à la formule de probabilités totales, on peut déduire que la différence entre estimateur et paramètre de la moyenne suit une loi normale centrée réduite:

$$\frac{2n}{\sigma^2}(\hat{\mu} - \mu) \sim N(0, \frac{4n^{3/2}}{\sigma^2})$$

Donc , d'après le lemme de Slutsky, l'intervalle de confiance asymptotique est :

$$I(\mu, n) = [\hat{\mu} - \frac{\sqrt{n}}{\hat{\sigma}}\mu_{1-\frac{\alpha}{2}}, \hat{\mu} - \frac{\sqrt{n}}{\hat{\sigma}}\mu_{\frac{\alpha}{2}}]$$

#Question2

On applique notre règle de décision pour faire le décompte

L'inverse de la fonction de répartition de la loi de Student associée est décroissante

```
##           [,1]
## [1,] 0.8609506
## [2,] 1.3277282
## [3,] 1.7291328
## [4,] 2.5394832
```

```
##      [,1] [,2]
## [1,]    0  100
## [2,]    0  100
## [3,]    0  100
## [4,]    8   92
```

Les résultats sont cohérents

#Question3

```
log_norm = function(mu, std, X) {
  n = length(X)
  return(-n*log(std)+0.5*n*log(2*pi)-(1/(2*std**2))*sum((X-mu)**2))
}
```

##TP de l'année dernière = mêmes questions dans un autre ordre (correspond à la suite des questions à partir de Q4 / ordre décalé vers Q12_ici -> Q9_sujet cette année) ## Maximum de vraisemblance pour plusieurs paramètres

Question 1

Pour simuler un tel échantillon, on a :

```
echantWeibull <- rweibull(10,2,3)
```

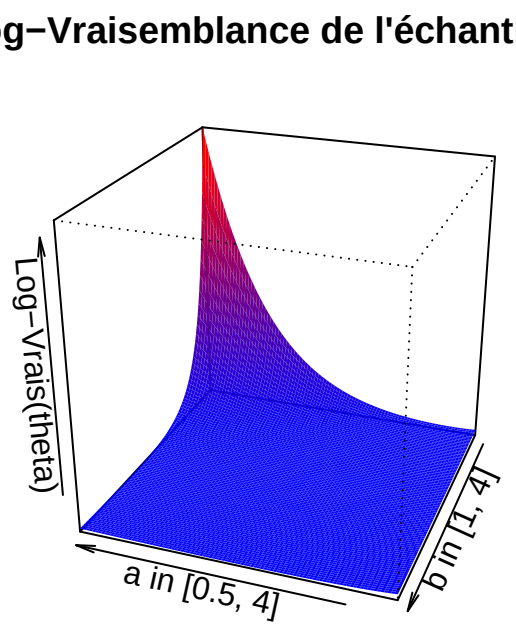
Nous avons implémenté la fonction demandée de la manière suivante :

```
nlogL_Weibull <- function(theta, x) {
  y <- log(dweibull(x,theta[1],theta[2]))
  s <- sum(y)
  return (-s)
}
```

Où `theta[1]` et `theta[2]` sont respectivement a et b dans $\theta = (a, b)$.

Question 2

Afin de faire une telle simulation, nous faisons varier a et b respectivement dans $[0.5, 4]$ et $[1, 4]$. Les résultats sont résumés dans le graphe 3D suivant, représentant la log-vraisemblance de l'échantillon précédemment simulé en fonction des variations de a et b :



Nous voyons que plus le couple (a, b) tend vers $(4, 1)$, plus le log-vraisemblance semble augmenter. Bien sûr, on remarque qu'aux 3 autres coins du graphe il y a une légère augmentation du log-vraisemblance.

On regarde alors pour quelles valeurs de a et b on a le minimum de vraisemblance :

```
## [1] "On a alors a =  2.55050505050505  et b =  3.15151515151515"
```

Ce qui souligne les résultats représentés sur le graphe.

Question 3

On utilise alors la méthode suivante afin de pouvoir avoir un résultat plus précis :

```
opt <- optim(par = c(2,3), nlogL_Weibull, x = echantWeibull)
```

```
## [1] "On obtient : a =  3.00022541365179  et b =  2.75060915265349"
```

La majorité du temps, les valeurs sont semblables, cependant la faible taille de l'échantillon empêche d'avoir des résultats qui soient toujours identiques.

Gérer les contraintes sur les paramètres

Question 4

```
nlogL_Weibull <- function(theta, x) {
  y <- log(dweibull(x,theta[1],theta[2]))
  s <- sum(y)
  if (s < -10^(6)) {
    return (10^6)
  }
  return (-s)

}


opt <- optim(par = c(2,3),nlogL_Weibull, x = echantWeibull, method = "L-BFGS-B",
             lower = c(0,10^(-4)), upper = c(100,100))
print(paste("On a : a = ", opt$par[1], " et b = ", opt$par[2]))
```

```
## [1] "On a : a =  3.00022625138608  et b =  2.75071296571833"
```

Question 5

```
#fonction etudiant la sensibilite des resultats pour un echantillon de taille n
#(en etudiant 1000 echantillons différents)
sensibiliteResult <- function(n) {
  y <- c()
  for (i in 1:1000) {
    y <- append(y, optim(c(2,3), nlogL_Weibull, x = rweibull(n, 2, 3), method = "L-BFGS-B",
                         lower = c(0,10^(-4))))
  }
  a <- c()
  b <- c()
  for (i in 1:1000) {
    a <- append(a, y[i]$par[1])
    b <- append(b, y[i]$par[2])
  }
  min_a <- min(a)
  max_a <- max(a)
  min_b <- min(b)
  max_b <- max(b)

  return(c(min_a, max_a, min_b, max_b))

}
```

On observe que plus la taille de nos échantillons est élevée, plus la sensibilité de nos résultats est faible. On voit que pour n=10, nos résultats ne sont pas très fiables et peuvent être éloignés du résultat théorique.

Normalité asymptotique de l'EMV et l'intervalle de confiance

Question 6

```
## [1] "intervalle de a= [ 2.62520873036633 , 3.37524209693724 ]"
```

```
## [1] "intervalle de b= [ 2.60616919482707 , 2.8950491104799 ]"
```

La vraie valeur n'est pas toujours incluse dans ses intervalles ceci est dû à la taille trop faible des échantillons, pour de faibles valeurs de n notre approximation n'est pas valable.

Question 7

Afin d'arriver à ce but, nous avons utilisé la méthode qui suit :

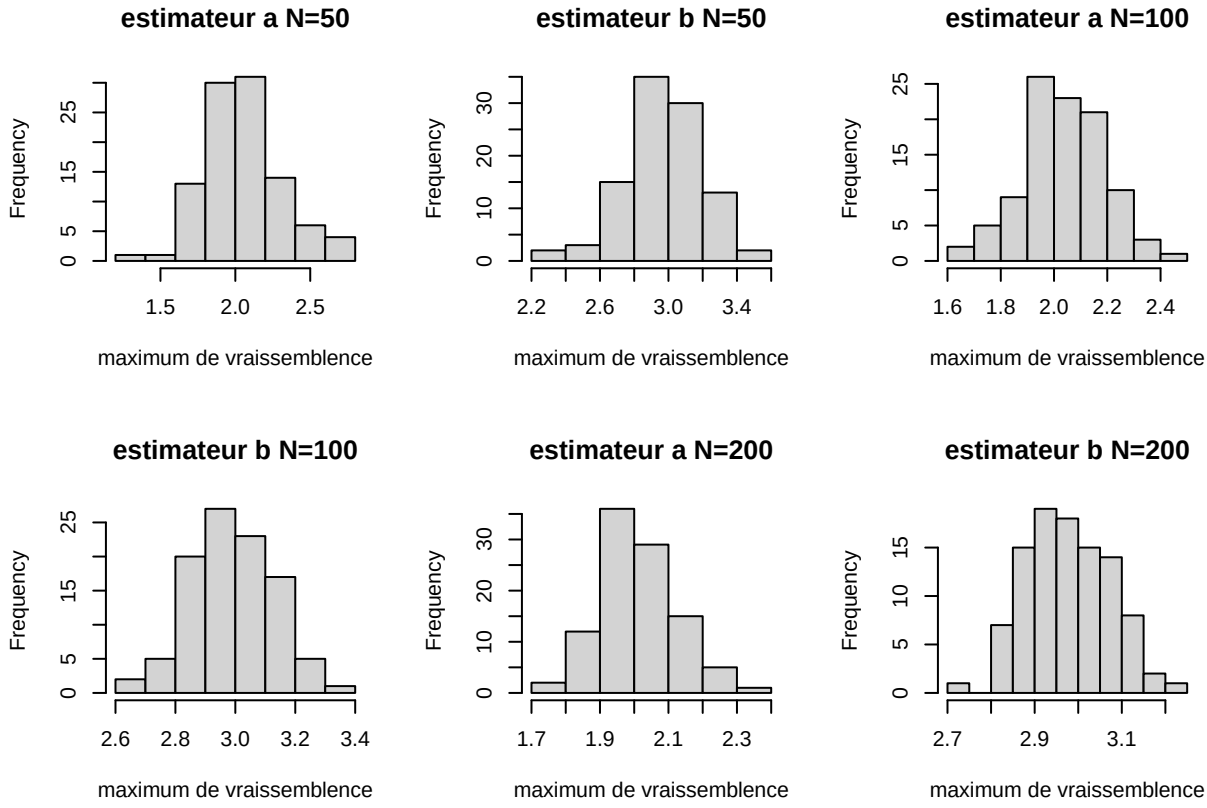
```
couverture_a <- intervalle_a
couverture_b <- intervalle_b
for(i in 2:100){
  echantWeibull <- rweibull(10,2,3)
  hes<-optim(par=c(2,3),nlogL_Weibull, x = echantWeibull, hessian=TRUE)


  matcov <- solve(hes$hessian)
  vara <- matcov[1]
  varb <- matcov[4]
  aexp <- hes$par[1]
  bexp <- hes$par[2]
  min_a <- min(c(aexp-(0.475*sqrt(vara)), aexp+(0.475*sqrt(vara)), couverture_a[1]))
  min_b <- min(c(bexp-(0.475*sqrt(varb)), bexp+(0.475*sqrt(varb)), couverture_b[1]))
  max_a <- max(c(aexp+(0.475*sqrt(vara)), aexp+(0.475*sqrt(vara)), couverture_a[2]))
  max_b <- max(c(bexp+(0.475*sqrt(varb)), bexp+(0.475*sqrt(varb)), couverture_b[2]))
  couverture_a <- c(min_a,max_a)
  couverture_b <- c(min_b,max_b)
}
```

Après application de la méthode précédente, nous obtenons les résultats suivants :

```
## [1] "couverture de a= [ 1.13526265392661 , 4.46494369636447 ]"
```

```
## [1] "couverture de b= [ 1.55560896873225 , 4.51575255543231 ]"
```

Question 8



Lorsque n tend vers la l'infinie la loi théorique des estimateurs a et b vaut: $\mathcal{N}(\begin{pmatrix} a \\ b \end{pmatrix}, V)$ où V est la matrice de covariance de la loi Weibull pour les paramètre a et b.

```
## [1] "N=50 couverture de a= [ 1.29715893524353 , 2.84383013073088 ]"
## [1] "N= 50 couverture de b= [ 2.1576884568588 , 3.67011273984604 ]"
## [1] "N=100 couverture de a= [ 1.60646929406668 , 2.49570632506909 ]"
## [1] "N= 100 couverture de b= [ 2.54359727732974 , 3.42513602976796 ]"
## [1] "N=200 couverture de a= [ 1.68875949286477 , 2.42384450127197 ]"
## [1] "N= 200 couverture de b= [ 2.6813997770079 , 3.25479077436117 ]"
```

Lorsque la taille de l'échantillon augmente, la taille de la couverture empirique des estimateurs diminue et les maximum de vraisemblances sont plus proche des valeurs théoriques de la loi de weibull qui sont donnés pour construire les échantillons.

Méthode delta

Question 9

Nous avons simulé un tel échantillon de la manière suivante :

```
#echantillon loi gamma
echantillonGamma <- rgamma(200,2,2)
```

Pour l'estimateur $\phi = \frac{\alpha}{\beta}$, on a implémenté :

```
#estimateur gamma
estimateur_phi_gamma <- function(x) {
  opt <- optim(par = c(2,3),nlogL_Gamma, x = echantillonGamma, method = "L-BFGS-B",
               lower = c(0,10^(-4)))
  return (g_gamma(opt$par))
}
```

Et pour son écart type, on a :

```
#ecart type de l'estimateur
ecart_type_estim_gamma <- function(x) {
  opt <- optim(par = c(2,3),nlogL_Gamma, x = echantillonGamma, method = "L-BFGS-B",
               lower = c(0,10^(-4)), hessian=TRUE)
  grad <-t(matrix(myderiv(g_gamma, opt$par),1,2))
  Io <- matrix(solve(opt$hessian),2,2)
  return (sqrt(t(grad) %*% Io %*% grad))
}
```

```
## [1] "L'intervalle de confiance à 95% de phi est : [ 1.05135373632293 1.09944347013823 ]"
```

Question10

Nous avons simulé un tel échantillon de la manière suivante :

```
echantillonCauchy<-rcauchy(2000, 10, 0.1)
```

(i) Comme $\mathbb{P}(X > 100) = \int_{100}^{\infty} f(x, x_0, \alpha)$,
on prend $g(x_0, \alpha) = \int_{100}^{\infty} f(x, x_0, \alpha)$:

```
g_cauchy <- function(theta) {
  auxiliaire <- function(x) {
    return (dcauchy(x, theta[1], theta[2]))
  }
  return (integrate(auxiliaire, lower = 100, upper = 50000)$value)
}
```

Pour l'estimateur de $\mathbb{P}(X > 100)$ on a :

```
estimateur_P <- function(y) {
  return (length(y[y>100]) / length(y))
}
```

Et, pour son écart type on a :

```
ecart_type_estim_cauchy <- function(y) {
  opt <- optim(par = c(2,3),nlogL_cauchy, x = y, method = "L-BFGS-B",
               lower = c(0,10^(-4)), hessian=TRUE)
  grad <-t(matrix(myderiv(g_cauchy, opt$par),1,2))
  Io <- matrix(solve(opt$hessian),2,2)
  return (sqrt(t(grad) %*% Io %*% grad))
}
```

```
## [1] "L'intervalle à 95% de P(X>100) vaut : [ -2.12277649346099e-05 2.12277649346099e-05 ]"
```

(ii) On a $\mathbb{P}(X \leq x) = 1 - \int_x^{\infty} f(x, x_0, \alpha)dx$
Ce qui nous donne : $Arctan(\frac{x-x_0}{\alpha}) = 0.49 * \pi$
Donc on obtient g en trouvant y tel que $Arctan(y) = 0.49 * \pi$ puis en calculant $x = y * \alpha + x_0$:

```
g <- function(x) {
  return (abs(atan(x) - 0.49*pi))
}

g_cauchy_v2 <- function(theta) {
  value <- optimize(f = g, c(0,10000), tol=.Machine$double.eps^0.5)$min
  return (theta[1] + theta[2]*value)
}
```

Pour l'estimateur de $\mathbb{P}(X \leq x)$ on a :

```
estimateur_x <- function(y) {
  opt <- optim(par = c(2,3),nlogL_cauchy, x = y, method = "L-BFGS-B",
               lower = c(0,10^(-4)), hessian=TRUE)
  return (g_cauchy_v2(opt$par))
}
```

Et, pour son écart type :

```
ecart_type_estim_cauchy_v2 <- function(y) {
  opt <- optim(par = c(2,3),nlogL_cauchy, x = y, method = "L-BFGS-B",
               lower = c(0,10^(-4)), hessian=TRUE)
  grad <-t(matrix(myderiv(g_cauchy_v2, opt$par),1,2))
  Io <- matrix(solve(opt$hessian),2,2)
  return (sqrt(t(grad) %*% Io %*% grad))
}
```

```
## [1] "L'intervalle à 95% de x tel que P(X>x) = 0.99 vaut : [ 12.9138768663346 13.2965732411049 ]"
```

Question 11

Les estimateurs de μ *et* θ valent:

```
## [1] "On a : mu =  2.25571168194956  et sigma =  1.20437788928803"
```

L'intervalle obtenu avec la méthode asymptotic est le suivant:

```
## [1] "intervalle de mu à 95% vaut : [ 1.36350871874195 , 2.85647101787515 ]"
```

L'intervalle obtenu avec la méthode bootstrap paramétriques est le suivant:

```
## [1] "L'intervalle de confiance à 95% de mu est : [ 1.57820795581837 2.83755931001519 ]"
```

L'intervalle obtenu avec la méthode non bootstrap paramétriques le suivant:

```
## [1] "L'intervalle de confiance à 95% de mu est : [ 1.50624365352005 2.88399613458994 ]"
```

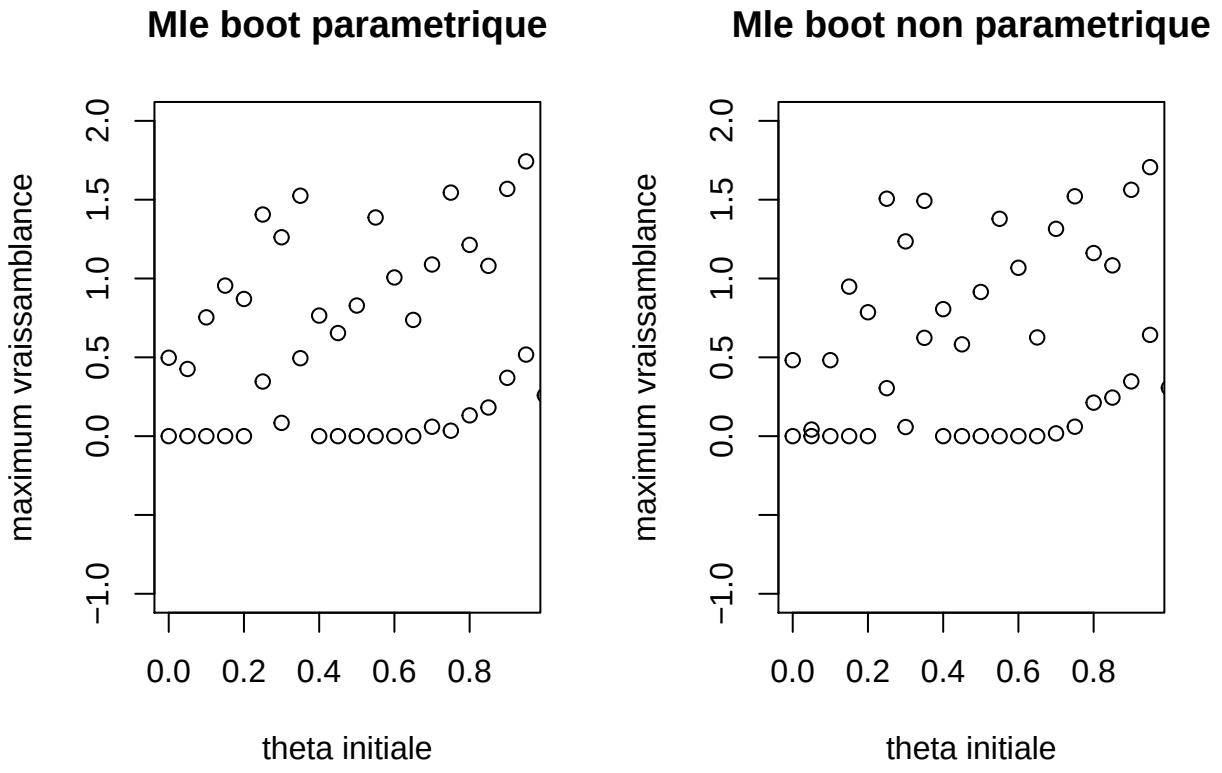
La valeur réelle de μ est bien contenu dans les intervalles obtenus par les methodes mais la méthode asymptotique donne un intervalle beaucoup plus grand que pour les autres méthodes. En revanche la différence entres les deux modèles bootstrap est très faible mais sur plusieurs essais la methodes non paramétrique est mieux centré sur la valeurs réel de μ mais son intervalle est plus petit voir trop petit.

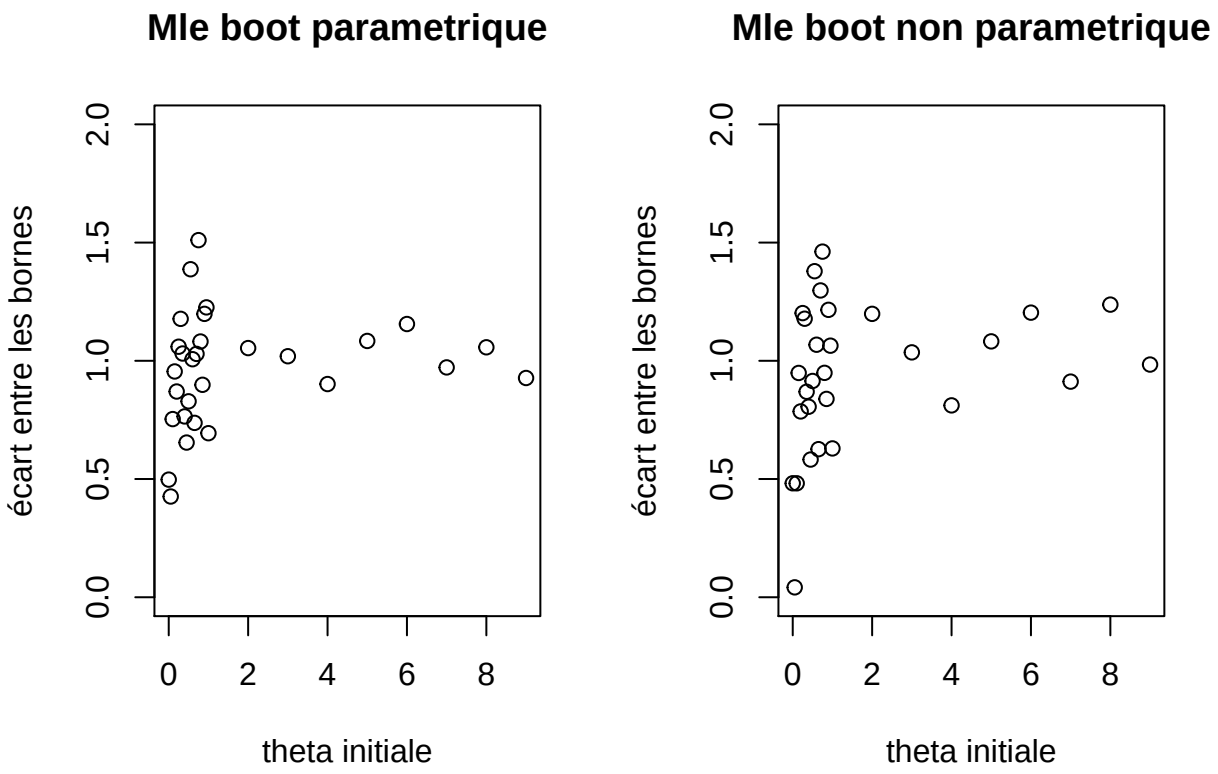
Question 12

Si $\overline{x} \geq 0$ c'est l'estimateur du maximum de vraisemblance de X_i. On sait que $\theta \geq 0$ donc si $\overline{x} < 0$ on sais que l'estimateur du maximum de vraisemblance est supérieur ou égale à 0 donc 0 est la valeur qui correspond le mieux à l'échantillions en prenant en compte la condition $\theta \geq 0$.

Ainsi le maximum de vraisemblance de X_i vaut $max(\overline{x}, 0)$

La fonction densité du maximum de vraissemblance de ce modèle n'est pas continue en 0 car tous les echantillons qui donnerais une valeurs de $\overline{x}$ négatif, ils attribuent au maximum de ressemblance un valeurs de 0. La probabilité que le mle =0 est non négligeable. Le modèle obtenu n'est donc pas régulier.





La taille des intervalles de confiance est réduite pour des faibles valeurs de θ car la born inférieur est écrasé sur la valeurs 0. On observe aucune différence significative entre les deux modèles bootstrap.

Question 13

Pour faire la couverture de l'échantillon de loi gamma, on reprend les fonctions faites en question 9 pour implémenter une fonction calculant la couverture par la méthode delta :

```
couverture_methode_delta <- function(n) {
  infs <- c()
  sups <- c()
  for (i in 1:n) {
    x <- rgamma(200, 2, 2)
    phi_gamma <- estimateur_phi_gamma(x)
    sigma_gamma <- ecart_type_estim_gamma(x)
    infs <- append(infs, phi_gamma - 0.475 * sigma_gamma)
    sups <- append(sups, phi_gamma + 0.475 * sigma_gamma)
  }
  inf <- min(infs)
  sup <- max(sups)
  return (c(inf,sup))
}
```

```
## La couverture pour la méthode delta est : [ 1.051354 , 1.099443 ]
```

On fait une fonction pour calculer la couverture avec la méthode bootstrap paramétrique :

```
couverture_bootstrap_parametric <- function(n) {
  infs <- c()
  sups <- c()
  for (i in 1:n) {
    x <- rgamma(200, 2, 2)
    intervalle <- bootstrap_parametric_gamma_phi(x)
    infs <- append(infs, intervalle[1])
    sups <- append(sups, intervalle[2])
  }
  inf <- min(infs)
  sup <- max(sups)
  return (c(inf,sup))
}
```

```
## La couverture pour la méthode bootstrap paramétrique est :[ 0.8115593 , 1.178614 ]
```

On fait une fonction pour calculer la couverture avec la méthode bootstrap non paramétrique :

```
couverture_bootstrap_nonParametric <- function(n) {
  infs <- c()
  sups <- c()
  for (i in 1:n) {
    x <- rgamma(200, 2, 2)
    intervalle <- bootstrap_nonParametric_gamma_phi(x)
    infs <- append(infs, intervalle[1])
    sups <- append(sups, intervalle[2])
  }
  inf <- min(infs)
  sup <- max(sups)
  return (c(inf,sup))
}
```

```
## La couverture pour la méthode bootstrap non paramétrique est : [ 0.4993494 , 1.636475 ]
```

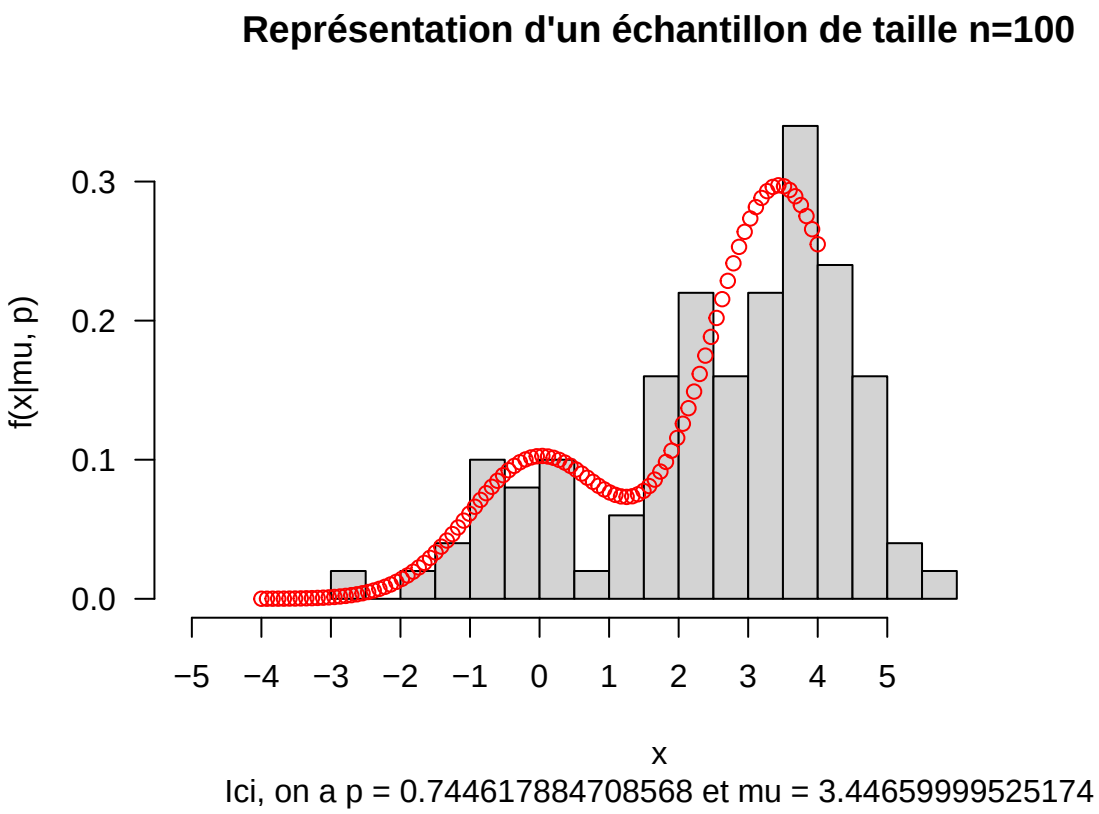
Question 14

On a maintenant une loi normale mixture définie par la densité :

$$f(x|\mu, p) = pN(x|\mu) + (1-p)N(x|0)\,,$$

où $N(x|\mu)$ est la densité de $N(\mu, 1)$ évaluée en x et $0 < p < 1$.

On a choisi ici de prendre $\mu \in [2, 5]$ afin de montrer un phénomène remarquable.



Nous cherchons maintenant à construire des intervalles bootstrap de confiance à 95% pour μ et p.
Nous allons donc implémenter les fonctions nécessaires, et de la même manière que précédemment.

Après application des fonctions bootstrap, nous obtenons :

```
## Pour p l'intervalle : [ 0.6757936 , 0.8550634 ], alors que p =  0.7446179
```

```
## Et pour mu, nous avons l'intervalle : [ 3.188818 , 3.580949 ], alors que mu =  3.4466
```

Nous voyons donc que les intervalles bootstrap fonctionnent bien, cette méthode semble fiable et plutôt optimisée.