Projet Applications du bootstrap et autres techniques de rééchantillonage

Pascal Jauffret et Alexis Rosuel

28 Avril 2017

Au cours de ce projet, nous avons décidé de nous interesser à un jeu de données fournit par le Texas Department of Justice, disponible ici : http://www.tdcj.state.tx.us/death_row/dr_executed_offenders.html. Ce dataset rassemble quelques informations basiques concernant tous les condamnés à la peine de mort et éxecutés depuis le 7 décembre 1982, jusqu'au 14 mars dernier. En particulier on connait leur âge, la date de l'exécution et leur ethnie.

Description du jeu de données

Commençons par importer le jeu de données

```
setwd('C:\\Users\\Alexis\\Google Drive\\Documents\\ENSAE\\Semestre 2\\Applications du bootstrap\\examen
data = read.csv('executed.csv', sep=';')
```

Affichons les 6 premières lignes

```
head(data)
```

```
##   Execution Last.Name  First.Name Age       Date     Race  County
## 1       542     Bigby       James  61  3/14/2017    White Tarrant
## 2       541      Ruiz     Rolando  44 03/07/2017 Hispanic   Bexar
## 3       540   Edwards       Terry  43  1/26/2017    Black  Dallas
## 4       539   Wilkins Christopher  48 01/11/2017    White Tarrant
## 5       538    Fuller      Barney  58 10/05/2016    White Houston
## 6       537   Vasquez       Pablo  38 04/06/2016 Hispanic Hidalgo
```

Par la suite, on va répondre aux questions suivante :
- la distribution de l'âge des condamnés est-elle issue d'une loi normale ? - quelle est leur moyenne d'âge ?
- le jour de leur exécution est-elle uniforme ? (ou bien il y en a plus le mardi que le dimanche par exemple ?)

Modélisation et justification théorique des hypothèses nécessaires

Afin d'appliquer les méthodes de rééchantillonage vues en cours, nous avons besoin de fixer un cadre théorique à notre étude. Notons $X_i \in R^k$ chaque individu. Ici, on a pas de raison de penser qu'il puisse y avoir de lien entre les individus, ni qu'ils ne soient pas issus de la même distribution, qu'on notera P.

Pour résumer, on a donc ceci : $\mathcal{X}_n = (X_1, ..., X_n) \sim P$ iid.

L'âge est une variable aléatoire bornée, tout comme le jour de l'éxecution. Ainsi, tous les moments des composantes des individus X_i sont bien définis. L'inégalité de Hoeffding pour les U-statistiques est donc aussi applicable, ce qui nous permet d'appliquer le subsampling.

$$E[\|X_n\|] < +\infty \text{ et } E[\|X_n\|^2] < +\infty$$

Par ailleurs, la condition (ii) du théorème de Lindeberg bien vérifiée.

Enfin, tous nos estimateur (moyenne, min, max, etc.) seront bien invariants par permutation.

Simulations

On charge la librairie 'boot' qui permet d'appliquer le bootstrap de manière efficace dans R

```
library(boot)
```

Dans tout ce qui suit, on utilisera B = 10000 simulations

```
B = 10000
```

On dispose de 542 observations dans notre dataset :

```
n = length(data$Age)
n
```

```
## [1] 542
```

Et on sous échantillonera des sous ensembles de taille b = 25 (on prend en fait $b \approx \sqrt{n}$)

```
b = 25
```

Concernant les tests, on se placera toujours au niveau $\alpha = 0.05$

```
alpha = 0.05
```

Enfin, et afin de pouvoir reproduire les résultats, on fixe :

```
set.seed(123)
```

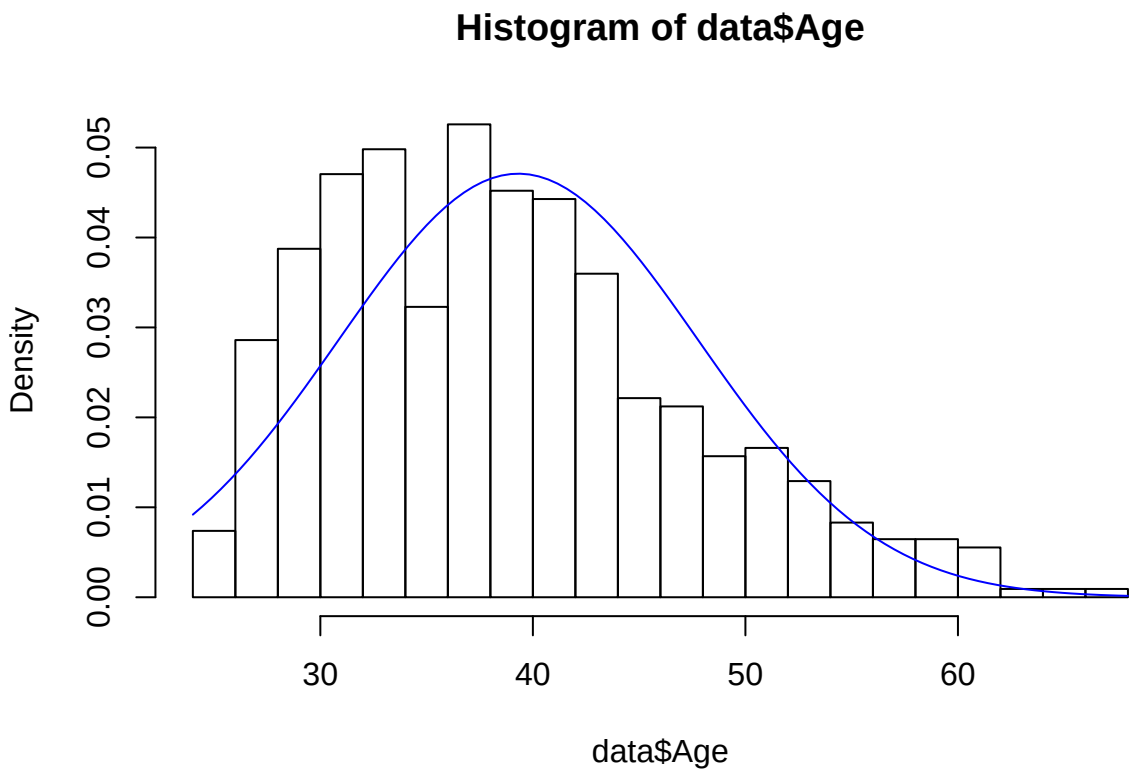
Application

Etude de la variable Age

Dans cette section, nous allons nous concentrer sur la variable âge. Les questions auxquelles nous allons essayer de répondre sont celles-ci:
1) Est ce que la distribution des âges est normale ?
1) Quel est un intervalle de confiance de la moyenne des condamnés au niveau 95% ? (bootstrap vs subsampling vs test de Fisher)
2) L'âge moyen est-il de 38 ans ? (bootstrap sous contrainte et par symmétrisation)

Commençons par le test de Kolmogorov-Smirnov en utilisant la méthode du bootstrap paramétrique. A première vue, on s'attend à répondre que la distribution réelle n'est pas issue d'une loi normale :

```
hist(data$Age, nclass=25, freq=FALSE)
curve(dnorm(x,mean(data$Age),sd(data$Age)), add=TRUE, col='blue')
```



Plus précisiement, on note $\hat{F}_n$ la fonction de répartition empirique associée. On note aussi $\mathcal{P}_0 = \{\mathcal{N}(\mu, \sigma^2), \mu \in R, \sigma^2 \in R^+\}$ la famille de lois dont on souhaite tester si nos données sont issues.

La statistique de test naturelle est $T_n(\mathcal{X}_n) = \sqrt{n}\|\hat{F}_n - F_0\|_\infty$. Enfin, on note la loi de $T_n(\mathcal{X}_n)$: $L_n(P_0, P_0) = \mathcal{L}(\sqrt{n}\|\hat{F}_n - F_0\|_\infty)$

On calcule $\hat{\theta}_n$ l'estimateur de θ :

```
mu = mean(data$Age)
sigma2 = sd(data$Age)
```

On calcule la statistique T_n :

```
norme_infinie <- function(Xn){
  Xn.sorted = sort(Xn)
  resultat = c()
  n = length(Xn)
  for(i in 1:n){
    resultat = c(resultat ,max(i/n - pnorm(Xn.sorted[i], mean=mu, sd=sigma2), pnorm(Xn.sorted[i], mean=m
  }
  return(max(resultat))
}

norme_infinie(data$Age)
```

```
## [1] 0.0780213
```

On peut d'ailleurs la comparer et la confirmer à la fonction ks.test :

```
ks.test(data$Age, "pnorm", mu, sigma2)$statistic
```

```
## Warning in ks.test(data$Age, "pnorm", mu, sigma2): ties should not be
## present for the Kolmogorov-Smirnov test
```

```
##         D
## 0.0780213
```

On calcule la valeur de la statique de test

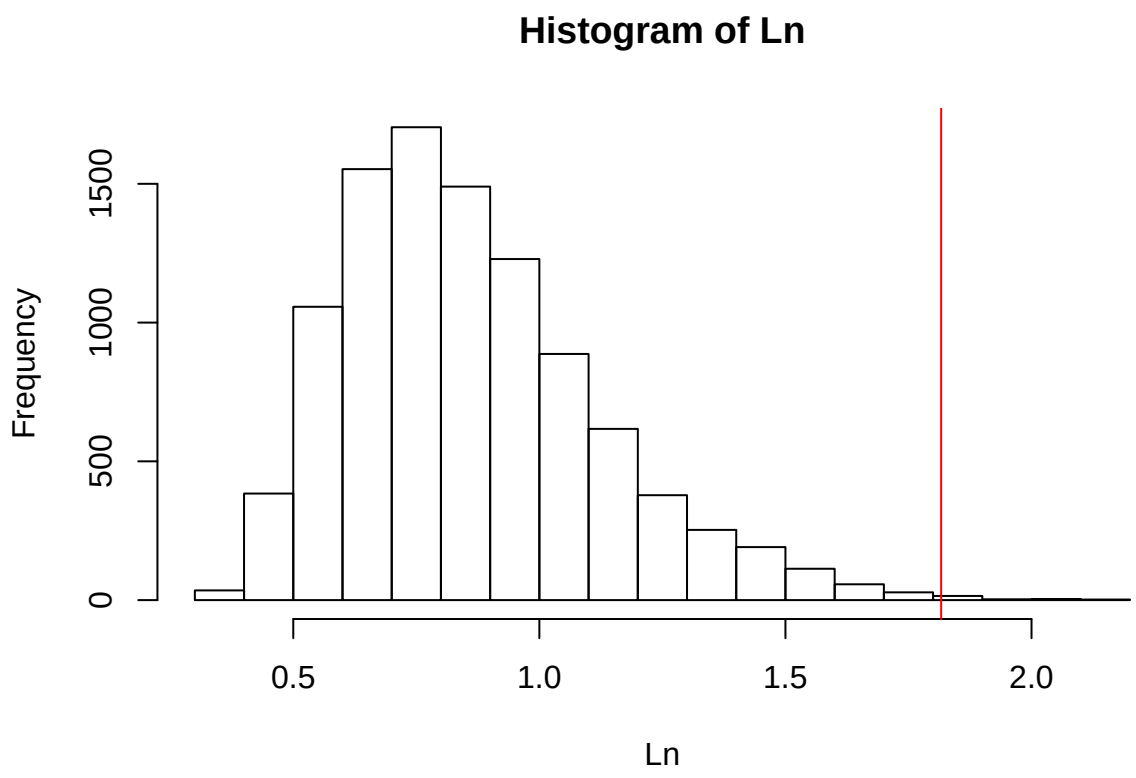
```
Tn = sqrt(n) * norme_infinie(data$Age)
Tn
```

```
## [1] 1.816406
```

On simule la loi selon B échantillons

```
simulation = matrix(rnorm(B*n, mean=mu, sd=sigma2), n, B)
Ln = sqrt(n) * apply(simulation, 2, norme_infinie)
hist(Ln)
abline(v=Tn, col='red', add=TRUE)
```

```
## Warning in int_abline(a = a, b = b, h = h, v = v, untf = untf, ...): "add"
## n'est pas un paramètre graphique
```



Enfin, on calcule la p-value

```
mean(Tn<Ln)
```

```
## [1] 0.0019
```

Elle vaut 0.0019, on rejette donc H0. Cette valeur est en accord avec la fonction ks.test

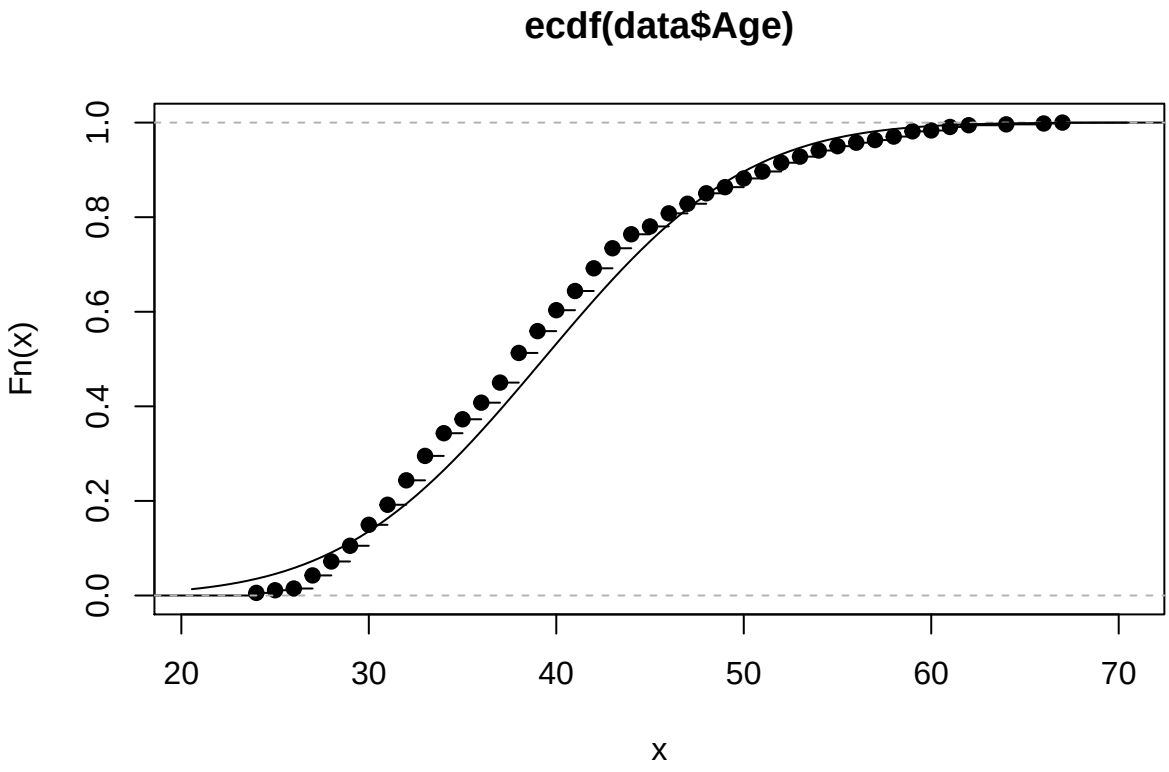
```
ks.test(data$Age, "pnorm", mu, sigma2)
```

```
## Warning in ks.test(data$Age, "pnorm", mu, sigma2): ties should not be
## present for the Kolmogorov-Smirnov test
```

```
##
##  One-sample Kolmogorov-Smirnov test
##
## data:  data$Age
## D = 0.078021, p-value = 0.002724
## alternative hypothesis: two-sided
```

Graphiquement, le “non ajustement” semble vérifié

```
plot(ecdf(data$Age))
curve(pnorm(x, mean=mu, sd=sigma2), add=TRUE)
```



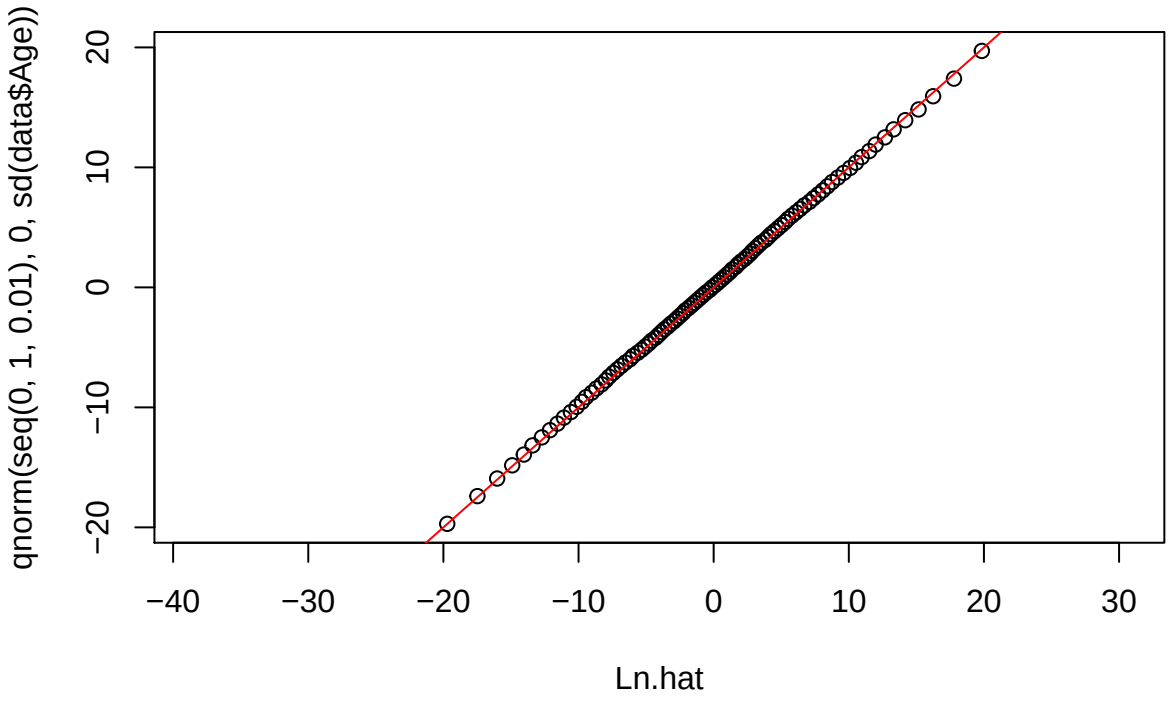
Maintenant qu’on a justifié le fait de reposer sur des hypothèses non-paramétriques, on va calculer des intervalles de confiances et des faire des tests d’hypothèses qui utilisent les méthode bootstrap.

On note dans ce paragraphe X_i la variable âge de l’individu i. Le paramètre sur lequel on fait l’inférence est $\theta(P) = E_P[X]$. Dans ce cas, on sait que la fonction de perte à condidérer est :

$$\gamma_n(\hat{\theta}_n) = \sqrt{n} * (\hat{\theta}_n - \theta)$$

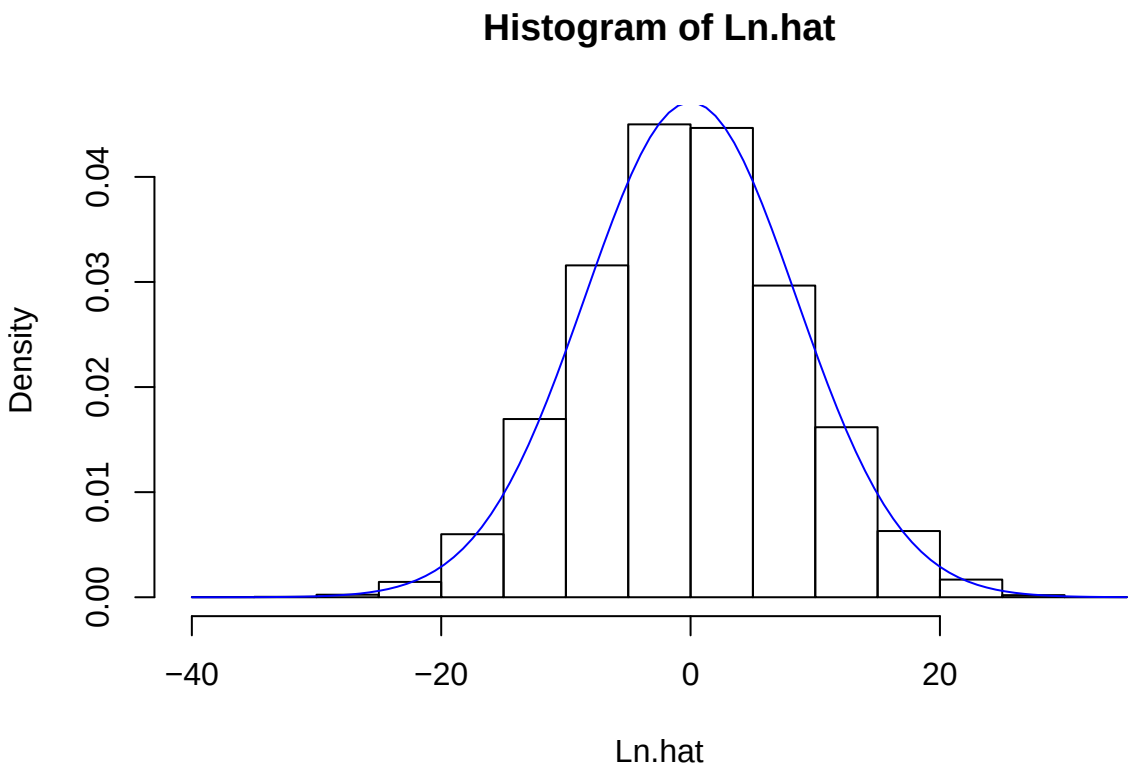
qui est bien une fonction continue de $\hat{\theta}_n$. Commençons par le calcul de l’intervalle de confiance pour la moyenne d’âge des condamnés.

```r
theta.hat=mean(data$Age)

bootdata.age = boot(data$Age, function(y,i) mean(y[i]), R=B, sim='ordinary')
Ln.hat = sqrt(n) * (bootdata.age$t - theta.hat)

qqplot(Ln.hat, qnorm(seq(0,1,0.01),0,sd(data$Age)))
abline(0,1, col='red')
```



Visuellement on confirme que $\hat{L}_n$ converge bien vers la quantité prévue.

```r
hist(Ln.hat, freq=FALSE)
curve(dnorm(x, sd=sd(data$Age)), add=TRUE, col='blue')
```



L'ajustement est OK. On calcule les quantiles 0.025 et 0.975

```
quantile(Ln.hat, prob=c(0.025, 0.975)) / sqrt(n)
```

```
##       2.5%      97.5%
## -0.7195572  0.7269373
```

D'où l'intervalle de confiance final à 95% pour la moyenne d'âge des condamnés :

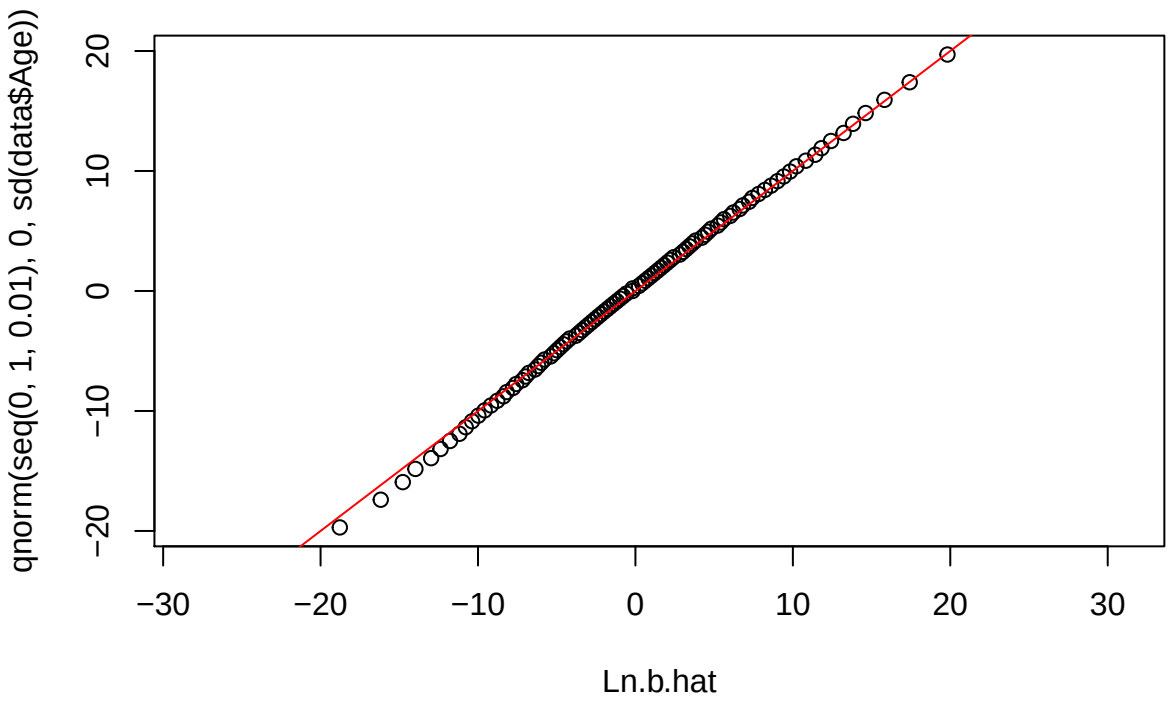
```
theta.hat + quantile(Ln.hat, prob=c(0.025, 0.975)) / sqrt(n)
```

```
##     2.5%    97.5%
## 38.59594 40.04244
```

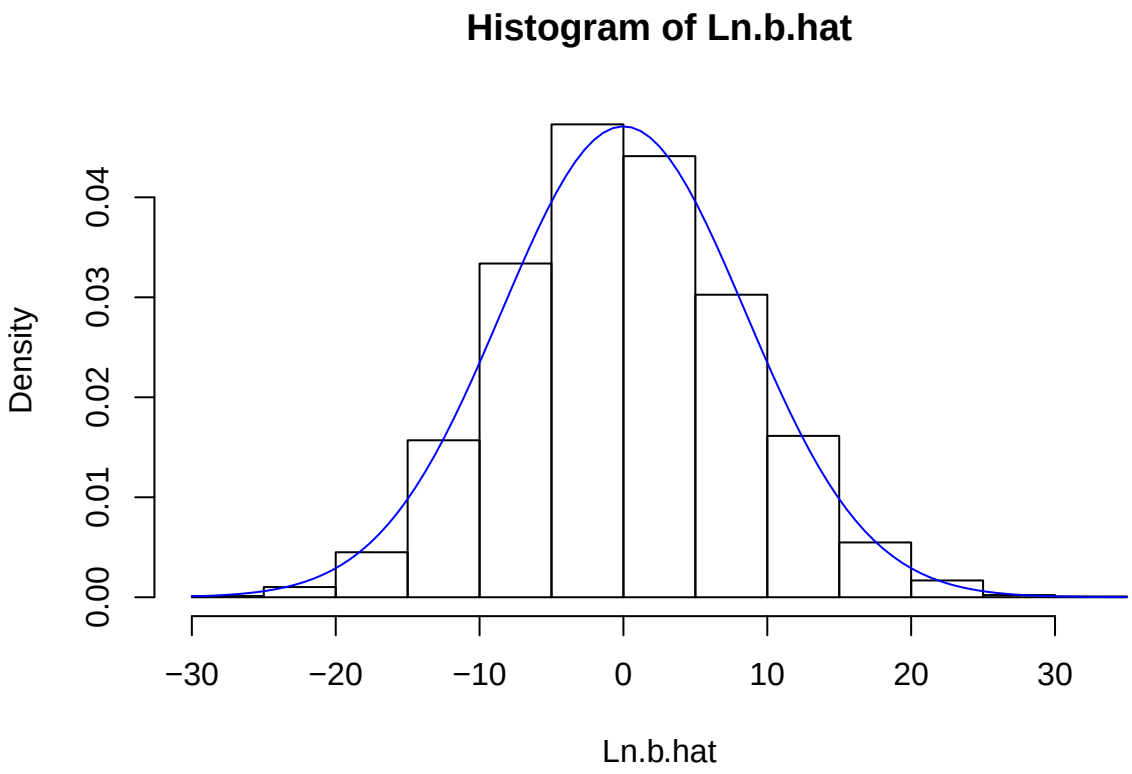
Passons au subsampling. On commence par écrire la fonction qui crée un sous-échantillon à partir des données :

```
subsample <- function(data)
{
  data.sampled = sample(data, size=b, replace=FALSE)
}

data.replicated = matrix(data$Age, nrow=n, ncol=B)

data.subsampled = apply(data.replicated, 2, subsample)
theta.subsampled.hat = apply(data.subsampled, 2, mean)

Ln.b.hat = sqrt(b) * (theta.subsampled.hat - theta.hat)
```

```
qqplot(Ln.b.hat, qnorm(seq(0,1,0.01), 0, sd(data$Age)))
abline(0,1, col='red')
```




```
hist(Ln.b.hat, freq=FALSE)
curve(dnorm(x, sd=sd(data$Age)), add=TRUE, col='blue')
```



On calcule les quantiles 0.025 et 0.975

```
quantile(Ln.b.hat, prob=c(0.025, 0.975)) / sqrt(b)
```

```
##      2.5%     97.5%
## -3.115498  3.324502
```

D'où l'intervalle de confiance final à 95% pour la moyenne d'âge des condamnés :

```
theta.hat + quantile(Ln.hat, prob=c(0.025, 0.975)) / sqrt(b)
```

```
##     2.5%    97.5%
## 35.96511 42.70025
```

Enfin, on compare ce résultat à un test de student (dans le cadre paramétrique)

```
t.test(data$Age)
```

```
##
##  One Sample t-test
##
## data:  data$Age
## t = 108.06, df = 541, p-value < 2.2e-16
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  38.6008 40.0302
## sample estimates:
## mean of x
##   39.3155
```

Comme prévu l'intervalle de confiance est plus petit, mais repose sur une hypothèse qui n'est dans notre cas pas vérifié.

Bootstrap	Subsampling	Fisher
38.60	35.97	38.60
40.04	42.70	40.03

C'est d'ailleurs une remarque générale que l'on peut faire sur les intervalles de confiance non paramétrique : ils sont toujours plus larges que ceux qui sont paramétriques, mais possèdent l'avantage de ne reposer sur aucune hypothèse de distribution des données. C'est pourquoi on préférera faire du non paramétrique lorsque la distribution des données est inconnue, et dès lors que l'on est suffisament sur que les données suivent une certaine distribution, on utilisera les méthodes paramétriques correspondantes.

Enfin, on va tester si la moyenne d'age est de 38 ans en appliquant les deux méthodes proposées plus haut.

Commençons par le test bootstrap sous contrainte. On teste : "H0 : $\theta = 38$" contre "H1 : $\theta \neq 38$". La statistique de test est $T_n = |\bar{X}_n|$, et on se fixe $\alpha = 0.05$. Afin de se ramener dans le cadre du test "H0 : $\theta = 0$", on va faire "data = data - 38"

```
data.38 = data$Age - 38
Tn.hat = sqrt(n) * abs(mean(data.38))
```

On génère l'échantillon bootstrap sous contrainte, puis on calcule les statistiques de test

```
data.Age.corrected = data.38 - mean(data.38)
bootdata.age = boot(data.Age.corrected, function(y,i) mean(y[i]), R=B, sim='ordinary')
Tn.star = sqrt(n) * abs(bootdata.age$t)
```

On calcule le quantile empirique qui correspond au niveau α fixé

```
c_alpha = quantile(Tn.star, prob=1-alpha)
```

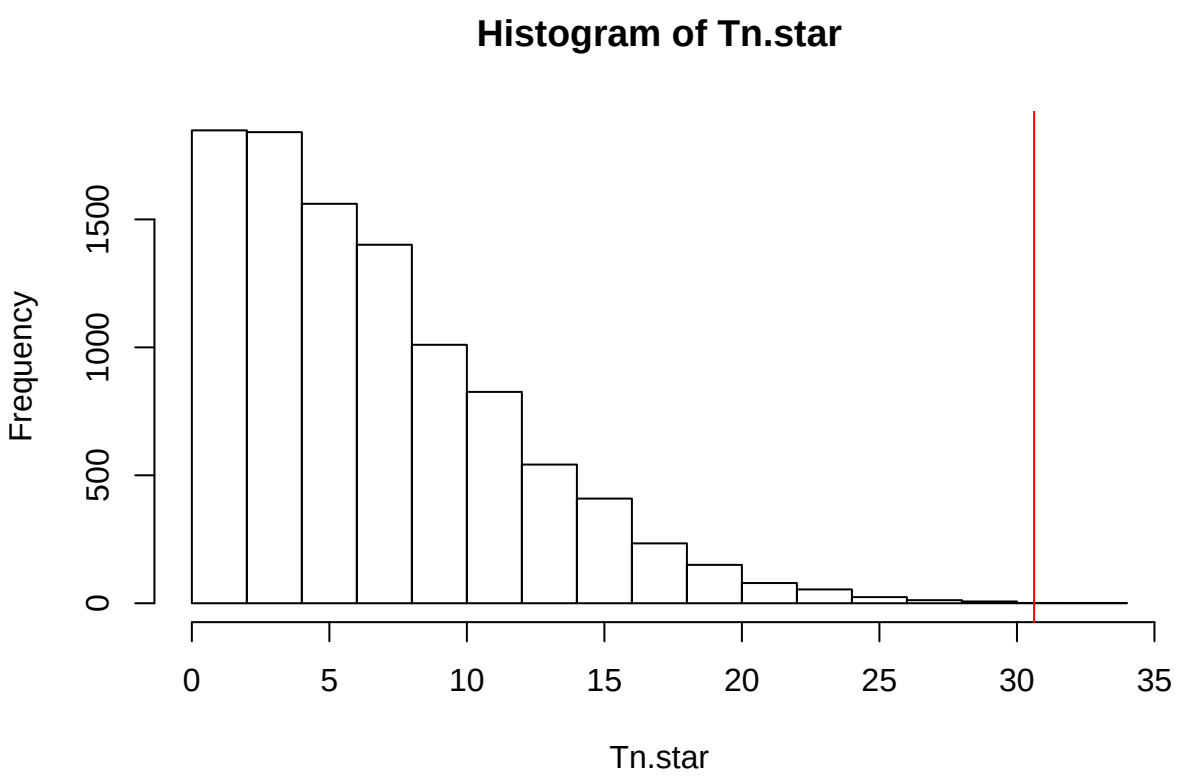
On calcule la p-valeur

```
mean(Tn.star > Tn.hat)
```

```
## [1] 1e-04
```

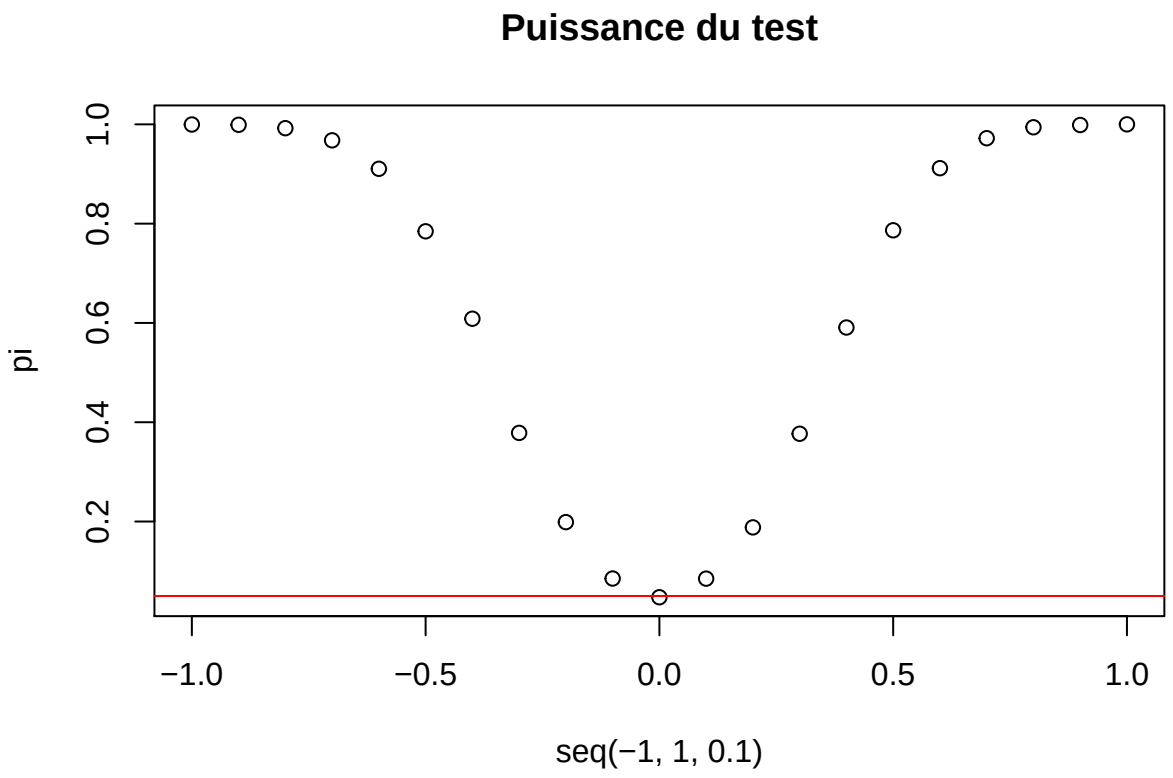
Elle est très faible, plus petite que α, on rejete donc H0

```
hist(Tn.star)
abline(v=Tn.hat, col='red')
```



Calculons la puissance du test. Pour ce faire, on va calculer c_α pour $\theta = 37, 37.1, ..., 39$ (ici on calcule donc pour $\theta = -1, -0.9, ..., 1$ car on a translaté les données de 38) :

```
pi = c()
for(theta in seq(-1,1,0.1)){
  data.Age.corrected = data.38 - mean(data.38) + theta
  bootdata.age = boot(data.Age.corrected, function(y,i) mean(y[i]), R=B, sim='ordinary')
  pi = c(pi, mean(sqrt(n) * bootdata.age$t > c_alpha - sqrt(n) * theta) + mean(sqrt(n) * bootdata.age$t
}
plot(seq(-1,1,0.1), pi, main="Puissance du test")
abline(h=alpha, col='red')
```



On confirme bien que la puissance du test vaut α pour $\theta = 38$, puis augmente et converge vers 1 en s'en éloignant.
Testons par permutation si la moyenne est égale à 38. Pour ce faire, on va faire data - 38, puis tester "H0 : $\theta = 0$" contre "H1 : $\theta \neq 0$"

```
data.38 = data$Age - 38
epsilon = rbinom(n, size=1, prob=1/2)
data.38.shuffled = data.38 * epsilon

Tn.hat = sqrt(n) * mean(data$Age - 38)

bootdata.age = boot(data.38.shuffled, function(y,i) mean(y[i]), R=B, sim='ordinary')
Tn.star = sqrt(n) * abs(bootdata.age$t)
```

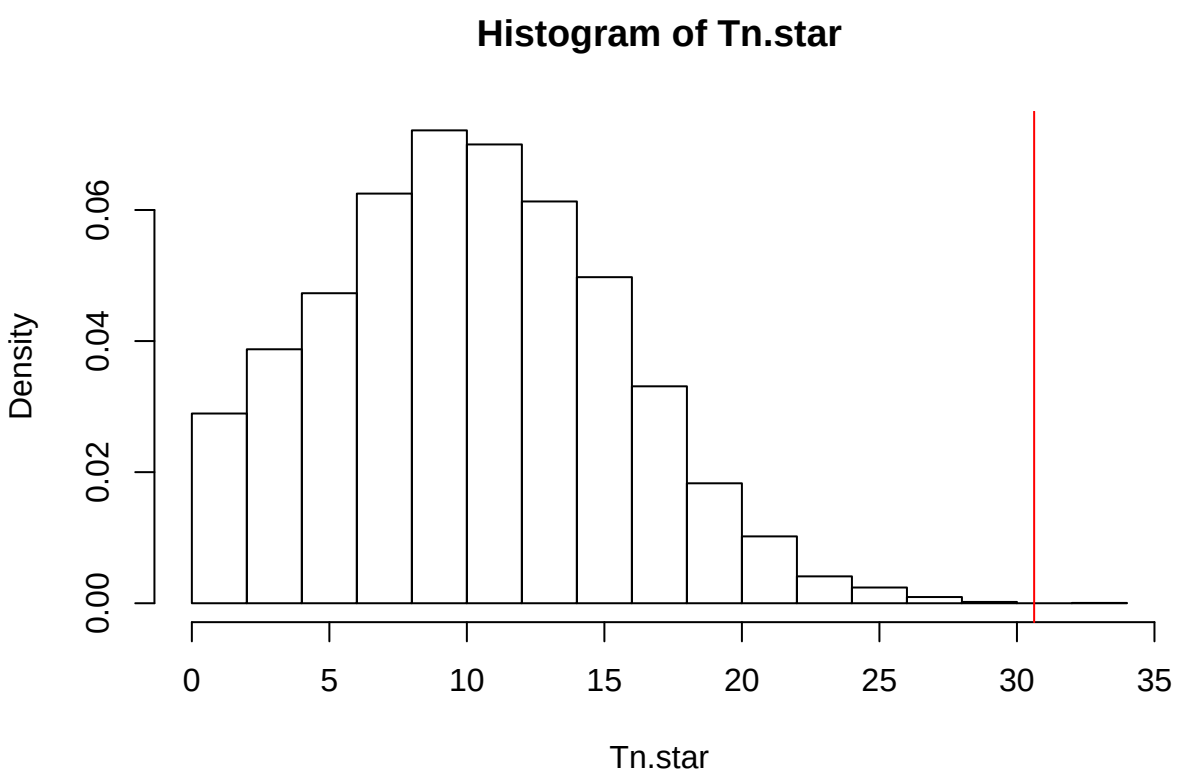
On calcule la p-valeur

```
mean(Tn.star > Tn.hat)
```

```
## [1] 1e-04
```

Elle est aussi très faible, on rejette H0 au niveau 5%

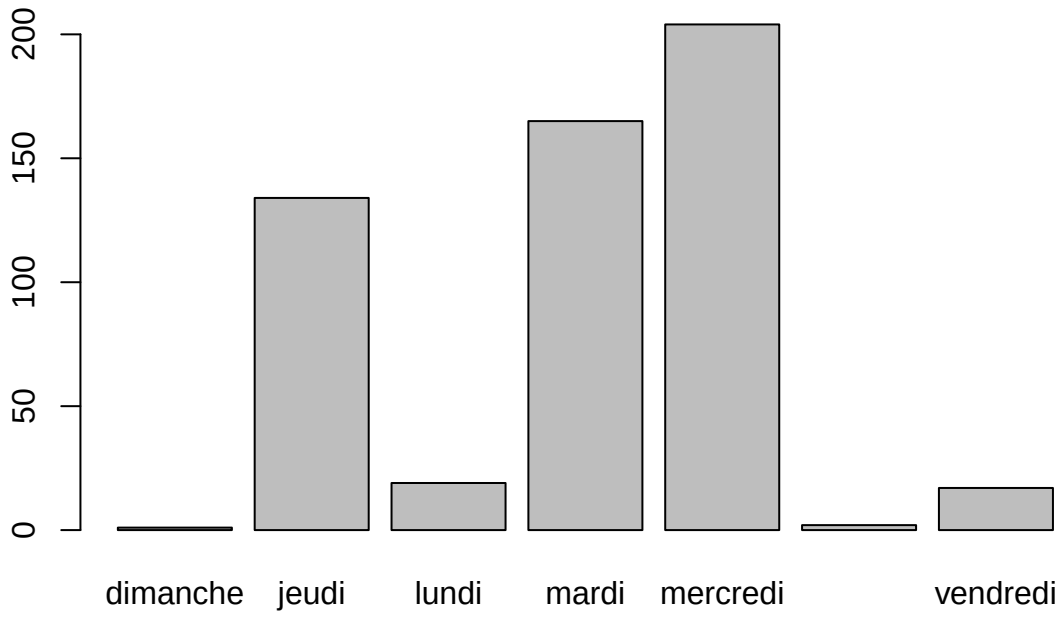
```
hist(Tn.star, freq=FALSE)
abline(v=Tn.hat, col='red')
```



Etude de la variable jour de la semaine

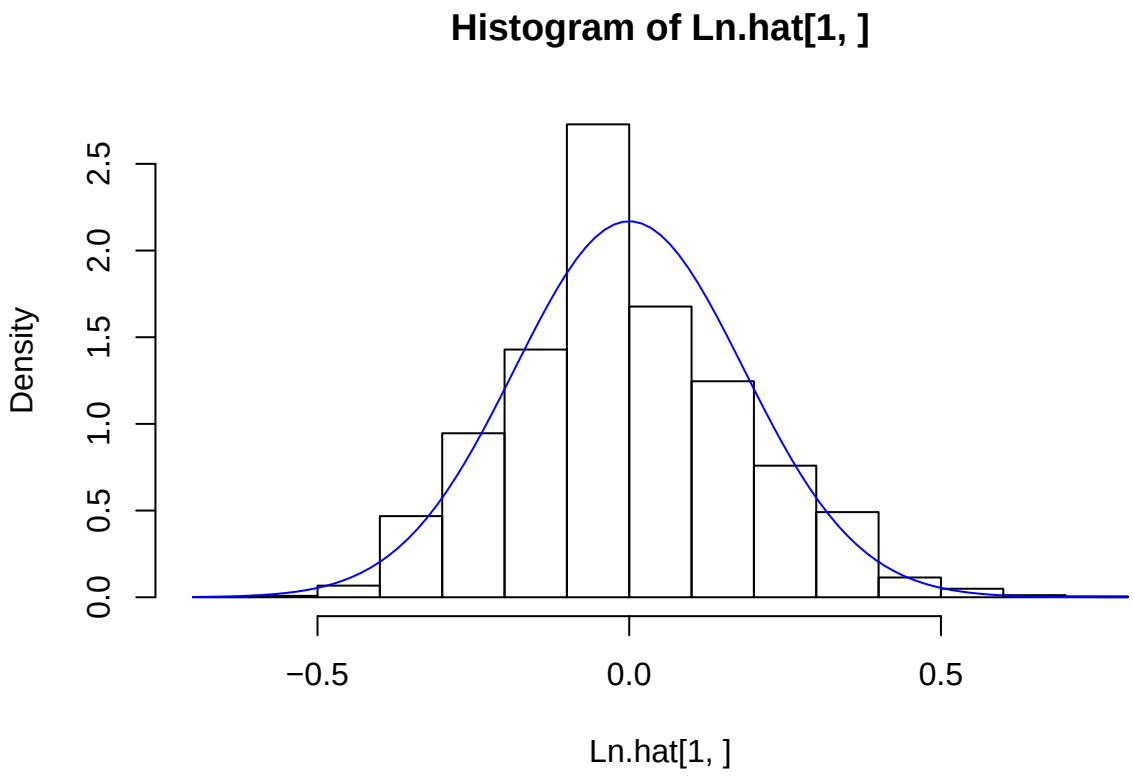
Dans cette dernière section nous allons nous interesser à la distribution de la variable 'jour de la semaine'. Plus précisement, voici le modèle que nous allons considérer :
Soient $X_1, ..., X_n \sim \mathcal{M}_\theta$ iid les valeurs relevés des individus dont on dispose. Ils sont donc tirés depuis une loi multinomiale de paramètre $\theta \in R^7$ avec $|\theta|_1 = 1$. L'objectif va être de pouvoir donner un intervalle de confiance sur la valeur des composantes de θ puis de tester si elles sont toutes égales à $\frac{1}{7}$. Commençons par une rapide visualisation. A priori la distribution ne semble pas être uniforme.

```
date = as.Date(data$Date, format='%m/%d/%Y')
date1 = as.factor(sapply(date, weekdays))
plot(date1)
```



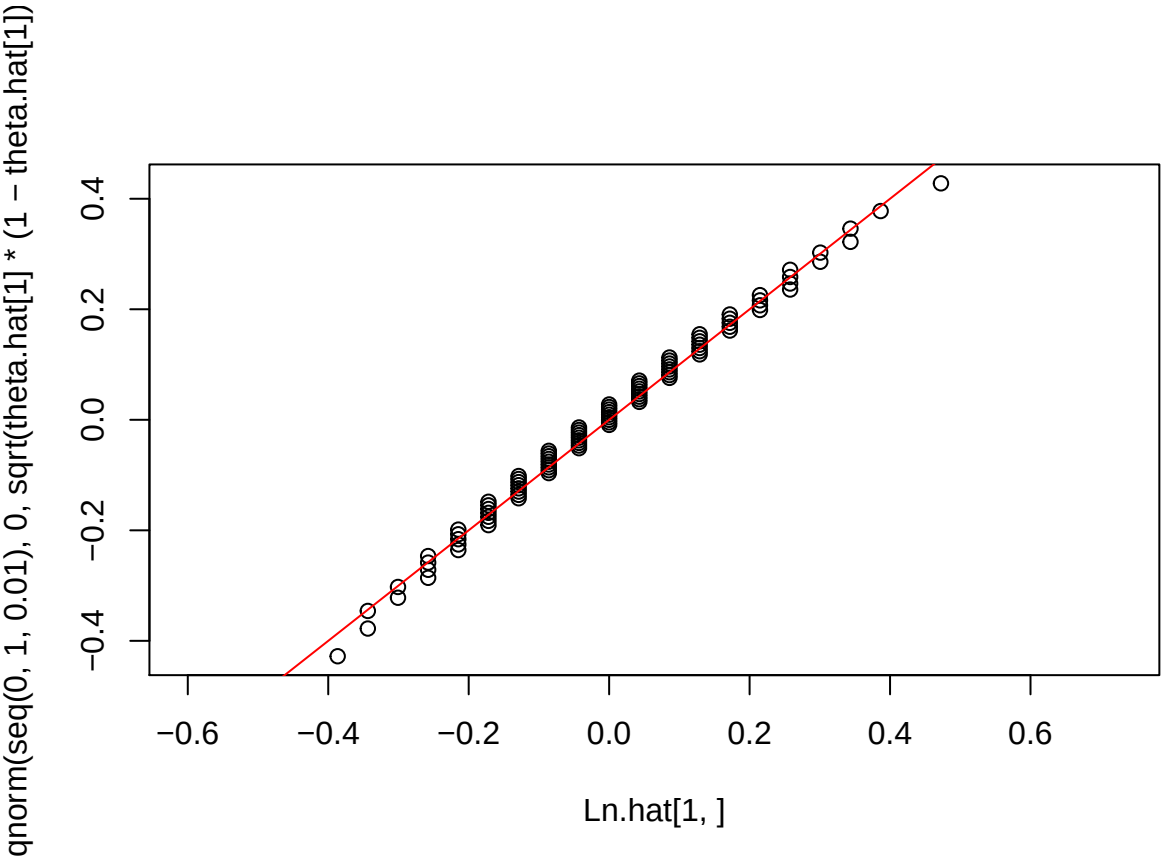
Testons le

```
theta.hat = c(mean(date1 == "lundi"), mean(date1 == "mardi"), mean(date1 == "mercredi"), mean(date1 == 

compute_theta <- function(y, i){
  return( c(mean(y[i] == "lundi"), mean(y[i] == "mardi"), mean(y[i] == "mercredi"), mean(y[i] == "jeudi
}

bootdata.date1 = boot(date1, compute_theta, R=B, sim='ordinary')

compute_Ln.hat <- function(boot){
  return(sqrt(n) * (boot - theta.hat))
}

Ln.hat = apply(bootdata.date1$t, 1, compute_Ln.hat)

hist(Ln.hat[1,], freq = FALSE)
curve(dnorm(x,sd=sqrt(theta.hat[1]*(1-theta.hat[1]))), add=TRUE, col='blue')
```



Le théorème central limite multidimensionnel nous assure que $\sqrt{n}(\hat{\theta}_n - \theta)$ converge vers une loi normale centrée de matrice de covariance $Diag(\theta_1(1-\theta_1), \theta_2(1-\theta_2), ..., \theta_7(1-`\theta_7))$.

```
qqplot(Ln.hat[1,], qnorm(seq(0,1,0.01), 0, sqrt(theta.hat[1]*(1-theta.hat[1]))))
abline(0,1, col='red')
```



On peut donc donner un intervalle de confiance sur les paramètres :

```
theta.hat[1] + quantile(Ln.hat[1,], prob=c(0.025,0.975))/sqrt(n)
```

```
##       2.5%      97.5%
## 0.02029520 0.05166052
```

```
theta.hat[2] + quantile(Ln.hat[2,], prob=c(0.025,0.975))/sqrt(n)
```

```
##      2.5%     97.5%
## 0.2656827 0.3431734
```

```
theta.hat[3] + quantile(Ln.hat[3,], prob=c(0.025,0.975))/sqrt(n)
```

```
##      2.5%     97.5%
## 0.3357934 0.4169742
```

```
theta.hat[4] + quantile(Ln.hat[4,], prob=c(0.025,0.975))/sqrt(n)
```

```
##      2.5%     97.5%
## 0.2121310 0.2841328
```

```
theta.hat[5] + quantile(Ln.hat[5,], prob=c(0.025,0.975))/sqrt(n)
```

```
##       2.5%      97.5%
## 0.01845018 0.04797048
```

```
theta.hat[6] + quantile(Ln.hat[6,], prob=c(0.025,0.975))/sqrt(n)
```

```
##        2.5%       97.5%
## 0.000000000 0.009225092
```

```
theta.hat[7] + quantile(Ln.hat[7,], prob=c(0.025,0.975))/sqrt(n)
```

```
##        2.5%       97.5%
## 0.000000000 0.005535055
```

Terminons avec le test : On veut tester "$H_0 : \theta = (\frac{1}{7}, ..., \frac{1}{7})^T$" contre "$H_1 : \theta \neq (\frac{1}{7}, ..., \frac{1}{7})^T$". On va utiliser le bootstrap sous contrainte.

```
Tn.hat = sqrt(n) * abs(theta.hat[1]-1/7)
```

On génère l'échantillon bootstrap sous contrainte. Ici, vu qu'on suppose connue la classe dans laquelle a été généré les données (une multinomiale), il suffit de générer un jeu $X_1^*, ..., X_n^* \sim b(1/7)$ (bernouilli) iid, puis de calculer la valeur de c_α tel que $P[Tn] puis on calcule les statistiques de test

```
bootdata.date = rbinom(B, size=n, prob=1/7)
theta.star = bootdata.date/n
Tn.star = sqrt(n) * abs(theta.star-1/7)
```

On calcule le quantile empirique qui correspond au niveau α fixé

```
c_alpha = quantile(Tn.star, prob=1-alpha)
```

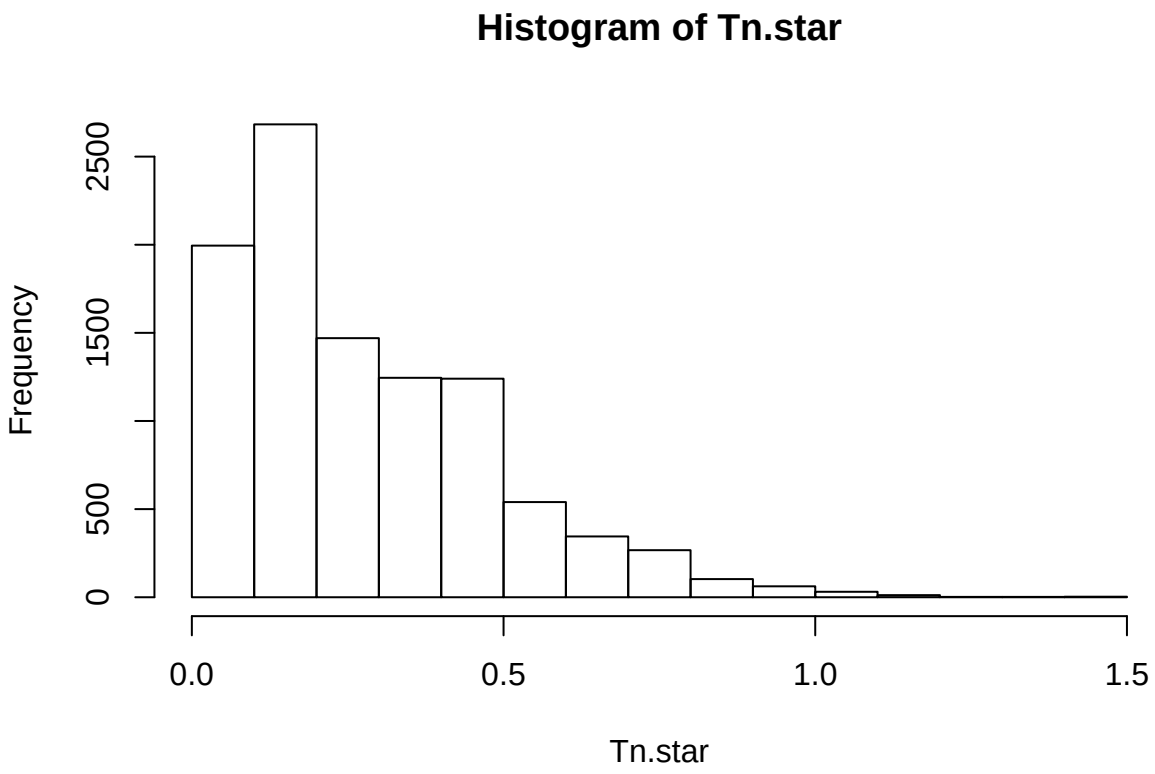
On calcule la p-valeur

```
mean(Tn.star > Tn.hat)
```

```
## [1] 0
```

Elle est très faible, plus petite que α, on rejete donc H0

```
hist(Tn.star)
abline(v=Tn.hat, col='red')
```



On pourrait faire ceci pour les 7 jours, tous les tests rejettent H_0.
Remarque : on n'a pas appliqué la procédure de Bonferroni ici car le nombre de test à effectuer est négligeable devant le nombre d'observations dont on dispose.