Projet Bayesien

Jianhui Luo, Mamadou Lamine Diamban, Khadidiatou AW, Adam Yimbi

2020/1/28

```
aci=read.csv("/Users/khadija/Desktop/donnees.txt",header=F)
head(aci)
```

```
##         V1
## 1 2.928524
## 2 3.910021
## 3 3.732896
## 4 3.688879
## 5 3.822098
## 6 3.735286
```

```
colnames(aci)="Acidite"
dim(aci)
```

```
## [1] 155   1
```

```
anyNA(aci)
```

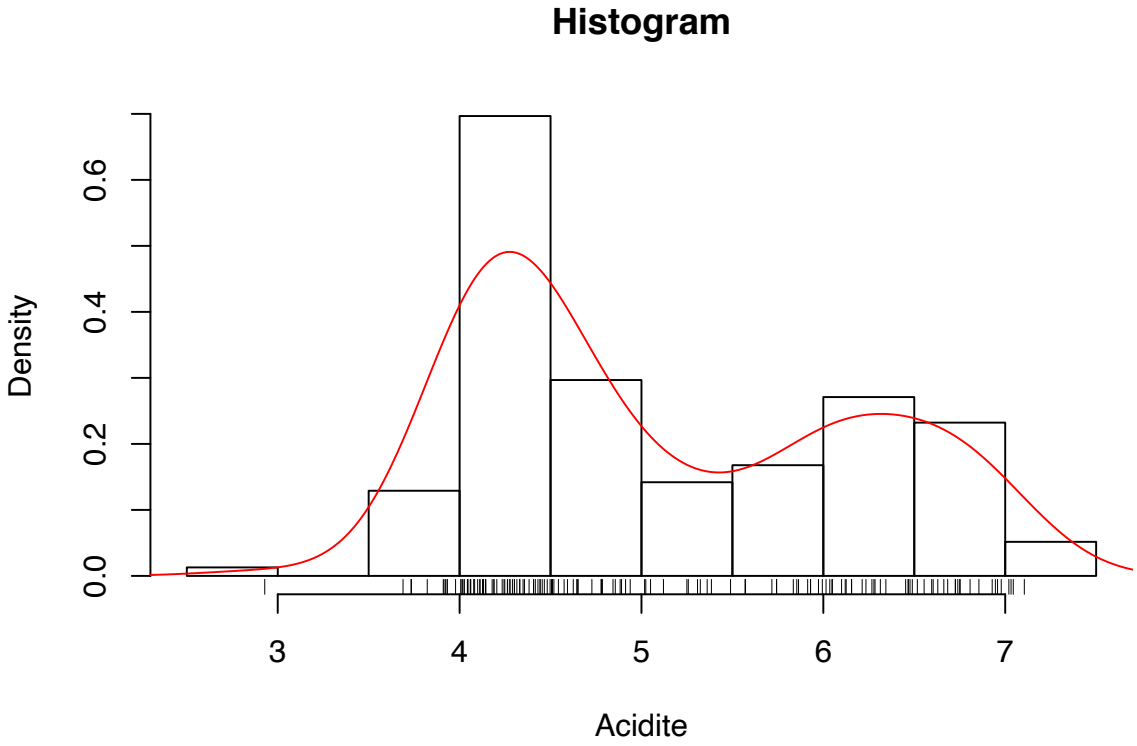
```
## [1] FALSE
```

```
summary(aci)
```

```
##     Acidite
##  Min.   :2.929
##  1st Qu.:4.219
##  Median :4.727
##  Mean   :5.105
##  3rd Qu.:6.075
##  Max.   :7.105
```

Analyse de visualisation des données

```
hist(aci$Acidite,xlab="Acidite",freq=F,main="Histogram")
lines(density(aci$Acidite),col="red")
rug(aci$Acidite)
```



Hypothèses du modèle

Comme le montre la figure, les données proviennent de deux distributions gaussiennes unidimensionnelles. Nous ajustons les données en utilisant un modèle de mélange gaussien,

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k) = \sum_{K=1}^{k} \pi_k N(x|u_k, \sigma_k)$$

Où $p(x|k) = N(x|u_k, \sigma_k)$ est la fonction de densité de probabilité du k-ème modèle gaussien, c'est-à-dire la probabilité que ce modèle produise x; $p(k) = {}_k$ est le k-ème modèle gaussien Le poids est appelé la probabilité antérieure de sélectionner k-ième modèle et satisfait $\sum_{k=1}^{K} \pi_k = 1$. Ici, nous savons déjà que K = 2, nous devons estimer π_k,u_k,σ_k.

L'algorithme EM pour comparaison

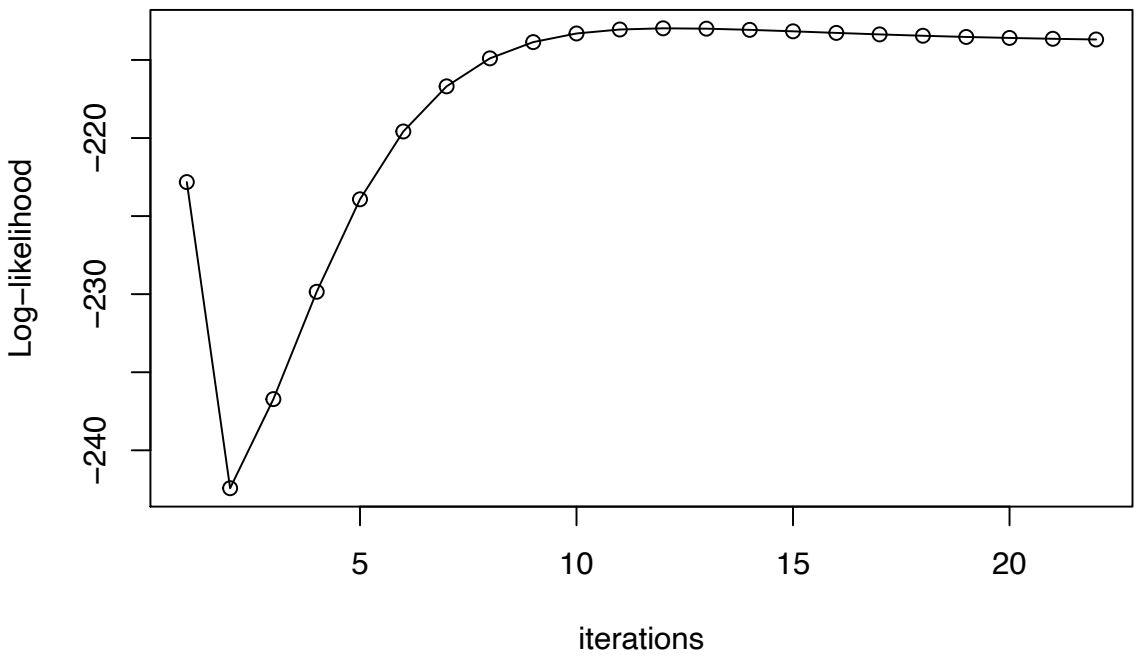
```
y=aci$Acidite

fnorm <- function(x, mu, sigma)
{
  return(1/(sqrt(2*pi)*sigma)*exp(-0.5*(x-mu)^2/sigma))
}

IterEM <- function(mu1, mu2, sigma1, sigma2, pi0, eps)
{
  cat('Start EM...\n')
  cat(paste0('pi = ', pi0, '\n'))
  iters = 0
```

```r
  ll = c()
  while(TRUE)
  {
    ## Expectation step: compute the responsibilities
    ## calculate the delta's expectation
    gamma = sapply(y, function(x) pi0*fnorm(x, mu2, sigma2)/((1-pi0)*fnorm(x, mu1, sigma1) + pi0*fnorm(
    ll = c(ll, sum((1-gamma)*log(fnorm(y,mu1,sigma1))+gamma*log(fnorm(y, mu2, sigma2))+(1-gamma)*log(1-
    ## Maximization Step: compute the weighted means and variances
    mu1.new = sum((1-gamma)*y)/sum(1-gamma)
    mu2.new = sum(gamma*y)/sum(gamma)
    sigma1.new = sqrt(sum((1-gamma)*(y-mu1.new)^2)/sum(1-gamma))
    sigma2.new = sqrt(sum((gamma)*(y-mu2.new)^2)/sum(gamma))
    pi0.new = sum(gamma)/length(y)
    cat(paste0('pi = ', pi0.new, '\n'))
    if (abs(pi0.new-pi)< eps || iters > 20)
    {
      cat('Finish!\n')
      cat(paste0('mu1 = ', mu1.new, '\n',
                 'mu2 = ', mu2.new, '\n',
                 'sigma1^2 = ', sigma1.new^2, '\n',
                 'sigma2^2 = ', sigma2.new^2))
      break
    }
    mu1 = mu1.new
    mu2 = mu2.new
    sigma1 = sigma1.new
    sigma2 = sigma2.new
    pi0 = pi0.new
    iters = iters + 1
  }
  return(ll)
}
## take initial guesses for the parameters
mu1 = 5; sigma1 = 0.5
mu2 = 7; sigma2 = 1
pi0 = 0.1
eps = 0.01
ll = IterEM(mu1, mu2, sigma1, sigma2, pi0, eps)
```

```
## Start EM...
## pi = 0.1
## pi = 0.112889529862459
## pi = 0.158106056845482
## pi = 0.206885930235993
## pi = 0.246433596612545
## pi = 0.275556988824228
## pi = 0.296236789172252
## pi = 0.310741464968297
## pi = 0.320873715854285
## pi = 0.327935991330984
## pi = 0.332846748024796
## pi = 0.33625107019964
## pi = 0.338602123514382
```

```
## pi = 0.340218286243699
## pi = 0.341323089576949
## pi = 0.342073261725348
## pi = 0.342578484805521
## pi = 0.342915334149386
## pi = 0.343137116099705
## pi = 0.34328080893111
## pi = 0.34337195659551
## pi = 0.343428115723726
## pi = 0.343461280432611
## Finish!
## mu1 = 4.49867612908765
## mu2 = 6.26429088020775
## sigma1^2 = 0.372801030825482
## sigma2^2 = 0.380499898073573
```

```
## Figure 8.6
plot(1:length(ll), ll, xlab = 'iterations', ylab = 'Log-likelihood', 'o')
```



MCMC pour l'échantillonnage du postérieur

On prend comme loi a priori sur $\mu, \mathcal{N}(\mu_0, \sigma_0^2)$ et sur σ^2 une loi inverse gamma de paramètres a et b.

On sait que les lois a posteriori sont :

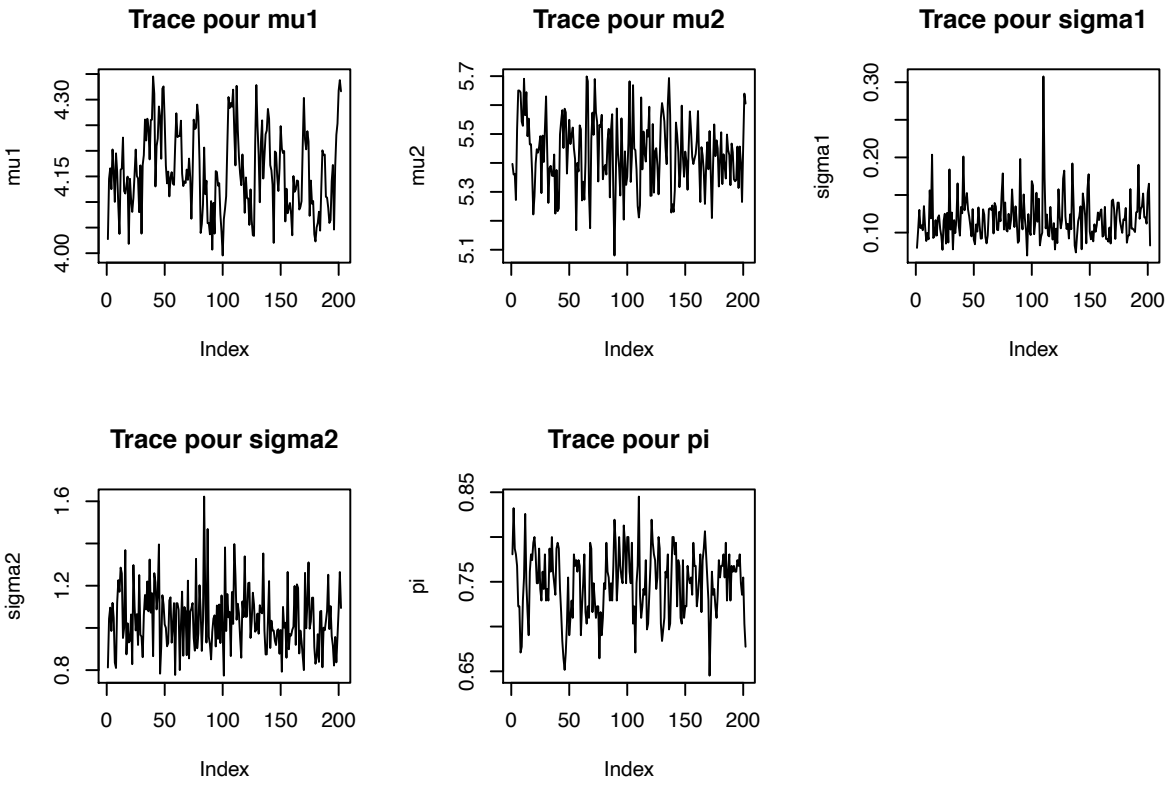
- $\pi(\mu|\sigma^2) = \mathcal{N}(\frac{\sigma_0^2 \sum x_i + \sigma^2 \mu_0}{n\sigma_0^2 + \sigma^2}; \frac{\sigma^2 \sigma_0^2}{n\sigma_0^2 + \sigma^2})$
- $\pi(\sigma^2|\mu) = IG(a + \frac{n}{2}; b + \frac{1}{2} \sum (x_i - \mu)^2)$

```r
library(MCMCpack)
mu1_0=4.5
mu2_0=6.3
sigma1_0=1
sigma2_0=1

mu1 = 5
mu2 = 7
sigma1 = 0.37
sigma2 = 0.38

moy.prior.sig<- 1
var.prior.sig<- 1
a<-2+moy.prior.sig^2/var.prior.sig
b<-moy.prior.sig*(a-1)

N = length(y)
t = 0
mu1.h = mu1
mu2.h = mu2
sigma1.h = sigma1
sigma2.h = sigma2
Delta = rep(c(0, 1), each = N/2)
pi0 = sum(Delta)/N
pi0.h = pi0
T = 50  # "burn in"
while(TRUE){
  t = t + 1
  gamma = sapply(1:N, function(i) pi0*dnorm(y[i], mu2, sigma2)/((1-pi0)*dnorm(y[i], mu1, sigma1)+pi0*dn
  ## sample Delta
  r = runif(N)
  Delta[gamma < r] = 0
  Delta[gamma >= r] = 1
  pi0 = sum(Delta)/N

  ## print info
  cat("t = ", t, "\n")
  for (i in 1:N)
    cat(Delta[i], " ")
  cat("\n")
  cat("mu1 = ", mu1,"sigma1 =", sigma1, " mu2 = ", mu2,"sigma2 =", sigma2, "pi0 = ", pi0, "\n")

  ## generate mu1 and mu2
  mu1 = rnorm(1,(sigma1_0*sum((1-Delta)*y)+sigma1*mu1_0)/(sum(1-Delta)*sigma1_0+sigma1),sqrt(sigma1*sig
  notnull=(1-Delta)*y
  sigma1 = rinvgamma(1,shape =a+sum(1-Delta)/2,scale=1/2*sum((notnull[notnull!=0]-mu1)^2)+b)
  mu2 = rnorm(1,(sigma2_0*sum(Delta*y)+sigma2*mu2_0)/(sum(Delta)*sigma2_0+sigma2),sqrt(sigma2*sigma2_0/
  notnull=Delta*y
  sigma2 = rinvgamma(1,shape =a+sum(Delta)/2,scale=1/2*sum((notnull[notnull!=0]-mu2)^2)+b)
  mu1.h = c(mu1.h, mu1)
  sigma1.h = c(sigma1.h, sigma1)
  mu2.h = c(mu2.h, mu2)
  sigma2.h = c(sigma2.h, sigma2)
```

```
  pi0.h = c(pi0.h, pi0)


  if (t > 250)
    break
}
```

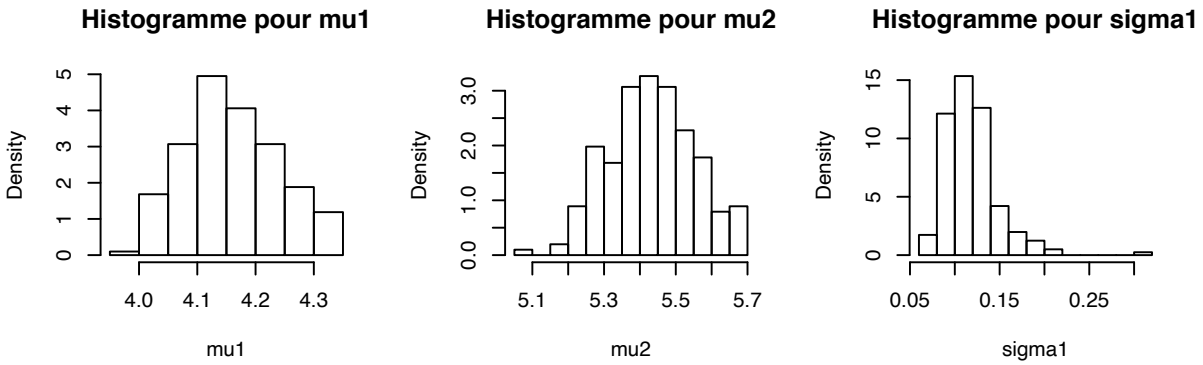
Trace de échantillon

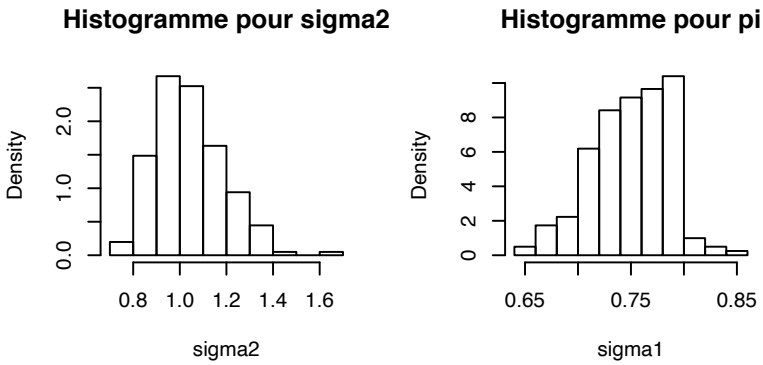
```
par(mfrow=c(2,3))
plot(mu1.h[-c(1:T)],type="l",ylab="mu1",main="Trace pour mu1")
plot(mu2.h[-c(1:T)],type="l",ylab="mu2",main="Trace pour mu2")
plot(sigma1.h[-c(1:T)],type="l",ylab="sigma1",main="Trace pour sigma1")
plot(sigma2.h[-c(1:T)],type="l",ylab="sigma2",main="Trace pour sigma2")
plot(pi0.h[-c(1:T)],type="l",ylab="pi",main="Trace pour pi")
```



Histogramme de échantillon

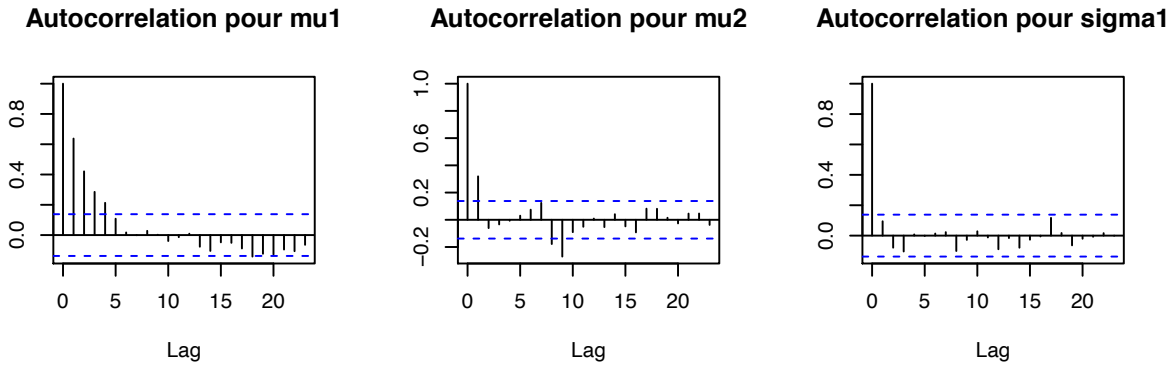
```
par(mfrow=c(2,3))
hist(mu1.h[-c(1:T)],freq=F,xlab="mu1",main="Histogramme pour mu1")
hist(mu2.h[-c(1:T)],freq=F,xlab="mu2",main="Histogramme pour mu2")
hist(sigma1.h[-c(1:T)],freq=F,xlab="sigma1",main="Histogramme pour sigma1")
hist(sigma2.h[-c(1:T)],freq=F,xlab="sigma2",main="Histogramme pour sigma2")
hist(pi0.h[-c(1:T)],freq=F,xlab="sigma1",main="Histogramme pour pi")
```

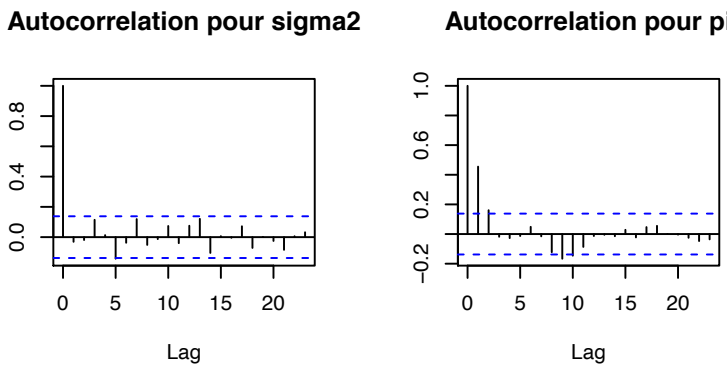




Autocorrelation de échantillon

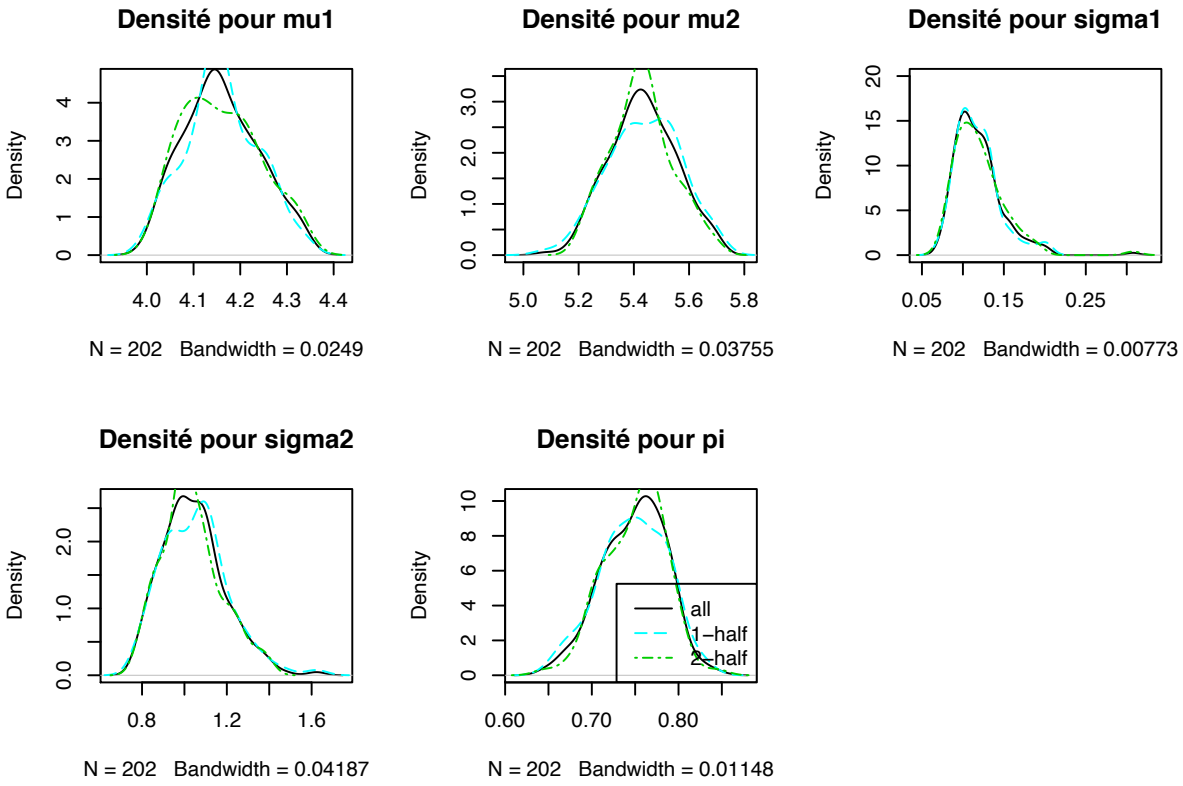
```
par(mfrow=c(2,3),pin=c(1.4,1))
acf(mu1.h[-c(1:T)],main="Autocorrelation pour mu1")
acf(mu2.h[-c(1:T)],main="Autocorrelation pour mu2")
acf(sigma1.h[-c(1:T)],main="Autocorrelation pour sigma1")
acf(sigma2.h[-c(1:T)],main="Autocorrelation pour sigma2")
acf(pi0.h[-c(1:T)],main="Autocorrelation pour pi")
```





Densité de échantillon

```
par(mfrow=c(2,3))
plot(density(mu1.h[-c(1:T)]),ylim=c(0,4.7),main="Densité pour mu1")
lines(density(mu1.h[-c(1:T,151:252)]),lty=5,col="cyan")
lines(density(mu1.h[-c(1:T,51:150)]),lty=6,col="green3")
plot(density(mu2.h[-c(1:T)]),ylim=c(0,3.5),main="Densité pour mu2")
lines(density(mu2.h[-c(1:T,151:252)]),lty=5,col="cyan")
lines(density(mu2.h[-c(1:T,51:150)]),lty=6,col="green3")
plot(density(sigma1.h[-c(1:T)]),ylim=c(0,20),main="Densité pour sigma1")
lines(density(sigma1.h[-c(1:T,151:252)]),lty=5,col="cyan")
lines(density(sigma1.h[-c(1:T,51:150)]),lty=6,col="green3")
plot(density(sigma2.h[-c(1:T)]),main="Densité pour sigma2")
lines(density(sigma2.h[-c(1:T,151:252)]),lty=5,col="cyan")
lines(density(sigma2.h[-c(1:T,51:150)]),lty=6,col="green3")
plot(density(pi0.h[-c(1:T)]),main="Densité pour pi")
lines(density(pi0.h[-c(1:T,151:252)]),lty=5,col="cyan")
lines(density(pi0.h[-c(1:T,51:150)]),lty=6,col="green3")
legend("bottomright",legend=c("all","1-half","2-half"),lty=c(1,5,6),col=c("black","cyan","green3"))
```


Densité pour mu1
Density
0 1 2 3 4
4.0 4.1 4.2 4.3 4.4
N = 202 Bandwidth = 0.0249
Densité pour mu2
Density
0.0 1.0 2.0 3.0
5.0 5.2 5.4 5.6 5.8
N = 202 Bandwidth = 0.03755
Densité pour sigma1
Density
0 5 10 15 20
0.05 0.15 0.25
N = 202 Bandwidth = 0.00773
Densité pour sigma2
Density
0.0 1.0 2.0
0.8 1.2 1.6
N = 202 Bandwidth = 0.04187
Densité pour pi
Density
0 2 4 6 8 10
0.60 0.70 0.80
all
1−half
2−half
N = 202 Bandwidth = 0.01148