Projet Statistique non paramétrique

AMOVIN-ASSAGBA M. Martial

06 Novembre 2018

On considère le modèle de régression,

$$Y_i = g(\frac{i}{n}) + \varepsilon_i, \quad 1 \leq i \leq n.$$

Avec $\varepsilon_1, \ldots, \varepsilon_n$ des variables aléatoires iid centrées et de variance σ^2. On définit, $\hat{g}$ l'estimateur de g, par

$$\hat{g} = \frac{1}{nh} \sum_{i=1}^{n} Y_i K\left(\frac{x - X_i}{h}\right)$$

h est la fenêtre et K est un noyau pair et à support compact. L'objectif de ce projet est d'étudier empiriquement un bon choix de la fenêtre h. On prendra par la suite

$$g(x) = \sin(2\pi x), \quad \varepsilon_i \sim \mathcal{N}(0, \sigma^2)$$

1

Représenter sur un même graphique le nuage des points $(i/n, Y_i)_{1 \leq i \leq n}$, la fonction g et l'estimateur $\hat{g}$ pour un choix de K et de σ^2 que vous préciserez

Sous R, l'estimateur de Nadaraya-Watson (qui est un estimateur par polynôme locaux de degré zéro) peut s'implémenter à partir de la fonction *locpoly* de la librairie *KernSmooth*.

L'estimateur de Nadaraya-Watson vu au cours est de la forme

$$\hat{g}^{NW}(x) = \frac{\sum_{i=1}^{n} Y_i K\left(\frac{x - X_i}{h}\right)}{\sum_{i=1}^{n} K\left(\frac{x - X_i}{h}\right)}$$

Pour trouver l'estimateur $\hat{g}$ définit précedemment, il faut donc multiplier $\hat{g}^{NW}(x)$ par

$$\hat{g}^{D} = \frac{1}{nh} \sum_{i=1}^{n} Y_i K\left(\frac{x - X_i}{h}\right)$$

$\hat{g}^{D}$ est l'estimateur de la densité. Sous R elle est obtenue grâce à la fonction *density*

Nous prendrons dans notre travail, K le noyau gaussien (elle est paire et infiniment dérivable) et $\sigma^2 = 0.3$.

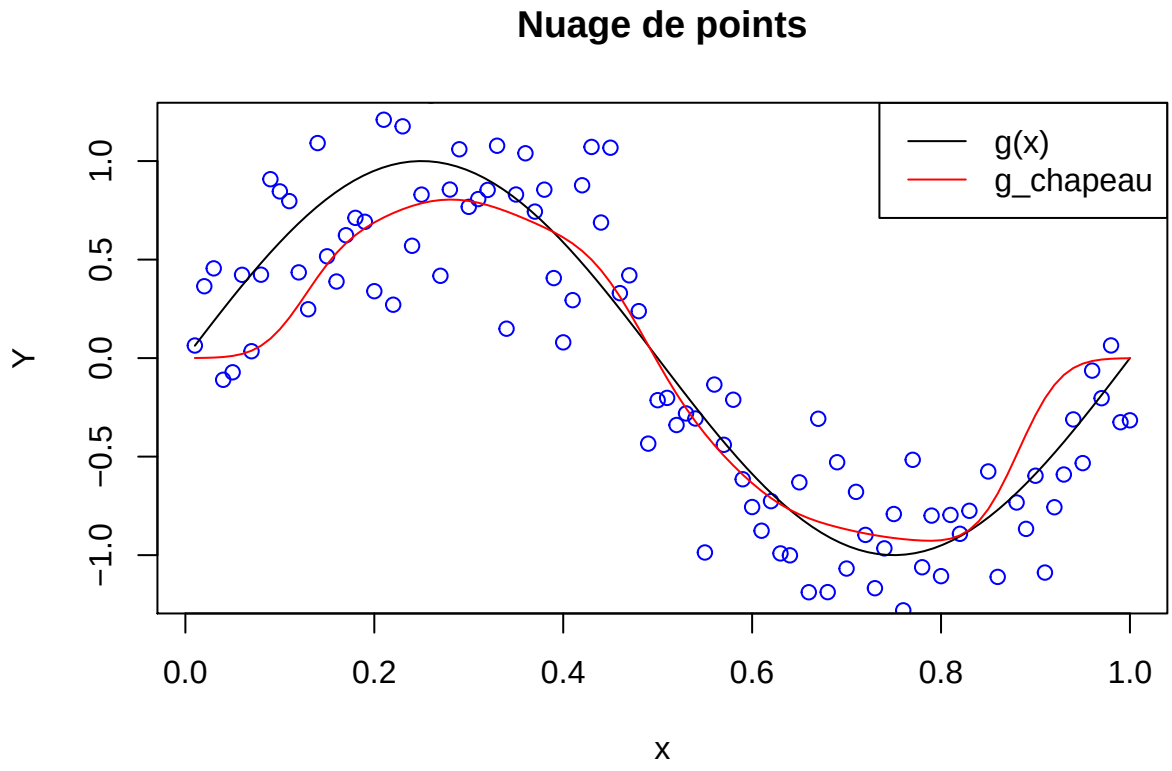
```
require(KernSmooth)
```

```
## Loading required package: KernSmooth
```

```
## KernSmooth 2.23 loaded
## Copyright M. P. Wand 1997-2009
```

```
set.seed(11111)

# fonction g
g<- function(x){sin(2*pi*x)}
#initialisation de n
```

```r
n<-100

#Création des vecteurs x et epsilon
x<-1:n/n;eps<-rnorm(n,0,0.3)

#graphique du nuage de ppoints
Y<-g(x)+eps
plot(x,Y,type="p",ylim=c(-1.2,1.2),ylab="Y",col="blue", main="Nuage de points")
#Ajoût de la courbe de g sur le même graphique
curve(g(x),add=TRUE)

#Estimation par polynômes locaux de g et tracée de la courbe
nw<-locpoly(x,Y,degree = 0,gridsize = n, bandwidth=0.05)$y*density(x,bw=0.05,kernel="gaussian",n=n)$y
lines(x,nw, type="l", col="red")
legend("topright", c("g(x)","g_chapeau"), col = c("black", "red"),lty=1)
```

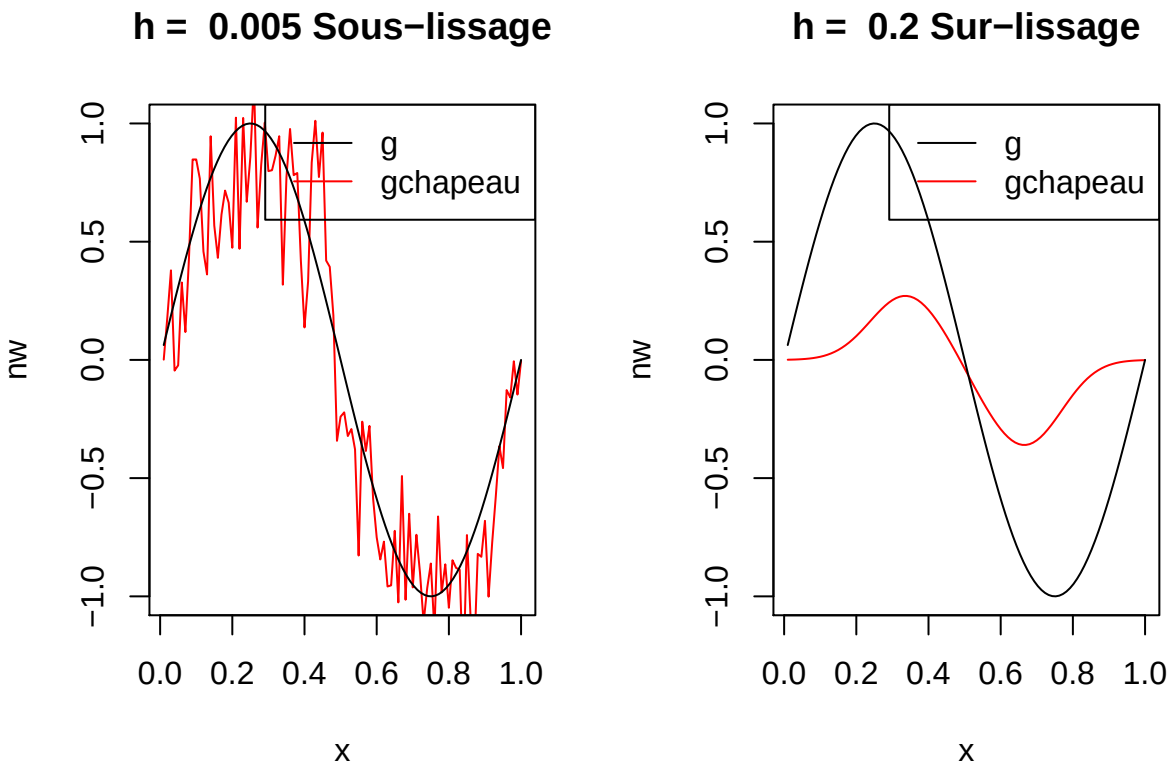


2

Visualisez, selon des différentes valeurs de h, la situation de sous et de sur-lissage.

```r
par(mfrow=c(1,2))
#Vecteur des fenêtres
h=c(0.005, 0.2)
p=c("Sous-lissage","Sur-lissage")
```

```r
for (i in 1:2){
  nw<-locpoly(x,Y,degree = 0, gridsize = n, bandwidth=h[i])$y*density(x,bw=h[i],kernel="gaussian",n=n)$y
  plot(x,nw, type="l",lty=1,lwd=1, col="red",ylim=c(-1,1), main=paste("h = ",h[i],p[i]))
  curve(g(x),add=TRUE)
  legend("topright", c("g","gchapeau"), col = c("black", "red"),lty=1)

}
```



On remarque bien avec les graphes ci-dessus que pour une valeur faible de la fenêtre, il y a un phénomène de sous-lissage (biais petit, variance grande). Au contraire quand la fenêtre est beaucoup plus grande, il ya un phénomène de sur-lissage qui se produit (biais grand, variance petite). Il est donc important de faire un compromis entre le biais et la variance, pour trouver un meilleur choix de la fenêtre.

3

Ecrire un programme qui calcule la valeur optimale du paramètre de lissage en fonction du ASE (Average square error)

$$ASE(h) = \frac{1}{n}\sum_{i=1}^{n}((\hat{r}(x_i) - r(x_i))^2)$$

$$\hat{h}_0 = argmin_{h>0} ASE(h)$$

Pour ce faire, nous allons écrire une fonction qui retourne un vecteur contenant la valeur optimale du

paramètre de lissage ($\hat{h}_0$) et $ASE(\hat{h}_0)$. Nous allons ensuite prendre pour la fenêtre, une séquence entre 0.005 et 0.2 avec un pas de 0.001 pour choisir parmi cette séquence, la fenêtre qui minimise ASE.

```
#Fonction qui calcule ASE
ase<-function(g_hat,g){
  mean((g_hat-g)^2)
}

#Fonction qui détermine la valeur optimale h_0 et ASE(h_0)
optimale_ASE <- function(x,Y,h,n){
  vect=c()
  for(i in h){

    nw=locpoly(x,Y,degree = 0,gridsize = n, bandwidth = i)$y*density(x,bw=i,kernel="gaussian",n=n)$y
    vect=c(vect,ase(nw,g(x)))
  }
  return(cbind(h,vect)[which.min(vect),])

}

#valeurs du paramètre de lissage
h<-seq(0.005,0.2,by=1/1000)

#Rentrée des paramètres dans la fonction pour
#obtenir le vecteur optimal h_0 et ASE(h_0)
opt<-optimale_ASE(x,Y,h,n=100)

cat("La valeur optimale du paramètre de lissage en fonction du ASE est h0 = ",opt[1], "\n")
```

```
## La valeur optimale du paramètre de lissage en fonction du ASE est h0 =  0.027
```

```
cat("Avec cette valeur optimale, on trouve ASE =",opt[2])
```

```
## Avec cette valeur optimale, on trouve ASE = 0.0213737
```

4

Même question, en remplacant ASE(h) pour le critère de validation croisée CV(h)

$$CV(h) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{\hat{r}(x_i) - Y_i}{1 - L_{i,i}}\right)^2 \quad \text{avec} \quad L_{i,i} = \frac{K(0)}{nh}$$

$$\hat{h} = argmin_{h>0} CV(h)$$

Puisque nous avons choisi le noyau gaussien au début de notre travail, $K(0) = 1/\sqrt{2\pi}$

```
#fonction CV(h)
cvh=function(g_hat,y,n,h){
  mean(((g_hat-y)/(1-(1/(sqrt(2*pi)*n*h))))^2)
}

#Fonction qui détermine la valeur optimale h_chapeau et CV(h_chapeau)
optimale_CV<-function(x,Y,h,n){
  vect1=c()
  for(i in h){
```

```
    nw=locpoly(x,Y,degree = 0, gridsize = n, bandwidth = i)$y*density(x,bw=i,kernel="gaussian",n=n)$y
    vect1=c(vect1,cvh(nw,Y,n,i))
  }

  return(cbind(h,vect1)[which.min(vect1),]) # Récupération de la valeur optimale
}

n=100
h=seq(0.005,0.2,by=1/1000)
opt2=optimale_CV(x,Y,h,n)

cat("La valeur optimale du paramètre de lissage en fonction du CV(h) est h_chapeau = ",opt2[1], "\n")
```

```
## La valeur optimale du paramètre de lissage en fonction du CV(h) est h_chapeau =  0.023
```

```
cat("Avec cette valeur optimale, on trouve CV =",opt2[2])
```

```
## Avec cette valeur optimale, on trouve CV = 0.1069961
```

5

Illustrer le comportement asymptotique lorsque n tend vers l'infini de:

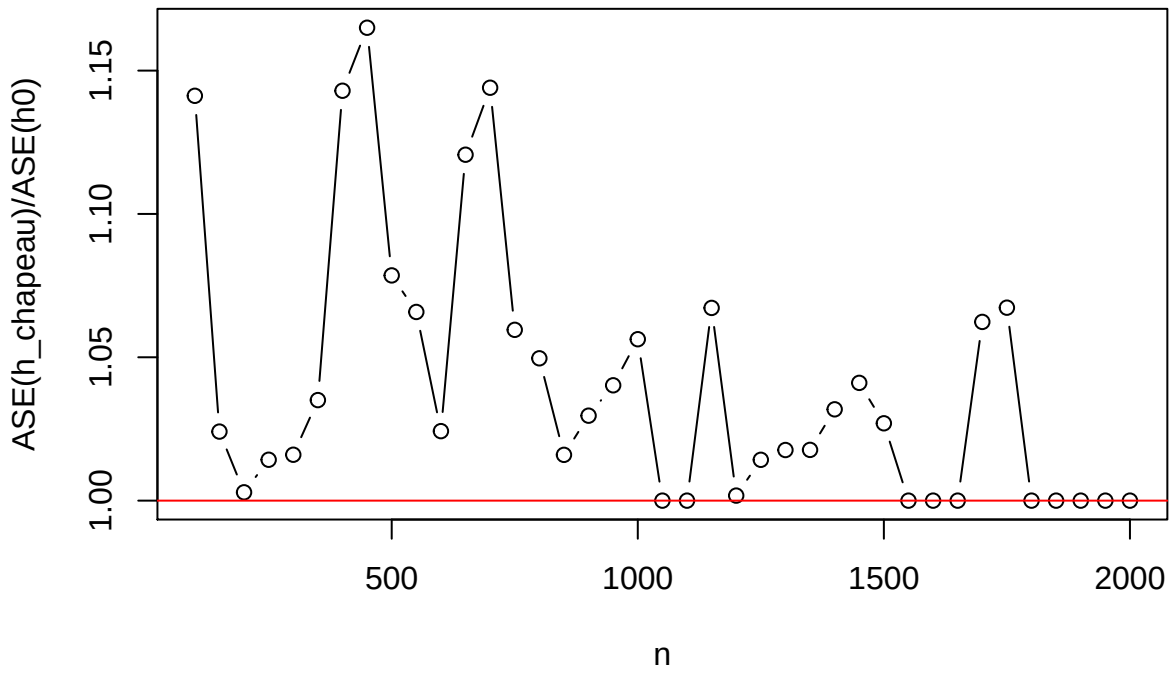
$$\frac{ASE(\hat{h})}{ASE(\hat{h}_0)}$$

```
#Séquence des vecteurs n et h
n<-seq(100,2000,by=50)
h<-seq(0.005,0.2,by=1/1000)

#création des vecteurs qui vont contenir respectivement les
#valeurs de h_0, h_chapeau, ASE(h_0) et ASE(h_chapeau)
tab_h0<- c()
tab_hchap<- c()
tab_ase_h0 <- c()
tab_ase_h_chap <-c()


for(i in n){

  set.seed(11111)

  #Création des vecteurs x et epsilon
  x<-1:i/i;eps<-rnorm(i,0,0.1)
  #Vecteur Y
  Y<-g(x)+eps

  #Calcul du vecteur optimal (h_0, ASE(h_0) )
  tab_h0<- c(tab_h0, optimale_ASE(x,Y,h,i)[1])
  tab_ase_h0<- c(tab_ase_h0, optimale_ASE(x,Y,h,i)[2])
  h_chap<- optimale_CV(x,Y,h,i)[1]
  tab_hchap<- c(tab_hchap, h_chap)
```

```
  #Calul du ASE avec h_chapeau
  nw=locpoly(x,Y,degree = 0,gridsize = i, bandwidth = h_chap)$y*density(x,bw=h_chap,kernel="gaussian",n=
  tab_ase_h_chap=c(tab_ase_h_chap,ase(nw,g(x)))

}
```

Tracée de la courbe de $ASE(\hat{h})/ASE(\hat{h}_0)$

```
plot(n,tab_ase_h_chap/tab_ase_h0, type="b", ylab="ASE(h_chapeau)/ASE(h0)")
abline(h=1, col="red")
```



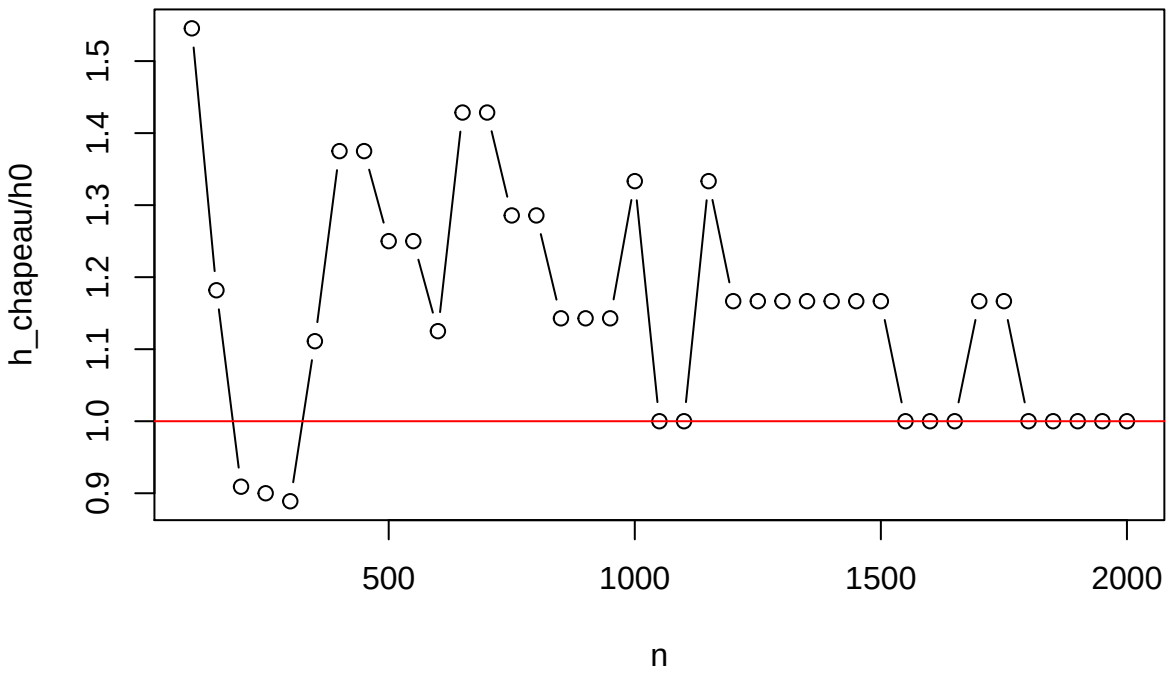
Conclusion: Quand n croît (tend vers l'infini), $ASE(\hat{h})/ASE(\hat{h}_0)$ tend vers 1. C'est à dire, avec un nombre croissant d'observations $ASE(\hat{h})$ approxime (ou est très proche de) $ASE(\hat{h}_0)$, et donc $\hat{h}$ est une bonne estimation de la fenêtre quand n est grand, (puisque $\hat{h}_0$ est théoriquement inconnu car l'ASE aussi l'est).

6

Illustrer le comportement asymptotique lorsque n tend vers l'infini de :

$$\frac{\hat{h}}{\hat{h}_o}$$

```
plot(n,tab_hchap/tab_h0, type="b", ylab="h_chapeau/h0")
abline(h=1, col="red")
```



Conclusion: Quand n croît (tend vers l'infini),$\hat{h}/\hat{h}_0$ tend vers 1, ce qui veut dire que l'approximation de $\hat{h}$ tend vers $\hat{h}_0$. Pour n grand Le critère de validation croisée donne une meilleure estimation de la fenêtre.

7

Vérifier, par simulations, que $n^{(3/10)}(\hat{h} - \hat{h}_0)$ a un comportement gaussien.

Nous fixons n à 1000 et à chaque itération, nous allons calculer la valeur de $\hat{h}$ et $\hat{h}_0$. Nous allons faire 50 itérations (on obtiendra alors un échantillon de taille 50).

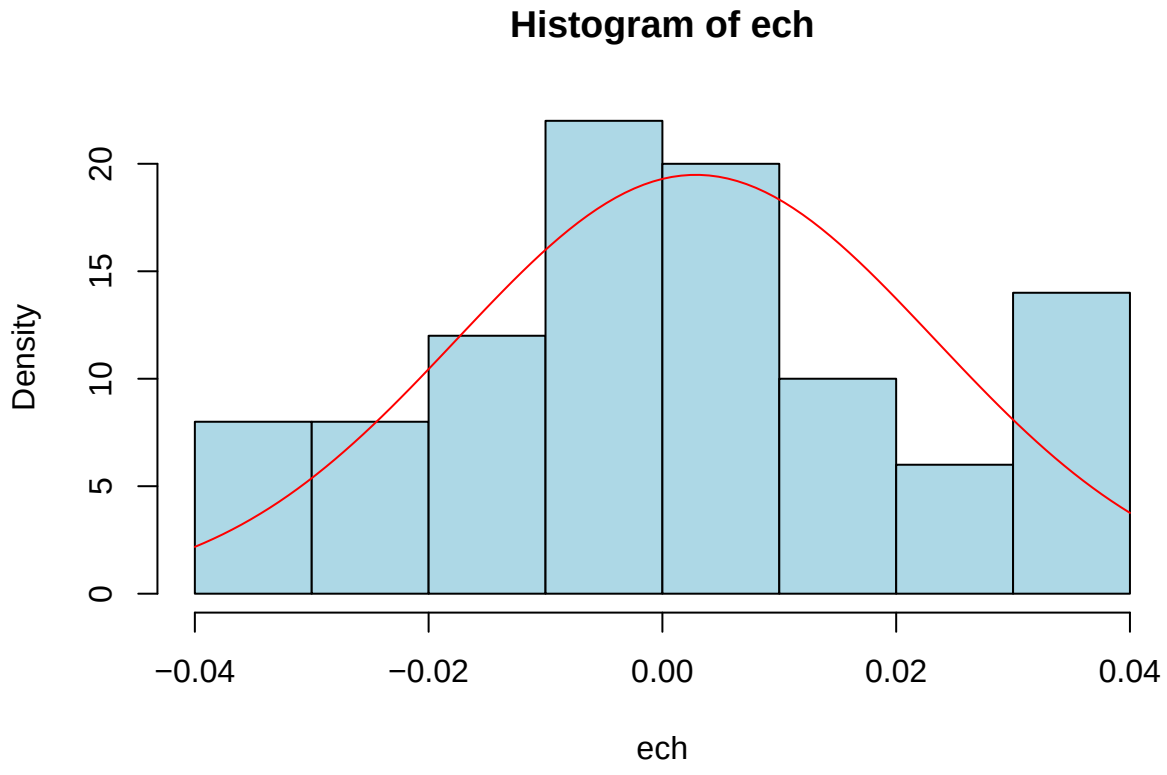
Ensuite nous allons et simuler tracer avec la fonction *dnorm* des variables aléatoires suivant la loi normale de moyenne: la moyenne de l'échantillon et d'écart type: celui de l'échantillon.

```
set.seed(12345)
#Séquence des vecteurs n et h
n<-1000
h<-seq(0.005,0.2,by=1/1000)

#création des vecteurs qui vont contenir respectivement à nouveau les
#valeurs de h_0, h_chapeau, ASE(h_0) et ASE(h_chapeau)
tab_h0<- c()
tab_hchap<- c()
tab_ase_h0 <- c()
tab_ase_h_chap <-c()
x<-1:n/n #vecteur x

for(i in 1:50){
```

```
  #Création du vecteur epsilon
  eps<-rnorm(n,0,0.3)
  #Vecteur Y
  Y<-g(x)+eps

  #Détermination du vecteur optimal (h_0, ASE(h_0) )
  tab_h0<- c(tab_h0, optimale_ASE(x,Y,h,n)[1])
  tab_ase_h0<- c(tab_ase_h0, optimale_ASE(x,Y,h,n)[2])
  h_chap<- optimale_CV(x,Y,h,n)[1]
  tab_hchap<- c(tab_hchap, h_chap)

  #Calul du ASE avec h_chapeau
  nw=locpoly(x,Y,degree = 0,gridsize = n, bandwidth = h_chap)$y*density(x,bw=h_chap,kernel="gaussian",n=
  tab_ase_h_chap=c(tab_ase_h_chap,ase(nw,g(x)))

}
```

Tracée de l'histogramme de $n^{(3/10)}(\hat{h} - \hat{h}_0)$

```
n=1000
ech= n^(3/10)*(tab_hchap - tab_h0)
hist(ech, col = "lightblue",  prob=TRUE)
curve(dnorm(x, mean=mean(ech), sd=sd(ech)), col=2,add=TRUE)
```



L'histogramme obtenu ressemble bien à celui d'une loi normale.

Faisons un test de shapiro pour vérifier de manière plus rigoureuse que $n^{(3/10)}(\hat{h} - \hat{h}_0)$ suit une loi normlale.

```r
shapiro.test(ech)
```

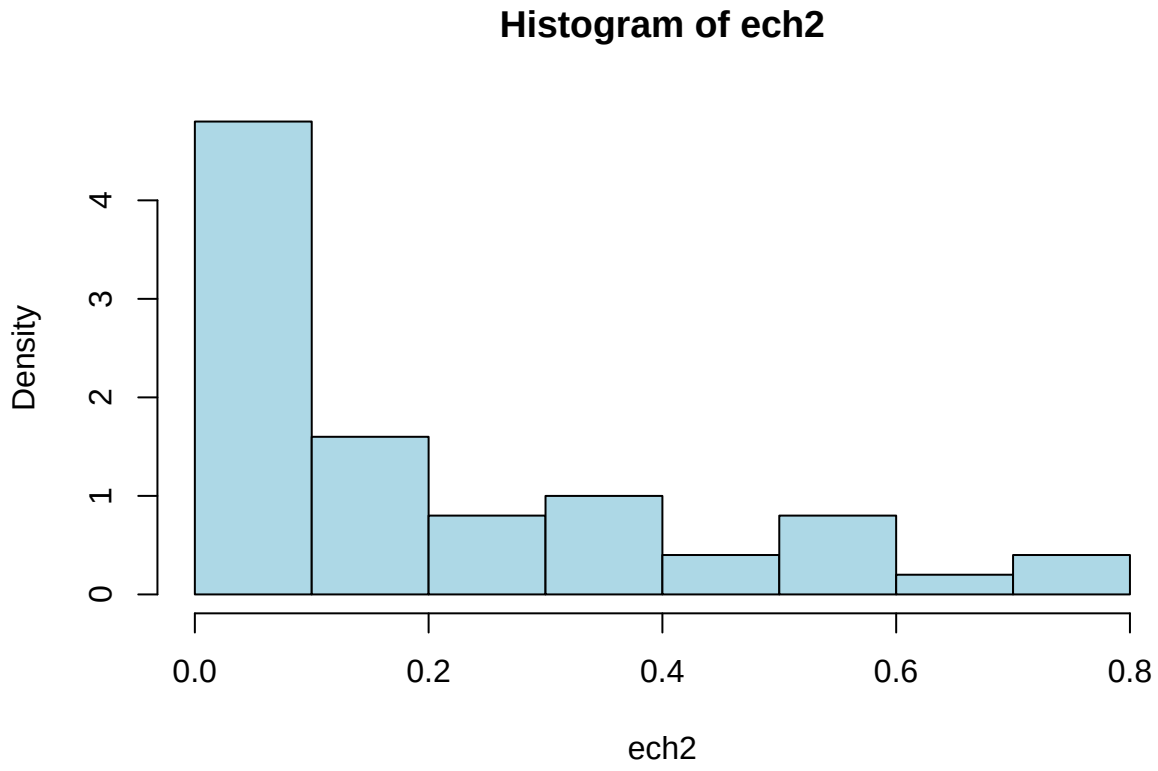
```
##
##  Shapiro-Wilk normality test
##
## data:  ech
## W = 0.95606, p-value = 0.0608
```

Le test de Shapiro, nous renvoie une p-value non significative (p-value > 0.05). On accepte l'hypothèse de normalité de $n^{(3/10)}(\hat{h} - h_0)$.

8.

Que peut être la loi asymptotique de $n(ASE(\hat{h}) - ASE(\hat{h}_0))$

```r
ech2=n*(tab_ase_h_chap - tab_ase_h0 )
#plot(n,ech, type="o")
hist(ech2,col = "lightblue", prob=TRUE)
```



L'histogramme obtenu ressemble à celui d'une loi de $\chi 2$

La loi asymptotique de $n(ASE(\hat{h}) - ASE(\hat{h}_0))$, peut être une loi de $\chi 2$.

9.

Conclure quant au critère CV(h).

En théorie, la fonction de régression r(x), est inconnue. C'est donc impossible de minimiser ASE, car il est aussi inconnu puisqu'il dépend de la fonction de régression. Dans notre TP, on constate qu'avec un nombre croissant d'observations, $\hat{h}$ est très proche de $\hat{h}_0$, et $(ASE(\hat{h})$ aussi très proche de $ASE(\hat{h}_0)$. Ainsi, choisir la fenêtre qui minimise le critère CV(h) est donc une très bonne règle de sélection de la fenêtre dans l'estimation de la régression par noyau.