Estimation fonctionnelle : estimation de la densité

Dhafer Malouche

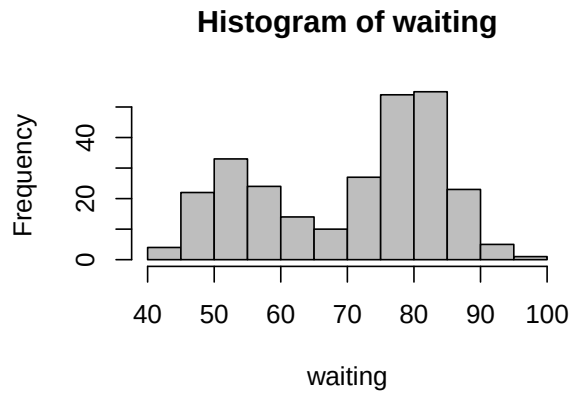
Ecole Supérieure de la Statistique
et de l'Analyse de l'Information de Tunis

3ième Année, 2012-2013

Exemple

- On considère les temps d'attentes entre deux éruptions
- Entre 01-08 et 15-08 1995 on a observé 272 temps d'attentes (en minutes) et durées de l'éruption
- Données `faithful` dans le package `datasets`

```
> library(datasets)
> data(faithful)
> faithful[1:3,]
  eruptions waiting
1     3.600      79
2     1.800      54
3     3.333      74
> attach(faithful)
> hist(waiting,col='gray')
```



Estimation de la densité

- C'est une "approximation" de la densité de probabilité (univariée ou multivariée)
- Deux méthodes déjà connues :
  - Tracer un histogramme est une méthode d'estimation de la densité.
  - Supposer que l'échantillon a été tiré selon un modèle paramétrique : estimer les parmétres à partir de l'échantillon.
- **Objectif** : utiliser une méthode dite "non-paramétrique" pour l'estimation de la densité.

La méthode des noyaux.

- On dit que la v.a. X a une densité de probabilité f si pour tout x

$$f(x) = \lim_{h \to 0} \frac{1}{2h} \mathbb{P}(x - h < X < x + h) \text{ existe.}$$

- Pour h fixé, un estimateur *naive* de $\mathbb{P}(x - h < X < x + h)$ est de "compter" le nombre des $x_1, x_2, \ldots, x_n$ qui appartiennent à $]x - h, x + h[$. Donc

$$\widehat{f}(x) = \frac{1}{2nh} \sum_{i=1}^{n} \mathbb{1}_{]x-h,x+h[}(x_i)$$

La méthode des noyaux.

- Si on pose

$$W(x) = \begin{cases} \frac{1}{2} & \text{si } |x| < 1 \\ 0 & \text{sinon} \end{cases}$$

alors

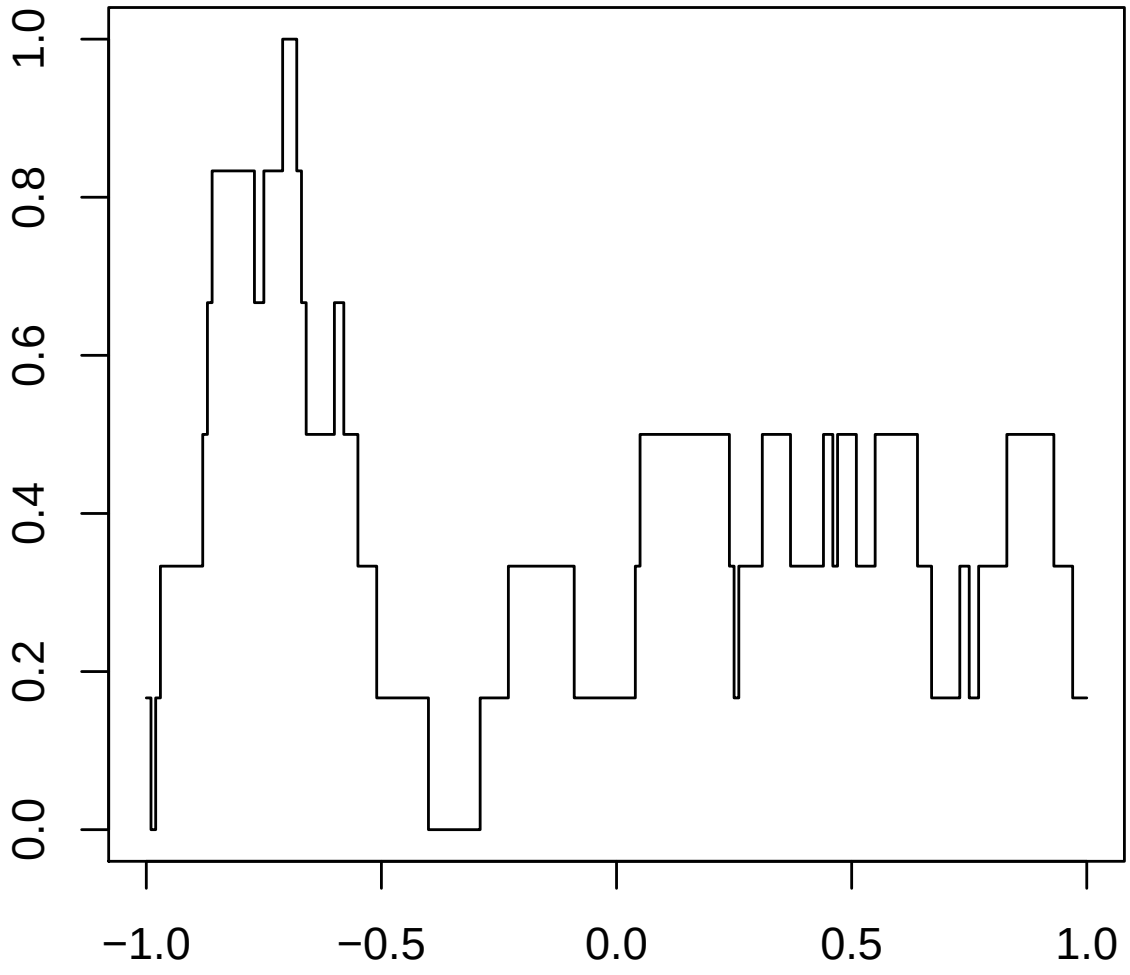
$$\widehat{f}(x) = \frac{1}{h} \sum_{i=1}^{n} \frac{1}{n} W\left(\frac{x - x_i}{h}\right)$$

- Remarques
  - W est la fonction densité d'une loi uniforme sur $[-1, 1]$.
  - $\widehat{f}$ n'est pas une fonction continue.

La méthode des noyaux.

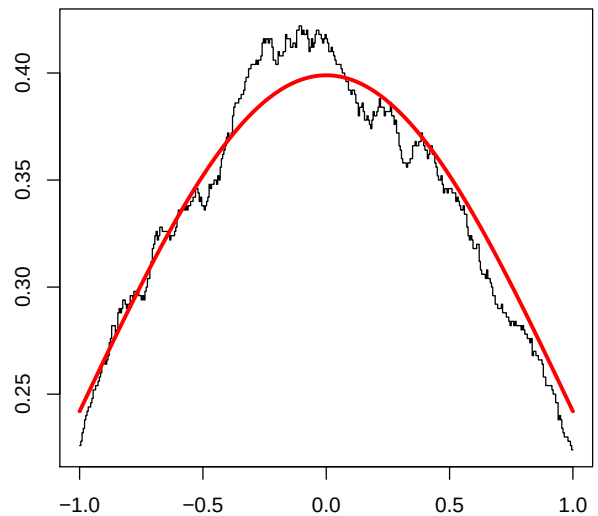
```
> n=30
> x=rnorm(n)
> W=function(x) 1/2*(abs(x)<1)
> fhat=function(y,x,h,n){
+   tt=(y-x)/h
+   sum(sapply(tt,function(t) W(t)))/n/h
+ }
> y=seq(-1,1,by=0.01)
> h=0.1
> n=30
> sapply(y,function(t) fhat(t,x,h,n))[1:5]

[1] 0.1666667 0.0000000 0.1666667 0.3333333 0.3333333

> plot(y,sapply(y,function(t) fhat(t,x,h,n)),
+      type='s',xlab='',ylab='')
```

La méthode des noyaux.



La méthode des noyaux.

```
> set.seed(123)
> n=500
> x=rnorm(n)
> h=.5
> y=seq(-1,1,by=0.005)
> yhat=sapply(y,function(t)
+     fhat(t,x,h,n))
> plot(y,yhat,
+      type='s',xlab='',ylab='')
> curve(dnorm(x),-1,1,lwd=3,
+       add=T,col='red')
```



Le noyau

L'estimateur par la méthode des noyaux (ou par noyau) de la densité f est définie par

$$\widehat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right)$$

où $h > 0$ est appelée la *fenêtre* (*bandwidth*) et K est la fonction *noyau* (*kernel*) qui vérifie :

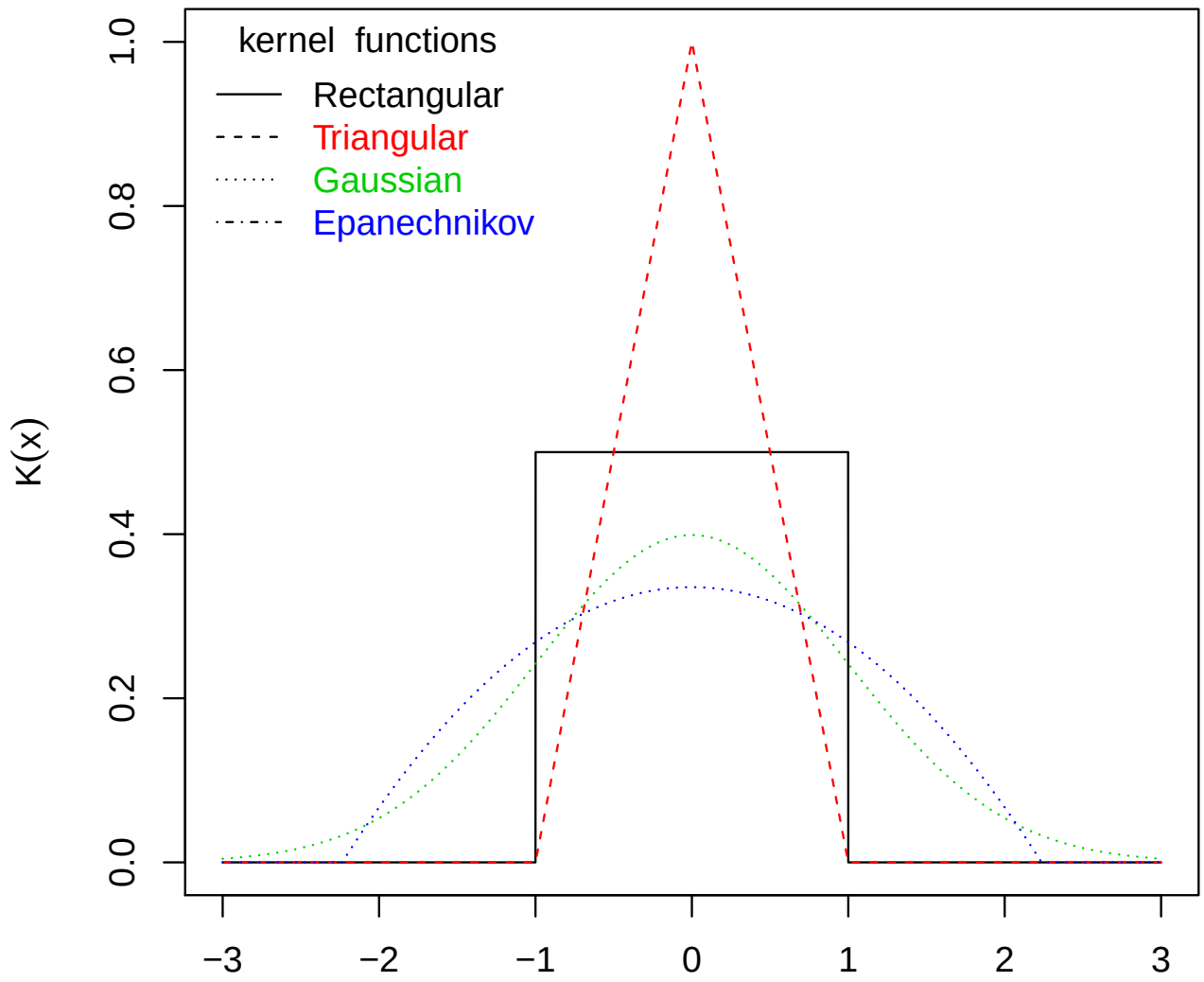
$$\begin{cases} K(x) \geq 0 \\ \int K(x)dx = 1 \end{cases}$$

et $\int xK(x)dx = 0$ et $\int x^2 K(x)dx < \infty$.

Exemples de noyaux

- Rectangulaire
$$K = (1/2)\mathbb{1}_{|x|<1}$$
- Triangulaire
$$K = (1 - |x|)\mathbb{1}_{|x|<1},$$
- Gaussienne
$$K(x) = (2\pi)^{-1/2} \exp(-x^2/2).$$
- Epanechnikov
$$K(x) = \frac{3}{4}(1 - \frac{1}{5}t^2)/\sqrt{5}\mathbb{1}_{|t|<\sqrt{5}}.$$

Représentation des noyaux



Mise en œuvre sous R, Noyau Gaussian

Utilision de la fonction density

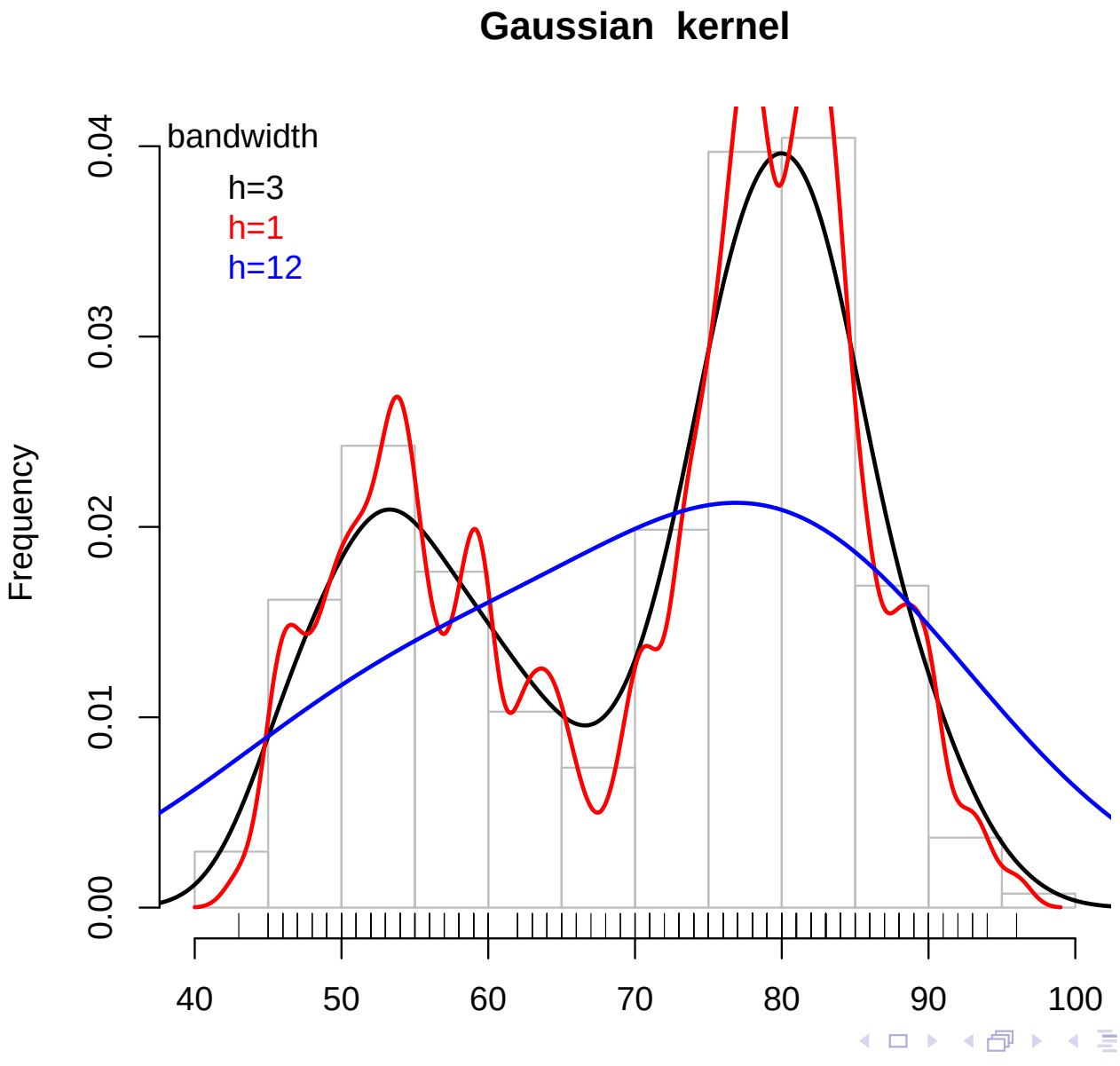
```
> x<-faithful$waiting
> xhat=density(x,bw=3,window="gaussian")
> xhat

Call:
        density.default(x = x, bw = 3, window = "gaussian")
Data: x (272 obs.);        Bandwidth 'bw' = 3
       x                    y
 Min.   : 34.00   Min.   :6.510e-06
 1st Qu.: 51.75   1st Qu.:3.191e-03
 Median : 69.50   Median :1.261e-02
 Mean   : 69.50   Mean   :1.407e-02
 3rd Qu.: 87.25   3rd Qu.:2.025e-02
 Max.   :105.00   Max.   :3.963e-02
```

Mise en œuvre sous R, Noyau Gaussian

Tracer l'estimation de la densité.

```
> hist(x, xlab = "Waiting times (in min.)",
+      ylab = "Frequency", probability = TRUE,
+      main = "Gaussian kernel",
+       border = "gray")
> lines(density(x, bw=3),lwd=2)
> lines(density(x, bw=1),lwd=2,col='red')
> lines(density(x, bw=12),lwd=2,col='blue')
> rug(x)
```

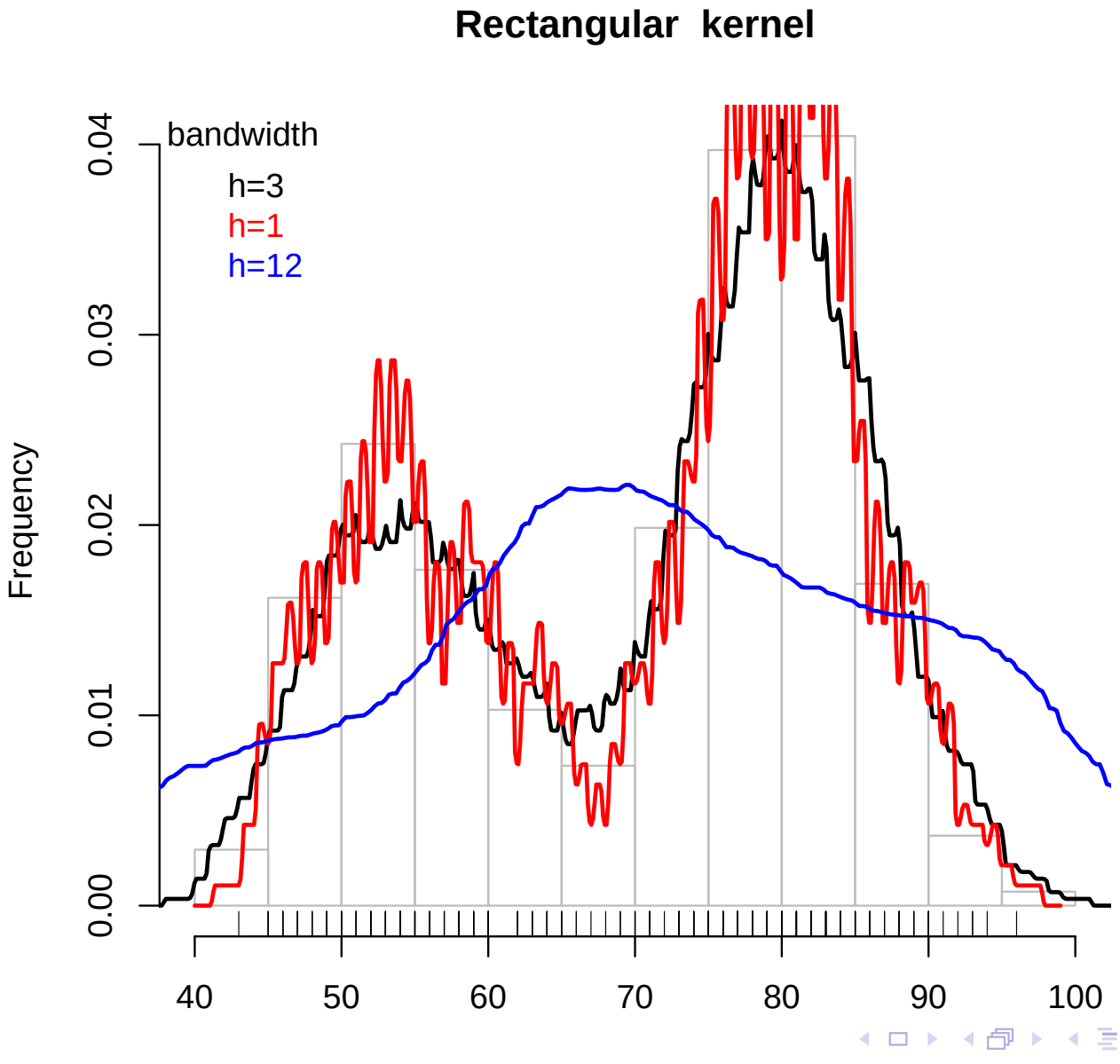
Mise en œuvre sous R, Noyau Gaussian



Mise en œuvre sous R, Noyau Rectangulaire

```
> x<-faithful$waiting
>  hist(x,xlab="Waiting  times  (in  min.)",ylab="Frequency
+  probability=TRUE,  main="Rectangular  kernel",
+  border  =  "gray")
> lines(density(x,bw=3,window='rectangular'),
+       lwd=2)
> lines(density(x,bw=1,window='rectangular'),
+       lwd=2,col='red')
> lines(density(x,bw=12,window='rectangular'),
+       lwd=2,col='blue')
> rug(x)
```

Mise en œuvre sous R, Noyau Rectangulaire



Biais de l'estimateur par noyau

On considère l'estimateur par noyau

$$\widehat{f}(x) = \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right), \quad \forall x$$

Proposition

Quand $h \to 0$ *et* $n \to +\infty$

$$E\left(\widehat{f}(x)\right) = f(x) + \frac{h^2}{2} f''(x) k_2 + o(h^2)$$

$$Var\left(\widehat{f}(x)\right) = \frac{1}{nh} f(x) j_2 + o\left(\frac{1}{nh}\right)$$

où $k_2 = \int z^2 K(z) dz$ *et* $j_2 = \int K(z)^2 dz$.

Précision de l'estimateur par noyau

- Erreur quadratique moyenne : (*mean square error*, MSE)

$$\begin{aligned} \text{MSE}\left(\widehat{f}(x)\right) &= E\left((\widehat{f}(x) - f(x))^2\right) \\ &= \text{Biais}^2\left(\widehat{f}(x)\right) + \text{Var}\left(\widehat{f}(x)\right) \end{aligned}$$

- Erreur quadratique moyenne intégrée : (*mean integrated squared error*, MISE)

$$\begin{aligned} \text{MISE}\left(\widehat{f}\right) &= E\left(\int (\widehat{f}(x) - f(x))^2 dx\right) \\ &= \int \text{Biais}^2\left(\widehat{f}(x)\right) dx + \int \text{Var}\left(\widehat{f}(x)\right) dx \end{aligned}$$

Quand $h \to 0$ et $n \to +\infty$

$$\text{MSE}\left(\widehat{f}(x)\right) \approx \frac{1}{4} h^4 k_2^2 f''(x)^2 + \frac{1}{nh} f(x) j_2$$

et en intégrant le MSE, on a

$$\text{MISE}\left(\widehat{f}\right) \approx \frac{1}{4} h^4 k_2^3 \beta(f) + \frac{1}{nh} j_2$$

où $\beta(f) = \int f''(x) dx$

Fenêtre Optimale

Si on dérive le MISE($\widehat{f}$) par rapport à h

$$\frac{d\text{MISE}}{dh} = h^3 k_2^2 \beta(f) - \frac{1}{nh} j_2.$$

Donc la fenêtre optimale

$$h_{opt} = \left(\frac{1}{n} \frac{\gamma(K)}{\beta(f)} \right)^{1/5}$$

où $\gamma(K) = j_2 k_2^{-2}$. et me MISE optimal

$$\text{MISE}_{opt}(\widehat{f}) = \frac{5}{4} \left(\frac{\beta(f) j_2^4 k_2^2}{n^4} \right)^{1/5}$$

Le noyau Optimal

- le MISE peut être minimisé par rapport au Noyau
- Wand et Jones 1995 montrent que le noyau d'Epanechnikov est optimal
- On mesure l'efficacité d'un noyau

$$\text{Eff}(K) = \left(\frac{\text{MISE}_{opt}(\widehat{f}) \text{ avec } K_{Ep}}{\text{MISE}_{opt}(\widehat{f}) \text{ avec } K} \right)^{5/4}.$$

- Exercice : Calculer l'efficacité de chaque noyau.

Sélection de la fenêtre

- Sélection subjectif : faire plusieurs choix de h.
- Sélection par rapport à un d.p. (le cas d'un noyau Gaussien) et par rapport à une d.p. Normale, Silverman (1986) propose

$$\widehat{h}_{opt} = \frac{0.9\widehat{\sigma}}{n^5}$$

où $\widehat{\sigma} = \min(s, \text{IQR}/1.34)$, où

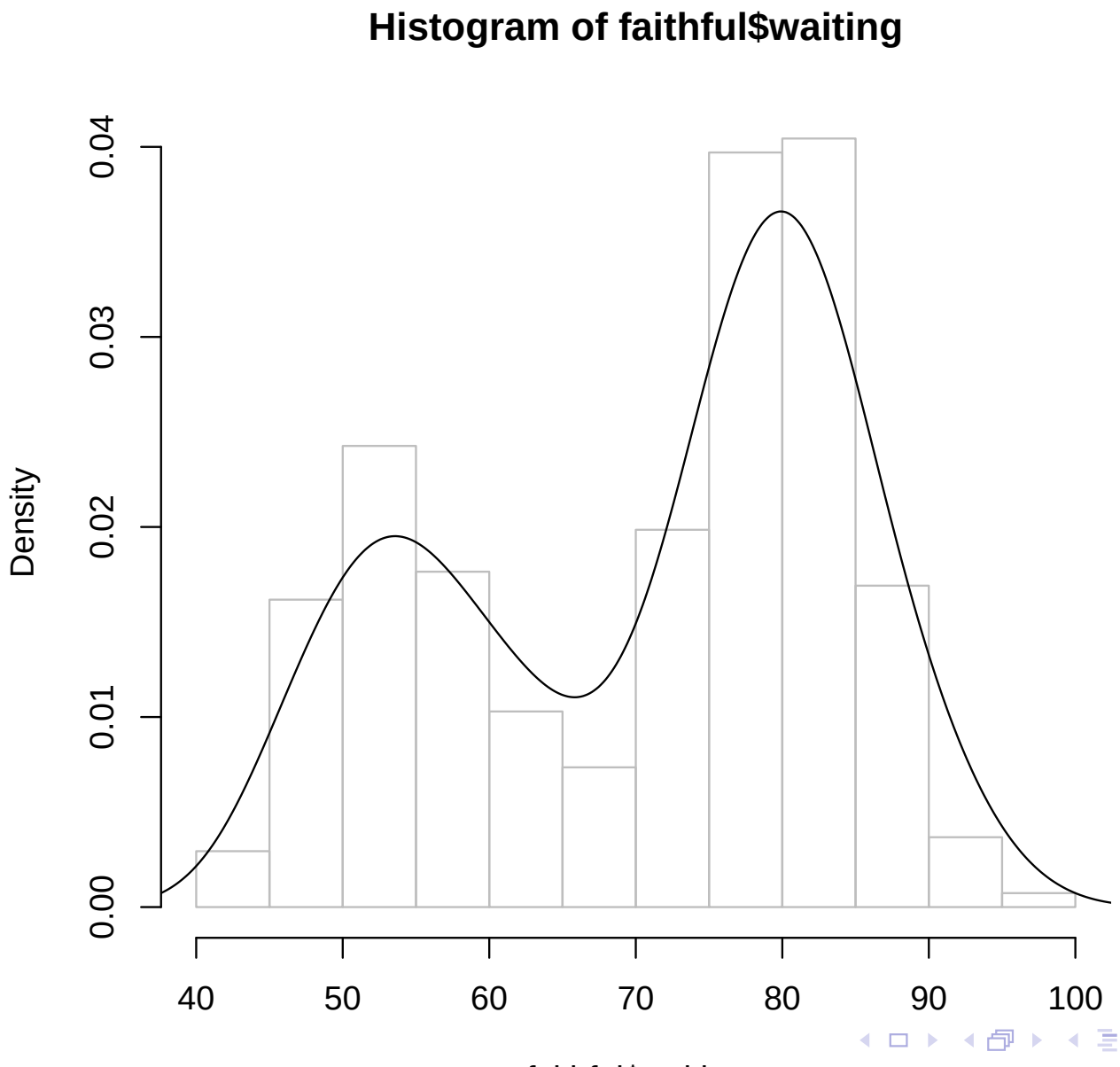
$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \overline{x})^2$$

Mise en œuvre sous R

- la fonction `density` : par défaut elle utilise le noyau Gaussien, le choix de la fenêtre est celui de Silverman (1986)
- application sur les données `faithful` :

```
> data(faithful)
> hist(faithful$waiting, probability=T,border="gray")
> lines(density(faithful$waiting))
```

Mise en œuvre sous R



Estimation de densité en 2D

- Maintenant, on veut donner une estimation de la densité du couple (`waiting`,`eruption`)
- On définit un estimateur par noyau 2D :

$$\widehat{f}(x,y) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x-x_i}{h}, \frac{y-y_i}{h}\right)$$

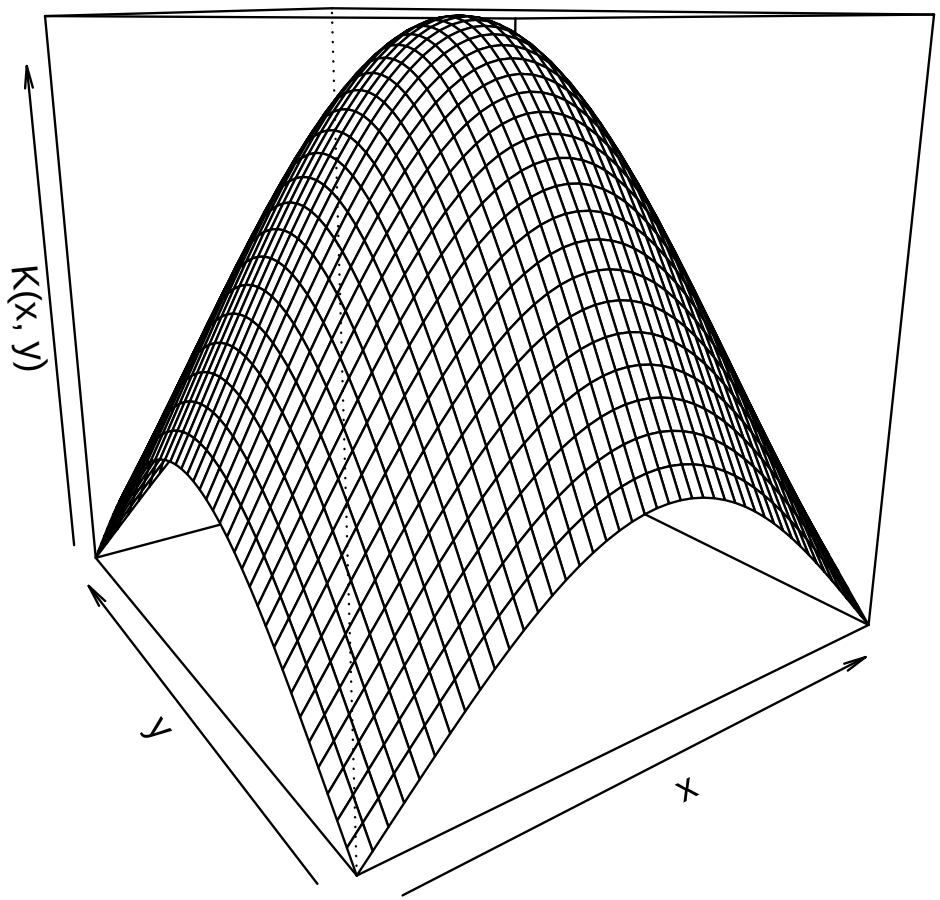
- Noyau Gaussien en 2D :

$$K(x,y) = \frac{1}{2\pi}\exp\left(-\frac{1}{2}(x^2+y^2)\right).$$

Le Noyau Gaussien en 2D

```
> gauss<-function(x,y)(1/pi)*exp(-.5*(x^2+y^2))
> x<-seq(from = -1.1, to = 1.1, by = 0.05)
> gaussvals<-sapply(x, function(a) gauss(a, x))
> persp(x = x, y = x, z = gaussvals,
+       xlab = "x", ylab = "y",
+       zlab = expression(K(x, y)),
+       theta = -35, axes = TRUE,
+       box = TRUE)
```

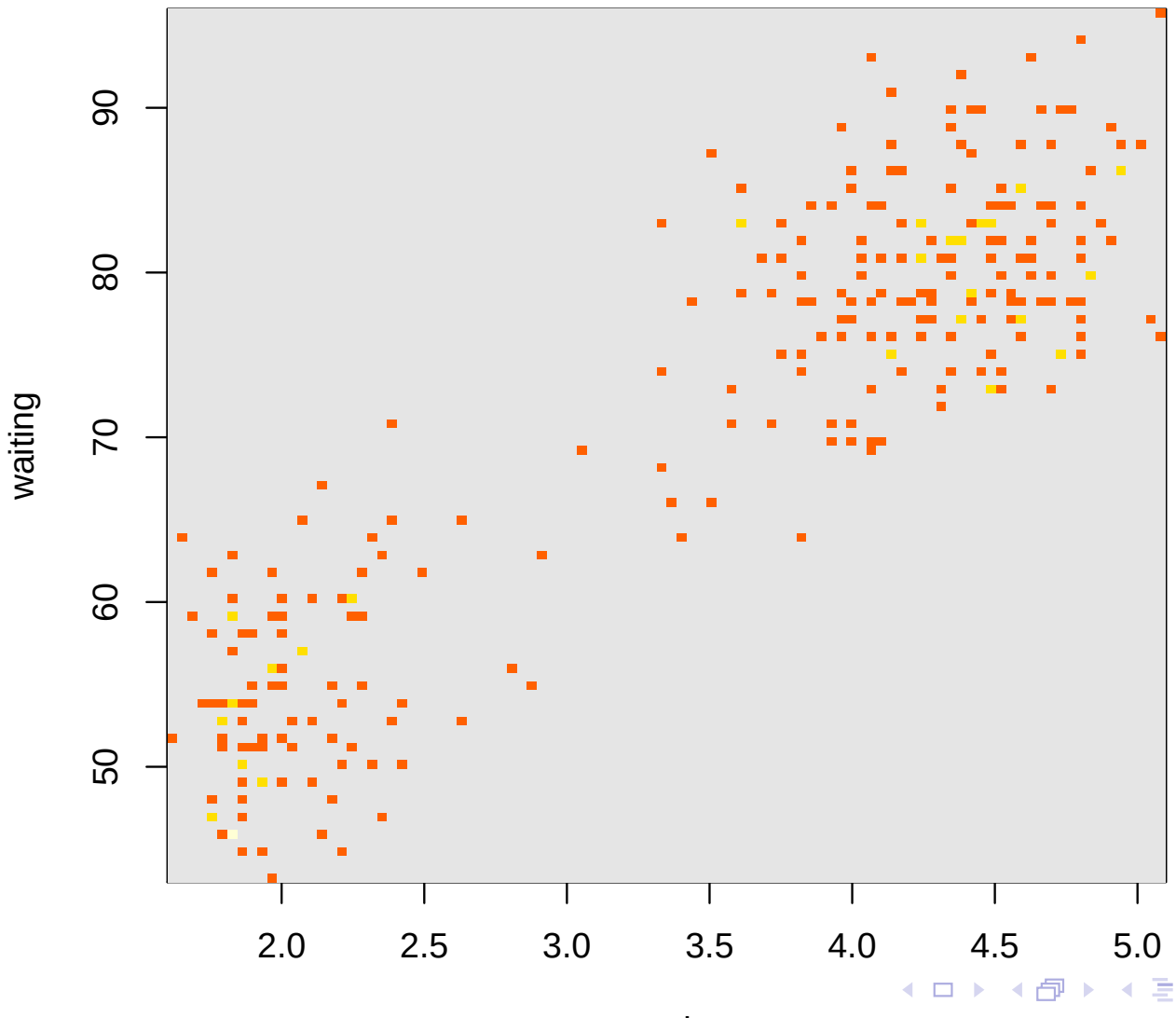
Le Noyau Gaussien en 2D



Mise en œuvre sous R

```
> library(KernSmooth)
> library(gplots)
> data(faithful)
> hist2d(faithful, same.scale=F,col=c('gray90',heat.colors(
+        nbins=100, xlab='eruption',ylab='waiting',show=T)

----------------------------
2-D Histogram Object
----------------------------
Number of data points:  272
Number of grid bins:  100 x 100
X range: ( 1.6 , 5.1 )
Y range: ( 43 , 96 )
```

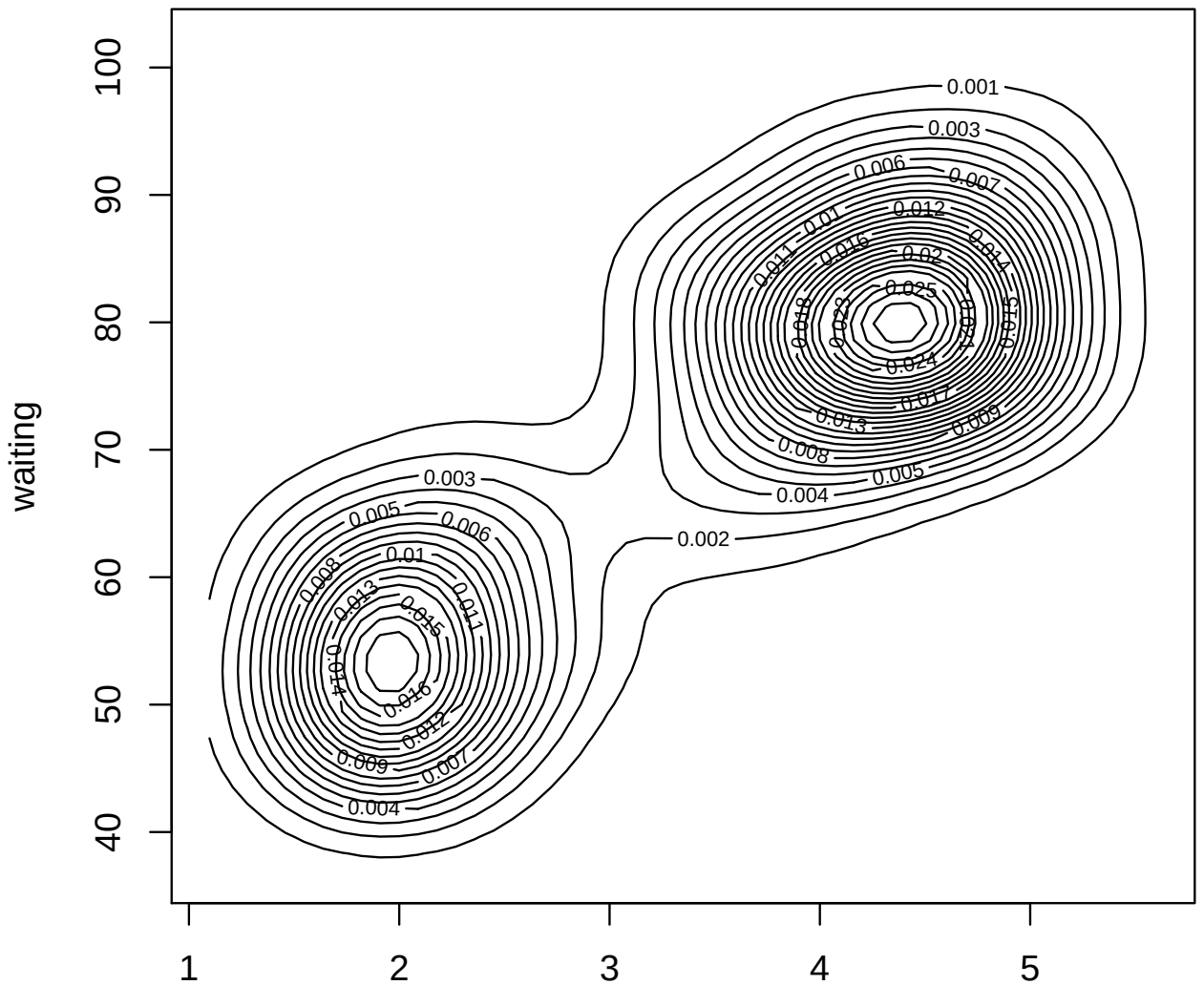
Mise en œuvre sous R



Mise en œuvre sous R

```
> f1hat=density(faithful$eruption)
> bw1=f1hat$bw
> f2hat=density(faithful$waiting)
> bw2=f2hat$bw
> fhat=bkde2D(faithful,bandwidth=c(bw1,bw2))
> contour(x=fhat$x1,y=fhat$x2,
+         z=fhat$fhat,xlab='eruption',ylab='waiting',axes=T
```

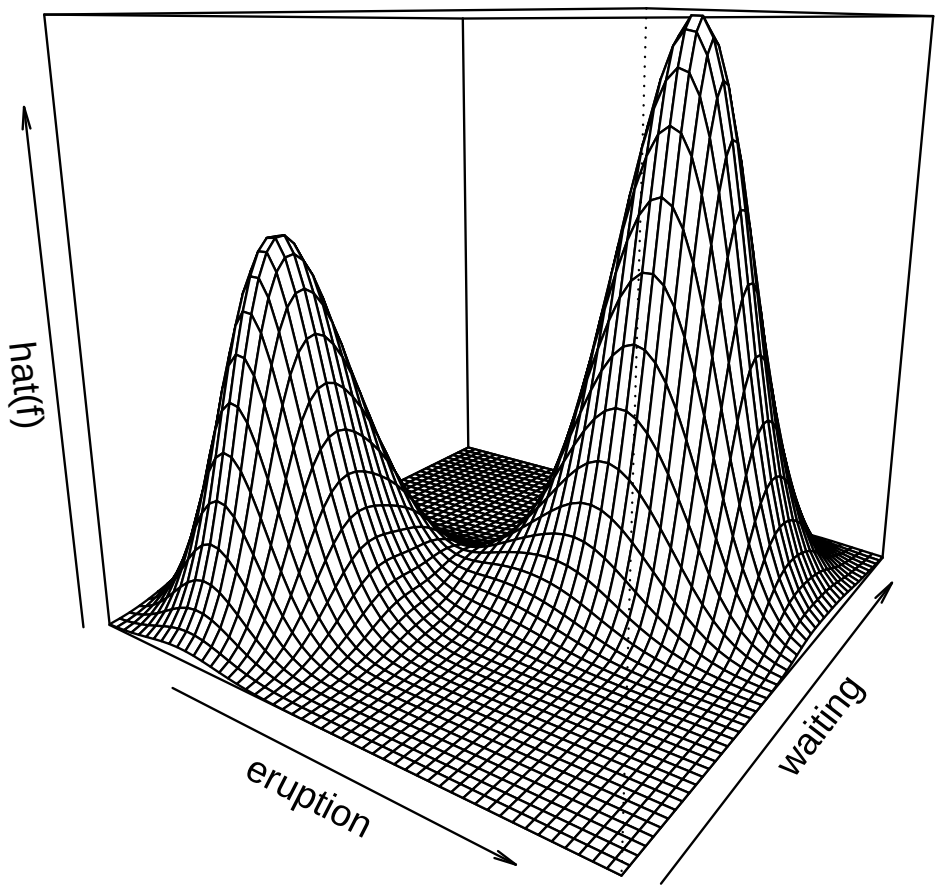
Mise en œuvre sous R



Mise en œuvre sous R

```
> persp(x=fhat$x1,y=fhat$x2,
+         z=fhat$fhat,xlab='eruption',ylab='waiting',
+       zlab=expression(hat(f)(x,y)),axes=T,box=T,theta=35)
```

Mise en œuvre sous R



Application : Estimation paramétrique d'une densité

- **L'objectif** : donner une estimation de la densité d'une mixture de la lois Normales, calculer des intervalles de confiance des paramètres (utilisant la méthode du *bootstrap*).
- Une v.a. suit une loi de probabilité dite mixture de lois Normales si densité s'écrit sous la forme

$$f(x) = \sum_{i=1}^{k} p_k \phi(x, \mu_k, \sigma_k)$$

où tous les $p_k \in [0, 1]$, $\sum_{i=1}^{k} p_i = i$ et

$$\phi(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2\right].$$

Application : Estimation paramétrique d'une densité

- L'observation de la variable `waiting` a montré qu'elle est bimodale donc on peut choisir $k = 2$ et on estimera les paramètres suivants : p, μ_i, σ_i, $i = 1, 2$.

- Deux approches :
  - Recherche EMV : recherche du maximum basé sur l'algorithme de Nelder-Mead.
  - EM-algorithme : `mclust` package

EMV des paramètres

Ecriture du log-vraisemblance

```
> logL <- function(param,x) {
+ d1<-dnorm(x,mean=param[2],sd=param[3])
+ d2<-dnorm(x,mean=param[4],sd=param[5])
+ -sum(log(param[1]*d1+(1-param[1])*d2))
+ }
```

On utiliser un algorithme d'optimisation pour caculé les EMV.

```
> startparam=c(p=0.5,mu1=50,sd1=3,mu2=80,sd2=3)
> emv<-optim(startparam,logL,x=faithful$waiting,
+ method = "L-BFGS-B", lower = c(0.01, rep(1, 4)),
+ upper = c(0.99, rep(200, 4)))
```

EMV des paramètres, résultats

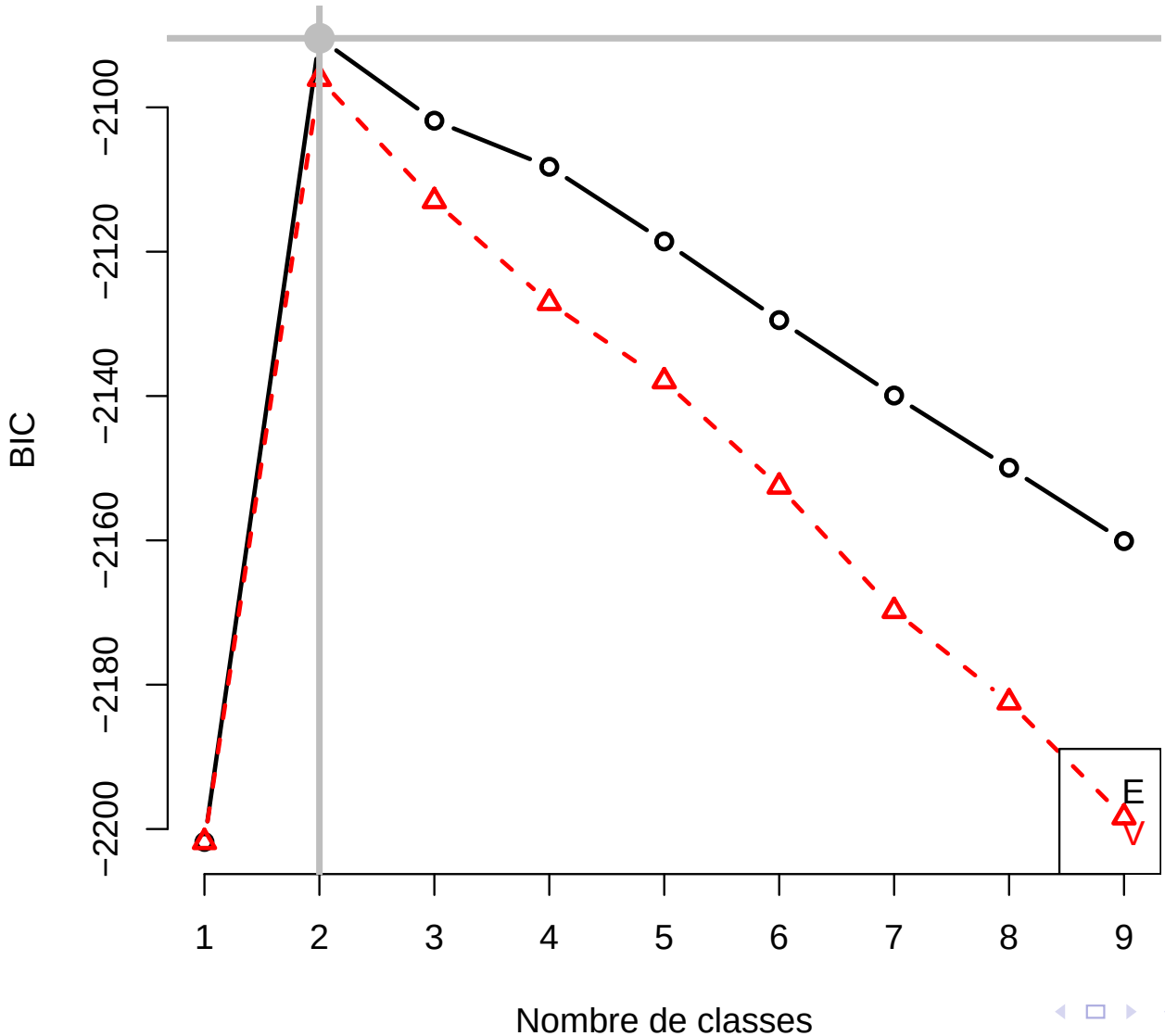
```
> emv

$par
         p         mu1         sd1         mu2         sd2
 0.3608912 54.6121396  5.8723774 80.0934102  5.8672823
$value
[1] 1034.002
$counts
function gradient
      55       55
$convergence
[1] 0
```

EM-algorithme.

- Méthode à voir dans le cours de classification : supposer qu'il existe une variable non-obersvée qui a permis de diviser l'échantillon en plusieurs sous-échantillons homogènes. Objectif : détermination du nombre des sous-échantillons et estimation des paramètres.
- Estimation.

```
> library("mclust")
> mc=Mclust(faithful$waiting)
> mc

'Mclust' model object:
  best model: univariate, equal variance (E)
+ with 2 components
```

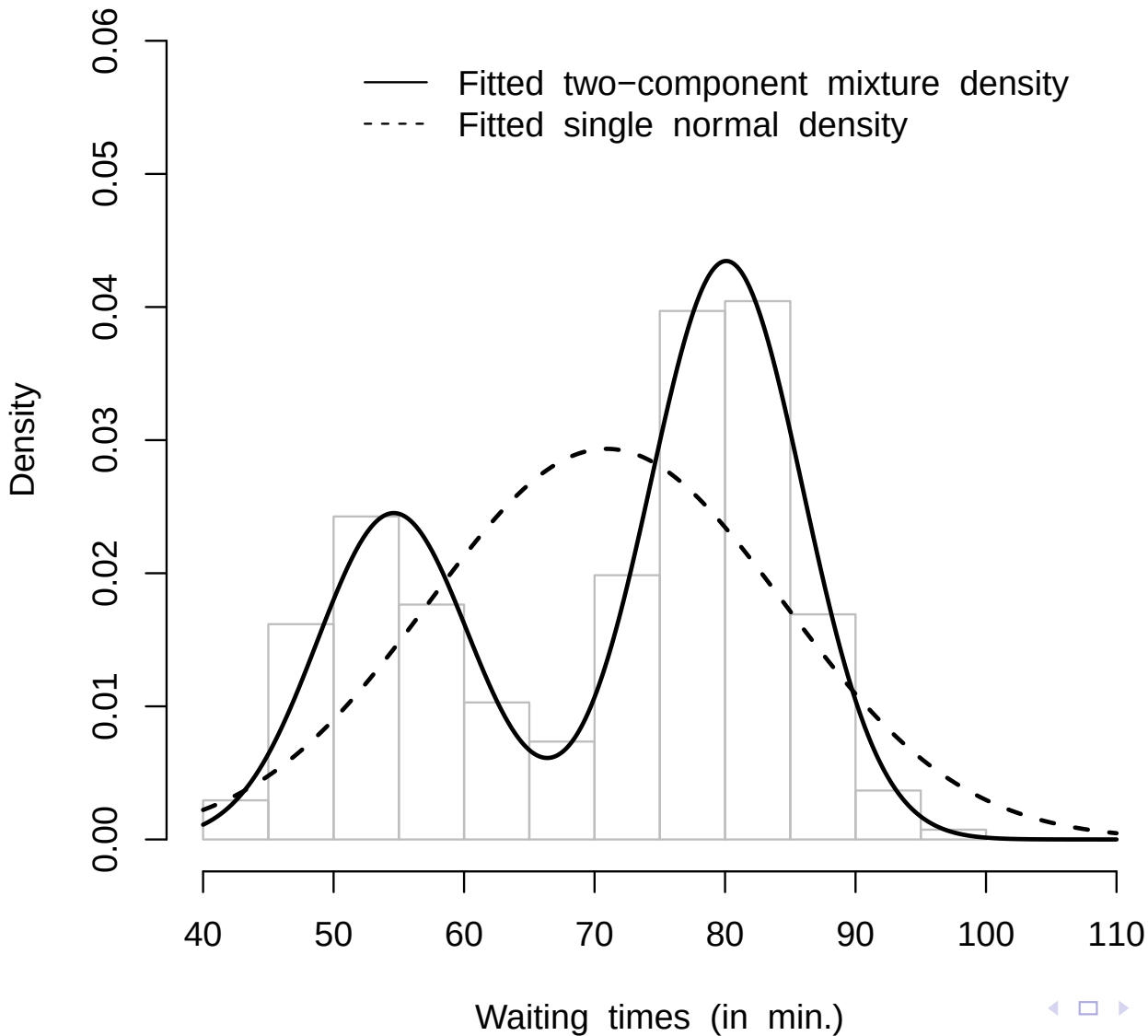
EM-algorithme, représentation du BIC.



EM-algorithme, résultats.

```
> mc$parameters$pro
[1] 0.3610159 0.6389841
> mc$parameters$mean
       1        2
54.61911 80.09384
> sqrt(mc$parameters$variance$sigmasq)
[1] 5.868479
```

EM-algorithme, représentation de la densité.



Estimation des intervalles de confiance : méthode bootstrap

- Déscription de la méthode bootstrap.
  1. On créé R échantillons bootstrap : tirer R fois n observations par les n observations, mais avec remplacement.
  2. Ajuster chaque échantillon avec un modèle de mixture de lois Normales
- code R

```
> library("boot")
> fit <-function(x,indx){
+ a  <-  Mclust(x[indx],minG=2,maxG=2)$parameters
+ if(a$pro[1]  <  0.5)
+ return(c(p=a$pro[1],mu1=a$mean[1],mu2=a$mean[2]))
+ return(c(p=1-a$pro[1],mu1=a$mean[2],mu2=a$mean[1]))
+ }
> bootpara<-boot(faithful$waiting,  fit,  R  =  1000)
```

Calcul des IC

```
> boot.ci(bootpara,type='bca',index=1)

Intervals :
Level       BCa
95%   ( 0.2924,  0.4228 )
Calculations and Intervals on Original Scale

> boot.ci(bootpara,type='bca',index=2)

Intervals :
Level       BCa
95%   (53.24, 55.95 )
Calculations and Intervals on Original Scale

> boot.ci(bootpara,type='bca',index=3)

Intervals :
Level       BCa
95%   (78.83, 81.03 )
Calculations and Intervals on Original Scale
```