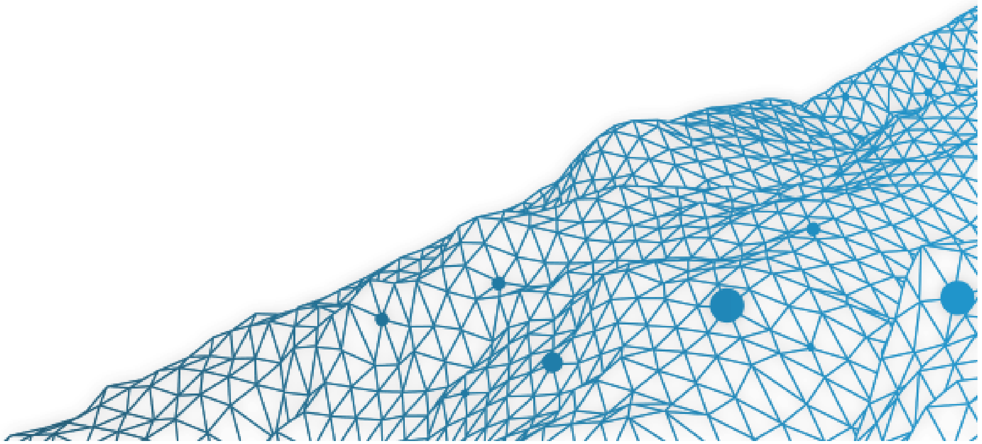
Advanced Econometrics #1 : Nonlinear Transformations

A. Charpentier (Université de Rennes 1)

Université de Rennes 1

Graduate Course, 2018.



Econometrics and ‘Regression’ ?

Galton (1870, Heriditary Genius, 1886, Regression towards mediocrity in hereditary stature) and Pearson & Lee (1896, On Telegony in Man, 1903 On the Laws of Inheritance in Man) studied genetic transmission of characterisitcs, e.g. the heigth.

On average the child of tall parents is taller than other children, but less than his parents.

“*I have called this peculiarity by the name of regression*”, Francis Galton, 1886.

Regression *towards* Mediocrity *in* Hereditary Stature.
By Francis Galton, F.R.S., &c.

Table 8.1. Galton's 1885 cross-tabulation of 928 adult children born of 205 midparents, by their height and their midparent's height.

Height of the mid-parent in inches	Height of the adult child <61.7	62.2	63.2	64.2	65.2	66.2	67.2	68.2	69.2	70.2	71.2	72.2	73.2	>73.7	Total no. of adult children	Total no. of mid-parents	Medians
> 73.0	—	—	—	—	—	—	—	—	—	—	—	1	3	—	4	5	—
72.5	—	—	—	—	—	—	—	1	2	1	2	7	2	4	19	6	72.2
71.5	—	—	—	—	1	3	4	3	5	10	4	9	2	2	43	11	69.9
70.5	1	—	1	—	1	1	3	12	18	14	7	4	3	3	68	22	69.5
69.5	—	—	1	16	4	17	27	20	33	25	20	11	4	5	183	41	68.9
68.5	1	—	7	11	16	25	31	34	48	21	18	4	3	—	219	49	68.2
67.5	—	3	5	14	15	36	38	28	38	19	11	4	—	—	211	33	67.6
66.5	—	3	3	5	2	17	17	14	13	4	—	—	—	—	78	20	67.2
65.5	1	—	9	5	7	11	11	7	7	5	2	1	—	—	66	12	66.7
64.5	1	1	4	4	1	5	5	—	2	—	—	—	—	—	23	5	65.8
< 64.0	1	—	2	4	1	2	2	1	1	—	—	—	—	—	14	1	—
Totals	5	7	32	59	48	117	138	120	167	99	64	41	17	14	928	205	—
Medians	—	—	66.3	67.8	67.9	67.7	67.9	68.3	68.5	69.0	69.0	70.0	—	—	—	—	—

Source: Galton (1886a).
Note: All female heights were multiplied by 1.08 before tabulation. Galton added an explanatory footnote to the table: "In calculating the Medians, the entries have been taken as referring to the middle of the squares in which they stand. The reason why the headings run 62.2, 63.2, &c., instead of 62.5, 63.5, &c., is that the observations are unequally distributed between 62 and 63, 63 and 64, &c., there being a strong bias in favour of integral inches. After careful consideration, I concluded that the headings, as adopted, best satisfied the conditions. This inequality was not apparent in the case of the Mid-parents." Galton republished these data in 1889, where they are referred to as the R.F.F. Data (Record of Family Faculties; he then noted that the first row must be in error (four children cannot have five sets of parents), but he claimed that "the bottom line, which looks suspicious, is correct" (p. 208).

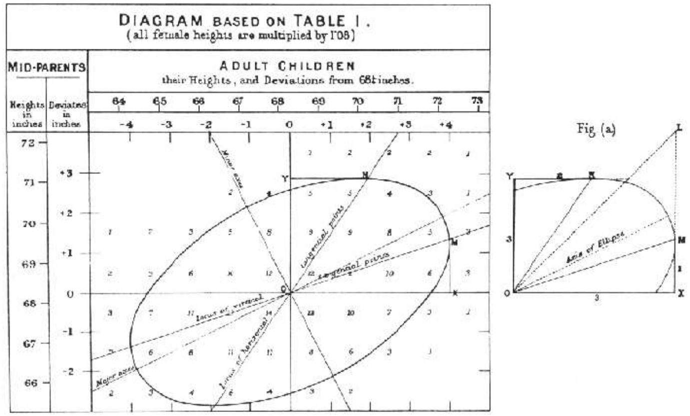
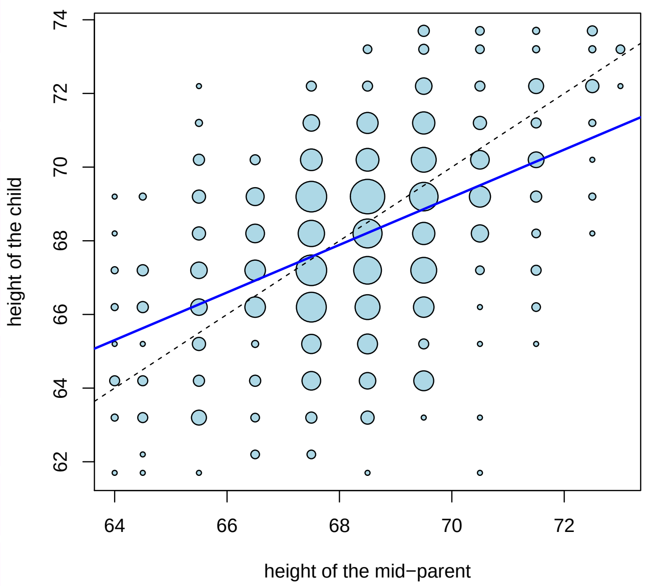


Figure 8.7. Galton's smoothed rendition of Table 8.1, with one of the "concentric and similar ellipses" drawn in. The geometric relationship of the two regression lines to the ellipse is also shown. (From Galton, 1886a.)

Econometrics and 'Regression' ?

```
> library(HistData)
> attach(Galton)
> Galton$count <- 1
> df <- aggregate(Galton, by=list(parent,
    child), FUN=sum)[,c(1,2,5)]
> plot(df[,1:2],cex=sqrt(df[,3]/3))
> abline(a=0,b=1,lty=2)
> abline(lm(child~parent,data=Galton))
>  coefficients(lm(child~parent,data=Galton)
    )[2]
   parent
0.6462906
```



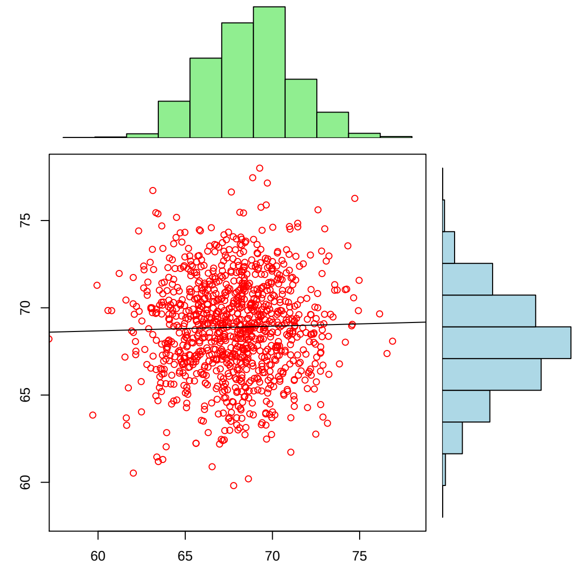
It is more an autoregression issue here :

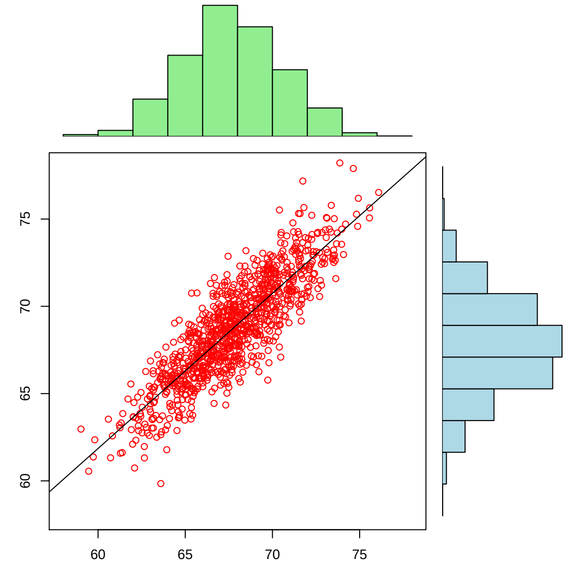
if $Y_t = \phi Y_{t-1} + \varepsilon_t$, then $\text{cor}[Y_t, Y_{t+h}] = \phi^h \to 0$ as $h \to \infty$.

Econometrics and ‘Regression’ ?

Regression is a correlation problem.

Overall, children are not smaller than parents





Overview

- Linear Regression Model: $y_i = \beta_0 + \boldsymbol{x}_i^\top \boldsymbol{\beta} + \varepsilon_i = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \varepsilon_i$
- Nonlinear Transformations : smoothing techniques

  $h(y_i) = \beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \varepsilon_i$

  $y_i = \beta_0 + \beta_1 x_{1,i} + h(x_{2,i}) + \varepsilon_i$
- Asymptotics vs. Finite Distance : boostrap techniques
- Penalization : Parcimony, Complexity and Overfit
- From least squares to other regressions : quantiles, expectiles, etc.

References

Motivation

Kopczuk, W. Tax bases, tax rates and the elasticity of reported income. JPE.

> income. I experiment with 10-piece splines in logarithms of both the $t-1$ income and the "transitory" component to allow for potential nonlinear effects. Nonlinearity in the permanent component allows me to account for trends in income varying across different income classes. In principle, the transitory component can be controlled for in a linear

References

Eubank, R.L. (1999) Nonparametric Regression and Spline Smoothing, CRC Press.

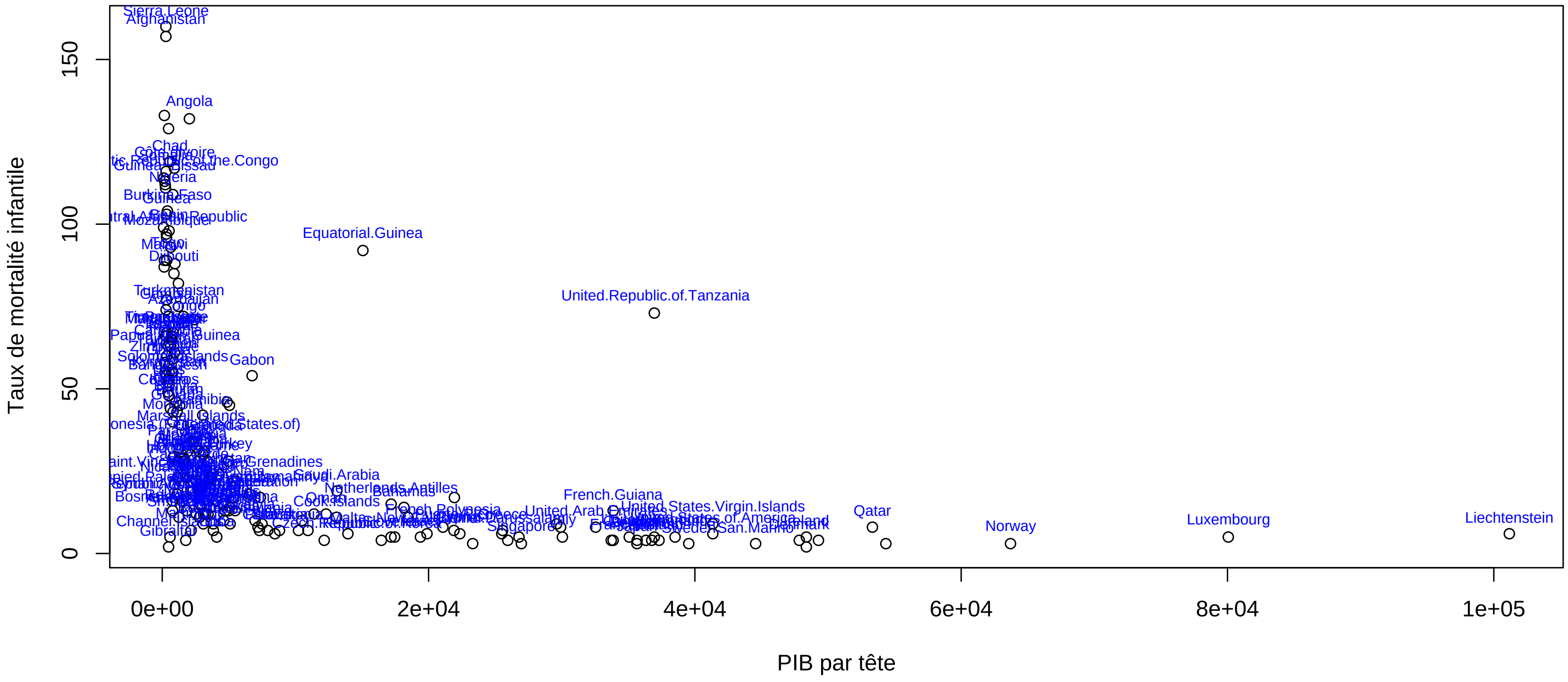
Fan, J. & Gijbels, I. (1996) Local Polynomial Modelling and Its Applications CRC Press.

Hastie, T.J. & Tibshirani, R.J. (1990) Generalized Additive Models. CRC Press

Wand, M.P & Jones, M.C. (1994) Kernel Smoothing. CRC Press

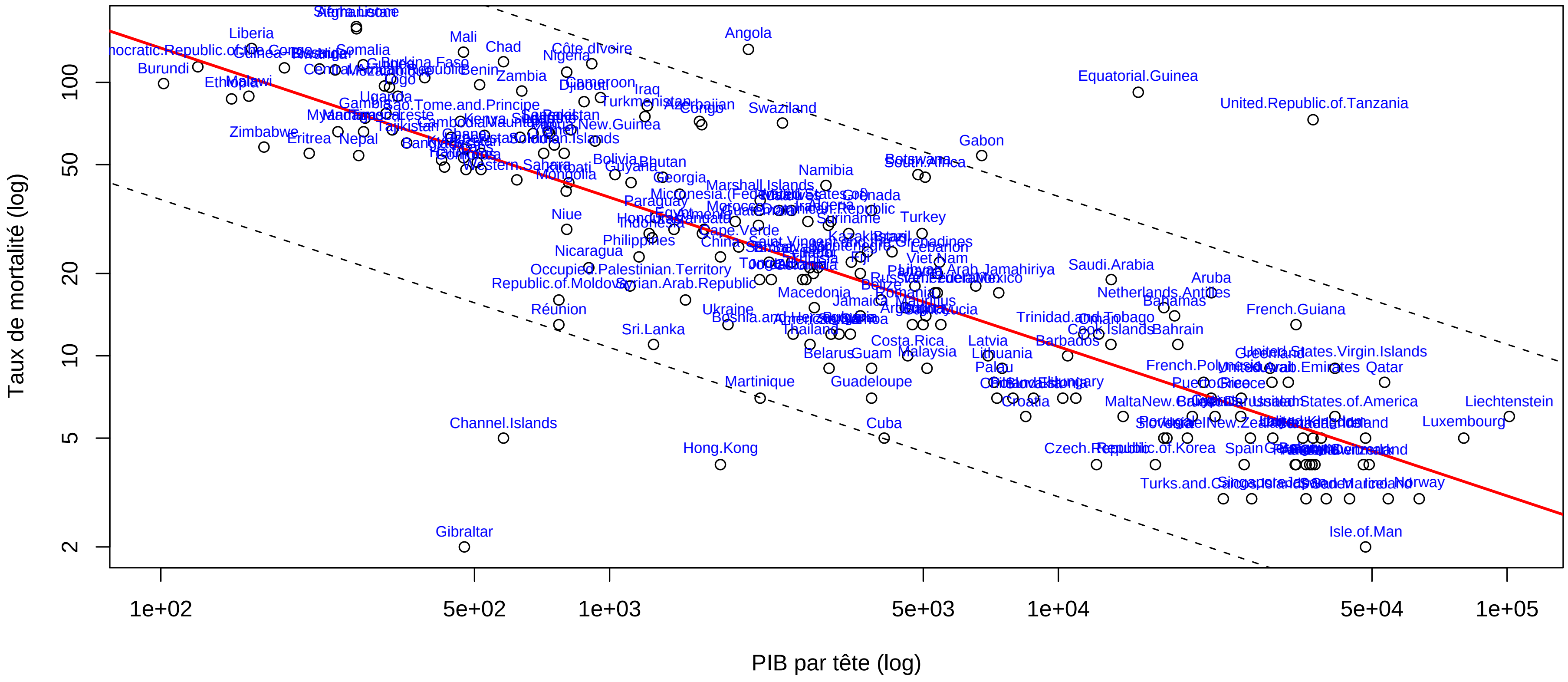
Deterministic or Parametric Transformations

Consider child mortality rate (y) as a function of GDP per capita (x).



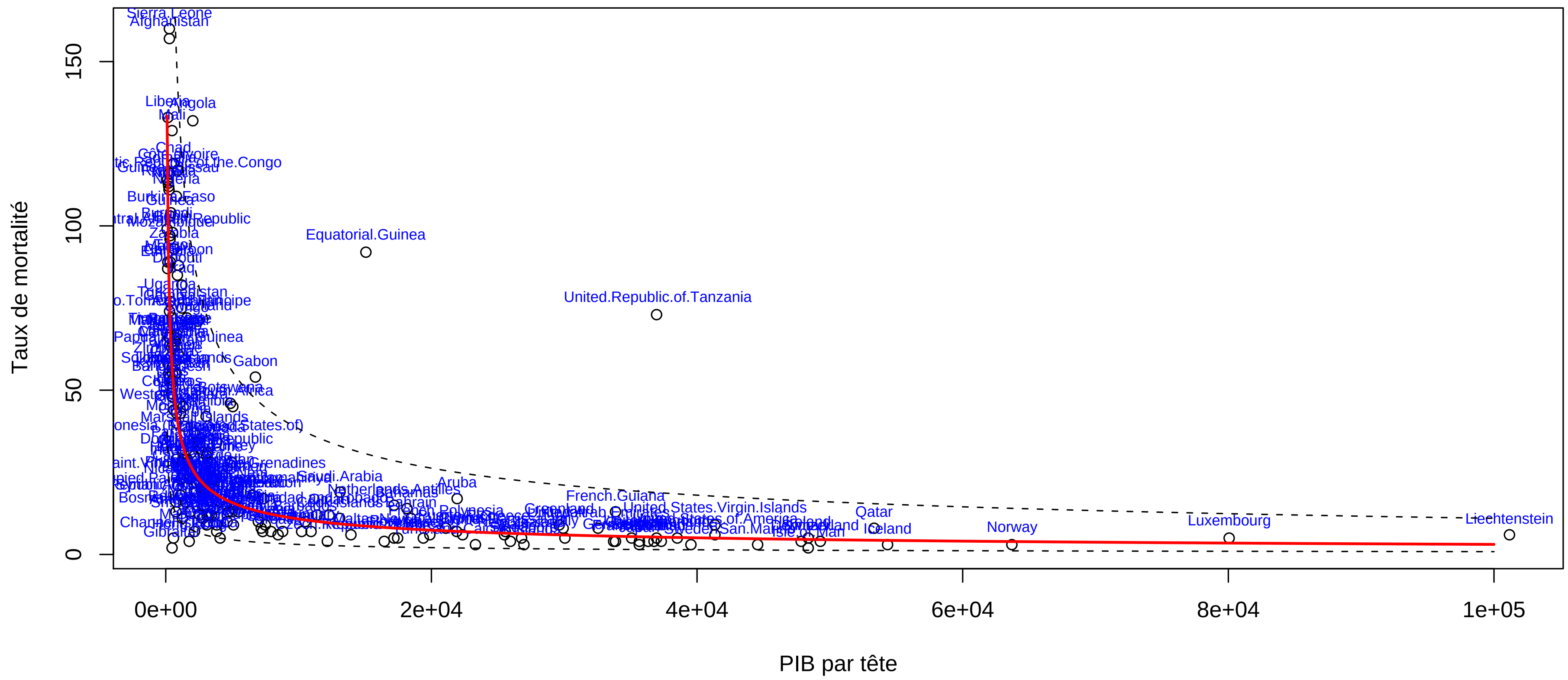
Deterministic or Parametric Transformations

Logartihmic transformation, $\log(y)$ as a function of $\log(x)$



Deterministic or Parametric Transformations

Reverse transformation



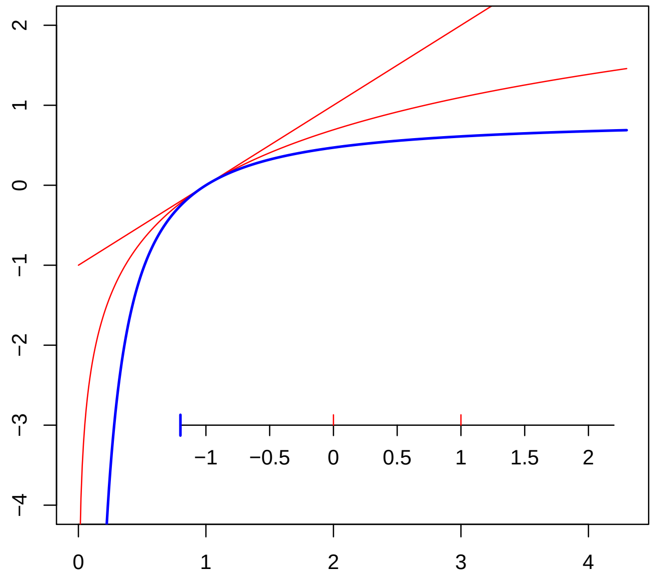
Box-Cox transformation

See Box & Cox (1964) An Analysis of Transformations ,

$$h(y,\lambda)=\begin{cases}\dfrac{y^\lambda-1}{\lambda}\text{ if }\lambda\neq 0\\ \log(y)\text{ if }\lambda=0\end{cases}$$

or

$$h(y,\lambda,\mu)=\begin{cases}\dfrac{[y+\mu]^\lambda-1}{\lambda}\text{ if }\lambda\neq 0\\ \log([y+\mu])\text{ if }\lambda=0\end{cases}$$



Profile Likelihood

In a statistical context, suppose that unknown parameter can be partitioned $\boldsymbol{\theta} = (\lambda, \boldsymbol{\beta})$ where λ is the parameter of interest, and $\boldsymbol{\beta}$ is a nuisance parameter.

Consider $\{y_1, \cdots, y_n\}$, a sample from distribution $F_{\boldsymbol{\theta}}$, so that the log-likelihood is

$$\log \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log f_{\boldsymbol{\theta}}(y_i)$$

$\widehat{\boldsymbol{\theta}}^{MLE}$ is defined as $\widehat{\boldsymbol{\theta}}^{MLE} = \text{argmax}\,\{\log \mathcal{L}(\boldsymbol{\theta})\}$

Rewrite the log-likelihood as $\log \mathcal{L}(\boldsymbol{\theta}) = \log \mathcal{L}_\lambda(\boldsymbol{\beta})$. Define

$$\widehat{\boldsymbol{\beta}}_\lambda^{pMLE} = \underset{\boldsymbol{\beta}}{\text{argmax}}\,\{\log \mathcal{L}_\lambda(\boldsymbol{\beta})\}$$

and then $\widehat{\lambda}^{pMLE} = \underset{\lambda}{\text{argmax}}\left\{\log \mathcal{L}_\lambda(\widehat{\boldsymbol{\beta}}_\lambda^{pMLE})\right\}$. Observe that

$$\sqrt{n}(\widehat{\lambda}^{pMLE} - \lambda) \xrightarrow{\mathcal{L}} \mathcal{N}(0, [\mathbb{I}_{\lambda,\lambda} - \mathbb{I}_{\lambda,\boldsymbol{\beta}}\mathbb{I}_{\boldsymbol{\beta},\boldsymbol{\beta}}^{-1}\mathbb{I}_{\boldsymbol{\beta},\lambda}]^{-1})$$

Profile Likelihood and Likelihood Ratio Test

The (profile) likelihood ratio test is based on

$$2\left(\max\left\{\mathcal{L}(\lambda,\boldsymbol{\beta})\right\}-\max\left\{\mathcal{L}(\lambda_0,\boldsymbol{\beta})\right\}\right)$$

If $(\lambda_0,\boldsymbol{\beta}_0)$ are the true value, this difference can be written

$$2\left(\max\left\{\mathcal{L}(\lambda,\boldsymbol{\beta})\right\}-\max\left\{\mathcal{L}(\lambda_0,\boldsymbol{\beta}_0)\right\}\right)-2\left(\max\left\{\mathcal{L}(\lambda_0,\boldsymbol{\beta})\right\}-\max\left\{\mathcal{L}(\lambda_0,\boldsymbol{\beta}_0)\right\}\right)$$

Using Taylor's expension

$$\left.\frac{\partial\mathcal{L}(\lambda,\boldsymbol{\beta})}{\partial\lambda}\right|_{(\lambda_0,\widehat{\boldsymbol{\beta}}_{\lambda_0})}\sim\left.\frac{\partial\mathcal{L}(\lambda,\boldsymbol{\beta})}{\partial\lambda}\right|_{(\lambda_0,\boldsymbol{\beta}_0)}-\mathbb{I}_{\boldsymbol{\beta}_0\lambda_0}\mathbb{I}^{-1}_{\boldsymbol{\beta}_0\boldsymbol{\beta}_0}\left.\frac{\partial\mathcal{L}(\lambda_0,\boldsymbol{\beta})}{\partial\boldsymbol{\beta}}\right|_{(\lambda_0,\boldsymbol{\beta}_0)}$$

Thus,

$$\frac{1}{\sqrt{n}}\left.\frac{\partial\mathcal{L}(\lambda,\boldsymbol{\beta})}{\partial\lambda}\right|_{(\lambda_0,\widehat{\boldsymbol{\beta}}_{\lambda_0})}\overset{\mathcal{L}}{\to}\mathcal{N}(0,\mathbb{I}_{\lambda_0\lambda_0})-\mathbb{I}_{\lambda_0\boldsymbol{\beta}_0}\mathbb{I}^{-1}_{\boldsymbol{\beta}_0\boldsymbol{\beta}_0}\mathbb{I}_{\boldsymbol{\beta}_0\lambda_0}$$

and $2\left(\mathcal{L}(\widehat{\lambda},\widehat{\boldsymbol{\beta}})-\mathcal{L}(\lambda_0,\widehat{\boldsymbol{\beta}}_{\lambda_0})\right)\overset{\mathcal{L}}{\to}\chi^2(\dim(\lambda))$.

Profile Likelihood and Likelihood Ratio Test

Consider some lognormal sample, and fit a Gamma distribution,

$$f(x;\alpha,\beta)=\frac{x^{\alpha-1}\beta^{\alpha}\,e^{-\beta\,x}}{\Gamma(\alpha)}\text{ with }x>0\text{ and }\boldsymbol{\theta}=(\alpha,\beta).$$

```
> x=exp(rnorm(100))
```

Maximum-likelihood, $\widehat{\boldsymbol{\theta}}=\text{argmax}\{\log\mathcal{L}(\boldsymbol{\theta})\}$.

```
> library(MASS)
> (F=fitdistr(x,"gamma"))
     shape        rate
  1.4214497   0.8619969
 (0.1822570) (0.1320717)
> F$estimate[1]+c(-1,1)*1.96*F$sd[1]
[1] 1.064226 1.778673
```

Profile Likelihood and Likelihood Ratio Test

See also

```
> log_lik=function(theta){
+    a=theta[1]
+    b=theta[2]
+    logL=sum(log(dgamma(x,a,b)))
+    return(-logL)
+ }
> optim(c(1,1),log_lik)
$par
[1] 1.4214116 0.8620311
```

We can also use profile likelihood,

$$\widehat{\alpha} = \text{argmax}\left\{\max_{\beta}\left\{\log\mathcal{L}(\alpha,\beta)\right\}\right\} = \text{argmax}\left\{\log\mathcal{L}(\alpha,\widehat{\beta}_{\alpha})\right\}$$

Profile Likelihood and Likelihood Ratio Test

```
> prof_log_lik=function(a){
+    b=(optim(1,function(z) -sum(log(dgamma(x,a,z)))))$par
+    return(-sum(log(dgamma(x,a,b))))
+ }

> vx=seq(.5,3,length=101)
> vl=-Vectorize(prof_log_lik)(vx)
> plot(vx,vl,type="l")
> optim(1,prof_log_lik)
$par
[1] 1.421094
```

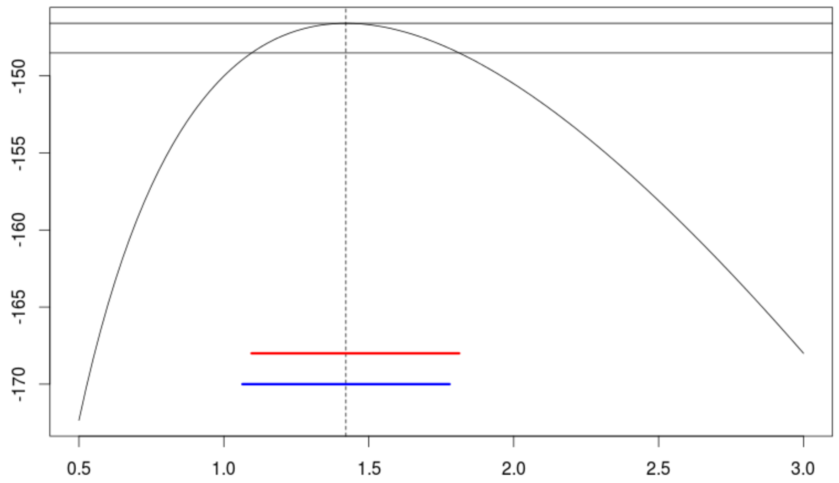
We can use the likelihood ratio test

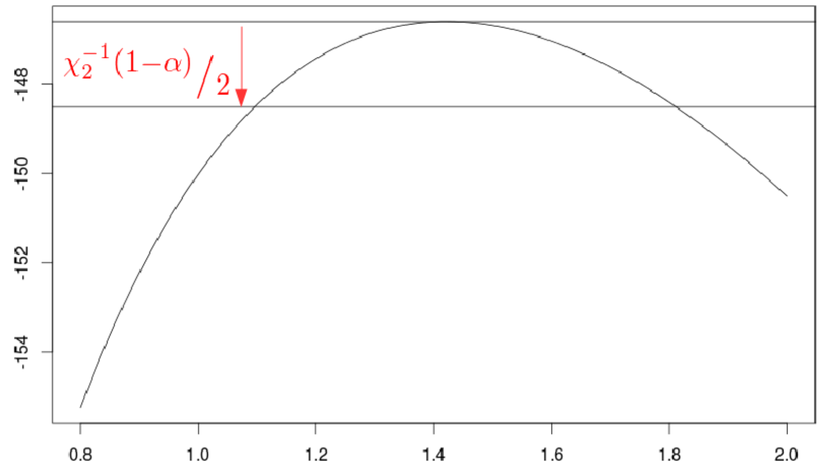
$$2\big(\log\mathcal{L}_p(\widehat{\alpha})-\log\mathcal{L}_p(\alpha)\big)\sim\chi^2(1)$$

Profile Likelihood and Likelihood Ratio Test

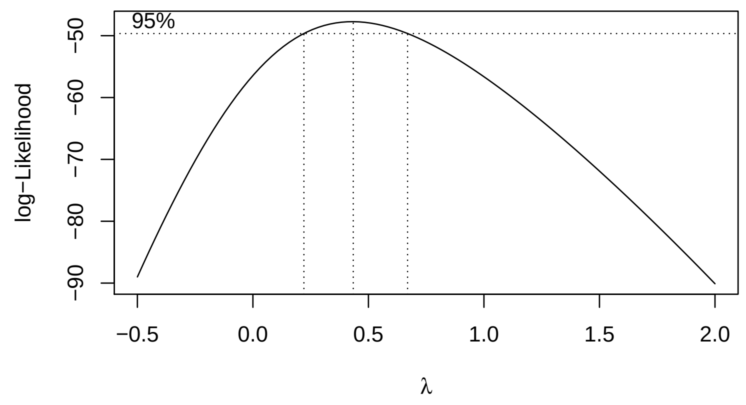
The implied 95% confidence interval is

```
> (b1=uniroot(function(z) Vectorize(prof_log_lik)(z)+borne,c(.5,1.5))
    $root)
[1] 1.095726
> (b2=uniroot(function(z) Vectorize(prof_log_lik)(z)+borne,c
    (1.25,2.5))$root)
[1] 1.811809
```





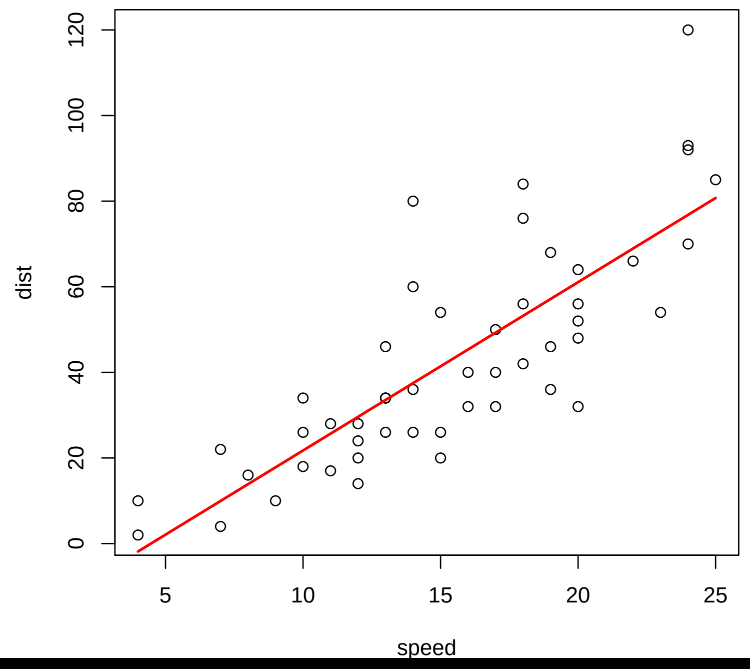
Box-Cox




```
1 > boxcox(lm(dist~speed,data=cars))
```

Here $h^* \sim 0.5$

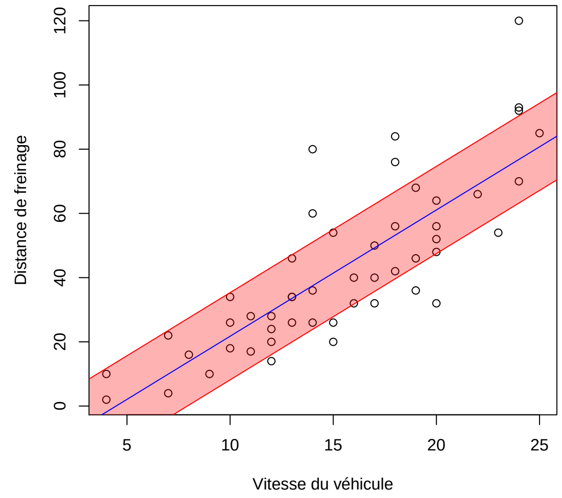
Heuristally, $y_i^{1/2} \sim \beta_0 + \beta_1 x_i + \varepsilon_i$
why not consider a quadratic regression...?

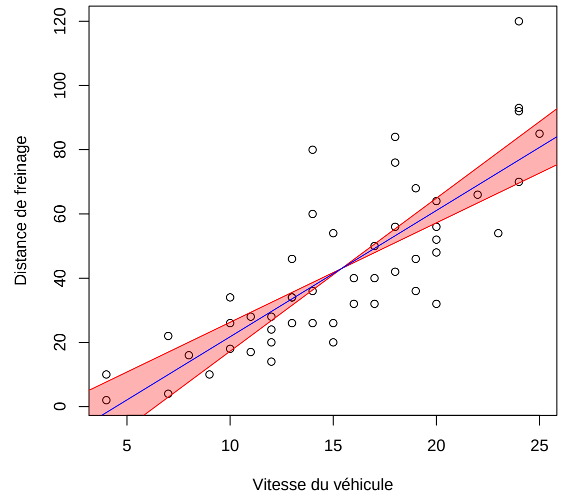


Uncertainty: Parameters vs. Prediction

Uncertainty on regression parameters (β_0, β_1)
From the output of the regression we can derive confidence intervals for β_0 and β_1, usually

$$\beta_k \in \left[\widehat{\beta}_k \pm u_{1-\alpha/2}\widehat{\text{se}}[\widehat{\beta}_k]\right]$$





Uncertainty: Parameters vs. Prediction

Uncertainty on a prediction, $y = m(\boldsymbol{x})$. Usually

$$m(\boldsymbol{x}) \in \left[\widehat{m}(\boldsymbol{x}) \pm u_{1-\alpha/2}\widehat{\text{se}}[m(\boldsymbol{x})]\right]$$

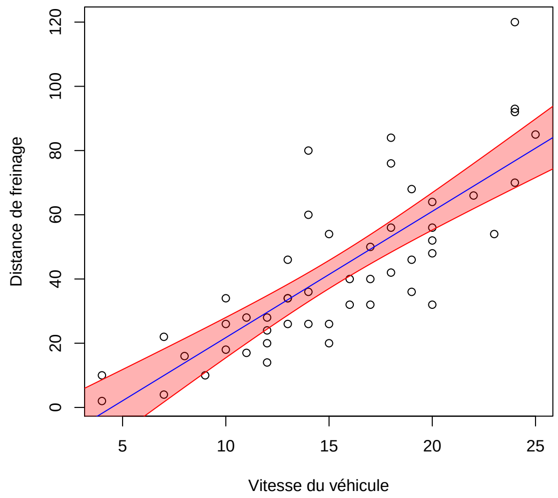
hence, for a linear model

$$\left[\boldsymbol{x}^\top\widehat{\boldsymbol{\beta}} \pm u_{1-\alpha/2}\widehat{\sigma}\sqrt{\boldsymbol{x}^\top[\boldsymbol{X}^\top\boldsymbol{X}]^{-1}\boldsymbol{x}}\right]$$

i.e. (with one covariate)

$$\text{se}^2[m(x)]^2 = \text{Var}[\widehat{\beta}_0 + \widehat{\beta}_1 x]$$

$$\text{se}^2[\widehat{\beta}_0] + \text{cov}[\widehat{\beta}_0, \widehat{\beta}_1]x + \text{se}^2[\widehat{\beta}_1]x^2$$




```
> predict(lm(dist~speed,data=cars),newdata=data.frame(speed=x),
    interval="confidence")
```

Least Squares and Expected Value (Orthogonal Projection Theorem)

Let $\boldsymbol{y} \in \mathbb{R}^d$, $\overline{y} = \underset{m\in\mathbb{R}}{\text{argmin}} \left\{ \sum_{i=1}^n \frac{1}{n} \big[\underbrace{y_i - m}_{\varepsilon_i}\big]^2 \right\}$. It is the empirical version of

$$\mathbb{E}[Y] = \underset{m\in\mathbb{R}}{\text{argmin}} \left\{ \int \big[\underbrace{y - m}_{\varepsilon}\big]^2 dF(y) \right\} = \underset{m\in\mathbb{R}}{\text{argmin}} \left\{ \mathbb{E}\big[(\underbrace{Y - m}_{\varepsilon})^2\big] \right\}$$

where Y is a ℓ_1 random variable.

Thus, $\underset{m(\cdot):\mathbb{R}^k\to\mathbb{R}}{\text{argmin}} \left\{ \sum_{i=1}^n \frac{1}{n} \big[\underbrace{y_i - m(\boldsymbol{x}_i)}_{\varepsilon_i}\big]^2 \right\}$ is the empirical version of $\mathbb{E}[Y|\boldsymbol{X} = \boldsymbol{x}]$.

The Histogram and the Regressogram

Connections between the estimation of $f(y)$ and $\mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}]$.

Assume that $y_i \in [a_1, a_{k+1})$, divided in k classes $[a_j, a_{j+1})$. The histogram is

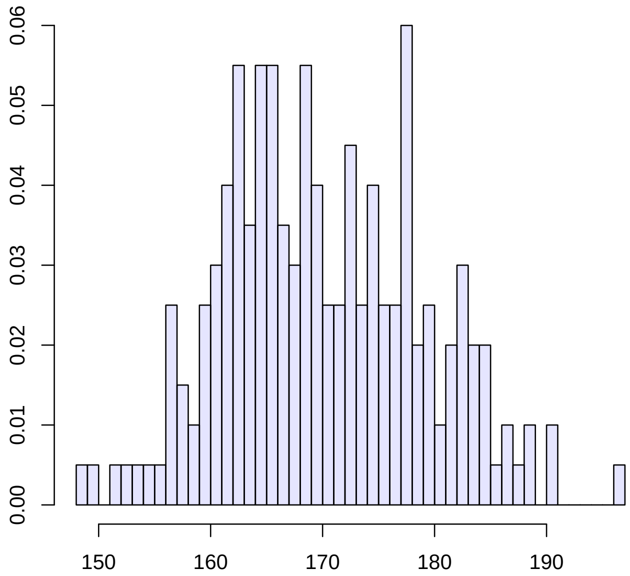
$$\hat{f}_{\boldsymbol{a}}(y) = \sum_{j=1}^{k} \frac{\mathbf{1}(t \in [a_j, a_{j+1}))}{a_{j+1} - a_j} \frac{1}{n} \sum_{i=1}^{n} \mathbf{1}(y_i \in [a_j, a_{j+1}))$$

Assume that $a_{j+1} - a_j = h_n$ and $h_n \to 0$ as $n \to \infty$ with $nh_n \to \infty$ then

$$\mathbb{E}\big[(\hat{f}_{\boldsymbol{a}}(y) - f(y))^2\big] \sim O(n^{-2/3})$$

(for an optimal choice of h_n).

```
> hist(height)
```



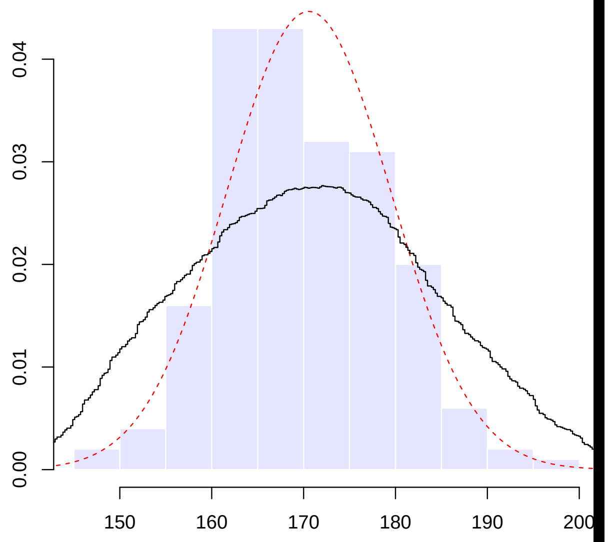
The Histogram and the Regressogram

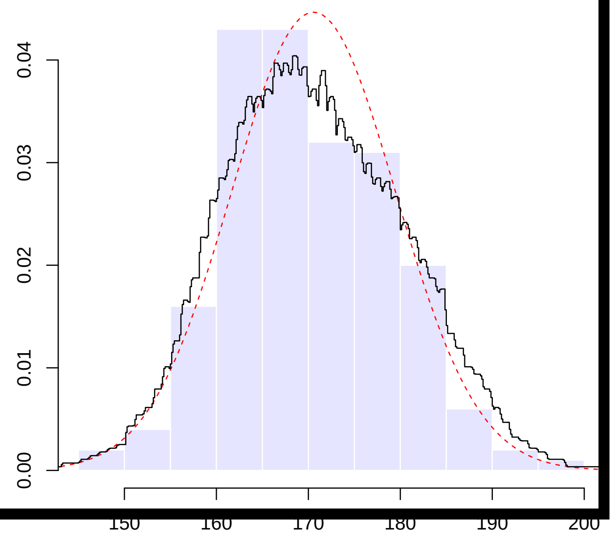
Then a moving histogram was considered,

$$\hat{f}(y) = \frac{1}{2nh_n}\sum_{i=1}^{n}\mathbf{1}(y_i \in [y \pm h_n)) \quad = \frac{1}{nh_n}\sum_{i=1}^{n} k\left(\frac{y_i - y}{h_n}\right)$$

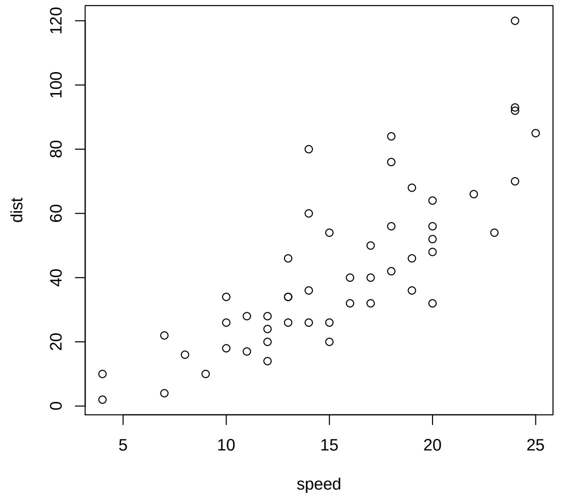
with $k(x) = \frac{1}{2}\mathbf{1}(x \in [-1, 1))$, which a (flat) kernel estimator.

```
> density(height, kernel = "rectangular")
```





The Histogram and the Regressogram

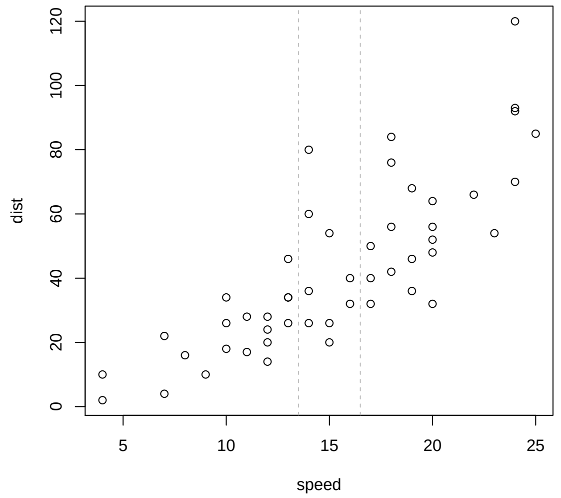


From Tukey (1961) Curves as parameters, and touch estimation, the regressogram is defined as

$$\hat{m}_{\boldsymbol{a}}(x) = \frac{\sum_{i=1}^{n} \mathbf{1}(x_i \in [a_j, a_{j+1}))y_i}{\sum_{i=1}^{n} \mathbf{1}(x_i \in [a_j, a_{j+1}))}$$

and the moving regressogram is

$$\hat{m}(x) = \frac{\sum_{i=1}^{n} \mathbf{1}(x_i \in [x \pm h_n])y_i}{\sum_{i=1}^{n} \mathbf{1}(x_i \in [x \pm h_n])}$$



Nadaraya-Watson and Kernels

Background: Kernel Density Estimator

Consider sample $\{y_1, \cdots, y_n\}$, $\widehat{F}_n$ empirical cumulative distribution function

$$\widehat{F}_n(y) = \frac{1}{n}\sum_{i=1}^{n}\mathbf{1}(y_i \leq y)$$

The empirical measure $\mathbb{P}_n$ consists in weights $1/n$ on each observation.

Idea: add (little) continuous noise to smooth $\widehat{F}_n$.

Let Y_n denote a random variable with distribution $\widehat{F}_n$ and define

$$\tilde{Y} = Y_n + hU \text{ where } U \perp\!\!\!\perp Y_n, \text{ with cdf } K$$

The cumulative distribution function of $\tilde{Y}$ is $\tilde{F}$

$$\tilde{F}(y) = \mathbb{P}[\tilde{Y} \leq y] = \mathbb{E}\big(\mathbf{1}(\tilde{Y} \leq y)\big) = \mathbb{E}\big(\mathbb{E}\big[\mathbf{1}(\tilde{Y} \leq y)\big|Y_n\big]\big)$$

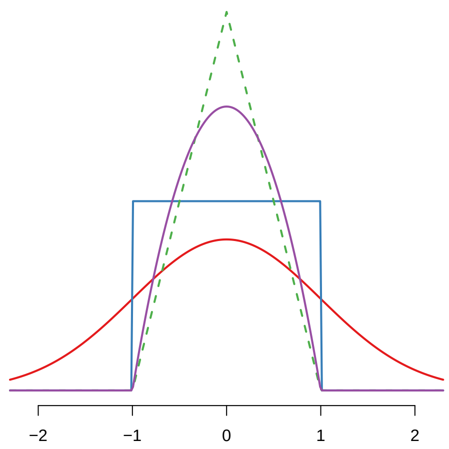
$$\tilde{F}(y) = \mathbb{E}\left(\mathbf{1}\left(U \leq \frac{y - Y_n}{h}\right)\Big|Y_n\right) = \sum_{i=1}^{n}\frac{1}{n}K\left(\frac{y - y_i}{h}\right)$$

Nadaraya-Watson and Kernels

If we differentiate

$$\tilde{f}(y) = \frac{1}{nh}\sum_{i=1}^{n} k\left(\frac{y-y_i}{h}\right)$$

$$= \frac{1}{n}\sum_{i=1}^{n} k_h\left(y-y_i\right) \text{ with } k_h(u) = \frac{1}{h}k\left(\frac{u}{h}\right)$$

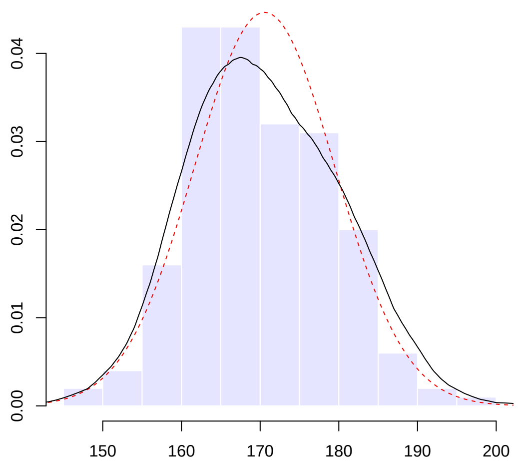


$\tilde{f}$ is the kernel density estimator of f, with kernel k and bandwidth h.

Rectangular, $k(u) = \frac{1}{2}\mathbf{1}(|u| \leq 1)$

Epanechnikov, $k(u) = \frac{3}{4}\mathbf{1}(|u| \leq 1)(1-u^2)$

Gaussian, $k(u) = \frac{1}{\sqrt{2\pi}}e^{-\frac{u^2}{2}}$




```
> density(height, kernel = "epanechnikov")
```

Kernels and Statistical Properties

Consider here an i.id. sample $\{Y_1, \cdots, Y_n\}$ with density f

Given y, observe that $\mathbb{E}[\tilde{f}(y)] = \displaystyle\int \frac{1}{h}k\left(\frac{y-t}{h}\right)f(t)dt = \int k(u)f(y-hu)du$. Use

Taylor expansion around $h=0$,$f(y-hu) \sim f(y) - f'(y)hu + \dfrac{1}{2}f''(y)h^2u^2$

$$
\begin{aligned}
\mathbb{E}[\tilde{f}(y)] &= \int f(y)k(u)du - \int f'(y)huk(u)du + \int \frac{1}{2}f''(y+\overline{h}u)h^2u^2k(u)du \\
&= f(y) + 0 + h^2\frac{f''(y)}{2}\int k(u)u^2du + o(h^2)
\end{aligned}
$$

Thus, if f is twice continuously differentiable with bounded second derivative,

$$
\int k(u)du = 1, \int uk(u)du = 0 \text{ and } \int u^2k(u)du < \infty,
$$

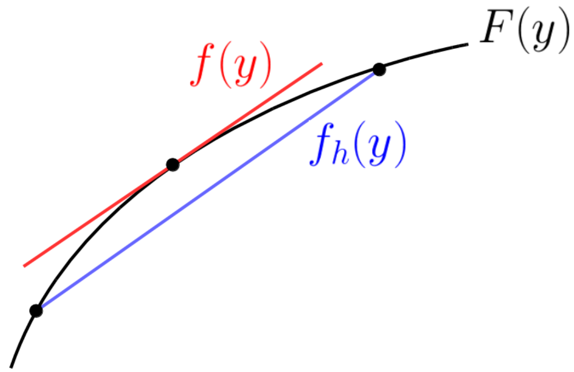
then $\mathbb{E}[\tilde{f}(y)] = f(y) + h^2\dfrac{f''(y)}{2}\displaystyle\int k(u)u^2du + o(h^2)$

Kernels and Statistical Properties

For the heuristics on that bias, consider a flat kernel,

and set

$$f_h(y) = \frac{F(y+h) - F(y-h)}{2h}$$



then the natural estimate is

$$\widehat{f}_h(y) = \frac{\widehat{F}(y+h) - \widehat{F}(y-h)}{2h} = \frac{1}{2nh}\sum_{i=1}^n \underbrace{\mathbf{1}(y_i \in [y \pm h])}_{Z_i}$$

where Z_i's are Bernoulli $\mathcal{B}(p_x)$ i.id. variables with $p_x = \mathbb{P}[Y_i \in [x \pm h]] = 2h \cdot f_h(x)$. Thus, $\mathbb{E}(\widehat{f}_h(y)) = f_h(y)$, while

$$f_h(y) \sim f(y) + \frac{h^2}{6} f''(y) \text{ as } h \sim 0.$$

Kernels and Statistical Properties

Similarly, as $h \to 0$ and $nh \to \infty$

$$\text{Var}[\tilde{f}(y)] = \frac{1}{n}\left(\mathbb{E}[k_h(z-Z)^2] - (\mathbb{E}[k_h(z-Z)])^2\right)$$

$$\text{Var}[\tilde{f}(y)] = \frac{f(y)}{nh}\int k(u)^2 du + o\left(\frac{1}{nh}\right)$$

Hence

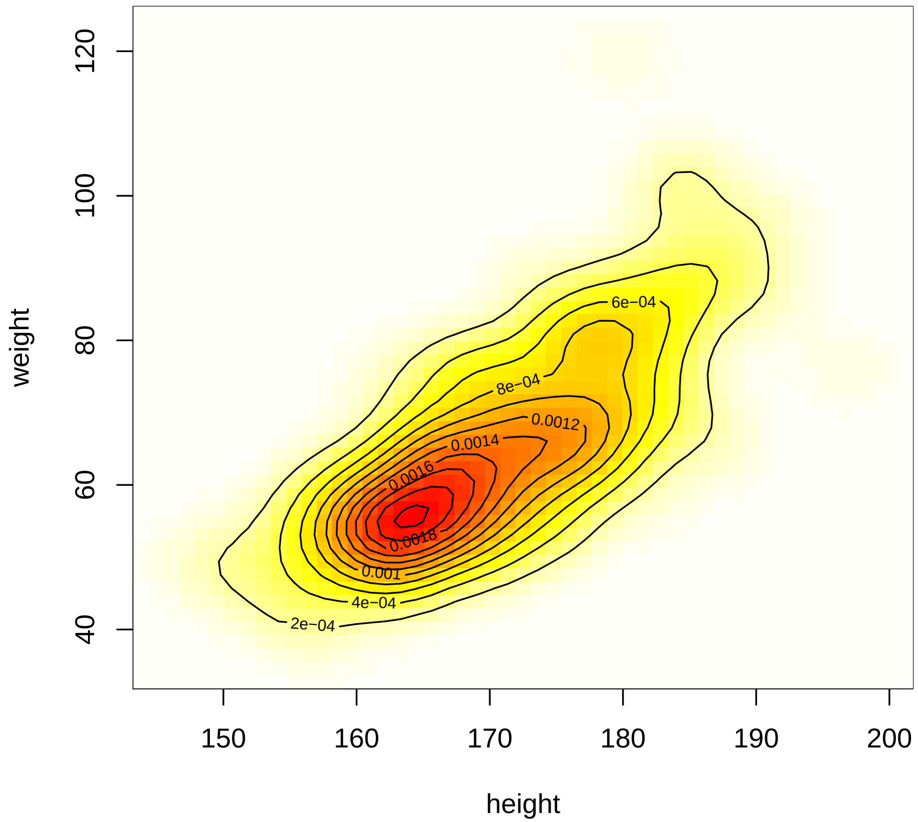
- if $h \to 0$ the bias goes to 0
- if $nh \to \infty$ the variance goes to 0

Kernels and Statistical Properties

Extension in Higher Dimension:

$$\tilde{f}(\boldsymbol{y}) = \frac{1}{n|\boldsymbol{H}|^{1/2}} \sum_{i=1}^{n} k\left(\boldsymbol{H}^{-1/2}(\boldsymbol{y} - \boldsymbol{y}_i)\right)$$

$$\tilde{f}(\boldsymbol{y}) = \frac{1}{nh^d|\boldsymbol{\Sigma}|^{1/2}} \sum_{i=1}^{n} k\left(\boldsymbol{\Sigma}^{-1/2}\frac{(\boldsymbol{y} - \boldsymbol{y}_i)}{h}\right)$$



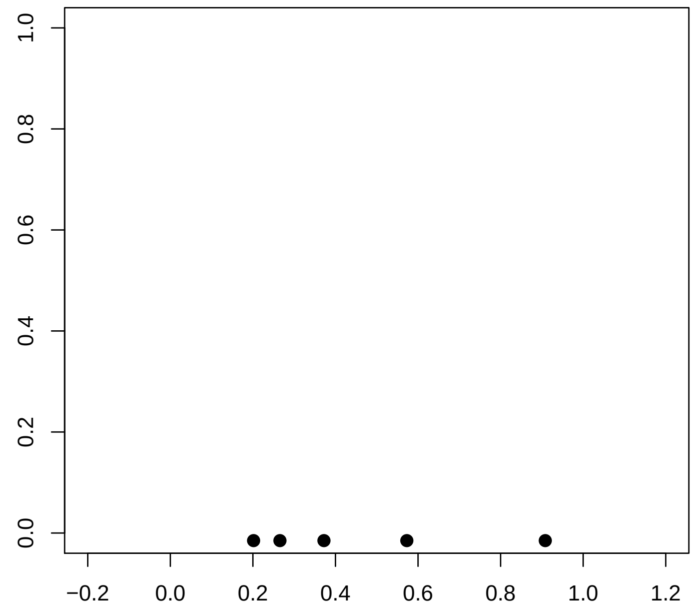
Kernels and Convolution

Given f and g, set

$(f\star g)(x)=\displaystyle\int_{\mathbb{R}} f(x-y)g(y)dy$

Then $\tilde f_h=(\hat f\star k_h)$, where

$$\hat f(y)=\frac{\hat F(y)}{dy}=\sum_{i=1}^n \delta_{y_i}(y)$$



Hence, $\tilde f$ is the distribution of $\widehat{Y}+\varepsilon$ where

$\widehat{Y}$ is uniform over $\{y_1,\cdots,y_n\}$ and $\varepsilon\sim k_h$ are independent

Nadaraya-Watson and Kernels

Here $\mathbb{E}[Y|X=x]=m(x)$. Write m as a function of densities

$$m(x)=\int yf(y|x)dy=\frac{\int yf(y,x)dy}{\int f(y,x)dy}$$

Consider some bivariate kernel k, such that

$$\int tk(t,u)dt=0 \text{ and } \kappa(u)=\int k(t,u)dt$$

For the numerator, it can be estimated using

$$\begin{aligned}\int y\tilde{f}(y,x)dy &= \frac{1}{nh^2}\sum_{i=1}^n\int yk\left(\frac{y-y_i}{h},\frac{x-x_i}{h}\right)\\ &= \frac{1}{nh}\sum_{i=1}^n\int y_ik\left(t,\frac{x-x_i}{h}\right)dt=\frac{1}{nh}\sum_{i=1}^n y_i\kappa\left(\frac{x-x_i}{h}\right)\end{aligned}$$

Nadaraya-Watson and Kernels

and for the denominator

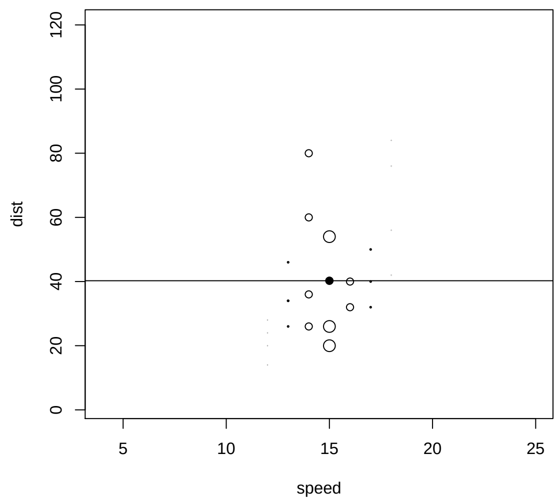
$$\int f(y,x)dy = \frac{1}{nh^2}\sum_{i=1}^{n}\int k\left(\frac{y-y_i}{h},\frac{x-x_i}{h}\right) = \frac{1}{nh}\sum_{i=1}^{n}\kappa\left(\frac{x-x_i}{h}\right)$$

Therefore, plugging in the expression for $g(x)$ yields

$$\tilde{m}(x) = \frac{\sum_{i=1}^{n} y_i\kappa_h(x-x_i)}{\sum_{i=1}^{n}\kappa_h(x-x_i)}$$

Observe that this regression estimator is a weighted average (see linear predictor section)

$$\tilde{m}(x) = \sum_{i=1}^{n}\omega_i(x)y_i \text{ with } \omega_i(x) = \frac{\kappa_h(x-x_i)}{\sum_{i=1}^{n}\kappa_h(x-x_i)}$$



Nadaraya-Watson and Kernels

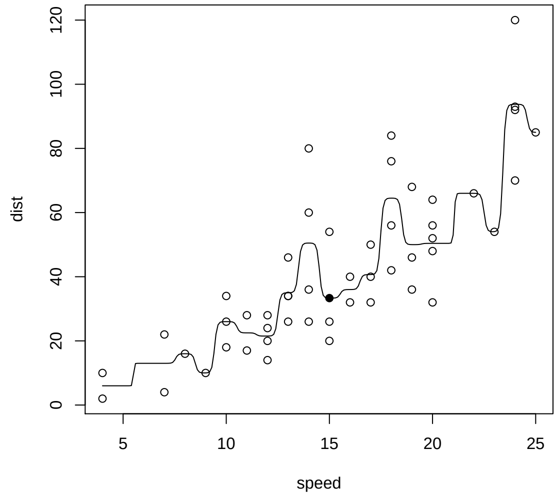
One can prove that kernel regression bias is given by

$$\mathbb{E}[\tilde{m}(x)] \sim m(x) + h^2 \left(\frac{C_1}{2} m''(x) + C_2 m'(x) \frac{f'(x)}{f(x)} \right)$$

while $\text{Var}[\tilde{m}(x)] \sim \frac{C_3}{nh} \frac{\sigma(x)}{f(x)}$. In this univariate case, one can easily get the kernel estimator of derivatives.

Actually, $\tilde{m}$ is a function of bandwidth h.

Note: this can be extended to multivariate $\boldsymbol{x}$.



Nadaraya-Watson and Kernels in Higher Dimension

Here $\widehat{m}_{\boldsymbol{H}}(\boldsymbol{x})=\dfrac{\sum_{i=1}^n y_i k_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})}{\sum_{i=1}^n k_{\boldsymbol{H}}(\boldsymbol{x}_i-\boldsymbol{x})}$ for some symmetric positive definite bandwidth matrix $\boldsymbol{H}$, and $k_{\boldsymbol{H}}(\boldsymbol{x})=\det[\boldsymbol{H}]^{-1}k(\boldsymbol{H}^{-1}\boldsymbol{x})$. Then

$$\mathbb{E}[\widehat{m}_{\boldsymbol{H}}(\boldsymbol{x})]\sim m(\boldsymbol{x})+\frac{C_1}{2}\text{trace}\big(\boldsymbol{H}^{\mathsf{T}}m''(\boldsymbol{x})\boldsymbol{H}\big)+C_2\frac{m'(\boldsymbol{x})^{\mathsf{T}}\boldsymbol{H}\boldsymbol{H}^{\mathsf{T}}\nabla f(\boldsymbol{x})}{f(\boldsymbol{x})}$$

while

$$\text{Var}[\widehat{m}_{\boldsymbol{H}}(\boldsymbol{x})]\sim\frac{C_3}{n\det(\boldsymbol{H})}\frac{\sigma(\boldsymbol{x})}{f(\boldsymbol{x})}$$

Hence, if $\boldsymbol{H}=h\mathbb{I}$, $h^\star\sim Cn^{-\frac{1}{4+\dim(\boldsymbol{x})}}$.

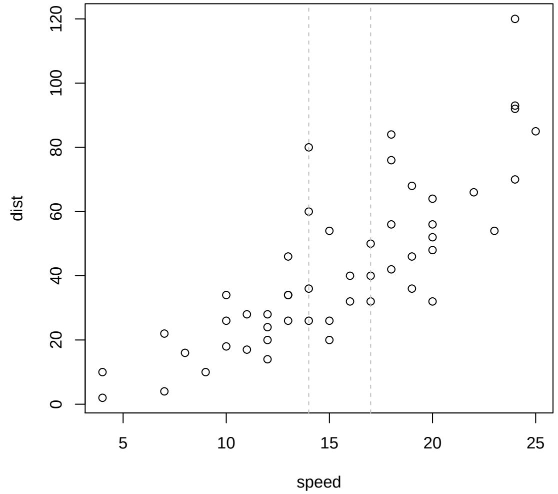
From kernels to k-nearest neighbours

An alternative is to consider

$$\tilde{m}_k(x) = \frac{1}{n}\sum_{i=1}^{n} \omega_{i,k}(x) y_i$$

where $\omega_{i,k}(x) = \dfrac{n}{k}$ if $i \in \mathcal{I}_x^k$ with

$\mathcal{I}_x^k = \{i : x_i \text{ one of the } k \text{ nearest observations to } x\}$



Lai (1977) Large sample properties of K-nearest neighbor procedures if $k \to \infty$ and $k/n \to 0$ as $n \to \infty$, then

$$\mathbb{E}[\tilde{m}_k(x)] \sim m(x) + \frac{1}{24 f(x)^3}\left[(m''f + 2m'f')(x)\right]\left(\frac{k}{n}\right)^2$$

while $\text{Var}[\tilde{m}_k(x)] \sim \dfrac{\sigma^2(x)}{k}$

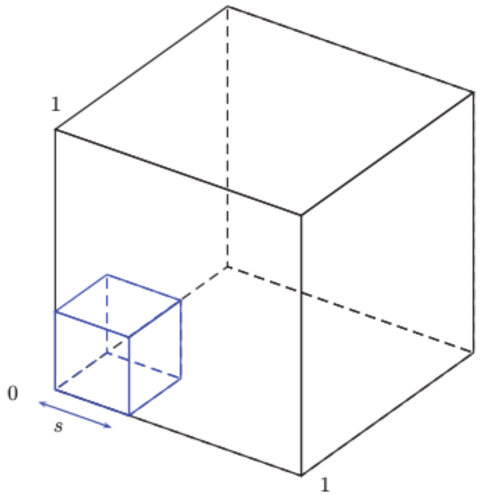
From kernels to k-nearest neighbours

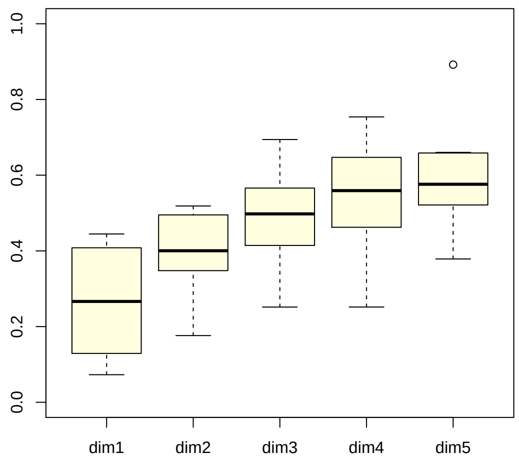
Remark: Brent & John (1985) Finding the median requires $2n$ comparisons considered some median smoothing algorithm, where we consider the median over the k nearest neighbours (see section #4).

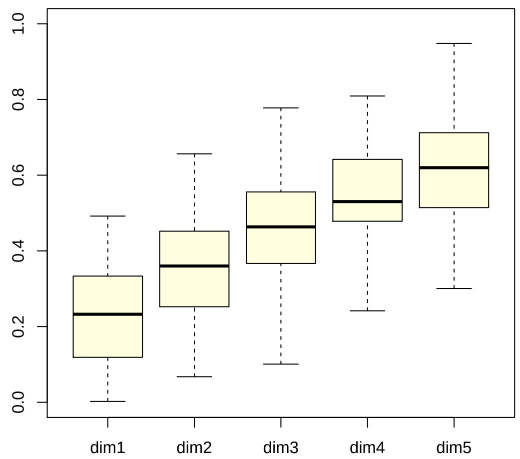
k-Nearest Neighbors and Curse of Dimensionality

The higher the dimension, the larger the distance to the closest neigbbor

$$\min_{i\in\{1,\cdots,n\}}\{d(\boldsymbol{a},\boldsymbol{x}_i)\},\boldsymbol{x}_i\in\mathbb{R}^{d}.$$







$n = 10$ $\qquad$ $n = 100$

Bandwidth selection : MISE for Density

$$MSE[\tilde{f}(y)] = \text{bias}[\tilde{f}(y)]^2 + \text{Var}[\tilde{f}(y)]$$

$$MSE[\tilde{f}(y)] = f(y)\frac{1}{nh}\int k(u)^2 du + h^4\left(\frac{f''(y)}{2}\int k(u)u^2 du\right)^2 + o\left(h^4 + \frac{1}{nh}\right)$$

Bandwidth choice is based on minimization of the asymptotic integrated MSE (over y)

$$MISE(\tilde{f}) = \int MSE[\tilde{f}(y)]dy \sim \frac{1}{nh}\int k(u)^2 du + h^4\int\left(\frac{f''(y)}{2}\int k(u)u^2 du\right)^2$$

Bandwidth selection : MISE for Density

Thus, the first-order condition yields

$$-\frac{C_1}{nh^2}+h^3\int f''(y)^2dyC_2=0$$

with $C_1=\int k^2(u)du$ and $C_2=\left(\int k(u)u^2du\right)^2$, and

$$h^\star=n^{-\frac{1}{5}}\left(\frac{C_1}{C_2\int f''(y)dy}\right)^{\frac{1}{5}}$$

$h^\star=1.06n^{-\frac{1}{5}}\sqrt{\text{Var}[Y]}$ from Silverman (1986) Density Estimation

```
> bw.nrd0(cars$speed)
[1] 2.150016
> bw.nrd(cars$speed)
[1] 2.532241
```

with Scott correction, see Scott (1992) Multivariate Density Estimation

Bandwidth selection : MISE for Regression Model

One can prove that

$$MISE[\widehat{m}_h] \sim \overbrace{\frac{h^4}{4}\left(\int x^2k(x)dx\right)^2\int\left[m''(x)+2m'(x)\frac{f'(x)}{f(x)}\right]^2dx}^{\text{bias}^2}$$
$$+\underbrace{\frac{\sigma^2}{nh}\int k^2(x)dx\cdot\int\frac{dx}{f(x)}}_{\text{variance}} \text{ as } n\to 0 \text{ and } nh\to\infty.$$

The bias is sensitive to the position of the x_i's.

$$h^\star = n^{-\frac{1}{5}}\left(\frac{C_1\int\frac{dx}{f(x)}}{C_2\int\left[m''(x)+2m'(x)\frac{f'(x)}{f(x)}\right]dx}\right)^{\frac{1}{5}}$$

Problem: depends on unknown $f(x)$ and $m(x)$.

Bandwidth Selection : Cross Validation

Let $R(h)=\mathbb{E}\big[(Y-\widehat{m}_h(\boldsymbol{X}))^2\big]$.

Natural idea $\widehat{R}(h)=\dfrac{1}{n}\displaystyle\sum_{i=1}^{n}(y_i-\widehat{m}_h(\boldsymbol{x}_i))^2$

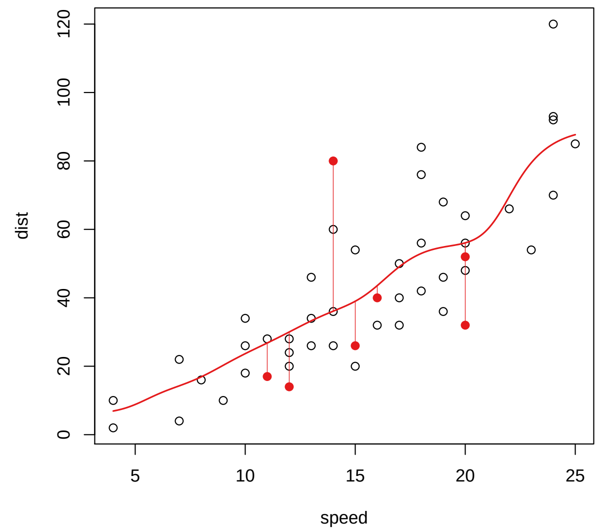
Instead use leave-one-out cross validation,

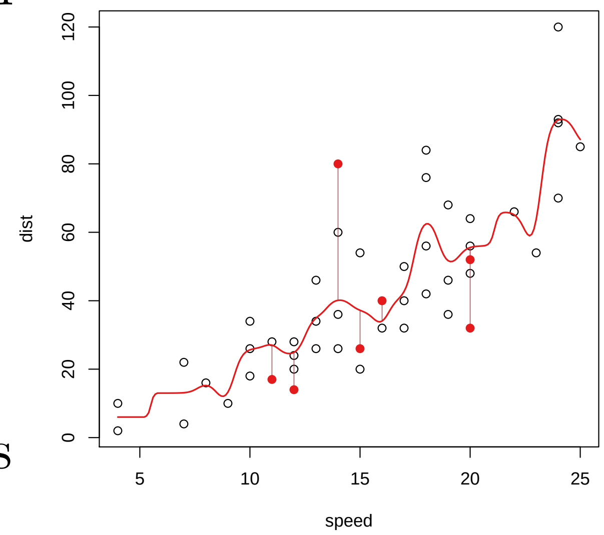
$$\widehat{R}(h)=\frac{1}{n}\sum_{i=1}^{n}\Big(y_i-\widehat{m}_h^{(i)}(\boldsymbol{x}_i)\Big)^2$$

where $\widehat{m}_h^{(i)}$ is the estimator obtained by omitting the ith pair $(y_i,\boldsymbol{x}_i)$ or k-fold cross validation,

$$\widehat{R}(h)=\frac{1}{n}\sum_{j=1}^{k}\sum_{i\in\mathcal{I}_j}\Big(y_i-\widehat{m}_h^{(j)}(\boldsymbol{x}_i)\Big)^2$$

where $\widehat{m}_h^{(j)}$ is the estimator obtained by omitting pairs $(y_i,\boldsymbol{x}_i)$ with $i\in\mathcal{I}_j$.



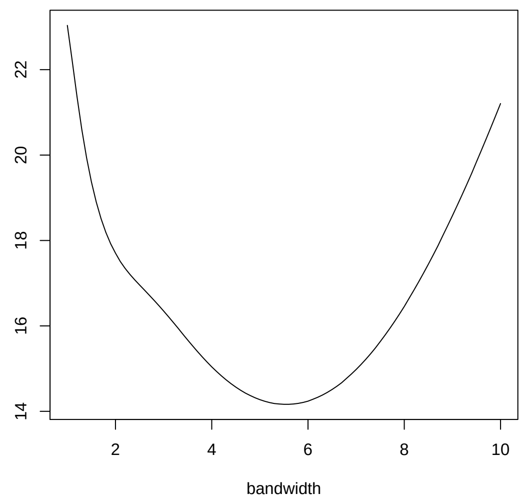


Bandwidth Selection : Cross Validation

Then find (numerically)

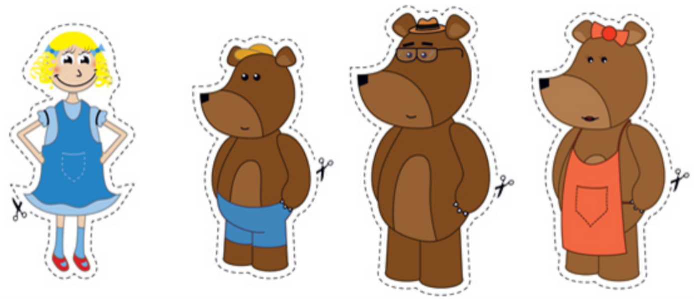
$$h^{\star} = \text{argmin}\{\widehat{R}(h)\}$$

In the context of density estimation, see Chiu (1991) Bandwidth Selection for Kernel Density Estimation



Usual bias-variance tradeoff, or Goldilock principle: h should be neither too small, nor too large

- undersmoothed: bias too large, variance too small
- oversmoothed: variance too large, bias too small



Local Linear Regression

Consider $\hat{m}(\boldsymbol{x})$ defined as $\hat{m}(\boldsymbol{x}) = \widehat{\beta}_0$ where $(\widehat{\beta}_0, \widehat{\boldsymbol{\beta}})$ is the solution of

$$\min_{(\beta_0,\boldsymbol{\beta})} \left\{ \sum_{i=1}^n \omega_i^{(\boldsymbol{x})} \left(y_i - [\beta_0 + (\boldsymbol{x} - \boldsymbol{x}_i)^\mathsf{T}\boldsymbol{\beta}]\right)^2 \right\}$$

where $\omega_i^{(\boldsymbol{x})} = k_h(\boldsymbol{x} - \boldsymbol{x}_i)$, e.g.

i.e. we seek the constant term in a weighted least squares regression of y_i's on $\boldsymbol{x} - \boldsymbol{x}_i$'s. If $\boldsymbol{X}_{\boldsymbol{x}}$ is the matrix $[\boldsymbol{1}\ (\boldsymbol{x} - \boldsymbol{X})^\mathsf{T}]$, and if $\boldsymbol{W}_{\boldsymbol{x}}$ is a matrix

$$\text{diag}[k_h(\boldsymbol{x} - \boldsymbol{x}_1), \cdots, k_h(\boldsymbol{x} - \boldsymbol{x}_n)]$$

then $\hat{m}(\boldsymbol{x}) = \boldsymbol{1}^\mathsf{T}(\boldsymbol{X}_{\boldsymbol{x}}^\mathsf{T}\boldsymbol{W}_{\boldsymbol{x}}\boldsymbol{X}_{\boldsymbol{x}})^{-1}\boldsymbol{X}_{\boldsymbol{x}}^\mathsf{T}\boldsymbol{W}_{\boldsymbol{x}}\boldsymbol{y}$

This estimator is also a linear predictor :

$$\hat{m}(\boldsymbol{x}) = \sum_{i=1}^n \frac{a_i(\boldsymbol{x})}{\sum a_i(\boldsymbol{x})} y_i$$

where

$$a_i(\boldsymbol{x}) = \frac{1}{n}k_h(\boldsymbol{x}-\boldsymbol{x}_i)\left(1 - s_1(\boldsymbol{x})^\mathsf{T} s_2(\boldsymbol{x})^{-1}\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)$$

with

$$s_1(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^n k_h(\boldsymbol{x}-\boldsymbol{x}_i)\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h} \text{ and } s_2(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^n k_h(\boldsymbol{x}-\boldsymbol{x}_i)\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right)$$

Note that Nadaraya-Watson estimator was simply the solution of

$$\min_{\beta_0}\left\{\sum_{i=1}^n \omega_i^{(\boldsymbol{x})}(y_i-\beta_0)^2\right\} \text{ where } \omega_i^{(\boldsymbol{x})} = k_h(\boldsymbol{x}-\boldsymbol{x}_i)$$

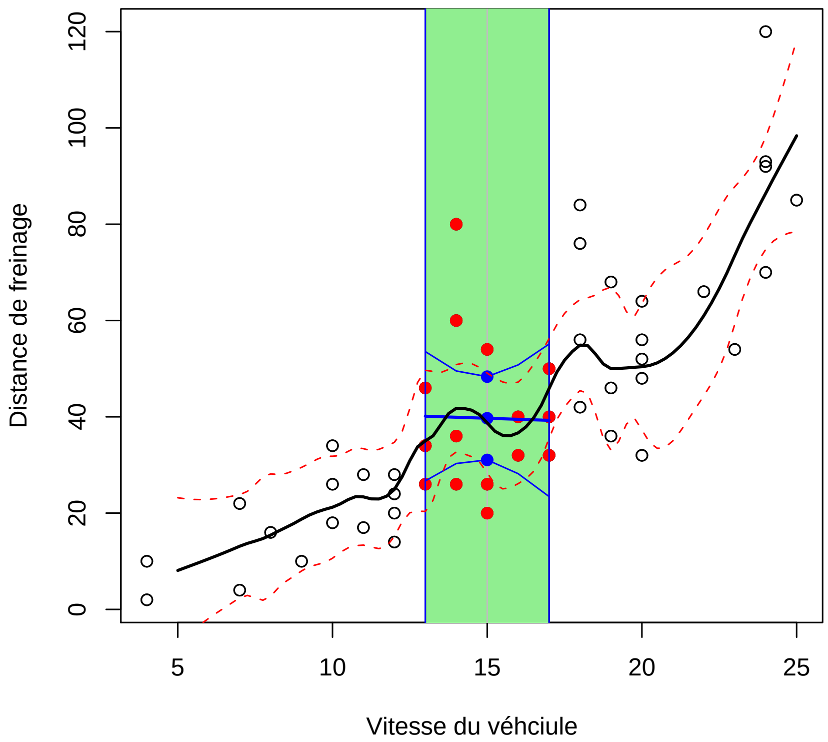
$$\mathbb{E}[\hat{m}(\boldsymbol{x})] \sim m(\boldsymbol{x}) + \frac{h^2}{2}m''(\boldsymbol{x})\mu_2 \text{ where } \mu_2 = \int k(u)u^2du.$$

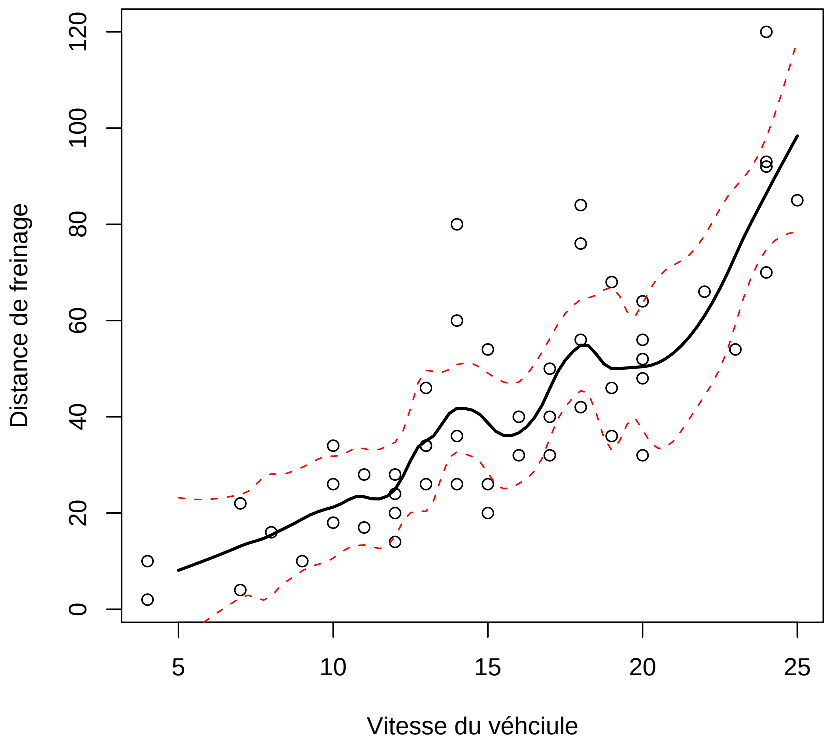
$$\text{Var}[\hat{m}(\boldsymbol{x})] \sim \frac{1}{nh}\frac{\nu\sigma_{\boldsymbol{x}}^2}{f(\boldsymbol{x})}$$

where $\nu = \int k(u)^2 du$

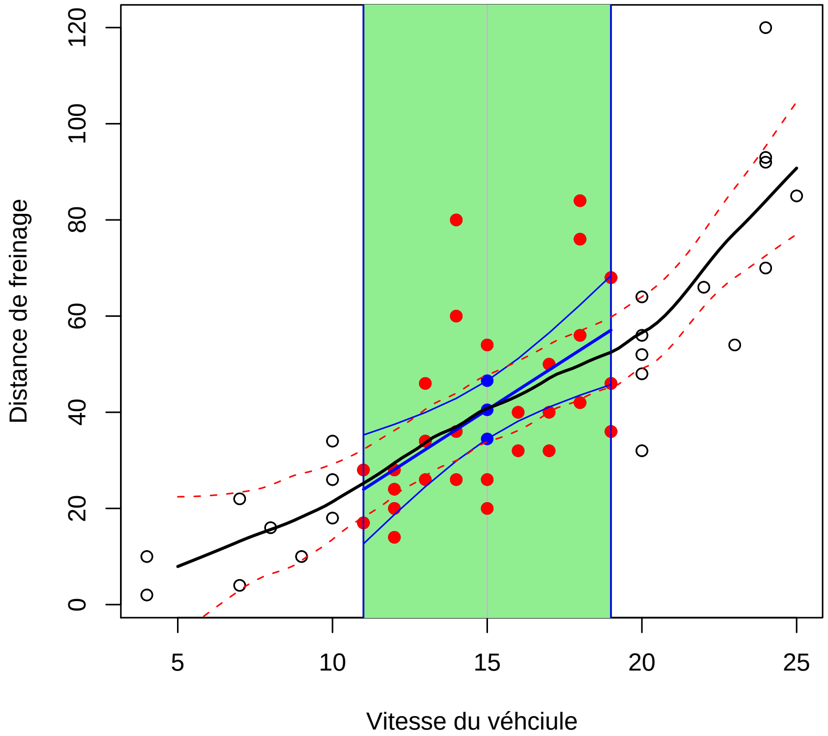
Thus, kernel regression MSE is

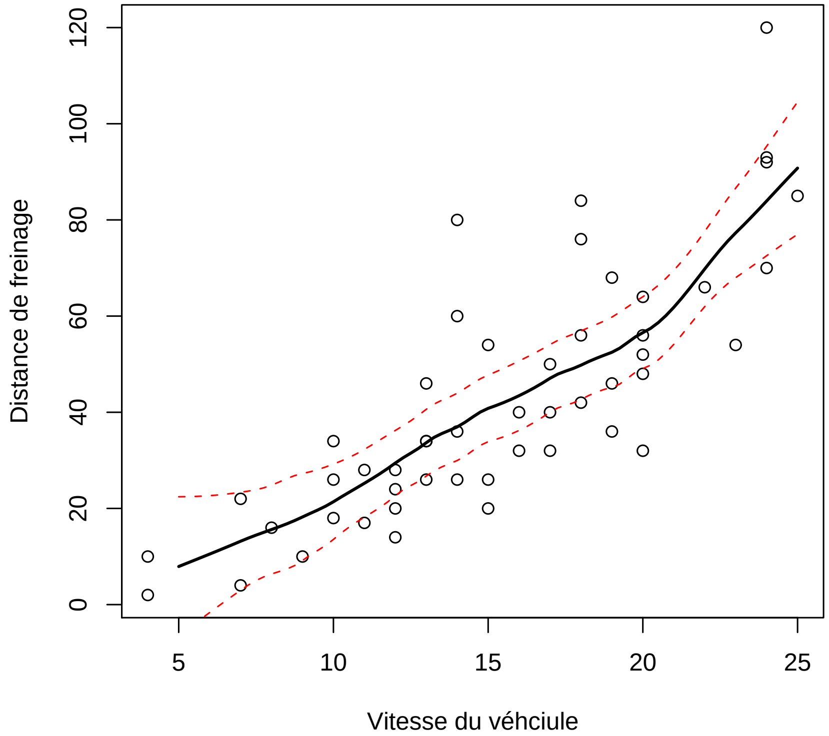
$$\frac{h^2}{4}\left(g''(x) + 2g'(x)\frac{f'(x)}{f(x)}\right)^2 \mu_2^2 + \frac{1}{nh}\frac{\nu\sigma_{\boldsymbol{x}}^2}{f(\boldsymbol{x})}$$





```
> loess(dist ~ speed, cars,span=0.75,degree=1)
> predict(REG, data.frame(speed = seq(5, 25, 0.25)), se = TRUE)
```





Local polynomials

One might assume that, locally, $m(x) \sim \mu_x(u)$ as $u \sim 0$, with

$$\mu_x(u) = \beta_0^{(x)} + \beta_1^{(x)} + [u-x] + \beta_2^{(x)} + \frac{[u-x]^2}{2} + \beta_3^{(x)} + \frac{[u-x]^3}{2} + \cdots$$

and we estimate $\boldsymbol{\beta}^{(x)}$ by minimizing $\displaystyle\sum_{i=1}^{n} \omega_i^{(x)} \big[y_i - \mu_x(x_i)\big]^2$.

If $\boldsymbol{X}_x$ is the design matrix $\left[1 \; x_i - x \; \dfrac{[x_i-x]^2}{2} \; \dfrac{[x_i-x]^3}{3} \; \cdots\right]$, then

$$\widehat{\boldsymbol{\beta}}^{(x)} = \left(\boldsymbol{X}_x^\top \boldsymbol{W}_x \boldsymbol{X}_x\right)^{-1} \boldsymbol{X}_x^\top \boldsymbol{W}_x \boldsymbol{y}$$

(weighted least squares estimators).

```
> library(locfit)
> locfit(dist~speed,data=cars)
```

Series Regression

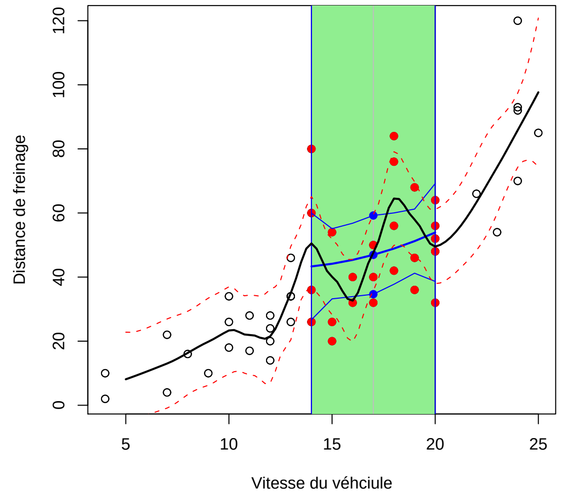
Recall that $\mathbb{E}[Y|X=x]=m(x)$.

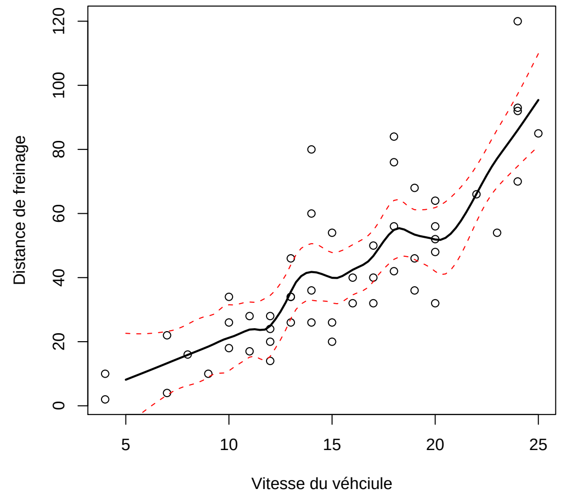
Why not approximate m by a linear combination of approximating functions $h_1(x),\cdots,h_k(x)$.

Set $\boldsymbol{h}(x)=(h_1(x),\cdots,h_k(x))$, and consider the regression of y_i's on $\boldsymbol{h}(x_i)$'s,

$$y_i=\boldsymbol{h}(x_i)^{\mathsf{T}}\boldsymbol{\beta}+\varepsilon_i$$

Then $\widehat{\boldsymbol{\beta}}=(\boldsymbol{H}^{\mathsf{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^{\mathsf{T}}\boldsymbol{y}$



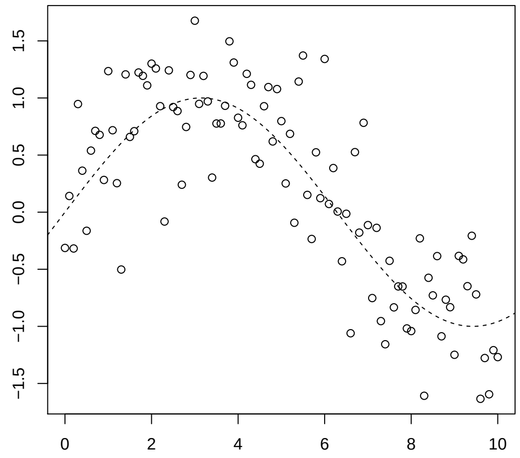


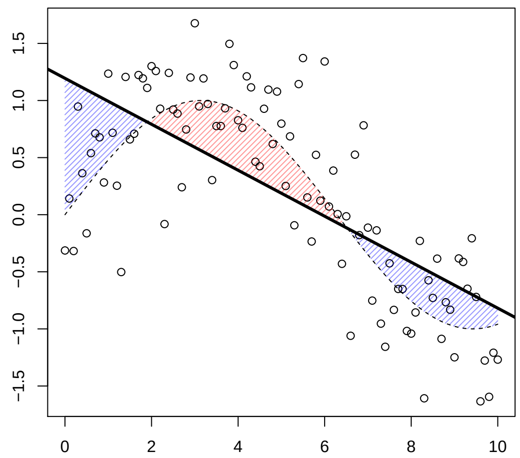
Series Regression : polynomials

Even if $m(x) = \mathbb{E}(Y|X = x)$ is not a polynomial function, a polynomial can still be a good approximation.

From Stone-Weierstrass theorem, if $m(\cdot)$ is continuous on some interval, then there is a uniform approximation of $m(\cdot)$ by polynomial functions.

```
> reg <- lm(y~x,data=db)
```

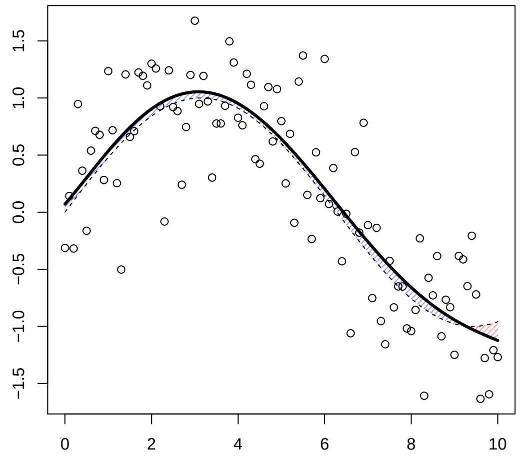


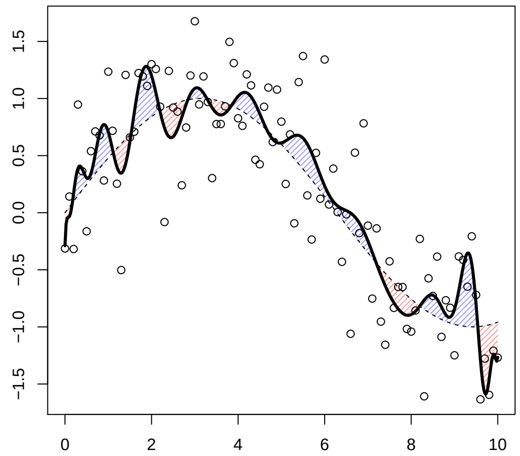


Series Regression : polynomials

Assume that $m(x) = \mathbb{E}(Y|X = x) = \sum_{i=0}^{k} \alpha_i x^i$, where parameters $\alpha_0, \cdots, \alpha_k$ will be estimated (but not k).

```
> reg <- lm(y~poly(x,5),data=db)
> reg <- lm(y~poly(x,25),data=db)
```





Series Regression : (Linear) Splines

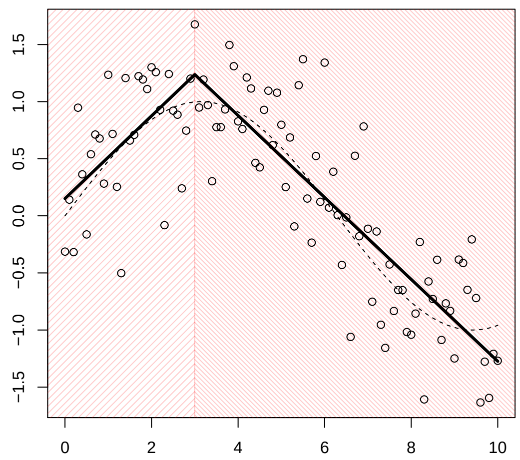
Consider $m+1$ knots on $\mathcal{X}$, $\min\{x_i\} \leq t_0 \leq t_1 \leq \cdots \leq t_m \leq \max\{x_n\}$, then define linear (degree $=1$) splines positive function,

$$b_{j,1}(x) = (x - t_j)_+ = \begin{cases} x - t_j \text{ if } x > t_j \\ 0 \text{ otherwise} \end{cases}$$

for linear splines, consider

$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 (X_i - s)_+ + \varepsilon_i$$

```
> positive_part <- function(x) ifelse(x>0,x,0)
> reg <- lm(Y~X+positive_part(X-s), data=db)
```



Series Regression : (Linear) Splines

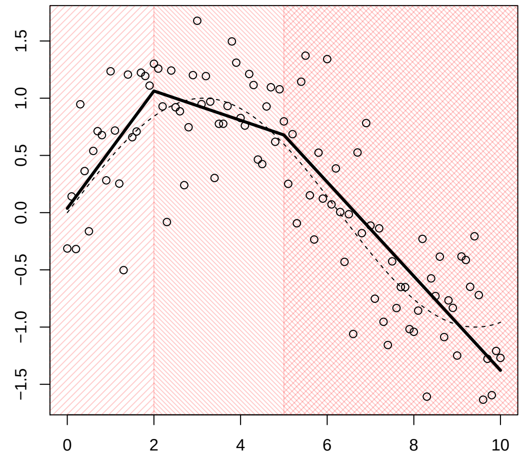
for linear splines, consider

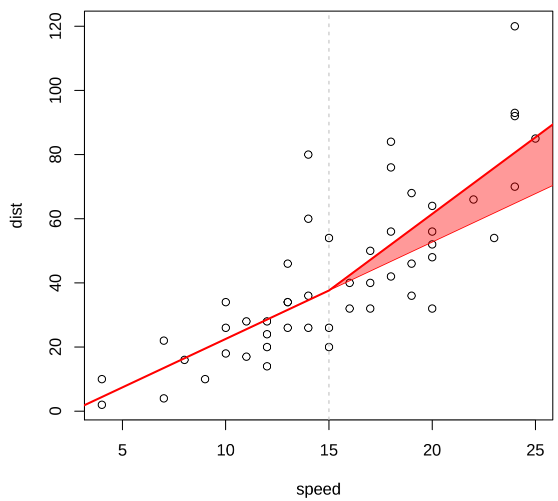
$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 (X_i - s_1)_+ + \beta_3 (X_i - s_2)_+ + \varepsilon_i$$

```
> reg <- lm(Y~X+positive_part(X-s1)+
                      positive_part(X-s2), data=db)
> library(bsplines)
```

A spline is a function defined by piecewise polynomials.

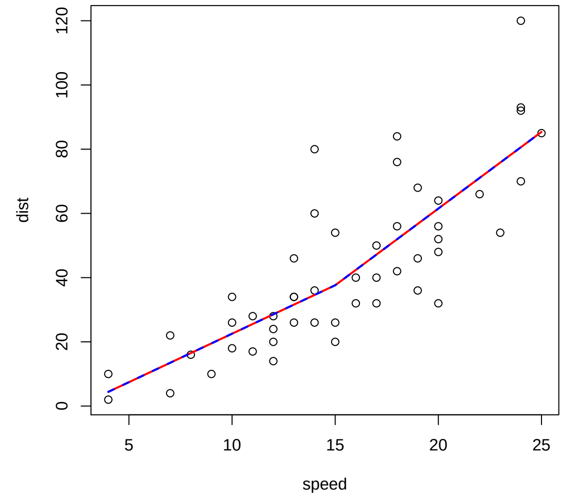
b-splines are defined recursively





b-Splines (in Practice)

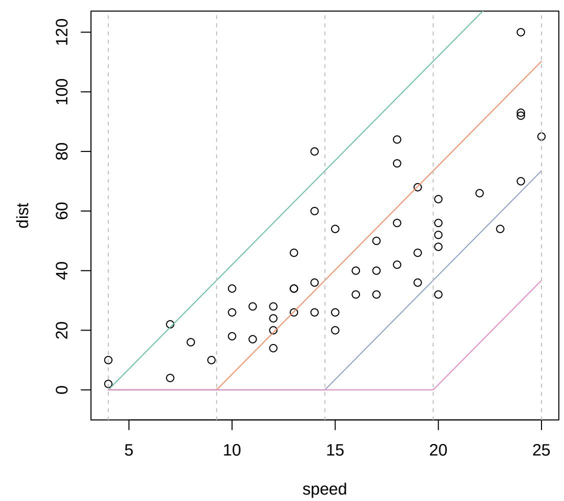
```
> reg1 <- lm(dist~speed+positive_part(speed-15),
    data=cars)
> reg2 <- lm(dist~bs(speed,df=2,degree=1), data=
    cars)
```



Consider $m+1$ knots on $[0,1]$, $0 \leq t_0 \leq t_1 \leq \cdots \leq t_m \leq 1$, then define recursively b-splines as

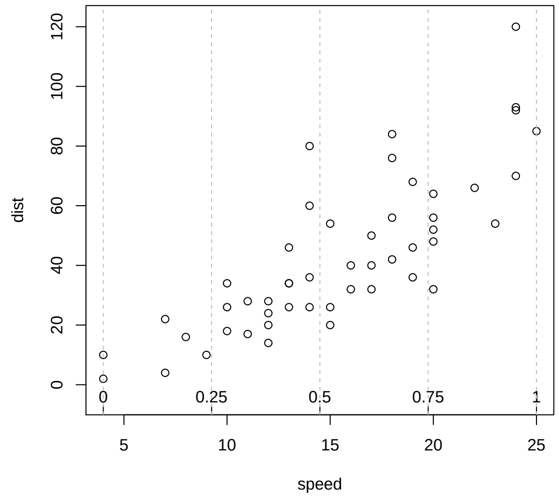
$$b_{j,0}(t) = \begin{cases} 1 \text{ if } & t_j \leq t < t_{j+1} \\ 0 \text{ otherwise, and} \end{cases}$$

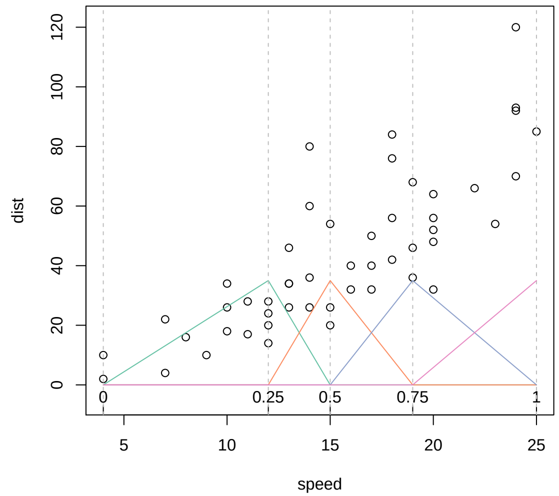
$$b_{j,n}(t) = \frac{t - t_j}{t_{j+n} - t_j} b_{j,n-1}(t) + \frac{t_{j+n+1} - t}{t_{j+n+1} - t_{j+1}} b_{j+1,n-1}(t)$$



b-Splines (in Practice)

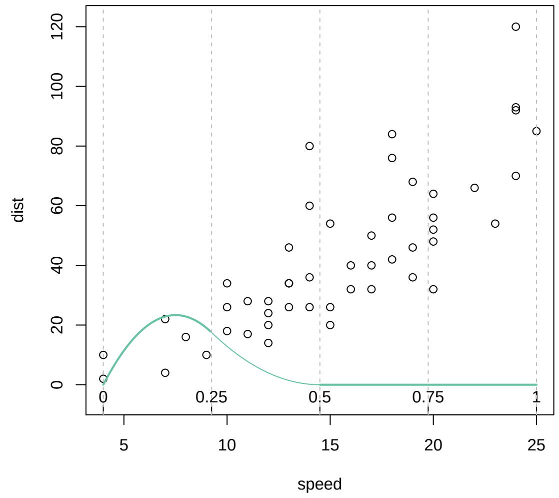
```
> summary(reg1)

Coefficients:
            Estimate Std Error t value Pr(>|t|)
(Intercept)-7.6519   10.6254   -0.720  0.475
speed        3.0186    0.8627   3.499  0.001 **
(speed-15)   1.7562    1.4551   1.207  0.233

> summary(reg2)

Coefficients:
           Estimate Std Error t value Pr(>|t|)
(Intercept) 4.423     7.343    0.602  0.5493
bs(speed)1 33.205     9.489    3.499  0.0012 **
bs(speed)2 80.954     8.788    9.211  4.2e-12 ***
```



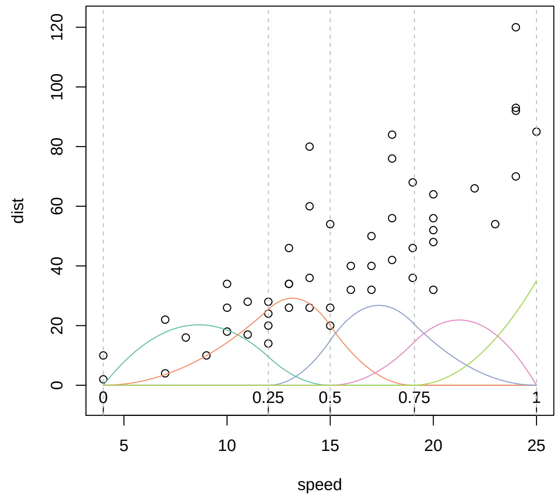


b and p-Splines

Note that those spline function define an orthonormal basis.



O'Sullivan (1986) A statistical perspective on ill-posed inverse problems suggested a penalty on the second derivative of the fitted curve (see #3).



$$m(x) = \text{argmin}\left\{ \sum_{i=1}^{n} \left(y_i - \boldsymbol{b}(x_i)^{\mathsf{T}}\boldsymbol{\beta}\right)^2 + \lambda \int_{\mathbb{R}} \boldsymbol{b}''(x_i)^{\mathsf{T}}\boldsymbol{\beta} \right\}$$

Adding Constraints: Convex Regression

Assume that $y_i = m(\boldsymbol{x}_i) + \varepsilon_i$ where $m : \mathbb{R}^d \to \infty\mathbb{R}$ is some convex function.

m is convex if and only if $\forall \boldsymbol{x}_1, \boldsymbol{x}_2 \in \mathbb{R}^d$, $\forall t \in [0,1]$,

$$m(t\boldsymbol{x}_1 + [1-t]\boldsymbol{x}_2) \leq tm(\boldsymbol{x}_1) + [1-t]m(\boldsymbol{x}_2)$$

Proposition (Hidreth (1954) Point Estimates of Ordinates of Concave Functions)

$$m^\star = \underset{m \text{ convex}}{\text{argmin}} \left\{ \sum_{i=1}^n \big(y_i - m(\boldsymbol{x_i})\big)^2 \right\}$$

Then $\boldsymbol{\theta}^\star = (m^\star(\boldsymbol{x_1}), \cdots, m^\star(\boldsymbol{x_n}))$ is unique.

Let $\boldsymbol{y} = \boldsymbol{\theta} + \boldsymbol{\varepsilon}$, then

$$\boldsymbol{\theta}^\star = \underset{\boldsymbol{\theta} \in \mathcal{K}}{\text{argmin}} \left\{ \sum_{i=1}^n \big(y_i - \theta_i)\big)^2 \right\}$$

where $\mathcal{K} = \{\boldsymbol{\theta} \in \mathbb{R}^n : \exists m \text{ convex }, m(\boldsymbol{x}_i) = \theta_i\}$. I.e. $\boldsymbol{\theta}^\star$ is the projection of $\boldsymbol{y}$ onto the (closed) convex cone $\mathcal{K}$. The projection theorem gives existence and unicity.

Adding Constraints: Convex Regression

In dimension 1: $y_i = m(x_i) + \varepsilon_i$. Assume that observations are ordered $x_1 < x_2 < \cdots < x_n$.

Here

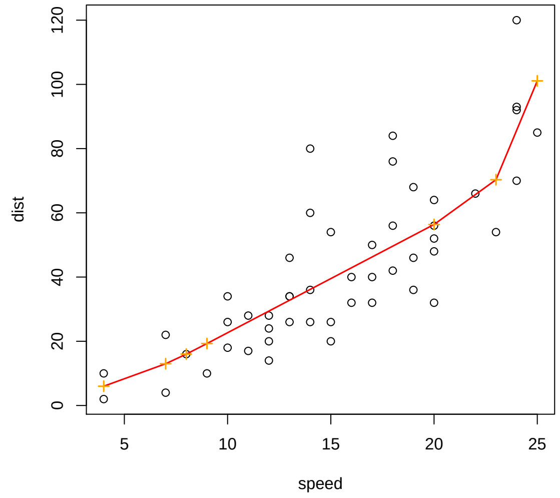
$$\mathcal{K} = \left\{ \boldsymbol{\theta} \in \mathbb{R}^n : \frac{\theta_2 - \theta_1}{x_2 - x_1} \leq \frac{\theta_3 - \theta_2}{x_3 - x_2} \leq \cdots \leq \frac{\theta_n - \theta_{n-1}}{x_n - x_{n-1}} \right\}$$

Hence, quadratic program with $n-2$ linear constraints.

$m^\star$ is a piecewise linear function (interpolation of consecutive pairs $(x_i, \theta_i^\star)$).

If m is differentiable, m is convex if

$$m(\boldsymbol{x}) + \nabla m(\boldsymbol{x}) \cdot [\boldsymbol{y} - \boldsymbol{x}] \leq m(\boldsymbol{y})$$



Adding Constraints: Convex Regression

More generally: if m is convex, then there exists $\xi_{\boldsymbol{x}} \in \mathbb{R}^n$ such that

$$m(\boldsymbol{x}) + \xi_{\boldsymbol{x}} \cdot [\boldsymbol{y} - \boldsymbol{x}] \leq m(\boldsymbol{y})$$

$\xi_{\boldsymbol{x}}$ is a subgradient of m at $\boldsymbol{x}$. And then

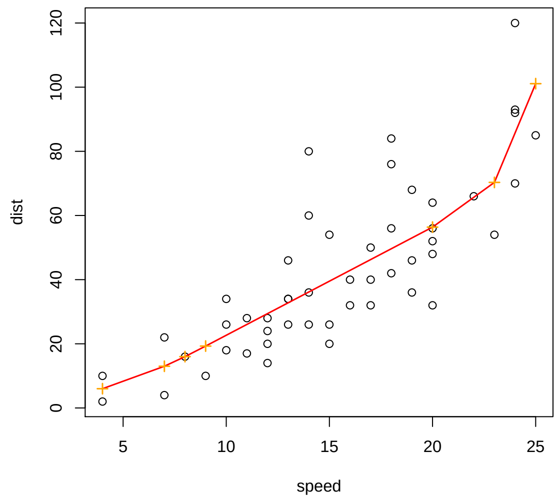
$$\partial m(\boldsymbol{x}) = \left\{m(\boldsymbol{x}) + \xi \cdot [\boldsymbol{y} - \boldsymbol{x}] \leq m(\boldsymbol{y}), \forall \boldsymbol{y} \in \mathbb{R}^n\right\}$$

Hence, $\boldsymbol{\theta}^\star$ is solution of

$$\text{argmin}\left\{\|\boldsymbol{y} - \boldsymbol{\theta}\|^2\right\}$$

$$\text{subject to } \theta_i + \xi_i[\boldsymbol{x}_j - \boldsymbol{x}_i] \leq \boldsymbol{\theta}_j, \ \forall i, j$$

and $\xi_1, \cdots, \xi_n \in \mathbb{R}^n$.



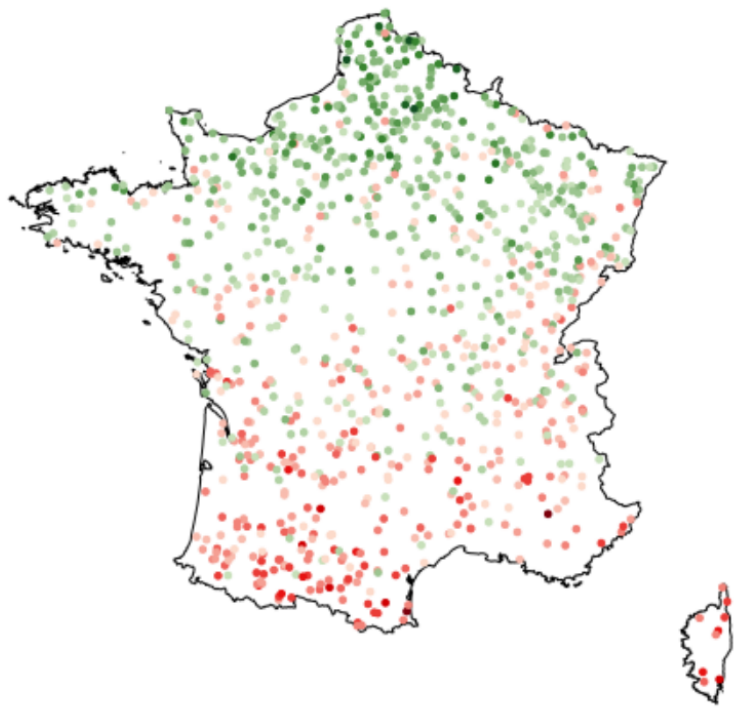
Spatial Smoothing

One can also consider some spatial smoothing, if we want to predict $\mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}]$ for some coordinate $\boldsymbol{x}$.

```
> library(rgeos)
> library(rgdal)
> library(maptools)
> library(cartography)
> download.file("http://bit.ly/2G3KIUG","zonier.RData")
> load("zonier.RData")
> cols=rev(carto.pal(pal1="red.pal",n1=10,pal2="green.pal",n2=10))
> download.file("http://bit.ly/2GSvzGW","FRA_adm0.rds")
> download.file("http://bit.ly/2FUZ0Lz","FRA_adm2.rds")
> FR=readRDS("FRA_adm2.rds")
> donnees_carte=data.frame(FRdata)
```

Spatial Smoothing

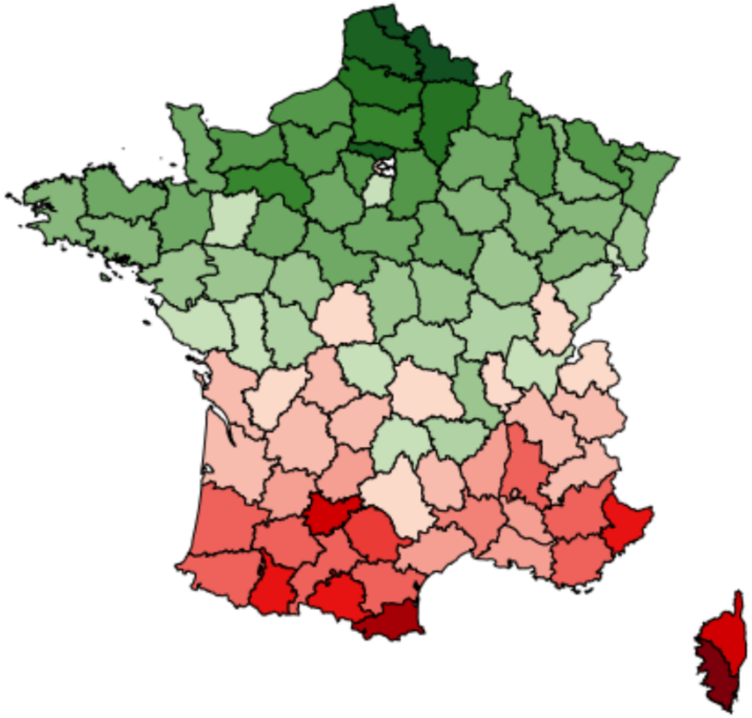
```
> FR0=readRDS("FRA_adm0.rds")
> plot(FR0)
> bk = seq(-5,4.5,length=21)
> cuty = cut(simbase$Y,breaks=bk,labels
    =1:20)
> points(simbase$long,simbase$lat,col=cols
    [cuty],pch=19,cex=.5)
```



One can consider a choropleth map (spatial version of the histogram).

Spatial Smoothing

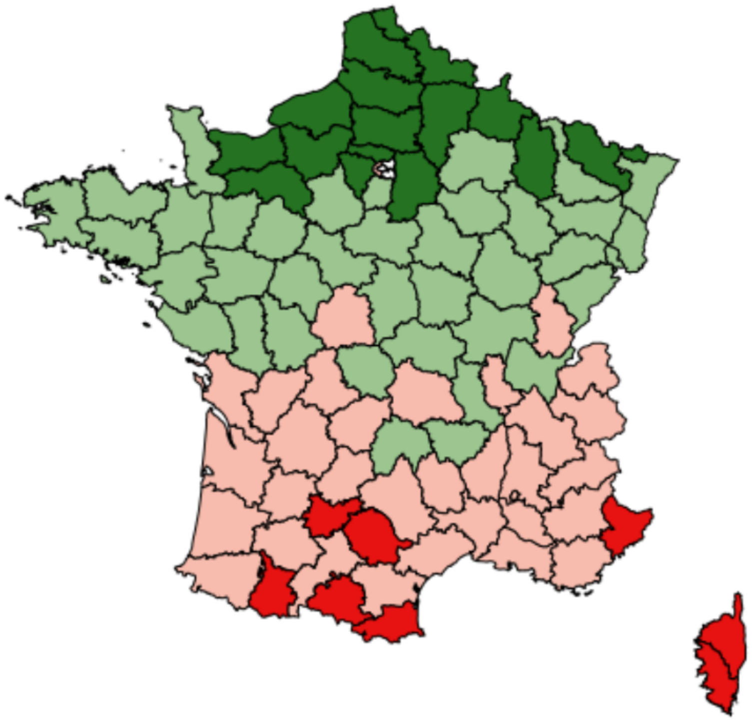
```
> A=aggregate(x = simbase$Y,by=list(
    simbase$dpt),mean)
> names(A)=c("dpt","y")
> d=donnees_carte$CCA_2
> d[d=="2A"]="201"
> d[d=="2B"]="202"
> donnees_carte$dpt=as.numeric(as.
    character(d))
> donnees_carte=merge(donnees_carte,A,all.
    x=TRUE)
> donnees_carte=donnees_carte[order(
    donnees_carte$OBJECTID),]
> bk=seq(-2.75,2.75,length=21)
> donnees_carte$cuty=cut(donnees_carte$y,
    breaks=bk,labels=1:20)
> plot(FR, col=cols[donnees_carte$cuty],
    xlim=c(-5.2,12))
```



Spatial Smoothing

Instead of a "continuous" gradient of colors, one can consider only 4 colors (4 levels) for the prediction.

```
> bk=seq(-2.75,2.75,length=5)
> donnees_carte$cuty=cut(donnees_carte$y,
    breaks=bk,labels=1:4)
> plot(FR, col=cols[c(3,8,12,17)][donnees_
    carte$cuty],xlim=c(-5.2,12))
```



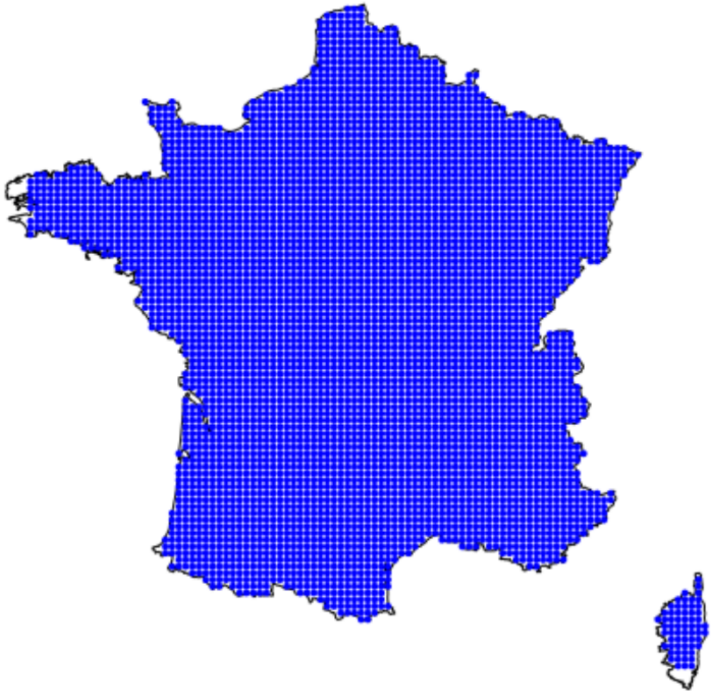
Spatial Smoothing

```
> P1 = FRO@polygons[[1]]@Polygons[[355]]@coords
> P2 = FRO@polygons[[1]]@Polygons[[27]]@coords
> plot(FRO,border=NA)
> polygon(P1)
> polygon(P2)
> grille<-expand.grid(seq(min(simbase$long),max(simbase$long),length=101),seq(min(simbase$lat),max(simbase$lat),length=101))
> paslong=(max(simbase$long)-min(simbase$long))/100
> paslat=(max(simbase$lat)-min(simbase$lat))/100
```

Spatial Smoothing

We need to create a grid (i.e. $\mathcal{X}$) on which we approximate $\mathbb{E}[Y|\boldsymbol{X}=\boldsymbol{x}]$

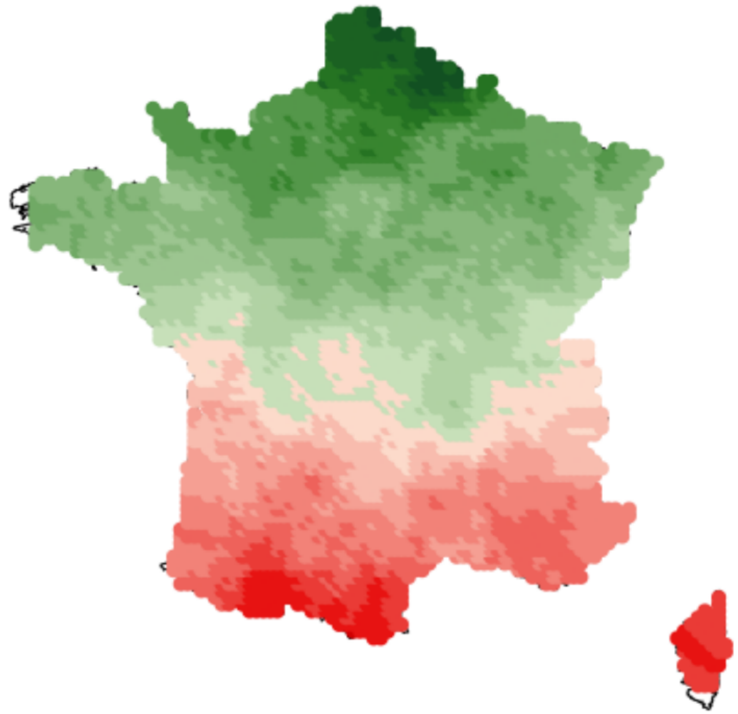
```
> f=function(i){ (point.in.polygon (grille
    [i, 1]+paslong/2 , grille[i, 2]+paslat/
    2 , P1[,1],P1[,2])>0)+(point.in.polygon
     (grille[i, 1]+paslong/2 , grille[i,
    2]+paslat/2 , P2[,1],P2[,2])>0) }
> indic=unlist(lapply(1:nrow(grille),f))
> grille=grille[which(indic==1),]
> points(grille[,1]+paslong/2,grille[,2]+
    paslat/2)
```



Spatial Smoothing

Consider here some k-NN, with $k = 20$

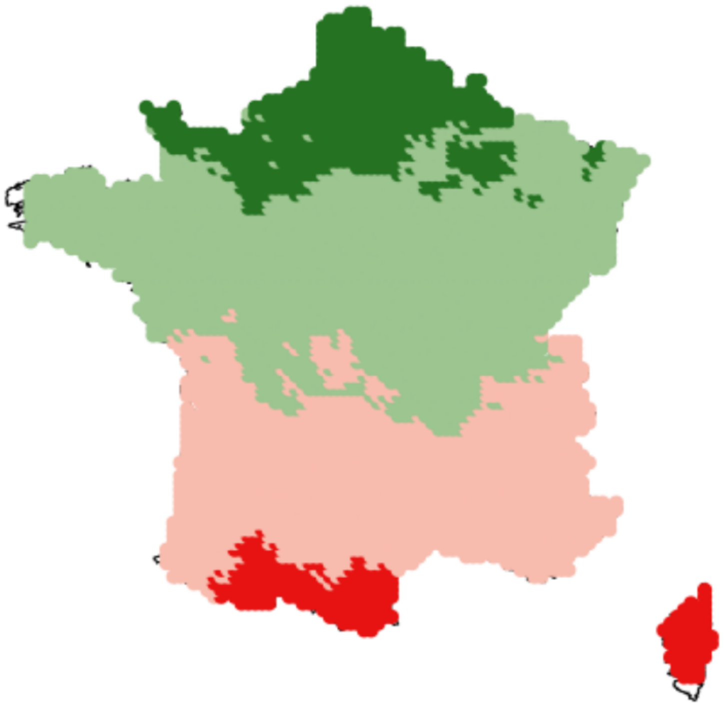
```
> library(geosphere)
> knn=function(i,k=20){
+ d=distHaversine(grille[i,1:2],simbase[,c
    ("long","lat")], r=6378.137)
+  r=rank(d)
+  ind=which(r<=k)
+  mean(simbase[ind,"Y"])
+ }
> grille$y=Vectorize(knn)(1:nrow(grille))
> bk=seq(-2.75,2.75,length=21)
> grille$cuty=cut(grille$y,breaks=bk,
    labels=1:20)
> points(grille[,1]+paslong/2,grille[,2]+
    paslat/2,col=cols[grille$cuty],pch=19)
```



Spatial Smoothing

Again, instead of a "continuous" gradient, we can use 4 levels,

```
> bk=seq(-2.75,2.75,length=5)
> grille$cuty=cut(grille$y,breaks=bk,
    labels=1:4)
> plot(FR0,border=NA)
> polygon(P1)
> polygon(P2)
> points(grille[,1]+paslong/2,grille[,2]+
    paslat/2,col=cols[c(3,8,12,17)][grille$
    cuty],pch=19)
```



Testing (Non-)Linearities

In the linear model,

$$\widehat{\boldsymbol{y}} = \boldsymbol{X}\widehat{\boldsymbol{\beta}} = \underbrace{\boldsymbol{X}[\boldsymbol{X}^\top\boldsymbol{X}]^{-1}\boldsymbol{X}^\top}_{\boldsymbol{H}}\boldsymbol{y}$$

$\boldsymbol{H}_{i,i}$ is the leverage of the ith element of this hat matrix.

Write

$$\widehat{y}_i = \sum_{j=1}^n [\boldsymbol{X}_i^\top[\boldsymbol{X}^\top\boldsymbol{X}]^{-1}\boldsymbol{X}^\top]_j y_j = \sum_{j=1}^n [\mathcal{H}(\boldsymbol{X}_i)]_j y_j$$

where

$$\mathcal{H}(\boldsymbol{x}) = \boldsymbol{x}^\top[\boldsymbol{X}^\top\boldsymbol{X}]^{-1}\boldsymbol{X}^\top$$

The prediction is

$$m(\boldsymbol{x}) = \mathbb{E}(Y|\boldsymbol{X}=\boldsymbol{x}) = \sum_{j=1}^n [\mathcal{H}(\boldsymbol{x})]_j y_j$$

Testing (Non-)Linearities

More generally, a predictor m is said to be linear if for all $\boldsymbol{x}$ if there is $\mathcal{S}(\cdot):\mathbb{R}^n\rightarrow\mathbb{R}^n$ such that

$$m(\boldsymbol{x})=\sum_{j=1}^n \mathcal{S}(\boldsymbol{x})_j y_j$$

Conversely, given $\widehat{y}_1,\cdots,\widehat{y}_n$, there is a matrix $\boldsymbol{S}$ $n\times n$ such that

$$\widehat{\boldsymbol{y}}=\boldsymbol{S}\boldsymbol{y}$$

For the linear model, $\boldsymbol{S}=\boldsymbol{H}$.

$\text{trace}(\boldsymbol{H})=\dim(\boldsymbol{\beta})$: degrees of freedom

$\dfrac{\boldsymbol{H}_{i,i}}{1-\boldsymbol{H}_{i,i}}$ is related to Cook's distance, from Cook (1977), Detection of Influential Observations in Linear Regression.

Testing (Non-)Linearities

For a kernel regression model, with kernel k and bandwidth h

$$S_{i,j}^{(k,h)} = \frac{k_h(x_i - x_j)}{\displaystyle\sum_{k=1}^{n} k_h(x_k - x_j)}$$

where $k_h(\cdot) = k(\cdot/h)$, while $\mathcal{S}^{(k,h)}(\boldsymbol{x})_j = \dfrac{K_h(\boldsymbol{x} - x_j)}{\displaystyle\sum_{k=1}^{n} k_h(\boldsymbol{x} - x_k)}$

For a k-nearest neighbor, $S_{i,j}^{(k)} = \dfrac{1}{k}\mathbf{1}(j \in \mathcal{I}_{\boldsymbol{x}_i})$ where $\mathcal{I}_{\boldsymbol{x}_i}$ are the k nearest observations to $\boldsymbol{x}_i$, while $\mathcal{S}^{(k)}(\boldsymbol{x})_j = \dfrac{1}{k}\mathbf{1}(j \in \mathcal{I}_{\boldsymbol{x}})$.

Testing (Non-)Linearities

Observe that trace($\boldsymbol{S}$) is usually seen as a degree of smoothness.

Do we have to smooth? Isn't linear model sufficent?

Define

$$T = \frac{\|\boldsymbol{S}\boldsymbol{y} - \boldsymbol{H}\boldsymbol{y}\|}{\text{trace}([\boldsymbol{S}-\boldsymbol{H}]^\mathsf{T}[\boldsymbol{S}-\boldsymbol{H}])}$$

If the model is linear, then T has a Fisher distribution.

Remark: In the case of a linear predictor, with smoothing matrix $\boldsymbol{S}_h$

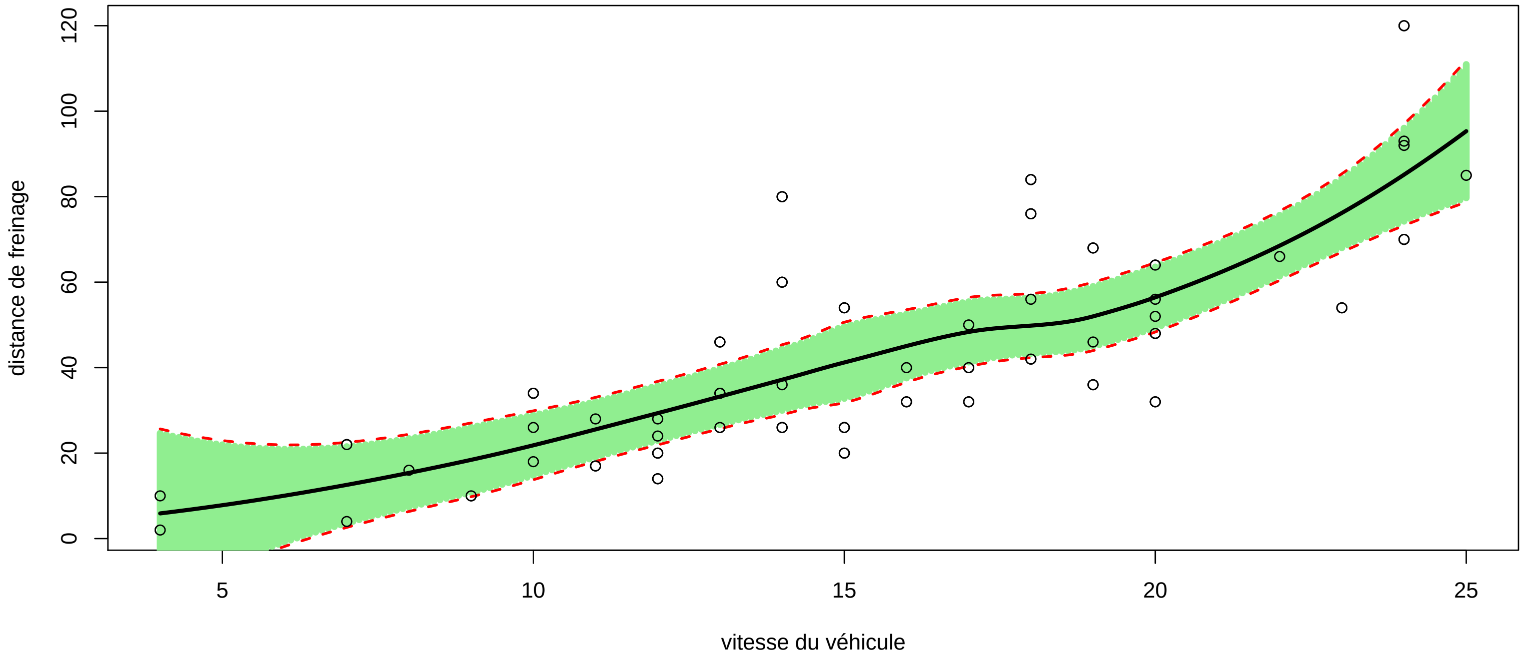
$$\widehat{R}(h) = \frac{1}{n}\sum_{i=1}^n (y_i - \widehat{m}_h^{(-i)}(\boldsymbol{x}_i))^2 = \frac{1}{n}\sum_{i=1}^n \left(\frac{Y_i - \widehat{m}_h(\boldsymbol{x}_i)}{1-[\boldsymbol{S}_h]_{i,i}}\right)^2$$

We do not need to estimate n models. One can also minimize

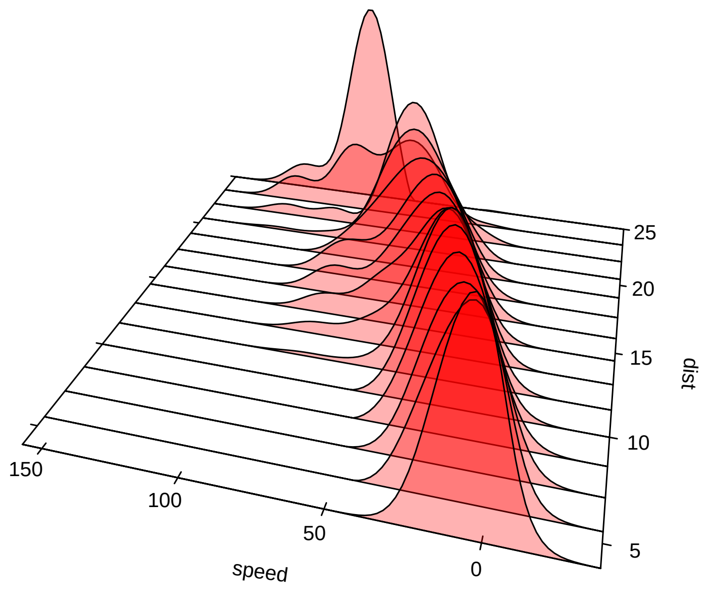
$$GCV(h) = \frac{n^2}{n^2 - \text{trace}(\boldsymbol{S})^2}\cdot\frac{1}{n}\sum_{i=1}^n (Y_i - \widehat{m}_h(\boldsymbol{x}_i))^2 \sim \text{Mallow's } C_p$$

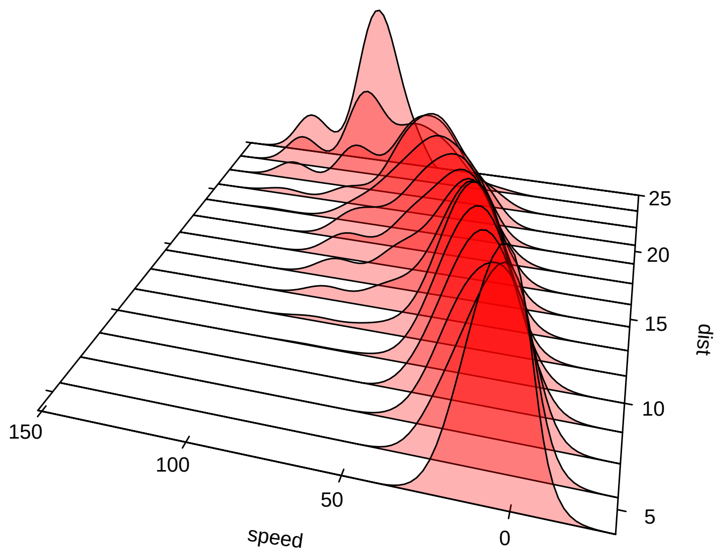
Confidence Intervals

If $\widehat{y} = \widehat{m}_h(\boldsymbol{x}) = S_h(\boldsymbol{x})\boldsymbol{y}$, let $\widehat{\sigma}^2 = \dfrac{1}{n}\sum_{i=1}^{n}(y_i - \widehat{m}_h(\boldsymbol{x}_i))^2$ and a confidence interval is, at $\boldsymbol{x}$ $\left[\widehat{m}_h(\boldsymbol{y}) \pm t_{1-\alpha/2}\widehat{\sigma}\sqrt{S_h(\boldsymbol{x})S_h(\boldsymbol{x})^\mathsf{T}}\right]$.



Confidence Bands





Confidence Bands

Also called variability bands for functions in Härdle (1990) Applied Nonparametric Regression.

From Collomb (1979) Condition nécessaires et suffisantes de convergence uniforme d'un estimateur de la régression, with Kernel regression (Nadarayah-Watson)

$$\sup\left\{|m(x)-\widehat{m}_h(x)|\right\}\sim C_1h^2+C_2\sqrt{\frac{\log n}{nh}}$$

$$\sup\left\{|m(\boldsymbol{x})-\widehat{m}_h(\boldsymbol{x})|\right\}\sim C_1h^2+C_2\sqrt{\frac{\log n}{nh^{\dim(\boldsymbol{x})}}}$$

Confidence Bands

So far, we have mainly discussed pointwise convergence with

$$\sqrt{nh}\,(\widehat{m}_h(x) - m(x)) \xrightarrow{\mathcal{L}} \mathcal{N}(\mu_x, \sigma_x^2).$$

This asymptotic normality can be used to derive (pointwise) confidence intervals

$$\mathbb{P}(IC^-(x) \leq m(x) \leq IC^+(x)) = 1 - \alpha \ \forall x \in \mathcal{X}.$$

But we can also seek uniform convergence properties. We want to derive functions $IC^{\pm}$ such that

$$\mathbb{P}(IC^-(x) \leq m(x) \leq IC^+(x) \ \forall x \in \mathcal{X}) = 1 - \alpha.$$

Confidence Bands

- Bonferroni's correction

Use a standard Gaussian (pointwise) confidence interval

$$IC_\star^{\pm}(x) = \widehat{m}(x) \pm \sqrt{nh}\widehat{\sigma}t_{1-\alpha/2}.$$

and take also into accound the regularity of m. Set

$$V(\eta) = \frac{1}{2}\left(\frac{2\eta+1}{n} + \frac{1}{n}\right)\|m'\|_{\infty,x}, \text{ for some } 0 < \eta < 1$$

where $\|\varphi'\|_{\infty,x}$ is on a neighborhood of x. Then consider

$$IC^{\pm}(x) = IC_\star^{\pm}(x) \pm V(\eta).$$

Confidence Bands

- Use of Gaussian processes

Observe that $\sqrt{nh}\,(\widehat{m}_h(x) - m(x)) \xrightarrow{\mathcal{D}} G_x$ for some Gaussian process (G_x).
Confidence bands are derived from quantiles of $\sup\{G_x, x \in \mathcal{X}\}$.

If we use kernel k for smoothing, Johnston (1982) Probabilities of Maximal Deviations for Nonparametric Regression Function Estimates proved that

$$G_x = \int k(x-t)dW_t, \text{ for some standard } (W_t) \text{ Wiener process}$$

is then a Gaussian process with variance $\int k(x)k(t-x)dt$. And

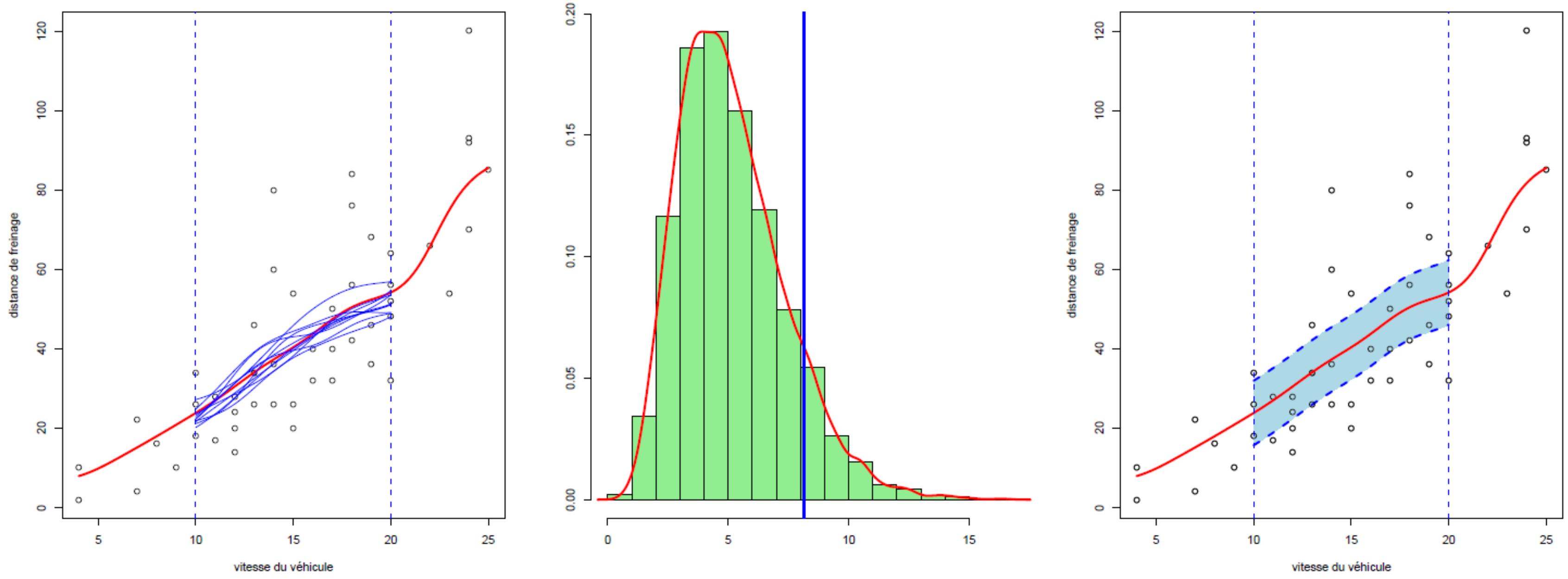
$$IC^{\pm}(x) = \widehat{\varphi}(x) \pm \left(\frac{q_\alpha}{\sqrt{2\log(1/h)}} + d_n\right)\frac{5}{7}\frac{\widehat{\sigma}^2}{\sqrt{nh}}$$

with $d_n = \sqrt{2\log h^{-1}} + \dfrac{1}{\sqrt{2\log h^{-1}}}\log\sqrt{\dfrac{3}{4\pi^2}}$, where $\exp(-2\exp(-q_\alpha)) = 1-\alpha$.

Confidence Bands

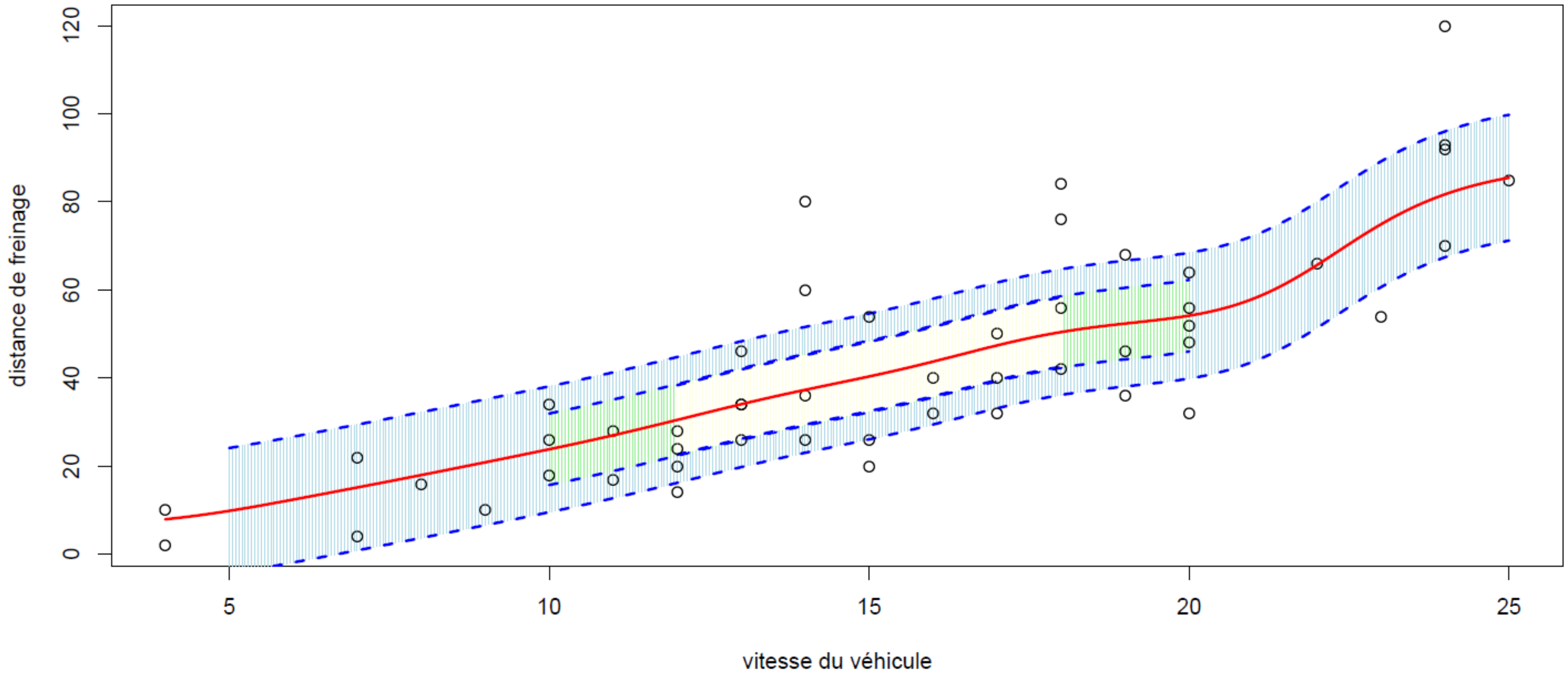
- Bootstrap (see #2)

Finally, McDonald (1986) Smoothing with Split Linear Fits suggested a bootstrap algorithm to approximate the distribution of $Z_n = \sup\{|\widehat{\varphi}(x) - \varphi(x)|, x \in \mathcal{X}\}$.



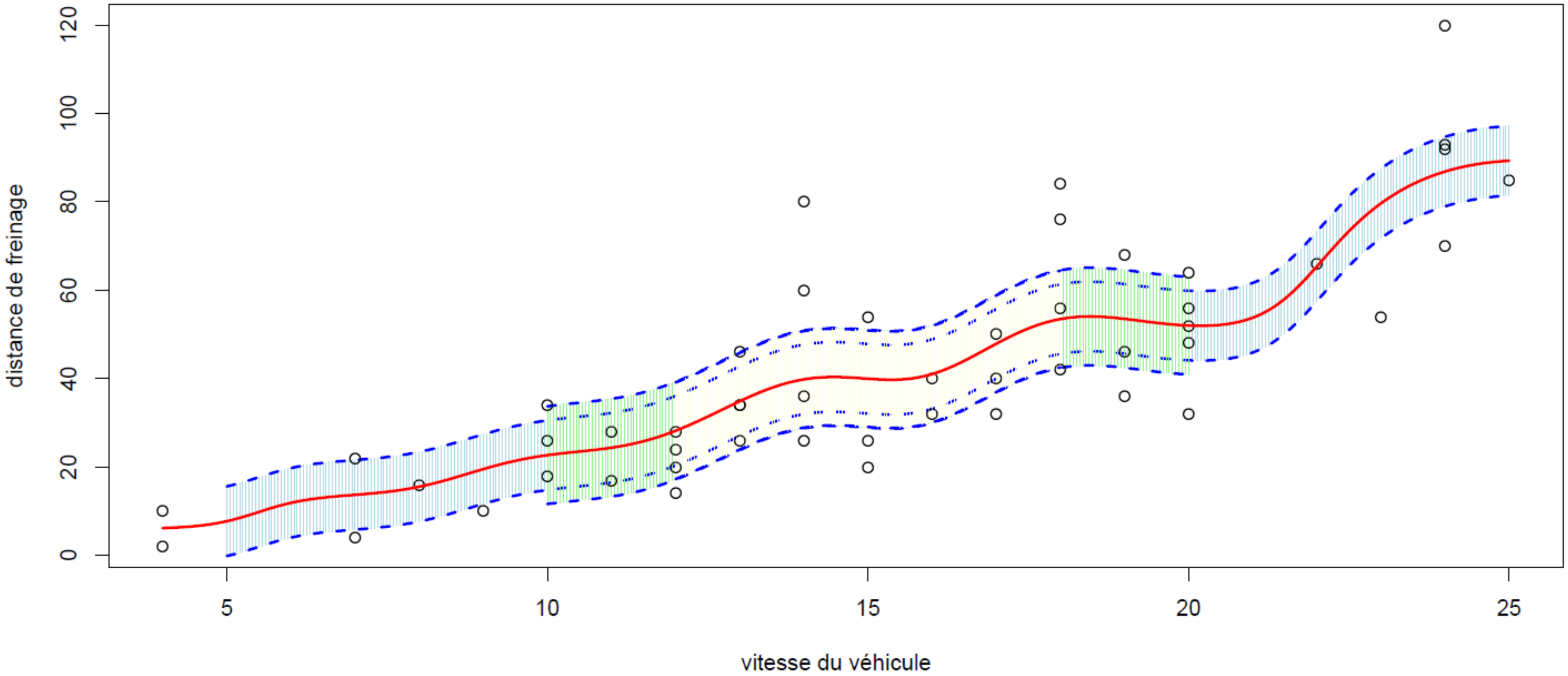
Confidence Bands

Depending on the smoothing parameter h, we get different corrections



Confidence Bands

Depending on the smoothing parameter h, we get different corrections



Boosting to Capture NonLinear Effects

We want to solve

$$m^\star = \text{argmin}\left\{\mathbb{E}\big[(Y - m(\boldsymbol{X}))^2\big]\right\}$$

The heuristics is simple: we consider an iterative process where we keep modeling the errors.

Fit model for $\boldsymbol{y}$, $h_1(\cdot)$ from $\boldsymbol{y}$ and $\boldsymbol{X}$, and compute the error, $\boldsymbol{\varepsilon}_1 = \boldsymbol{y} - h_1(\boldsymbol{X})$.

Fit model for $\boldsymbol{\varepsilon}_1$, $h_2(\cdot)$ from $\boldsymbol{\varepsilon}_1$ and $\boldsymbol{X}$, and compute the error, $\boldsymbol{\varepsilon}_2 = \boldsymbol{\varepsilon}_1 - h_2(\boldsymbol{X})$, etc. Then set

$$m_k(\cdot) = \underbrace{h_1(\cdot)}_{\sim y} + \underbrace{h_2(\cdot)}_{\sim \varepsilon_1} + \underbrace{h_3(\cdot)}_{\sim \varepsilon_2} + \cdots + \underbrace{h_k(\cdot)}_{\sim \varepsilon_{k-1}}$$

Hence, we consider an iterative procedure, $m_k(\cdot) = m_{k-1}(\cdot) + h_k(\cdot)$.

Boosting

$h(\boldsymbol{x}) = \boldsymbol{y} - m_k(\boldsymbol{x})$, which can be interpreted as a residual. Note that this residual is the gradient of $\dfrac{1}{2}[y - m_k(\boldsymbol{x})]^2$

A gradient descent is based on Taylor expansion

$$\underbrace{f(\boldsymbol{x}_k)}_{\langle f,\boldsymbol{x}_k\rangle} \sim \underbrace{f(\boldsymbol{x}_{k-1})}_{\langle f,\boldsymbol{x}_{k-1}\rangle} + \underbrace{(\boldsymbol{x}_k - \boldsymbol{x}_{k-1})}_{\alpha} \underbrace{\nabla f(\boldsymbol{x}_{k-1})}_{\langle \nabla f,\boldsymbol{x}_{k-1}\rangle}$$

But here, it is different. We claim we can write

$$\underbrace{f_k(\boldsymbol{x})}_{\langle f_k,\boldsymbol{x}\rangle} \sim \underbrace{f_{k-1}(\boldsymbol{x})}_{\langle f_{k-1},\boldsymbol{x}\rangle} + \underbrace{(f_k - f_{k-1})}_{\beta} \underbrace{?}_{\langle f_{k-1},\nabla \boldsymbol{x}\rangle}$$

where ? is interpreted as a 'gradient'.

Boosting

Construct iteratively

$$m_k(\cdot) = m_{k-1}(\cdot) + \underset{h\in\mathcal{H}}{\text{argmin}}\left\{\sum_{i=1}^n (y_i - [m_{k-1}(\boldsymbol{x}_i) + h(\boldsymbol{x}_i)])^2\right\}$$

$$m_k(\cdot) = m_{k-1}(\cdot) + \underset{h\in\mathcal{H}}{\text{argmin}}\left\{\sum_{i=1}^n ([y_i - m_{k-1}(\boldsymbol{x}_i)] - h(\boldsymbol{x}_i)])^2\right\}$$

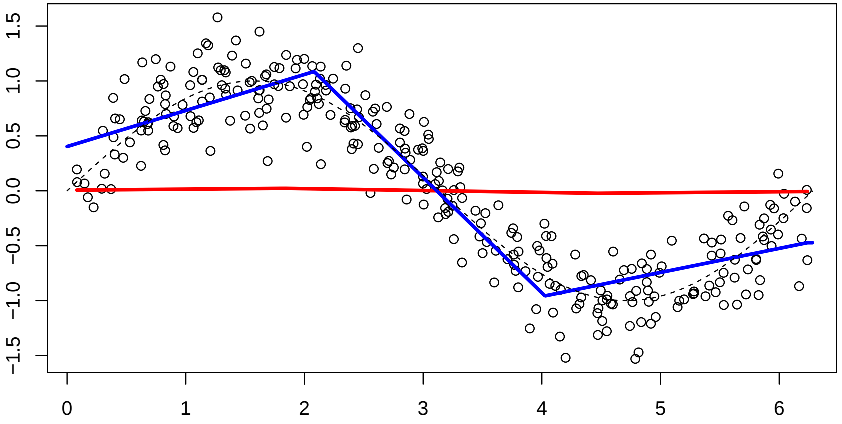
where $h \in \mathcal{H}$ means that we seek in a class of weak learner functions.

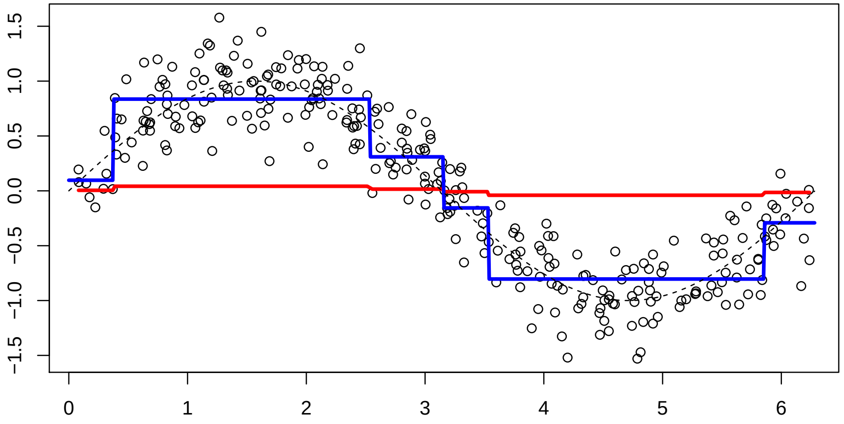
If learner are two strong, the first loop leads to some fixed point, and there is no learning procedure, see linear regression $y = \boldsymbol{x}^\mathsf{T}\boldsymbol{\beta} + \varepsilon$. Since $\varepsilon \perp \boldsymbol{x}$ we cannot learn from the residuals.

In order to make sure that we learn weakly, we can use some shrinkage parameter ν (or collection of parameters ν_j).

Boosting with Piecewise Linear Spline & Stump Functions

Instead of $\boldsymbol{\varepsilon}_k = \boldsymbol{\varepsilon}_{k-1} - h_k(\boldsymbol{x})$, set $\boldsymbol{\varepsilon}_k = \boldsymbol{\varepsilon}_{k-1} - \nu \cdot h_k(\boldsymbol{x})$





Remark : bumps are related to regression trees (see 2015 course).

Ruptures

One can use Chow test to test for a rupture. Note that it is simply Fisher test, with two parts,

$$\boldsymbol{\beta} = \begin{cases} \boldsymbol{\beta}_1 \text{ for } i = 1, \cdots, i_0 \\ \boldsymbol{\beta}_2 \text{ for } i = i_0 + 1, \cdots, n \end{cases} \quad \text{and test} \quad \begin{cases} H_0 : \boldsymbol{\beta}_1 = \boldsymbol{\beta}_2 \\ H_1 : \boldsymbol{\beta}_1 \neq \boldsymbol{\beta}_2 \end{cases}$$

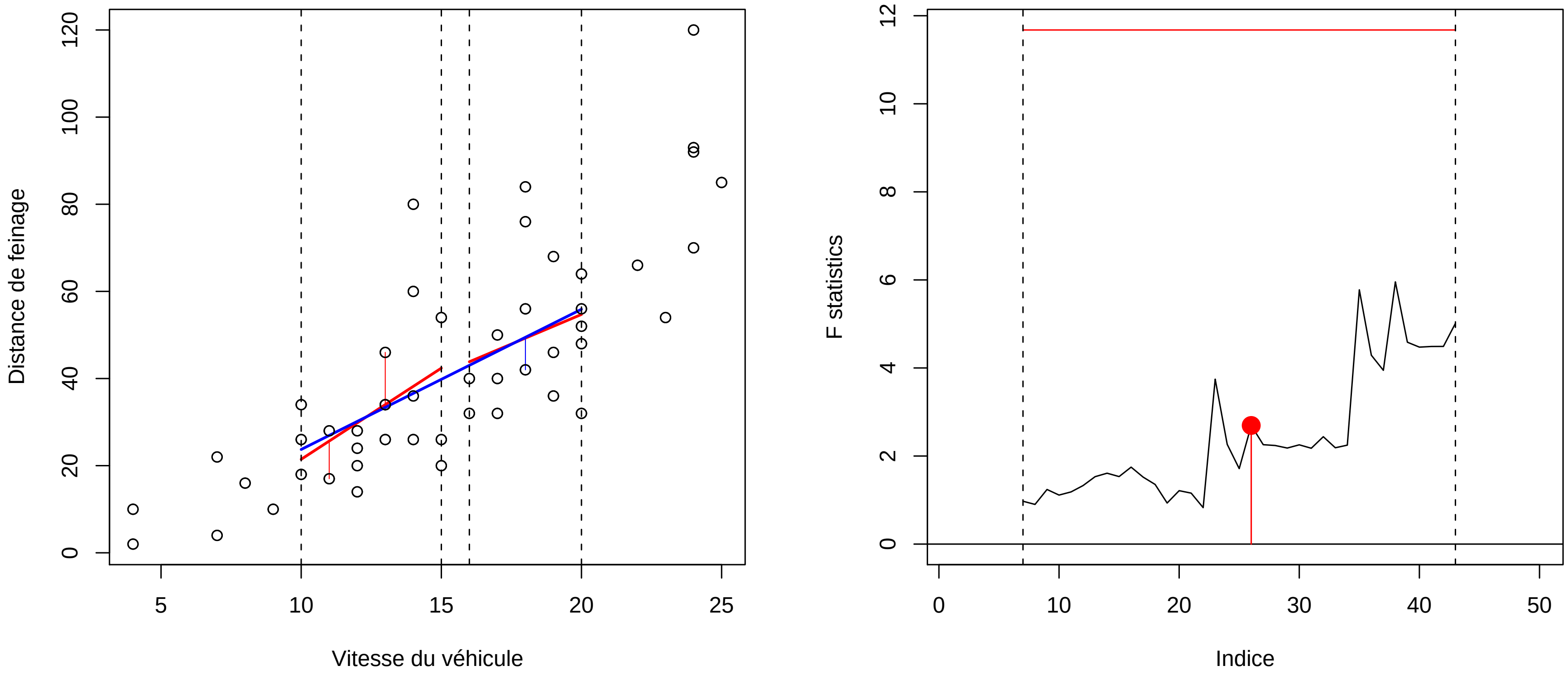
i_0 is a point between k and $n - k$ (we need enough observations). Chow (1960) Tests of Equality Between Sets of Coefficients in Two Linear Regressions suggested

$$F_{i_0} = \frac{\widehat{\boldsymbol{\eta}}^\mathsf{T}\widehat{\boldsymbol{\eta}} - \widehat{\boldsymbol{\varepsilon}}^\mathsf{T}\widehat{\boldsymbol{\varepsilon}}}{\widehat{\boldsymbol{\varepsilon}}^\mathsf{T}\widehat{\boldsymbol{\varepsilon}}/(n - 2k)}$$

where $\widehat{\varepsilon}_i = y_i - \boldsymbol{x}_i^\mathsf{T}\widehat{\boldsymbol{\beta}}$, and $\widehat{\eta}_i = \begin{cases} Y_i - \boldsymbol{x}_i^\mathsf{T}\widehat{\boldsymbol{\beta}}_1 \text{ for } i = k, \cdots, i_0 \\ Y_i - \boldsymbol{x}_i^\mathsf{T}\widehat{\boldsymbol{\beta}}_2 \text{ for } i = i_0 + 1, \cdots, n - k \end{cases}$

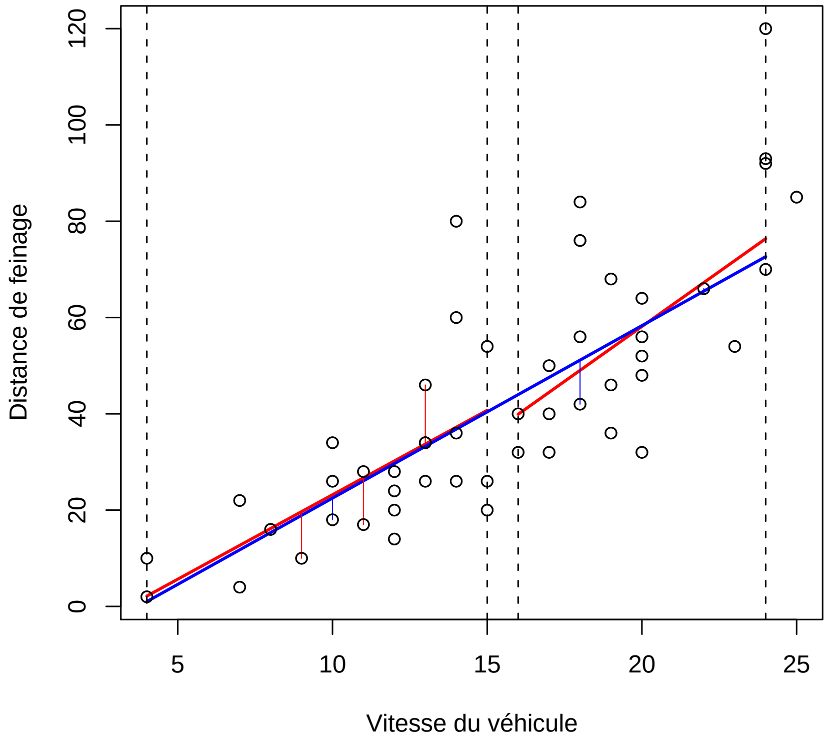
Ruptures

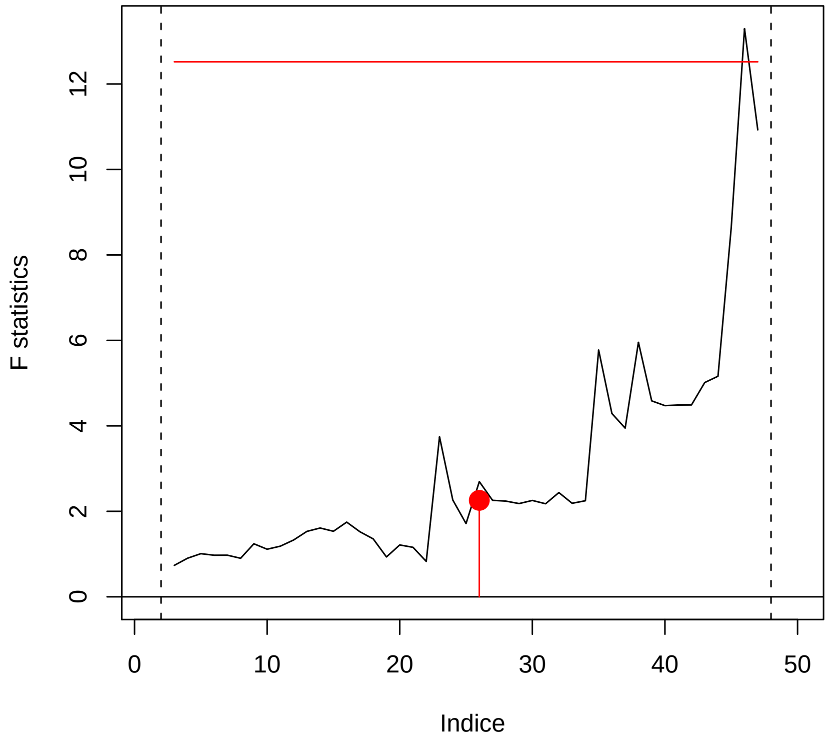
```
> Fstats(dist ~ speed,data=cars,from=7/50)
```



Ruptures

```
> Fstats(dist ~ speed,data=cars,from=2/50)
```





Ruptures

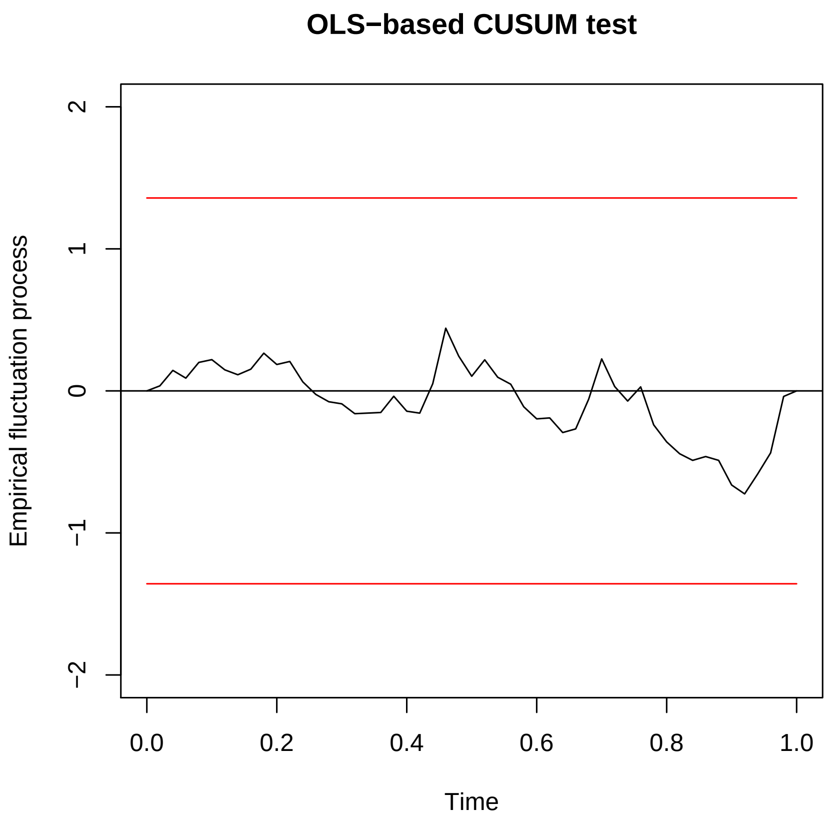
If i_0 is unknown, use CUSUM types of tests, see Ploberger & Krämer (1992) The Cusum Test with OLS Residuals. For all $t \in [0,1]$, set

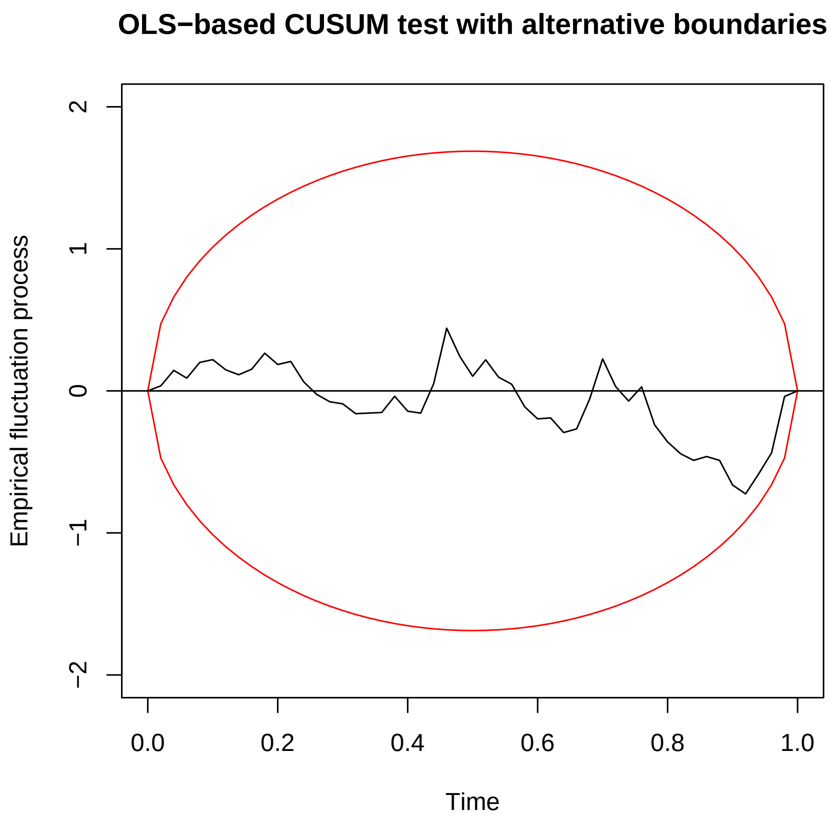
$$W_t = \frac{1}{\widehat{\sigma}\sqrt{n}} \sum_{i=1}^{\lfloor nt \rfloor} \widehat{\varepsilon}_i.$$

If α is the confidence level, bounds are generally $\pm\alpha$, even if theoretical bounds should be $\pm\alpha\sqrt{t(1-t)}$.

```
> cusum <- efp(dist ~ speed, type = "OLS-CUSUM",data=cars)
> plot(cusum,ylim=c(-2,2))
> plot(cusum, alpha = 0.05, alt.boundary = TRUE,ylim=c(-2,2))
```

Ruptures


OLS−based CUSUM test
Empirical fluctuation process
Time
−2
−1
0
1
2
0.0
0.2
0.4
0.6
0.8
1.0


OLS−based CUSUM test with alternative boundaries
Empirical fluctuation process
Time
−2
−1
0
1
2
0.0
0.2
0.4
0.6
0.8
1.0

From a Rupture to a Discontinuity

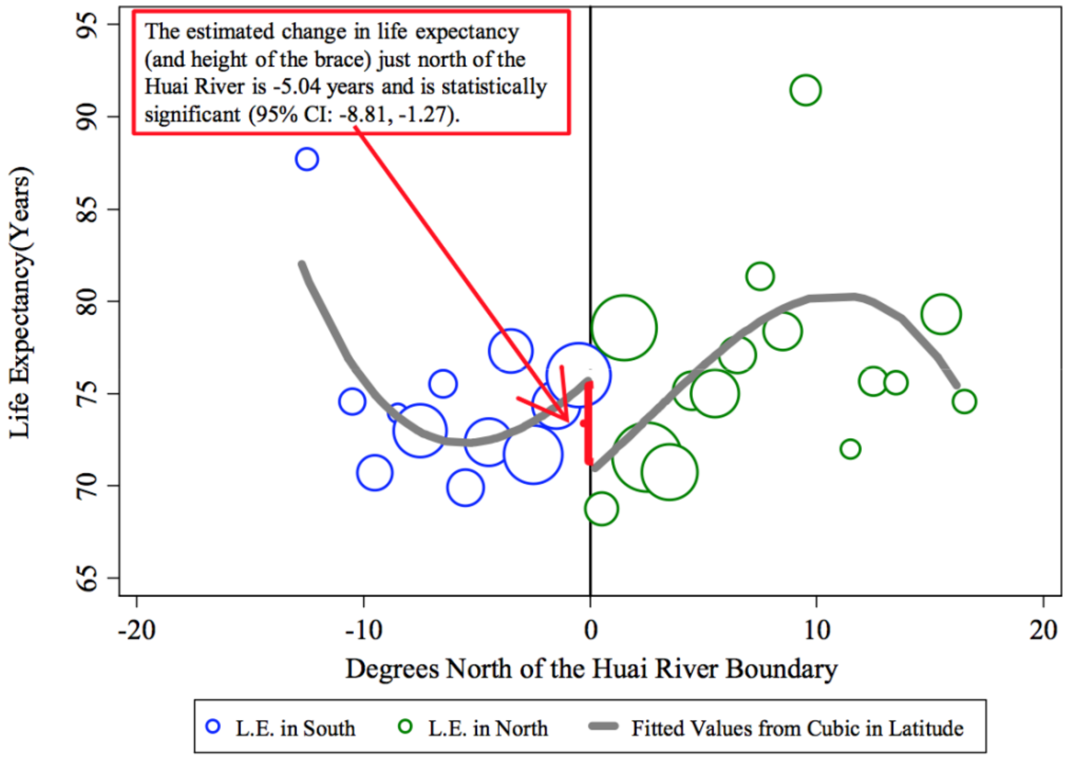


Fig. 3. The plotted line reports the fitted values from a regression of life expectancy on a cubic in latitude using the sample of DSP locations, weighted by the population at each location.

See Imbens & Lemieux (2008) Regression Discontinuity Designs.

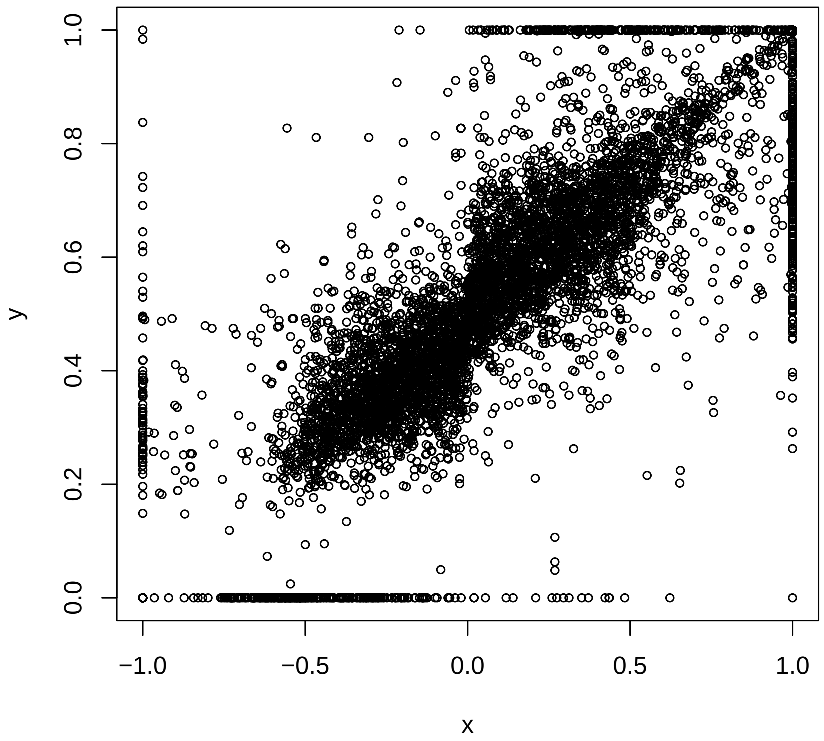
From a Rupture to a Discontinuity

Consider the dataset from Lee (2008) Randomized experiments from non-random selection in U.S. House elections.

```
> library(RDDtools)
> data(Lee2008)
```

We want to test if there is a discontinuity in 0.

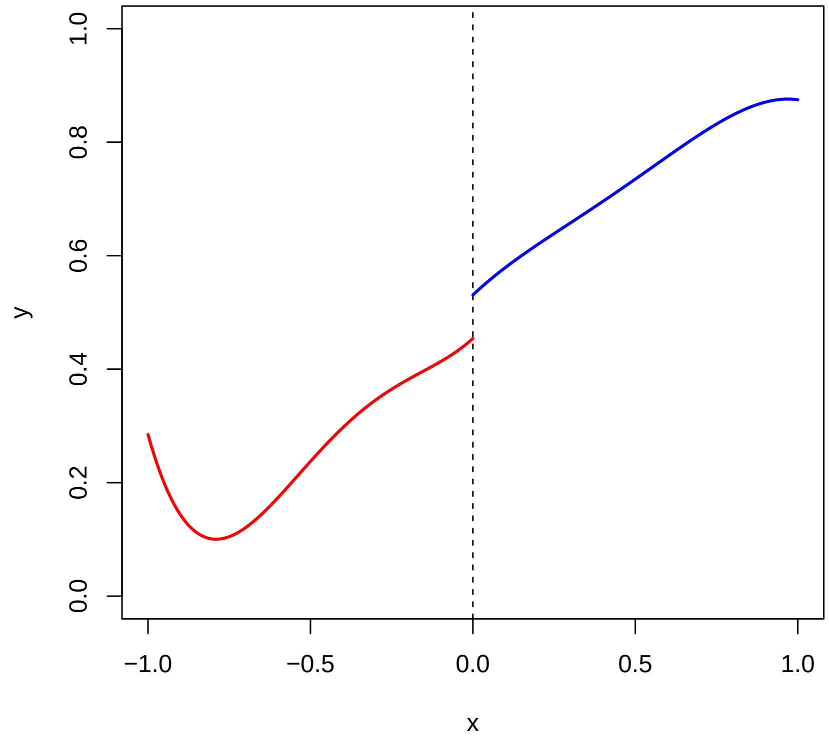
- with parametric tools
- with nonparametric tools



Testing for a rupture

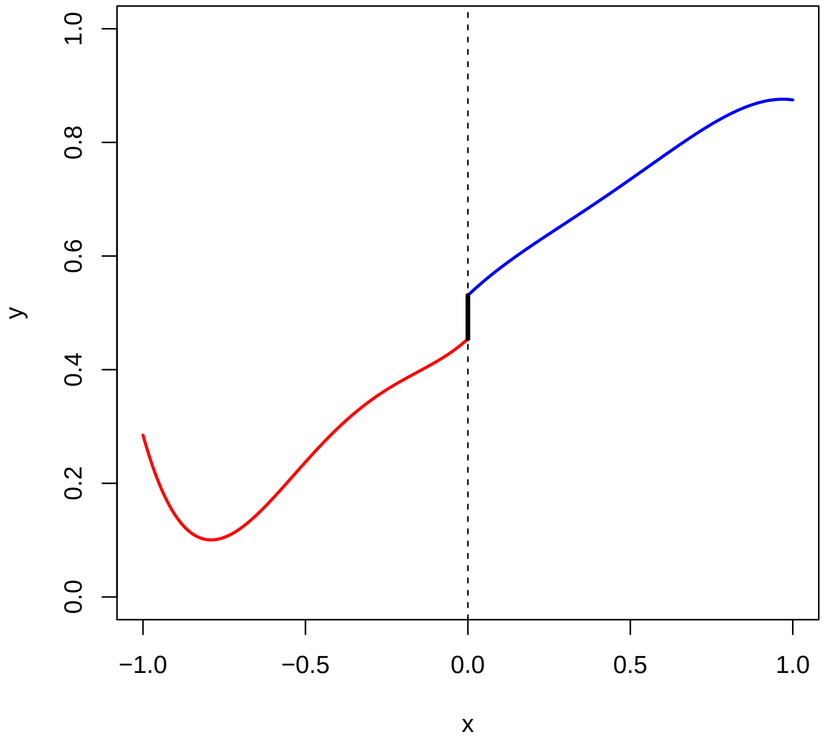
Use some 4th order polynomial, on each part

```
> idx1 = (Lee2008$x>0)
> reg1 = lm(y~poly(x,4),data=Lee2008[
    idx1,])
> idx2 = (Lee2008$x<0)
> reg2 = lm(y~poly(x,4),data=Lee2008[
    idx2,])
> s1=predict(reg1,newdata=data.frame(x
    =0))
> s2=predict(reg2,newdata=data.frame(x
    =0))
> abs(s1-s2)
         1
0.07659014
```



Testing for a rupture

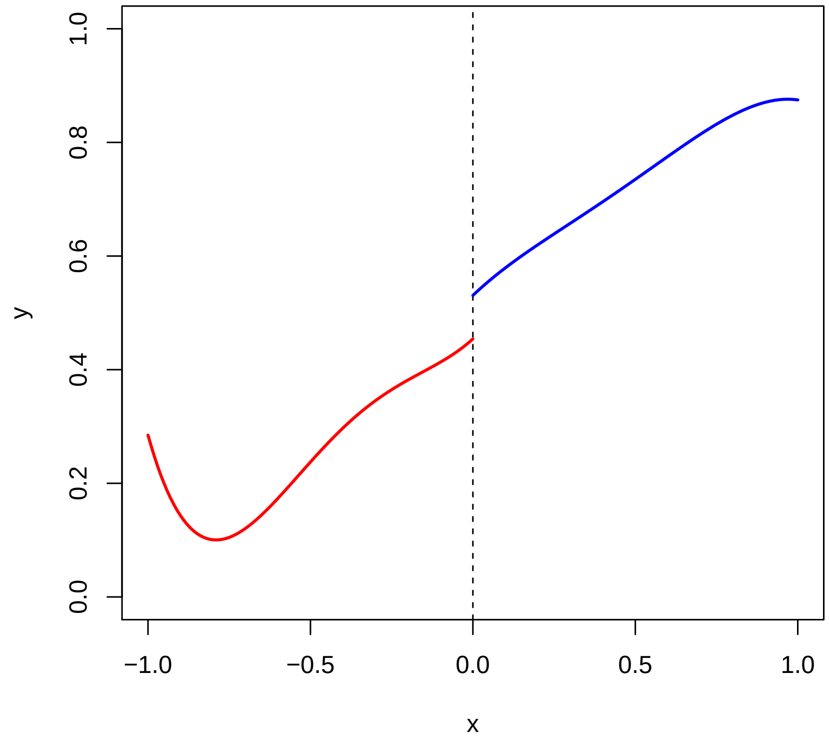
```
> reg_para <- RDDreg_lm(RDDdata(y =
    Lee2008$y, x = Lee2008$x, cutpoint
    = 0), order = 4)
> reg_para
### RDD regression: parametric ###
  Polynomial order:  4
  Slopes:  separate
  Number of obs: 6558 (left: 2740,
    right: 3818)

  Coefficient:
  Estimate Std. Error t value  Pr(>|t|)
D 0.076590   0.013239  5.7851 7.582e-09
     ***
```



Testing for a rupture

or use a simple local regression, see Imbens & Kalyanaraman (2012).

```
> reg1 = ksmooth(Lee2008$x[idx1],
    Lee2008$y[idx1], kernel = "normal",
     bandwidth = 0.1)
> reg2 = ksmooth(Lee2008$x[idx2],
    Lee2008$y[idx2], kernel = "normal",
     bandwidth = 0.1)
> s1 = reg1$y[1]
> s2 = reg2$y[length(reg2$y)]
> abs(s1-s2)
[1] 0.09883813
```



Testing for a rupture

```
> reg_nonpara <- RDDreg_np(RDDobject = Lee2008_rdd, bw = .1)
> print(reg_nonpara)
### RDD regression: nonparametric local linear
  Bandwidth:  0.1
  Number of obs: 1209 (left: 577, right: 632)

  Coefficient:
  Estimate Std. Error z value  Pr(>|z|)
D 0.059397   0.014119   4.207 2.588e-05 ***
```