Nonparametric Smoothing

Postgraduate Training Week

Jochen Einbeck

Department of Mathematical Sciences, Durham University

jochen.einbeck@durham.ac.uk

Durham, 10th th to 14th May 2010

Durham University

"Smoothing"

According to TheFreeDictionary, "smoothing" can have the following meanings:

1. To make (something) even, level, or unwrinkled.
2. To rid of obstructions, hindrances, or difficulties.
3. To soothe or tranquilize; make calm.
4. To make less harsh or crude; refine.

"Smoothing"

According to TheFreeDictionary, "smoothing" can have the following meanings:

1. To make (something) even, level, or unwrinkled.
2. To rid of obstructions, hindrances, or difficulties.
3. To soothe or tranquilize; make calm.
4. To make less harsh or crude; refine.

in short....

Example 1

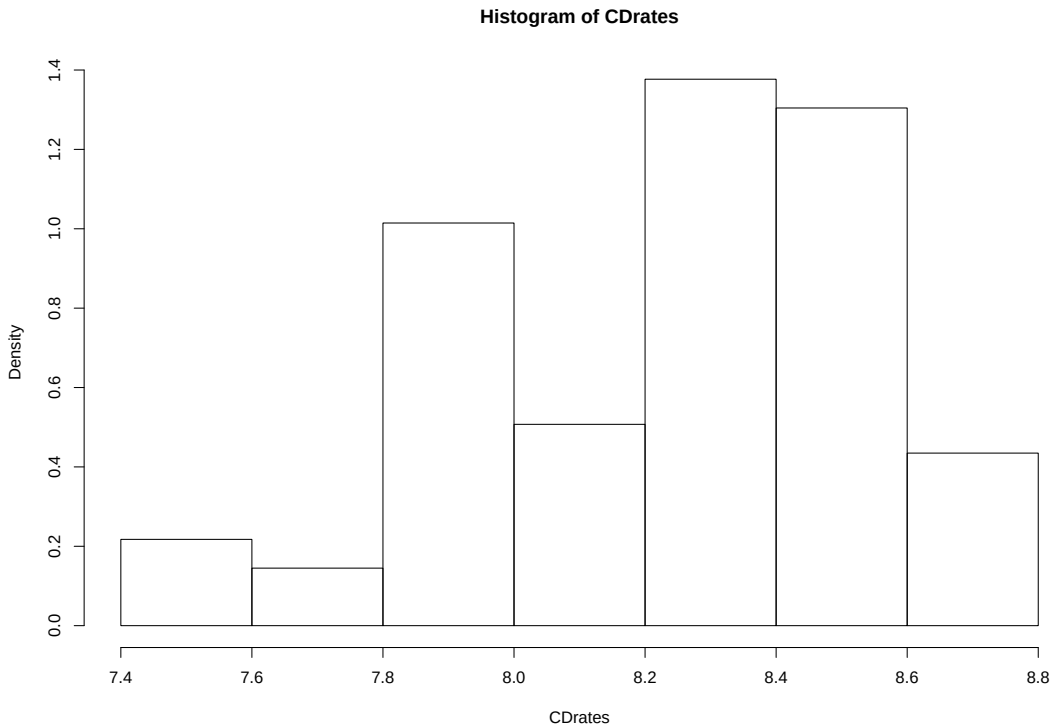
"Certificate of deposit" (CD) rates for 69 Long Island banks [S]

```
7.56 7.57 7.71 7.82 7.82 7.90 8.00 8.00 8.00 8.00
8.00 8.00 8.00 8.05 8.05 8.06 8.11 8.17 8.30 8.33
8.33 8.40 8.50 8.51 8.55 8.57 8.65 8.65 8.71
7.51 7.75 7.90 8.00 8.00 8.00 8.15 8.20 8.25 8.25
8.30 8.30 8.33 8.33 8.34 8.35 8.36 8.40 8.40 8.40
8.40 8.40 8.40 8.45 8.49 8.49 8.49 8.50 8.50 8.50
8.50 8.50 8.50 8.50 8.50 8.50 8.52 8.70 8.75 8.78
```

How would you visualize these data?

Example 1 (cont.)

Straightforward idea: Histogram.

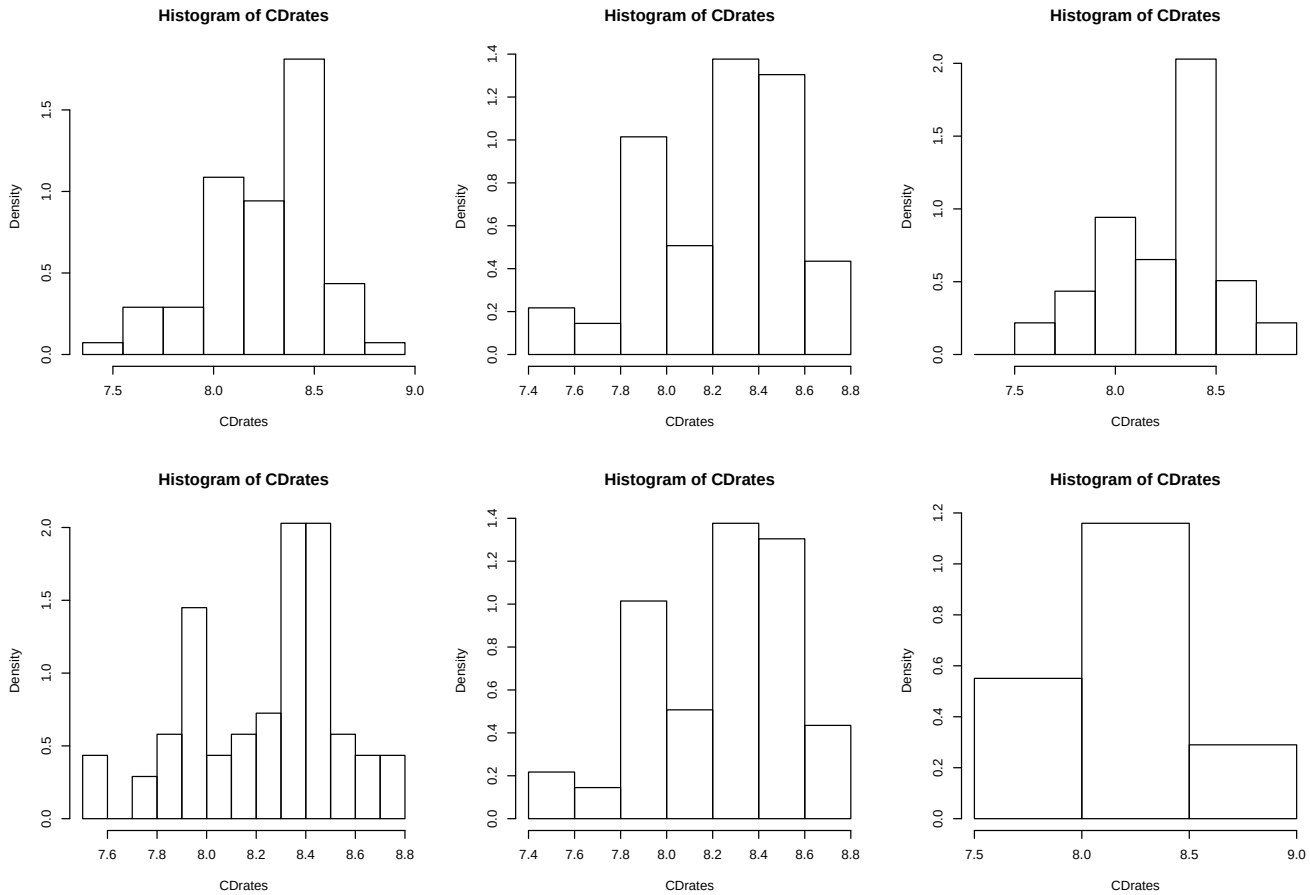



```
> hist(CDrates, freq=FALSE)
```

Does this adequately represent the distribution of the data?

Example 1 (cont.)

Vary anchor point (top) or number of bins (bottom):




```
> hist(CDrates, freq=FALSE, breaks=...)
```

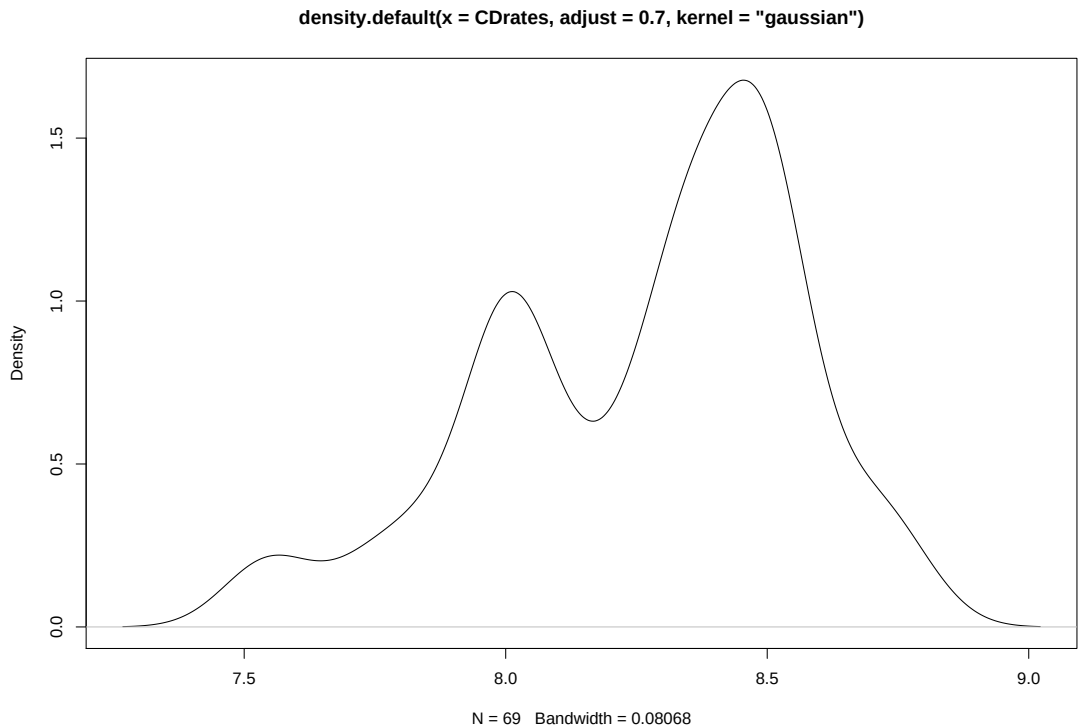
Example 1 (cont.)

Observations:

- Histogram shapes can be quite different for the same data and even for the same number of bins.
- Histograms are not "smooth" (but the true underlying density supposedly is!)
- If we try to make them "smoother" by reducing the number of bins, we discard important information contained in the data.
- Can one do that better?

Example 1 (cont.)

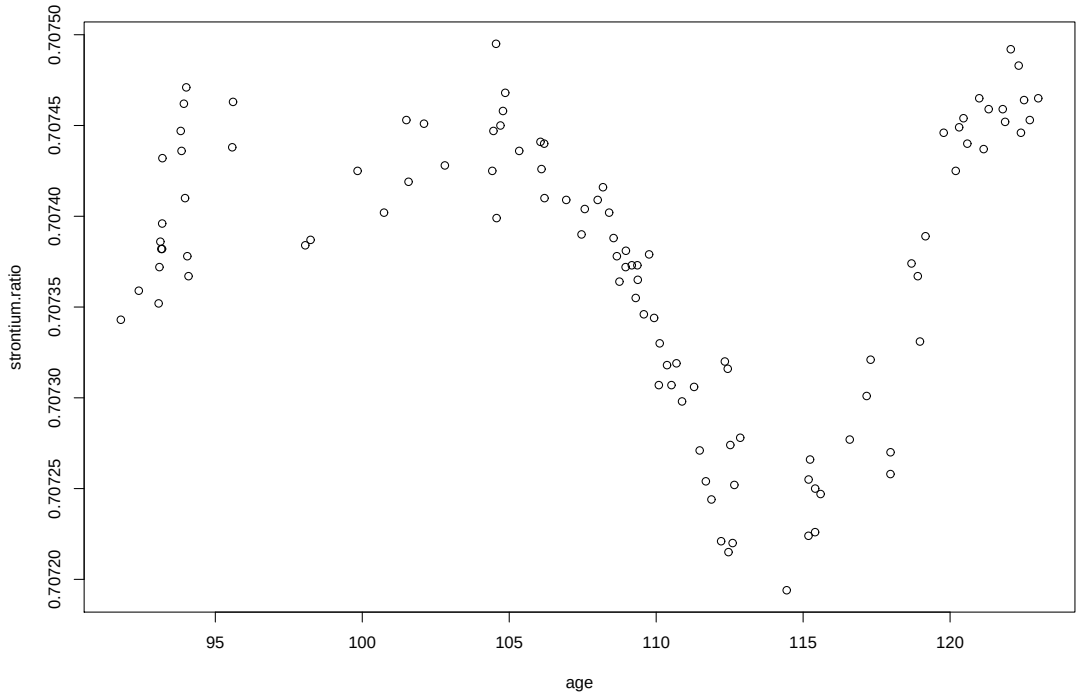
Kernel density estimation ("Kernel smoothing"):




```
> density(CDrates, kernel="gaussian", adjust=0.7)
```

Example 2

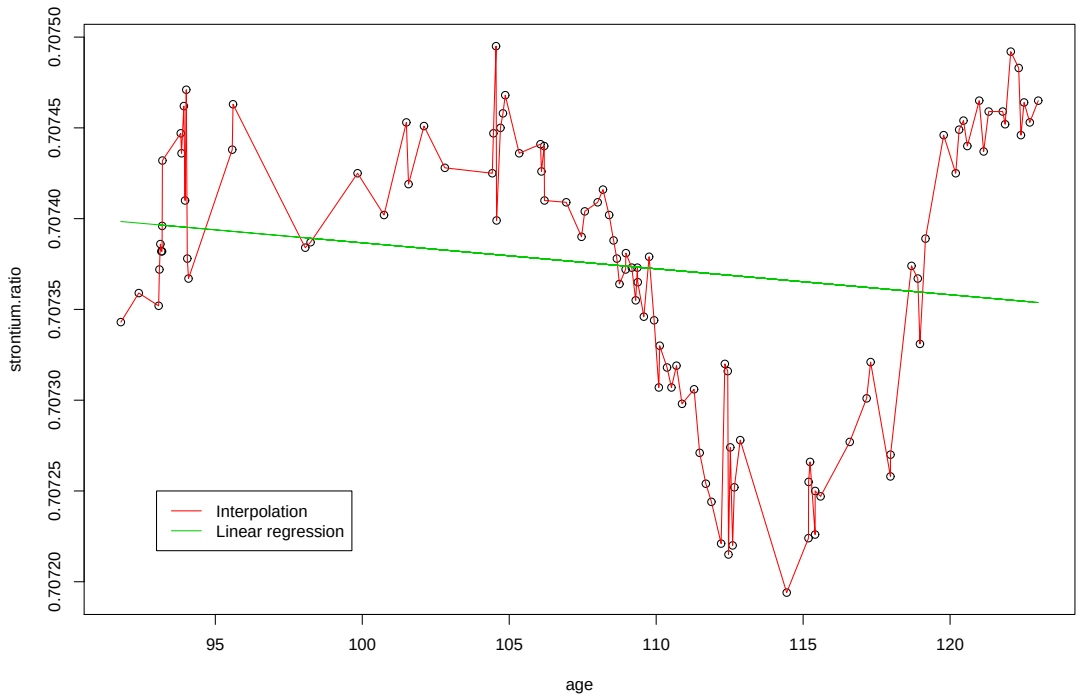
Strontium isotopes found in $n = 106$ fossil shells versus age [RWC].




```
> data(fossil, package="SemiPar"); attach(fossil)
> plot(age, strontium.ratio)
```

Example 2 (cont.)

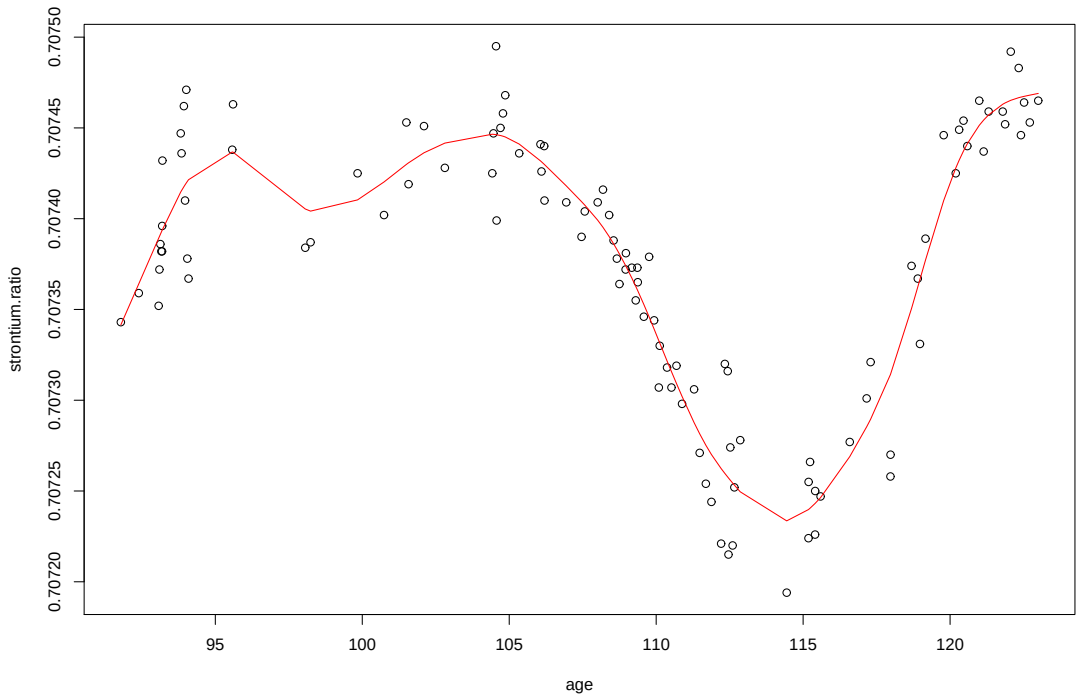
Interpolation (using grid of size 200) vs. Linear regression:



Actually, both are not satisfactory!

Example 2 (cont.)

Smooth nonparametric regression:

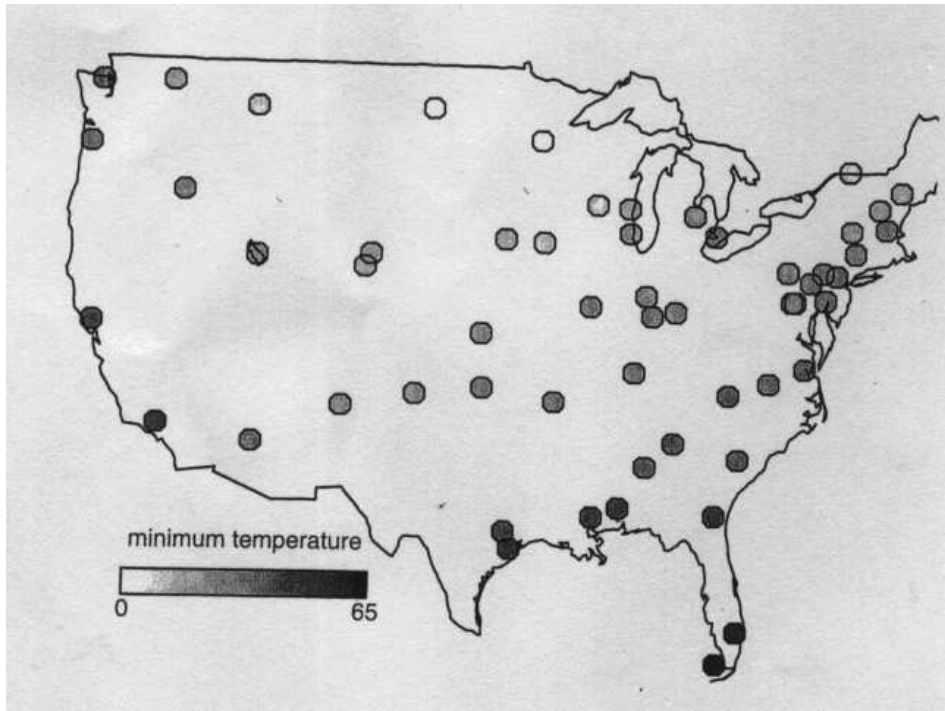



```
> library(pspline)
> fossil.spline <- sm.spline(age, strontium.ratio)
```

This fit is based on so-called *smoothing splines*.

Example 3

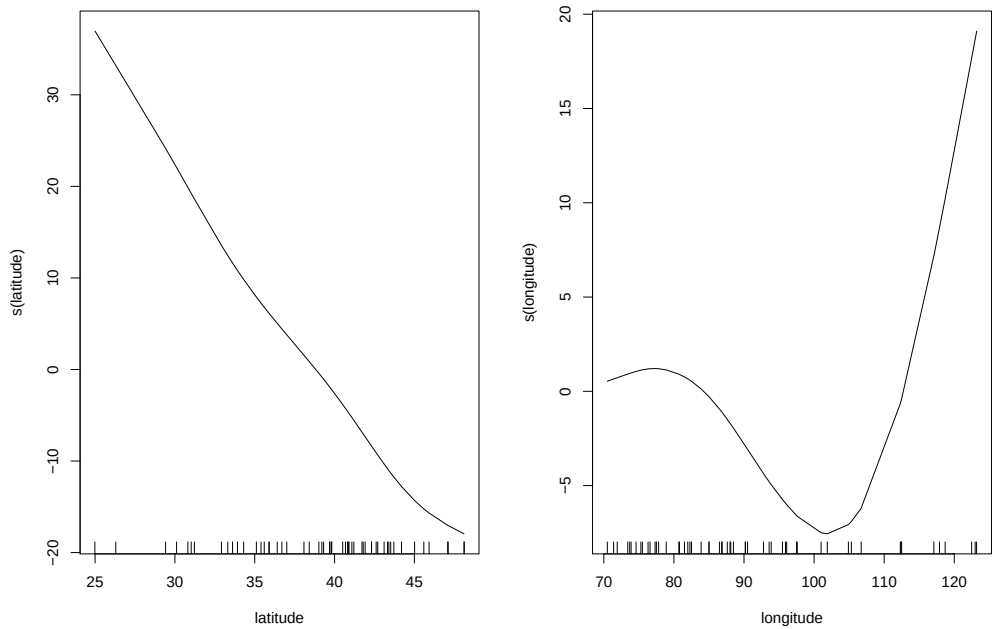
- Temperature data for 56 cities in the US [RWC]:



- Variables:
  - `min.temp` $\equiv$ average minimum January temperature
  - `latitude` $\equiv$ degrees latitude
  - `longitude`$\equiv$ negative degrees longitude

Example 3 (cont.)

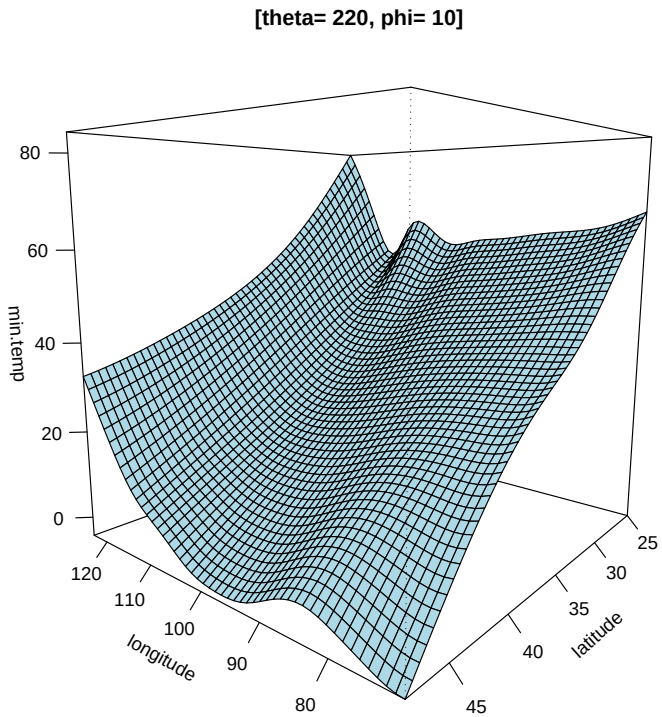
- Additive model:
  $\texttt{min.temp} = m_1(\texttt{latitude}) + m_2(\texttt{longitude})$
- is modelling a separate influence of `latitude` and `longitude` on `min.temp`:




```
> library(gam); data(ustemp, package="SemiPar"); attach(ustemp)
> gam.us <- gam(min.temp)~s(latitude)+s(longitude)); plot(gam.us)
```

Example 3 (cont.)

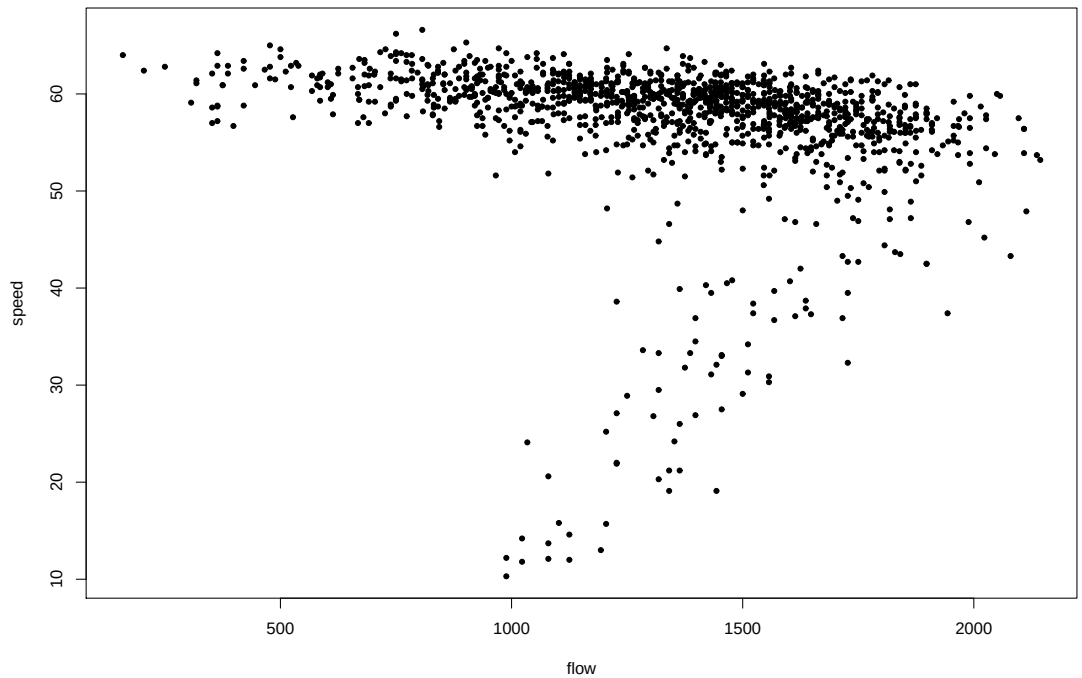
- Spatial model: `min.temp` $= m($`latitude, longitude`$)$
- is modelling an interacting influence of `latitude` and `longitude` on `min.temp`:




```
> library(np)
> us2dim <- npreg(min.temp~longitude+latitude, regtype= "ll",
bws=c(5,5))
> npplot(us2dim$bws, theta=220, view="fixed")
```

Example 4

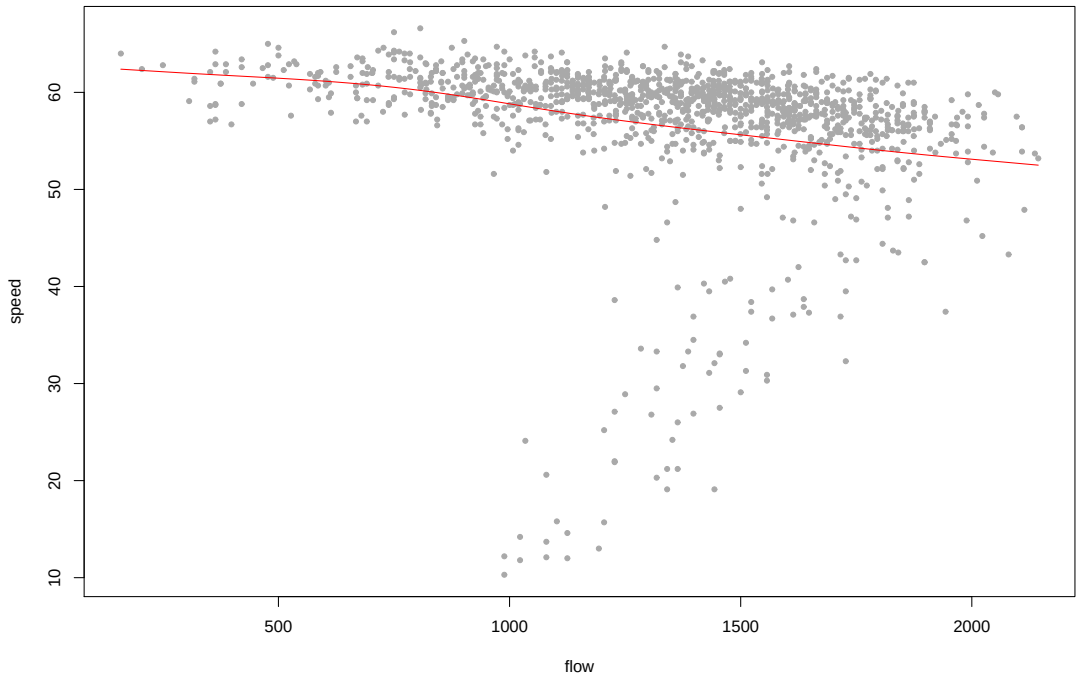
- Speed-Flow diagram recorded on a Californian Freeway.
- Each point corresponds to average flow and speed recorded at a certain location within a 30-seconds-interval.




```
> data(lane2, package="hdrcde"); plot(lane2)
```

Example 4 (cont.)

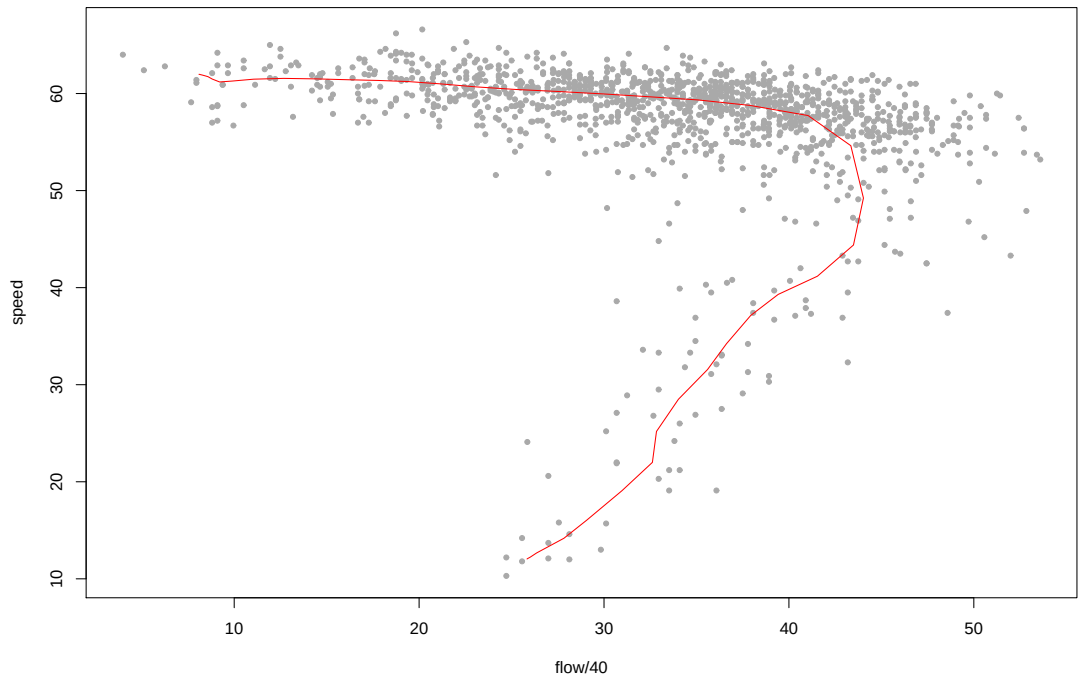
- Nonparametric regression?



- The fitted nonparametric regression curve estimates the *expected speed given flow*, but does it give the best *description* of the data set?

Example 4 (cont.)

- Alternative: principal curves, "smooth curves through the middle of the data cloud":



- Principal curves form the nonparametric analogue to principal components.

Summary

We understand Smoothing as a term for statistical methods including

1. Nonparametric (kernel) density estimation
2. Nonparametric regression, including
   - Univariate regression
   - Surface smoothing
   - Additive models
   - Semiparametric models
   - Spatial models
3. Principal curves (and the like)

The main emphasis of this course is on item 2.

Content

Roughly, the course is structured into four major blocks

- Kernel based methods
- Spline based methods
- Bayesian and partially Bayesian methods
- Principal curves

This categorization is quite artificial and arbitrary. For example, what is said in the "kernel based method" section on linear smoothers and bandwidth selection holds equally well for the splines. Further, the Bayesian methods that we investigate are actually a variant of spline based methods, too.

Literature

BOWMAN, A.W. and AZZALINI, A. (1997) [BA]: *Applied Smoothing Techniques for Data Analysis*, Oxford: Clarendon.

FAHRMEIR, L., and TUTZ, G. (2001) [FT]: *Multivariate Statistical Modelling based on Generalized Linear Models*, Springer.

FAN, J., and GIJBELS, I. (1996)[FG]: *Local Polynomial Modelling and its Applications*, London: Chapman & Hall/CRC.

GREEN, P.J. and SILVERMAN, B.W. (1994)[GS]: *Nonparametric Regression and Generalized linear models*, London: Chapman & Hall/CRC.

GORBAN, A., KÉGL, B., WUNSCH, D.C., and ZINOVYEV, A.(Eds.) (2008) [G]: *Principal Manifolds for Data Visualization and Dimension Reduction*, Springer.

Literature (cont.)

HASTIE, T., TIBSHIRANI, R. , and FRIEDMAN, R. (2001)[HTF]: *The Elements of Statistical Learning*, Springer.

LOADER, C. (1999)[L]: *Local Regression and Likelihood*, Springer.

RUPPERT, D., WAND, M.P., and CARROLL, R.J. (2003)[RWC]: *Semiparametric regression*, Camebridge University Press.

SILVERMAN, B.W. (1986)[Si]: *Density estimation*, London: Chapman & Hall.

SIMONOFF, J.S. (1994)[S]: *Smoothing Methods in Statistics*, Springer.

WAND, M.P., and JONES, M.C. (1995)[WJ]: *Kernel Smoothing*, London: Chapman & Hall/CRC.

Software

We will work with two publicly available and free software packages:

- **R**, available at

  `http://cran.r-project.org/`

- **BayesX**, available at

  `http://www.stat.uni-muenchen.de/~bayesx/bayesx.html`

Section 1: Kernel based methods

Scope of this section:

- Kernel functions and kernel density estimation;
- Kernel regression;
- Local linear and local polynomial regression;
- Derivative estimation;
- MSE, Bias, and Variance;
- Linear smoothers;
- Confidence and prediction bands;
- Smoothing parameter selection;
- Feature extraction;
- Robust smoothing;
- Bivariate smoothing and "the curse of dimensionality".

Density estimation

Consider the problem of estimating a density $f(\cdot)$ from data $x_1, \ldots, x_n$. Note that

$$f(x) = \frac{d}{dx} F(x) = \lim_{h \longrightarrow 0} \frac{F(x+h) - F(x-h)}{2h}, \tag{1}$$

where $F(\cdot)$ can be estimated by the empirical distribution function $\hat{F}(x) = \frac{\{\#x_i : x_i \leq x\}}{n}$. Plugging this into (1), one obtains for fixed h

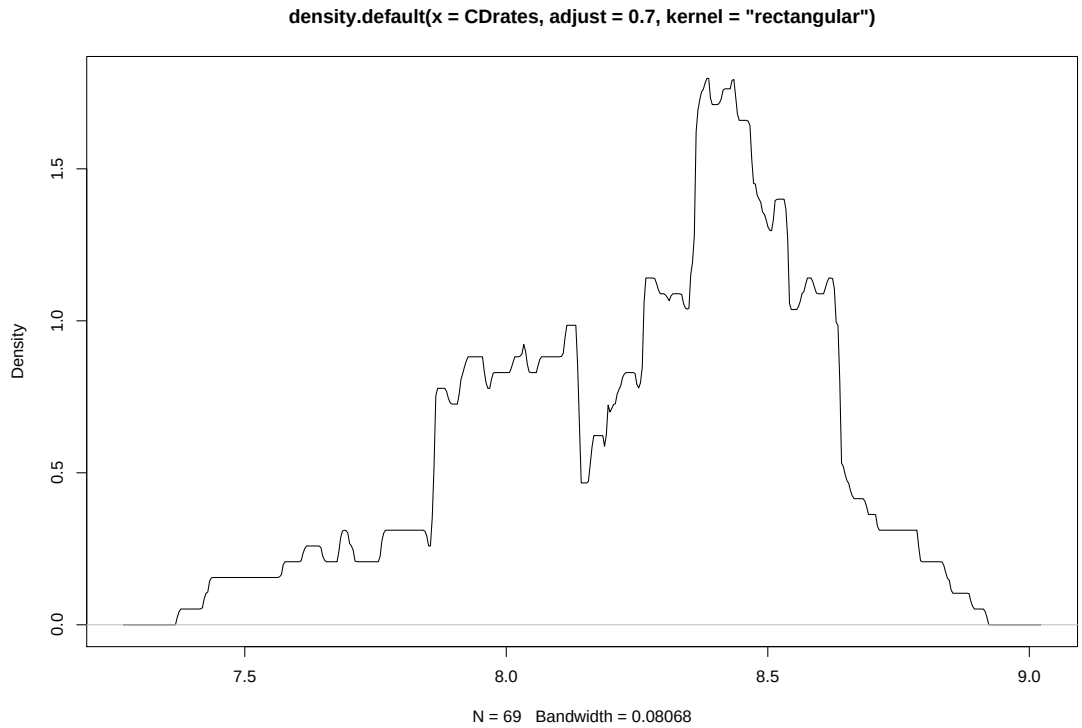
$$\begin{aligned} \hat{f}(x) &= \frac{1}{2h} \frac{\{\#x_i : x_i \in (x-h, x+h]\}}{n} = \frac{1}{nh} \sum_{i=1}^{n} \frac{1}{2} 1_{\{(x-h, x+h]\}}(x_i) = \\ &= \frac{1}{nh} \sum_{i=1}^{n} K\left(\frac{x_i - x}{h}\right) \end{aligned}$$

using the uniform kernel: $K(u) = \frac{1}{2}$ if $-1 < u \leq 1$ and 0 otherwise.

Density estimation (cont.)

- We apply this estimator on the CD rates data:

```
> plot(density(CDrates, kernel="rectangular", adjust=0.7))
```



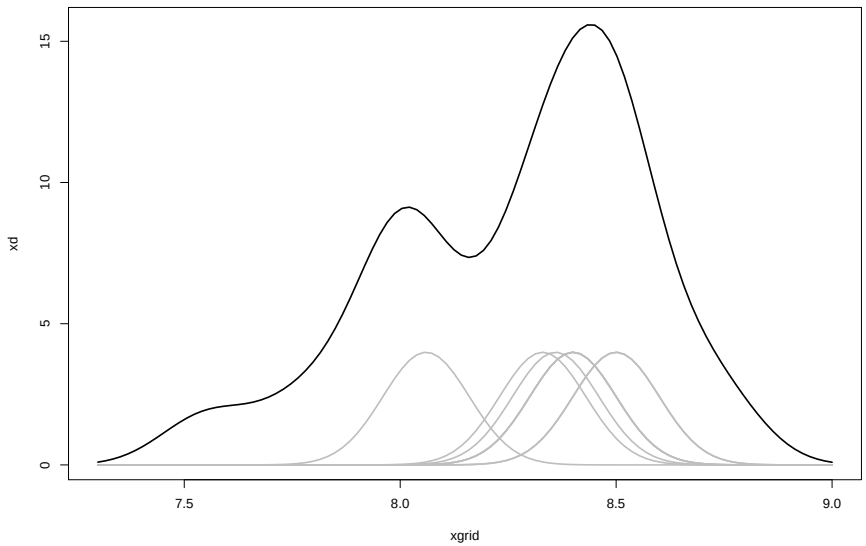
- This is quite wiggly! Problem: The kernel that we used is "unsmooth".
- Consequence: We need better (smoother) kernels!

Kernel density estimation

- For instance, we may use a Gaussian density for K.
- Generally, a kernel function K is symmetric, bounded, and non-negative, with $\int K(u)\,du = 1$. (Exceptions exist!)
- The kernel density estimator

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

estimates the density by re-distributing the point mass $\frac{1}{n}$ smoothly to its vicinity.

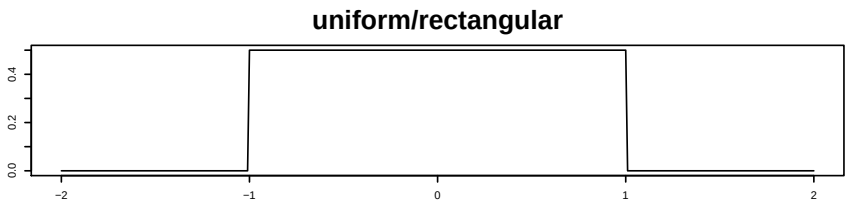


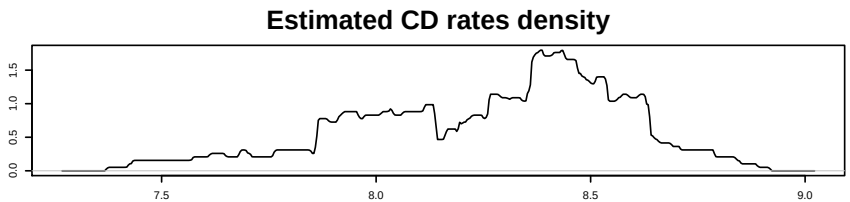
Kernel functions

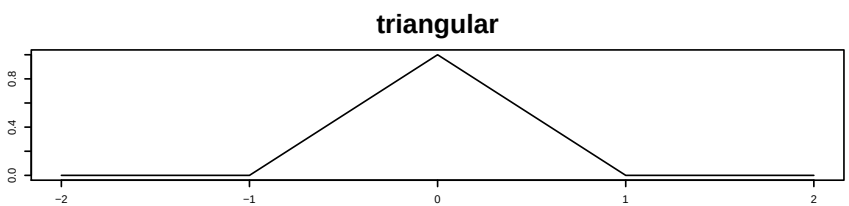
Commonly used kernel functions are summarized in the following table:

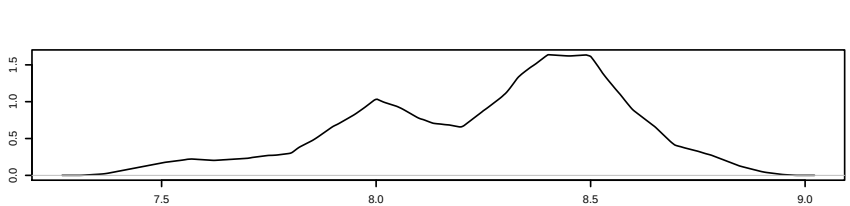
Kernel name	$K(u) =$	option `kernel=''...''` in R function `density`
Uniform	$\frac{1}{2} \cdot 1_{[-1,1]}(u)$	`rectangular`
Triangular	$(1 - \lvert u \rvert) \cdot 1_{[-1,1]}(u)$	`triangular`
Gaussian	$\frac{1}{\sqrt{2\pi}} e^{-u^2/2}$	`gaussian`
Epanechnikov	$\frac{3}{4}(1 - u^2) \cdot 1_{[-1,1]}(u)$	`epanechnikov`
Biweight	$\frac{15}{16}(1 - u^2)^2$	`biweight`

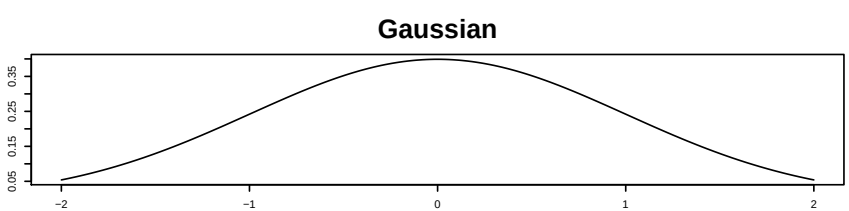
Comparison of kernel functions

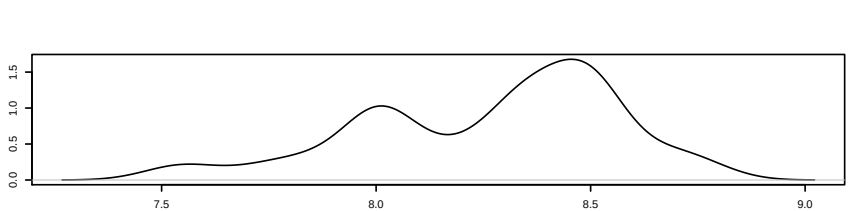


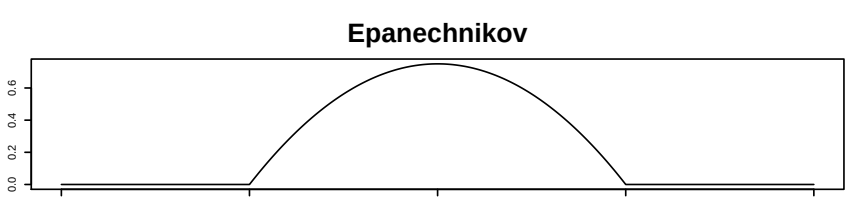


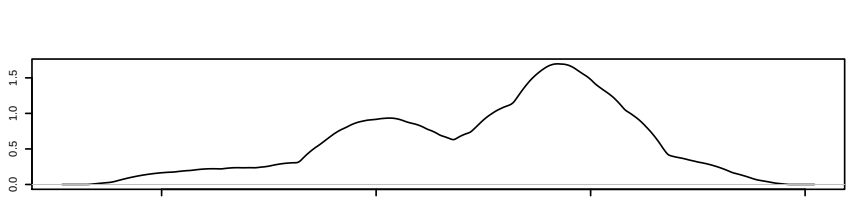


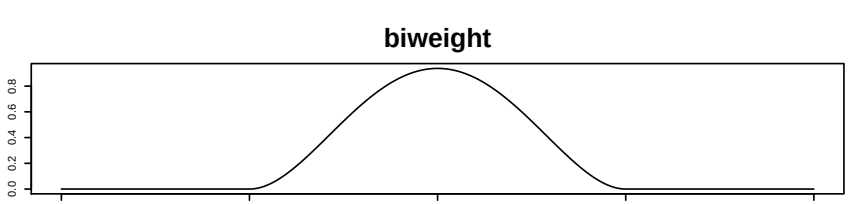


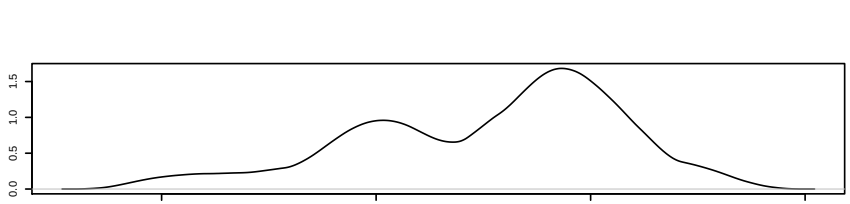






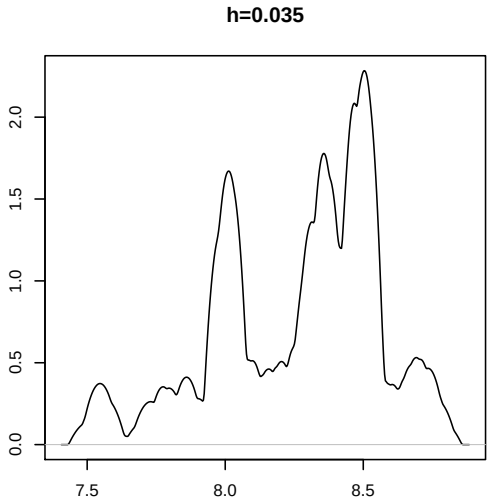






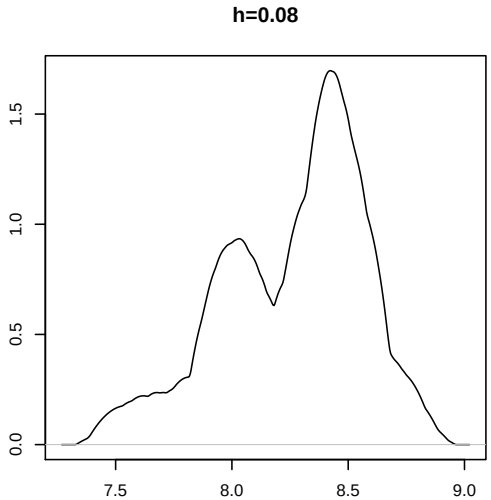
Comparison of bandwidths

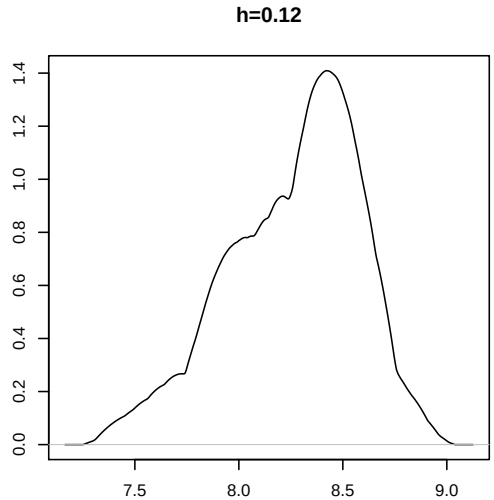
- Hence, for the same bandwidth h, the results are more or less similar as long as the kernel is "smooth".
- Far more important than the choice of the kernel is the choice of the bandwidth h:



h too small

undersmoothing





h too big

oversmoothing

Bandwidth selection

- A simple rule of thumb was provided by [Si, p.48]:

$$h_{opt} = 0.9An^{-1/5}$$

with $A = \min(\text{st.dev.}, \text{IQR}/1.34)$.

- This formula is based on asymptotic considerations (i.e. $h \longrightarrow 0, nh \longrightarrow \infty$) and is optimal for a normally distributed density ("normal reference"). The hybrid measure of spread, A, is used to account for multimodal distributions.
- There exist a large number of alternative bandwidth selection methods, but this one is the simplest, computationally fastest, and works generally well.
- We consider bandwidth selection, including its theoretical background, in more detail in the context of nonparametric regression.

Bivariate density estimation

- For bivariate data $x_1, \ldots, x_n, x_i \in \mathbb{R}^2$ we need a bivariate kernel $K : \mathbb{R}^2 \longrightarrow \mathbb{R}$, which can either be realized through a *product kernel* (generated from a univariate kernel K):

$$K^P(u_1, u_2) = K(u_1)K(u_2)$$

or through a *radially symmetric* kernel

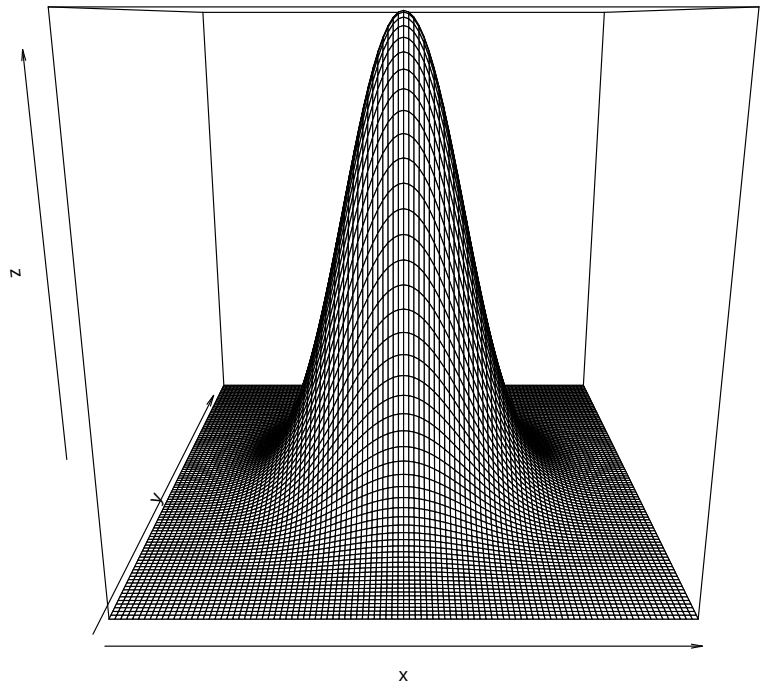
$$K^S(u_1, u_2) = \text{const} \cdot K\left(\sqrt{u_1^2 + u_2^2}\right)$$

- If K is a Gaussian kernel, then K^P and K^S are equivalent, and one gets

$$K(u_1, u_2) = \frac{1}{2\pi} \exp(-\frac{1}{2}(u_1^2 + u_2^2)$$

Bivariate density estimation (cont.)

- Bivariate Gaussian kernel:



- Straightforward extension to higher dimensional kernels:

$$K(\mathbf{u}) = (2\pi)^{-d/2} \exp(-\frac{1}{2}\mathbf{u}^T\mathbf{u})$$

Bivariate density estimation (cont.)

- Instead of a single bandwidth h, we need then two bandwidths h_1 and h_2 controlling the smoothness of the fit in direction of the corresponding axis. The estimate of the "true" density $f(x_1, x_2)$ is given by

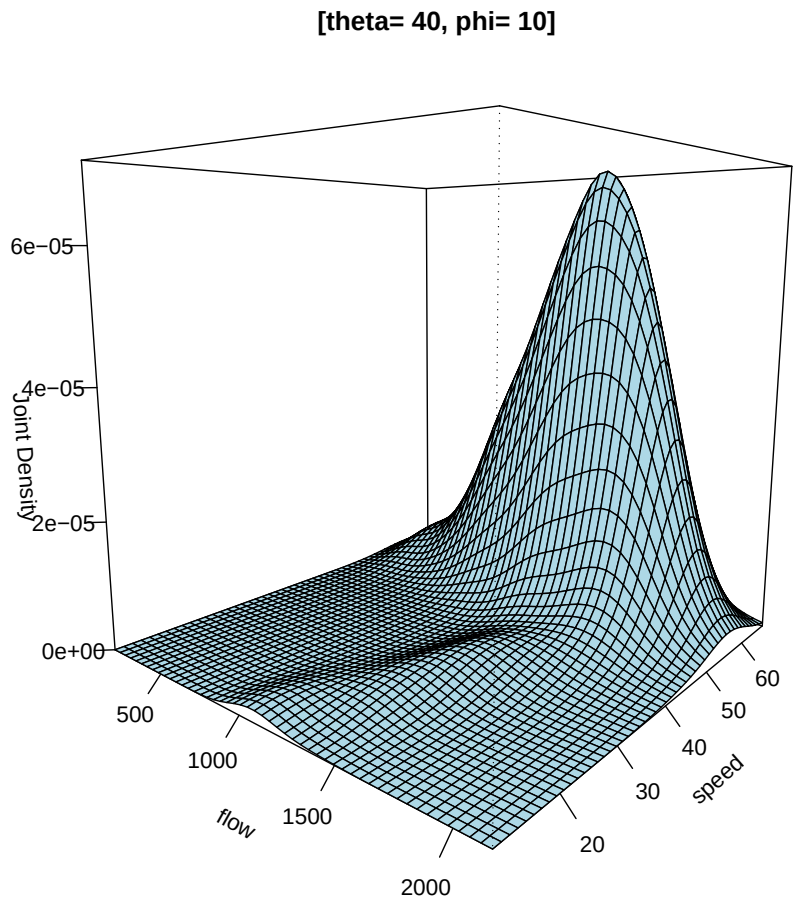
$$\hat{f}(x_1, x_2) = \frac{1}{nh_1h_2} \sum_{i=1}^{n} K\left(\frac{x_1 - X_{i1}}{h_1}, \frac{x_2 - X_{i2}}{h_2}\right)$$

- $d-$variate case: $\hat{f}(x) = \frac{1}{n}\sum_{i=1}^{n} K_{\mathbf{H}}(\mathbf{x} - \mathbf{X}_i)$ with $K_{\mathbf{H}}(\mathbf{x}) = |\mathbf{H}|^{-1/2} K(\mathbf{H}^{-1/2}\mathbf{x})$, with bandwidth matrix $\mathbf{H} \in \mathbb{R}^{\mathbf{d},\mathbf{d}}$. For $d = 2$,

$$\mathbf{H} = \begin{pmatrix} h_1^2 & 0 \\ 0 & h_2^2 \end{pmatrix}.$$

Bivariate density estimation (cont.)

- Kernel density estimate of traffic data ($h_1 = 100, h_2 = 5$):




```
> library(np)
> sf.npdens <- npudens(tdat =lane2[,c("flow","speed")], bws=c(100,5))
> plot(sf.npdens, view="fixed", theta=40)
```

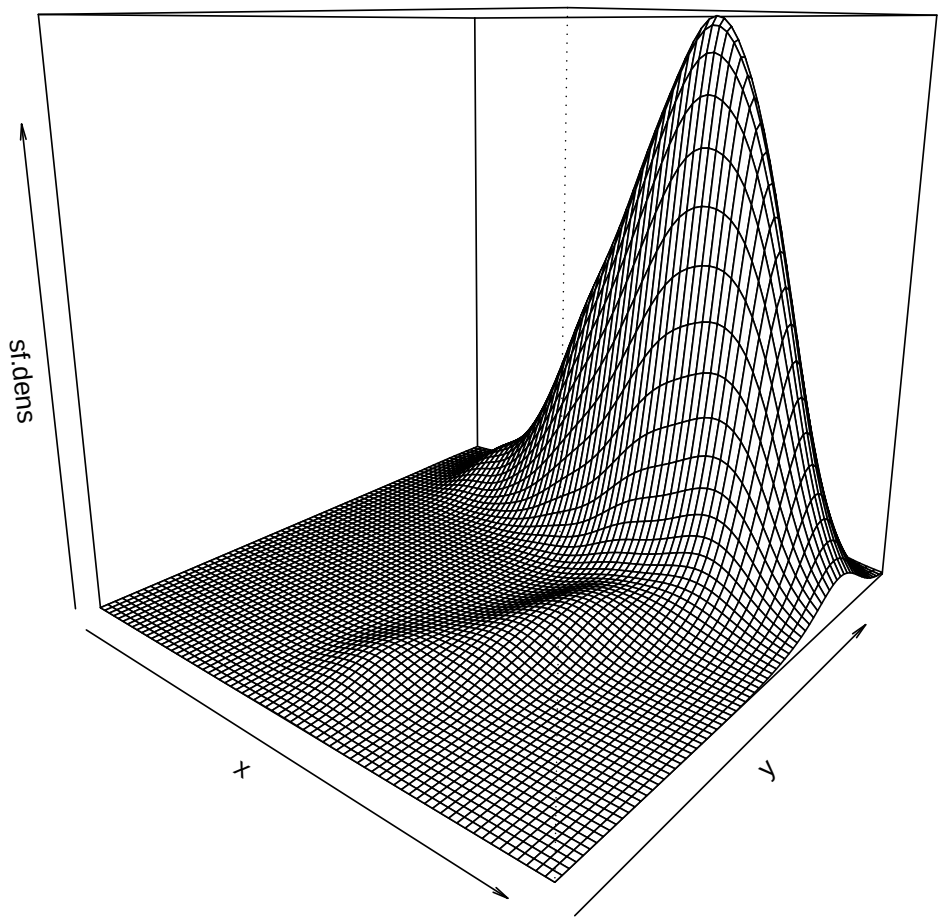
Bivariate density estimation (cont.)

- It is more enlightening to implement it by hand. With $K(\cdot) =$dnorm$(\cdot)$,

```
> bidens<- function(X,Y, xgrid, ygrid, h1,h2){
+  n<-length(X); n1 <-length(xgrid); n2<-length(ygrid)
+  dens <- matrix(0,n1,n2)
+  for (i in 1:n1){
+   for (j in 1:n2){
+    dens[i,j]<-(n*h1*h2)^(-1)*
        sum(dnorm((X-xgrid[i])/h1)*dnorm((Y-ygrid[j])/h2) )
+   }
+ }
+ return(dens)
+ }
> x<-seq(0,2000,by=20)
> y<- 0:70
> sf.dens <-bidens(lane2$flow, lane2$speed, x, y, 100, 5)
```

Bivariate density estimation (cont.)

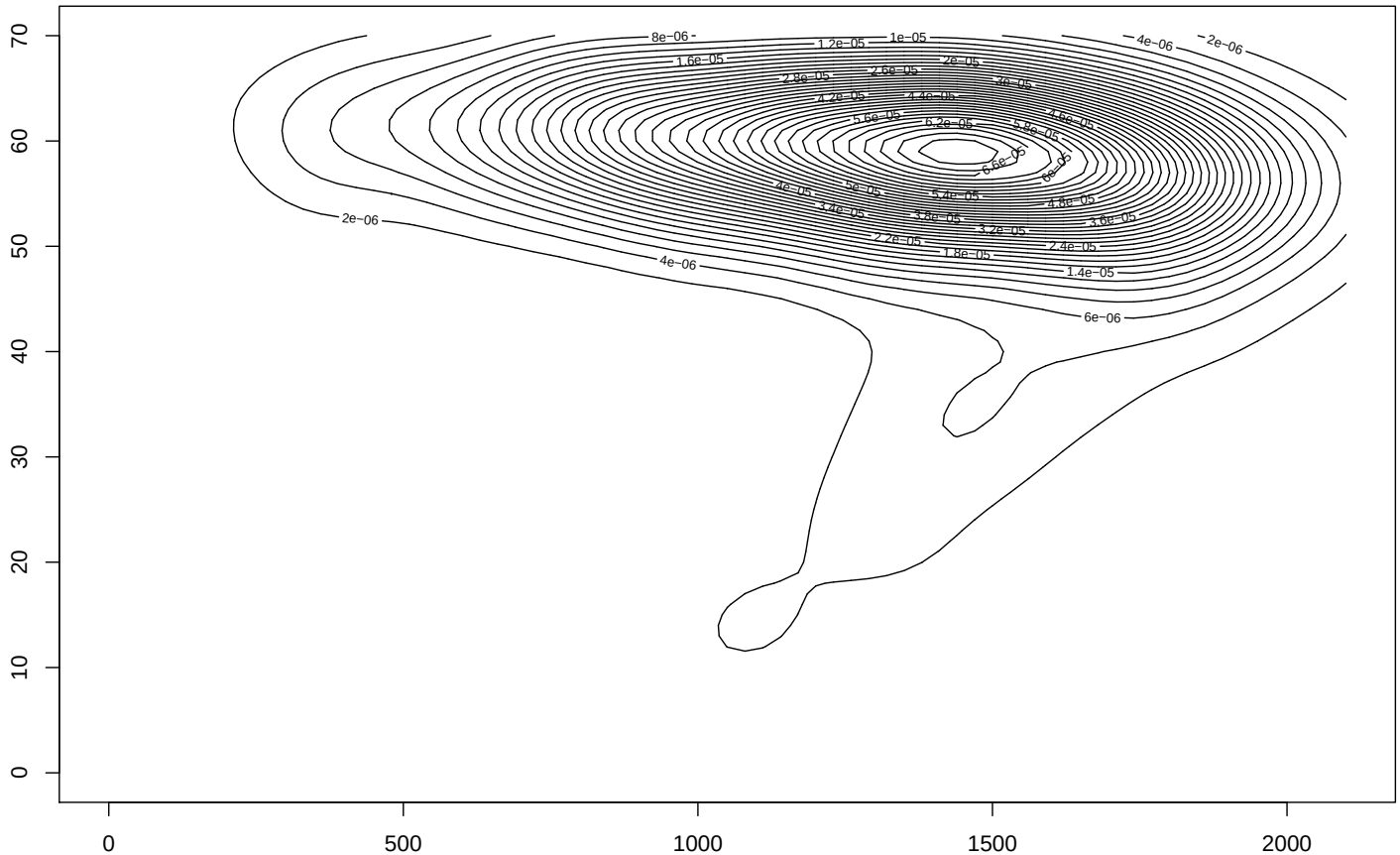
- Perspective plot:




```
> persp(x,y, sf.dens,theta=40)
```

Bivariate density estimation (cont.)

- A more attractive way of visualizing kernel density estimates is often to use contour plots:




```
> contour(x,y, sf.dens,nlevels=30)
```

Regression

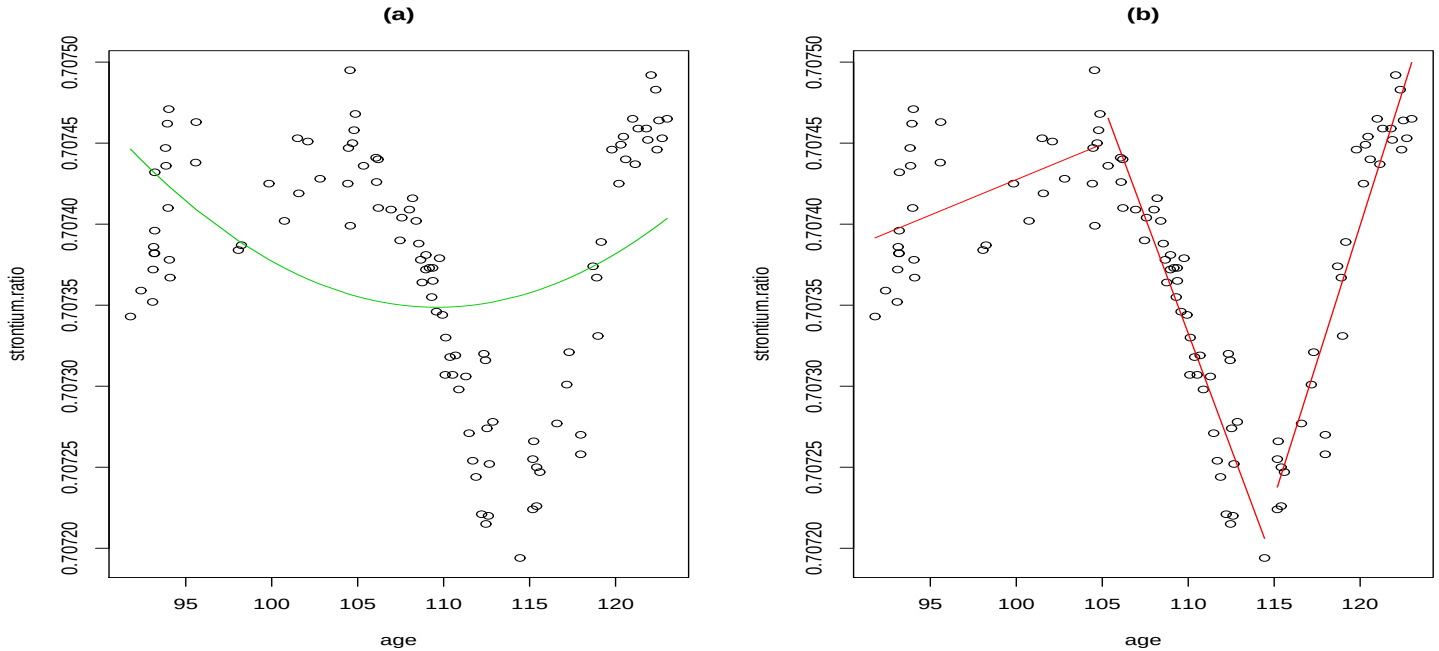
- Having observed data on two variables X and Y, the main objective is often to construct a model for Y given X, enabling us to predict the response for a future observation $X = x$.
- We write such a model in its most general form as

$$Y = m(X) + \varepsilon$$

  where m is the regression function or signal, and ε some noise (e.g. measurement errors).
- As both variables play now a particular, non-intercheangeable role, this implies an asymmetric relationship between X and Y.
- Estimating the function m from $(X_1, Y_1), \ldots, (X_n, Y_n)$ is called regression. If a parametric (e.g. linear) model is pre-specified for m, then we talk about parametric regression, otherwise nonparamatric regression.

Polynomial and piecewise regression

- Consider again fossil data.
- As seen before (Example 2), a linear regression line $\hat{y} = \hat{a} + \hat{b}x$ is clearly inappropriate to fit these data.
- Possible remedies:
  - (a) Fit a higher order polynomial, e.g. $\hat{y} = \hat{a} + \hat{b}x + \hat{c}x^2$
  - (b) Fit piecewise line segments, e.g. split at $x = 105$ and 115.

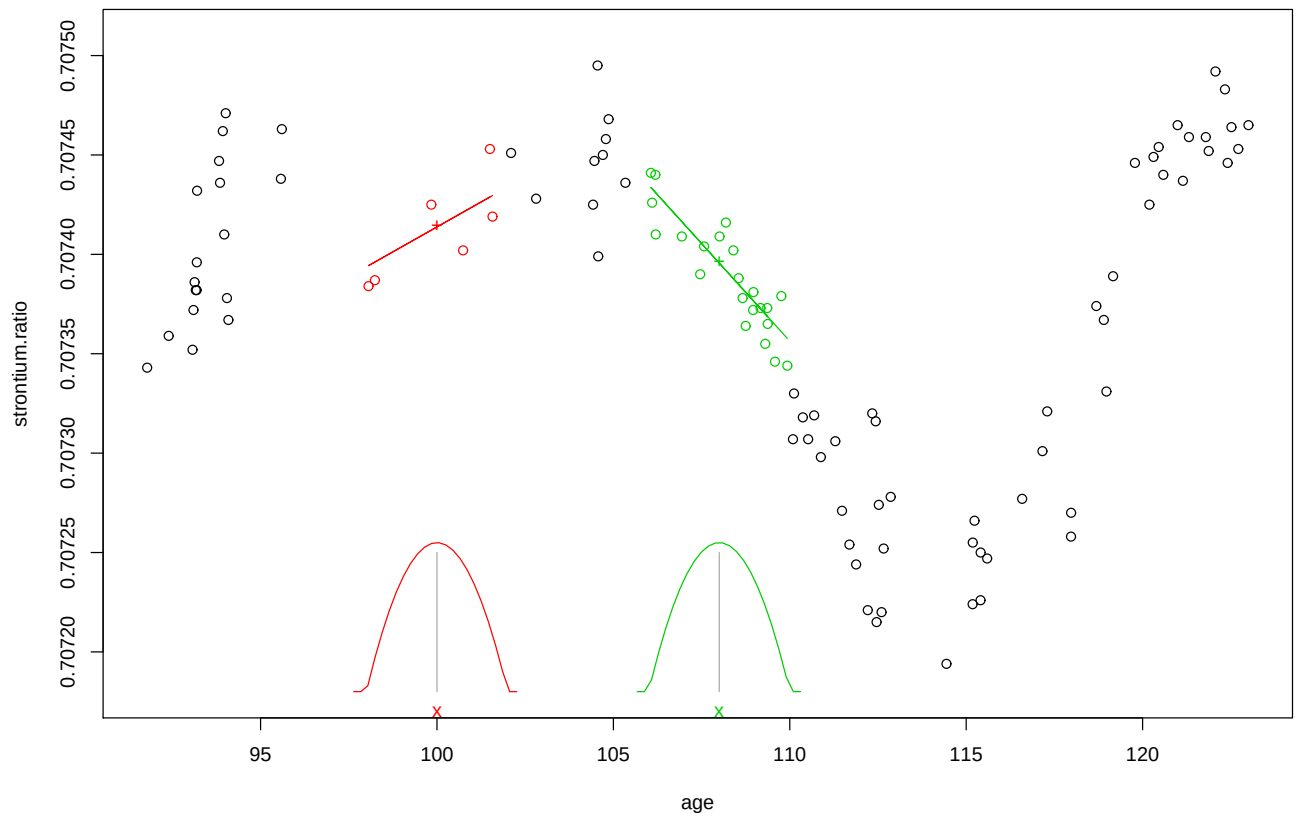


From parametric to nonparametric regression

- Clearly, the higher order polynomials used in (a) are a dead end: We would need a huge polynomial degree to fit the data adequately, e.g. $p \geq 10$, which would entail a large variability of the fit.
- The attempt in (b) seems to be the way to go. However, the fit has to be smoother, and the localization should happen automatically.
- Concretely, when estimating some regression function $m(x)$ at a certain target point x, data (X_i, Y_i) with X_i located close to x are more relevant than data situated far from x.
- Hence, some local weighting is required and it turns out that kernel functions provide a convenient way to achieve this.

Local linear regression

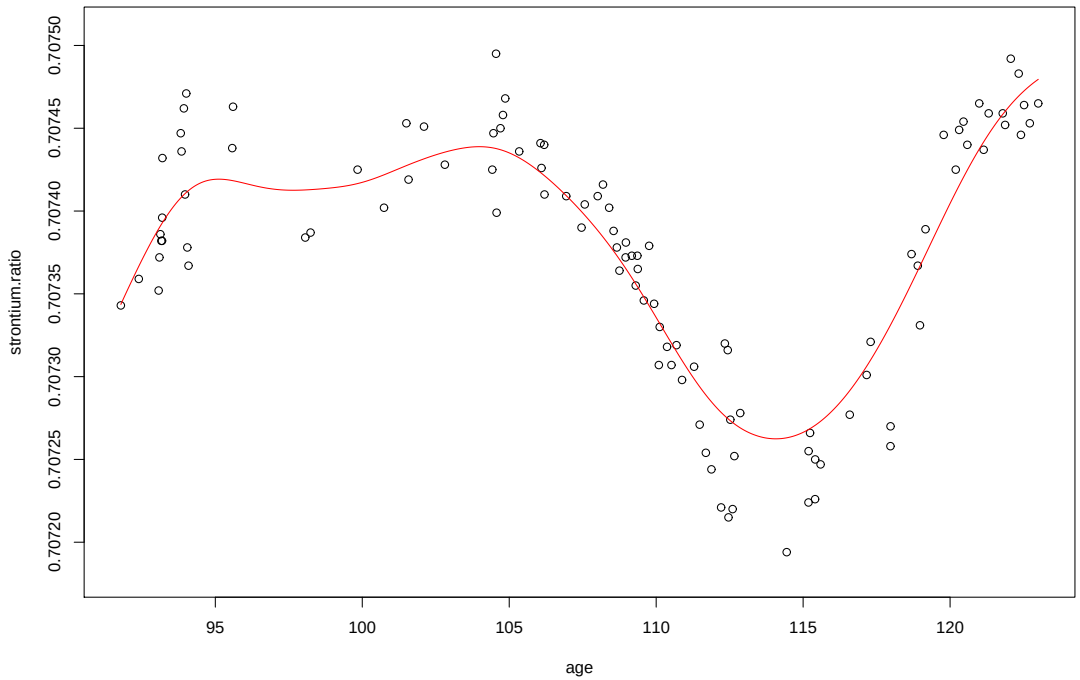
Example of two Epanechnikov kernel functions placed at $x = 100$ and $x = 108$. For estimation at x, only the colored data points are considered (respectively), with the kernel function serving as a weight function.



The estimates $\hat{m}(x)$ are symbolized by + and +, respectively.

Local linear regression (cont.)

This procedure is carried out at every single point x, and one gets the resulting curve, a "local linear fit" (using $h = 2$):




```
> library(KernSmooth)
> fossil.locpoly<-locpoly(age, strontium.ratio, bandwidth=2)
```

Apparently there is still room for improvement w.r.t. the choice of h (look at the last bump), and we get back to this issue later.

The nonparametric regression model

- We consider data $(X_i, Y_i) \in \mathbb{R}^2$, $i = 1, \ldots, n$, forming an iid sample from a population (X, Y).
- We assume that predictor X_i and response Y_i are related through

$$Y_i = m(X_i) + \sigma(X_i)\varepsilon_i$$

where

  - $m(x) = E(Y_i|X_i = x)$ is a "smooth" (i.e., twice continuously differentiable) underlying regression function
  - $\sigma^2(x) = \mathrm{Var}(Y_i|X_i = x)$ is a variance function.
  - ε_i is some iid noise with $E(\epsilon_i) = 0$, $\mathrm{Var}(\epsilon_i) = 1$, which is independent of the X_i.
- We further denote the "design density" of X by $f(\cdot)$.

Homoscedasticity and Heteroscedasticity

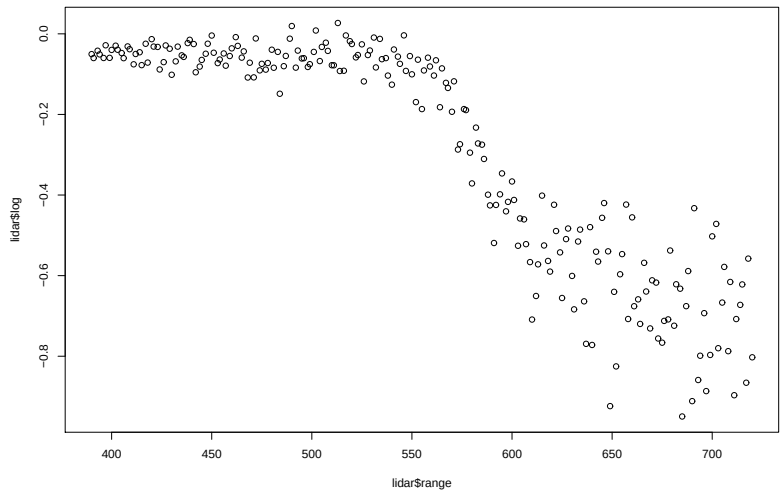
- The variance function $\sigma^2(x) = \mathrm{Var}(Y_i|X_i = x)$ allows explicitly for an error variance varying over the predictor domain, a characteristic which is referred to as heteroscedasticity. As one then models *location* $m(x)$ and *scale* $\sigma(x)$ simultaneously, one also speaks of a *location-scale model* in this context.
- An often assumed (and often approximately met, as for the fossil data) condition is homoscedasticity

$$\sigma^2(X_i) \equiv \sigma^2,\ i = 1, \ldots, n$$

- In nonparametric regression, it is irrelevant for the function-fitting process if the data are homoscedastic or heteroscedastic. However, it plays a role for the variability of the fitted curve, i.e. $\mathrm{Var}(\hat{m}(x))$, and therefore for the calculation of confidence bands etc.
- We assume the more general case of heteroscedastic data if not stated differently.

Homoscedasticity and Heteroscedasticity (cont.)

- Example for heteroscedastic data: LIDAR (light detection and ranging) data [RWC].
- LIDAR uses the reflection of laser-emitted light to detect chemical compounds in the athmosphere.
- Variables:
  - `range:` The distance travelled before light is reflected back to its source (X).
  - `logratio:` The logarithm of the ratio of received light from two laser sources (Y).



Local linear regression estimation

- Local linear regression works by fitting, for each value of x, a linear regression subject to the kernel weights

$$w_i(x) = K_h(X_i - x) \equiv \frac{1}{h} K\left(\frac{X_i - x}{h}\right).$$

- For improved stability and ease of asymptotic calculations, we center the local linear regession at x, yielding the local model

$$m(X_i) = \beta_0(x) + \beta_1(x)(X_i - x) + \epsilon_i. \quad (2)$$

  For ease of notation, we write $\beta_0 \equiv \beta_0(x)$ and $\beta_1 \equiv \beta_1(x)$.

- Then the *locally weighted least squares problem* takes the form

$$\sum_{i=1}^{n} (Y_i - \beta_0 - \beta_1(X_i - x))^2 K\left(\frac{X_i - x}{h}\right) = Q_1(\beta_0, \beta_1), \quad (3)$$

Local linear regression estimation (cont.)

which is minimized by solving

$$\frac{\partial Q_1(\beta_0,\beta_1)}{\partial \beta_0} = -2\sum_{i=1}^{n}\left(Y_i-\beta_0-\beta_1(X_i-x)\right)w_i(x)=0$$

$$\frac{\partial Q_1(\beta_0,\beta_1)}{\partial \beta_1} = -2\sum_{i=1}^{n}\left(Y_i-\beta_0-\beta_1(X_i-x)\right)(X_i-x)w_i(x)=0$$

with respect to $\hat{\beta}_0$ and $\hat{\beta}_1$, yielding

$$\hat{\beta}_0 = \frac{\sum_{i=1}^{n} s_i(x)Y_i}{\sum_{i=1}^{n} s_i(x)}$$

with $s_i(x) = w_i(x)(S_{n,2}-(X_i-x)S_{n,1})$; $S_{n,j}=\sum_{i=1}^{n} w_i(x)(X_i-x)^j$

Local linear regression estimation (cont.)

- From the estimated coefficients, we get, for every x, the estimate of the regression function $m(x)$ using the model (2):

$$\hat{m}(x) = \hat{\beta}_0(x)$$

- This procedure has to be carried out for every value x of interest (e.g. an equidistant grid, or all values $x = X_i, i = 1, \ldots, n$.)
- Special case, $\beta_1 \equiv 0$: This is *local constant regression* and leads to the Nadaraya-Watson estimator

$$\hat{m}(x) = \hat{\beta}_0(x) = \frac{\sum_{i=1}^{n} w_i(x) Y_i}{\sum_{i=1}^{n} w_i(x)}$$

Local polynomial regression

- For general polynomial order $p \geq 0$, we consider the least squares problem

$$\sum_{i=1}^{n} \left(Y_i - \sum_{j=0}^{p} \beta_j(x)(X_i - x)^j \right)^2 K\left(\frac{X_i - x}{h} \right)$$

  yielding estimates $\hat{\beta}_0(x), \ldots, \hat{\beta}_p(x)$.

- Note that a Taylor expansion of m at x of order p yields

$$m(z) = \sum_{j=0}^{p} \frac{m^{(j)}(x)}{j!}(z - x)^j \equiv \sum_{j=0}^{p} \beta_j(x)(z - x)^j$$

Derivative estimation

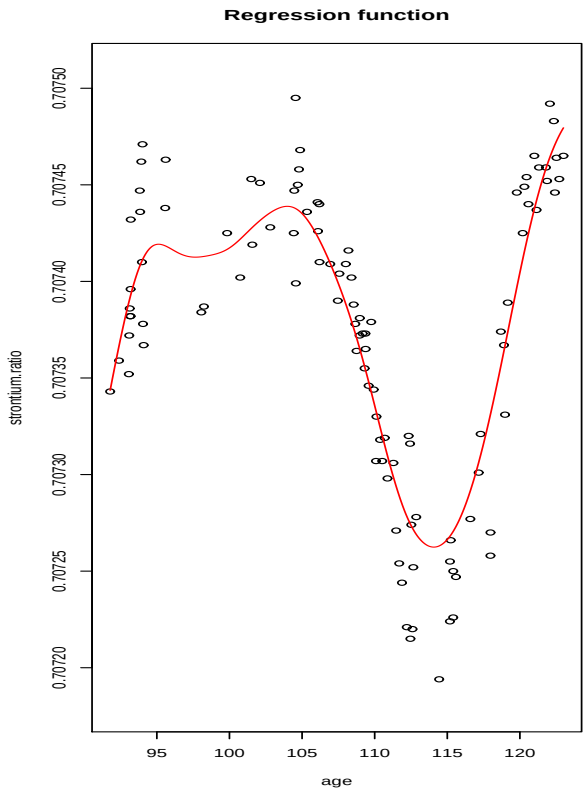
- Hence, the coefficients $\beta_j \equiv \beta_j(x)$ $(j > 1)$ capture the information contained in the derivatives.
- Generally, one has for the estimate of the $j - th$ derivative

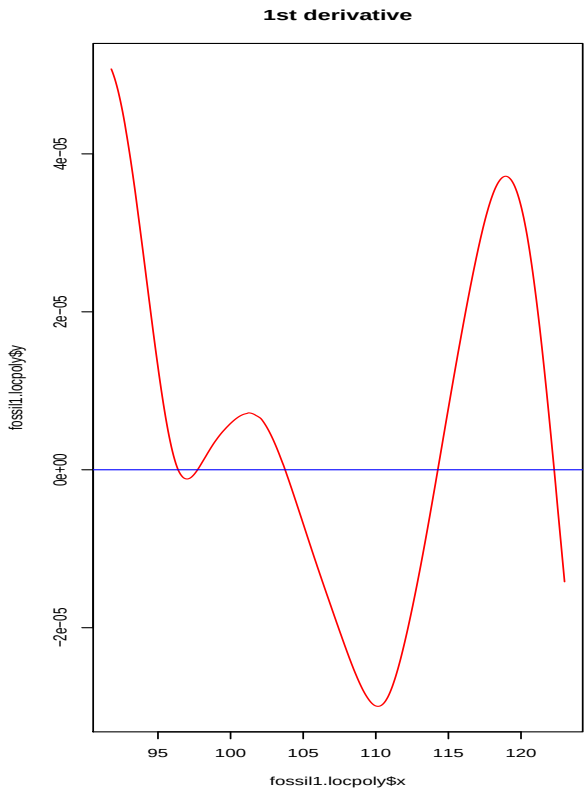
$$\hat{m}^{(j)}(x) = j!\hat{\beta}_j(x) \quad (0 \leq j \leq p)$$

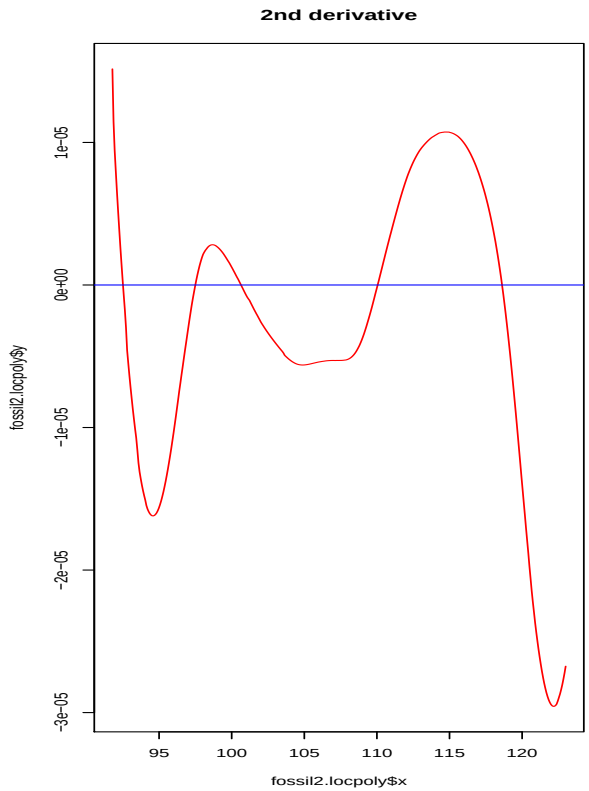
- Technically, an estimate of the $j - th$ derivative requires $p \geq j$.
- Practically, one should choose at least $p \geq j + 1$.
- Derivative estimates are often more variable than estimates of the regression function. The higher the derivative, the higher tends to be the "optimal" (in a sense to be defined later) bandwidth h.

Derivative estimation (cont.)

Example: Fossil data (all fits using $h = 2$)







$$p = 1,\ j = 0$$
$$\hat{m}(x) = \hat{\beta}_0(x)$$

$$p = 2,\ j = 1$$
$$\hat{m}'(x) = \hat{\beta}_1(x)$$

$$p = 3,\ j = 2$$
$$\hat{m}''(x) = 2\hat{\beta}_2(x)$$

Matrix notation

Let

$$\mathbf{X} = \begin{pmatrix} 1 & X_1 - x & \dots & (X_1 - x)^p \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_n - x & \dots & (X_n - x)^p \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} Y_1 \\ \vdots \\ Y_n \end{pmatrix}$$

and $\boldsymbol{\beta} = (\beta_0, \dots, \beta_p)$, $\mathbf{W} = \text{diag}\{w_1(x), \dots, w_n(x)\}$. Then

$$\sum_{i=1}^{n} w_i(x) \left(Y_i - \sum_{j=0}^{p} \beta_j (X_i - x)^j \right)^2 = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^T \mathbf{W} (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

with the straightforward weighted least squares solution

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} \mathbf{y} \tag{4}$$

Evaluation of performance

- Having an estimator $\hat{m}(x)$ of $m(x)$, one is interested in assessing its performance.
- The criterion usually employed here is the mean squared error

$$\text{MSE}(x) = E\left[(\hat{m}(x) - m(x))^2 | \mathbb{X}\right]$$

with $\mathbb{X} = (X_1, \ldots, X_n)$.

- The integral-version evaluates the entire curve and is given by the mean integrated squared error

$$\text{MISE} = \int \text{MSE}(x) w(x)\, dx$$

where a weight function, e.g. $w(x) = f(x)$, may be employed.

- Note that both $\text{MSE}(x)$ and MISE are random variables (as they depend on $X_1, \ldots, X_n$) and that the expectation is taken with respect to the conditional distribution $Y_1, \ldots, Y_n | \mathbb{X}$.

Bias-variance decomposition

We have

$$\begin{aligned}
\mathrm{MSE}(x) &= E\left[(\hat{m}(x) - m(x))^2 | \mathbb{X}\right] = \\
&= E[\hat{m}^2(x)|\mathbb{X}] - 2m(x)E[\hat{m}(x)|\mathbb{X}] + m^2(x) + \\
&\quad + \{E[\hat{m}(x)|\mathbb{X}]\}^2 - \{E[\hat{m}(x)|\mathbb{X}]\}^2 = \\
&= \{E[\hat{m}(x)|\mathbb{X}] - m(x)\}^2 + E[\hat{m}^2(x)|\mathbb{X}] - \{E[\hat{m}(x)|\mathbb{X}]\}^2 = \\
&\equiv (\mathrm{Bias}[\hat{m}(x)|\mathbb{X}])^2 + \mathrm{Var}[\hat{m}(x)|\mathbb{X}]
\end{aligned}$$

with

$$\begin{aligned}
\mathrm{Bias}[\hat{m}(x)|\mathbb{X}] &= E[\hat{m}(x)|\mathbb{X}] - m(x) \\
\mathrm{Var}[\hat{m}(x)|\mathbb{X}] &= E[\hat{m}^2(x)|\mathbb{X}] - \{E[\hat{m}(x)|\mathbb{X}]\}^2.
\end{aligned}$$

Bias and variance calculation

- Firstly, note that $\hat{m}(x) = \mathbf{e}_1^T \hat{\boldsymbol{\beta}}$, with $\mathbf{e}_1 = (1, 0, \ldots, 0)^T \in \mathbb{R}^p$.
- One can show that bias and variance of $\hat{m}(x)$ can be written in the following form:

$$\begin{aligned} \text{Bias}(\hat{m}(x)|\mathbb{X}) &= \mathbf{e}_1^T (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} (\mathbf{m} - \mathbf{X}\boldsymbol{\beta}) \\ \text{Var}(\hat{m}(x)|\mathbb{X}) &= \mathbf{e}_1^T (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} (\mathbf{X}^T \boldsymbol{\Sigma} \mathbf{X}) (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{e}_1 \end{aligned}$$

  with $\mathbf{m} = (m(X_1), \ldots, m(X_n))^T$, $\boldsymbol{\Sigma} = \text{diag}\{w_i^2(x)\sigma^2(x_i)\}_{1 \le i \le n}$.
- These formulas contain unknown quantities and are hard to use and interpret.
- This is why *asymptotical* versions of these formulas are developed.

Kernel Asymptotics

- We assume that the bandwidth is small ($h \longrightarrow 0$), but that the number of observations increases to infinity more rapidly than the bandwidth falls ($nh \longrightarrow \infty$).
- We define the *kernel moments*
  $\mu_j = \int u^j K(u)\, du$, $\nu_j = \int u^j K^2(u)\, du$.
- One can show that under these conditions the following asymptotic approximations hold:

polynomial order	Bias$(\hat{m}(x)\|\mathbb{X}) \approx$	Var$(\hat{m}(x)\|\mathbb{X}) \approx$
$p = 0$	$\frac{h^2\mu_2}{2}\{m''(x) + 2m'(x)\frac{f'(x)}{f(x)}\}$	$\frac{\nu_0\sigma^2(x)}{nhf(x)}$
$p = 1$	$\frac{h^2\mu_2}{2}m''(x)$	$\frac{\nu_0\sigma^2(x)}{nhf(x)}$

- Literature on kernel asymptotics: Fan & Gijbels, 1996, Wand & Jones, 1995.

Kernel Asymptotics (cont.)

Interpretation:

- With increasing bandwidth h,
  - the bias increases,
  - the variance falls.
- The bias of the local linear fit does not depend on the design density $f(x)$ (one says, the local linear fit is design-adaptive).
- The local linear fit is preferable to the local constant fit, because it reduces the bias compared to a local constant fit, but does not lead to an increase in variance.

Linear smoothers

- Recalling the matrix notation, the fitted values for a local polynomial fit are given by

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} = \mathbf{X}(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}\mathbf{y} \equiv \mathbf{L}\mathbf{y}$$

  with

$$\mathbf{L} = \mathbf{X}(\mathbf{X}^T\mathbf{W}\mathbf{X})^{-1}\mathbf{X}^T\mathbf{W}.$$

- Smoothers of the form $\hat{\mathbf{y}} = \mathbf{L}\mathbf{y}$, for some $n \times n$ matrix $\mathbf{L}$, are called linear smoothers.
- $\mathbf{L}$ is generally referred to as the hat matrix and in the smoothing context as the smoother matrix.
- The degrees of freedom of a linear smoother are given by

$$df_{\text{fit}} = \text{tr}(\mathbf{L})$$

  and represent the "equivalent" number of parameters used.

Prediction

- We have a linear smoother $\hat{\mathbf{y}} = \mathbf{L}\mathbf{y}$, i.e.

$$\begin{pmatrix} \hat{y}_1 \\ \vdots \\ \hat{y}_n \end{pmatrix} = \begin{pmatrix} \boldsymbol{\ell}_{X_1}^T \\ \vdots \\ \boldsymbol{\ell}_{X_n}^T \end{pmatrix} \begin{pmatrix} y_1 \\ \vdots \\ y_n \end{pmatrix}$$

  where $\boldsymbol{\ell}_{X_i}^T$ is the row of $\mathbf{L}$ responsible for estimation of $m(\cdot)$ at target point X_i.

- For any arbitrary value x, the estimate of $m(x)$ can then be written as

$$\hat{m}(x) = \boldsymbol{\ell}_x^T \mathbf{y}$$

- In case of the local linear fit,

$$\boldsymbol{\ell}_x^T = \frac{1}{\sum_{i=1}^n s_i(x)} \left(s_1(x) \dots s_n(x)\right)$$

Confidence bands

- The variance of $\hat{m}(x)$ is given by

$$\widehat{\text{Var}}\{\hat{m}(x)\} = \boldsymbol{\ell}_x^T \text{Cov}(\mathbf{y}) \boldsymbol{\ell}_x.$$

- Assuming homoscedasticity for the moment (i.e. $\sigma(X_1) = \ldots = \sigma(X_i) = \sigma$), one has $\text{Cov}(\mathbf{y}) = \sigma^2 \mathbf{I}$, and hence

$$\widehat{\text{st.dev.}}\{\hat{m}(x)\} = \hat{\sigma} ||\boldsymbol{\ell}_x||$$

  for some suitable estimate $\hat{\sigma}$ of σ.

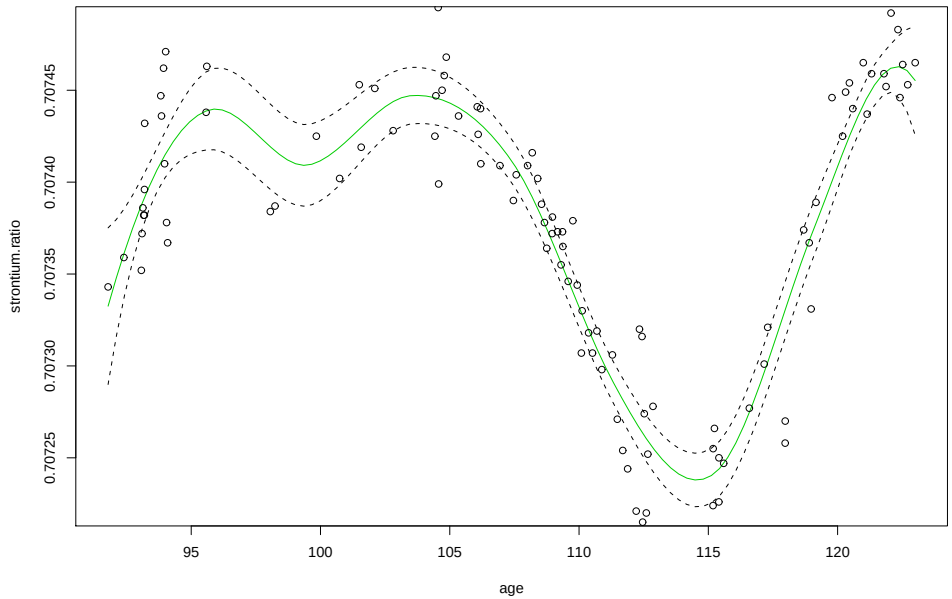
- A very rough 95% pointwise confidence interval for $m(x)$ is given by

$$\hat{m}(x) \pm 2 \times \widehat{\text{st.dev.}}\{\hat{m}(x)\} = \boldsymbol{\ell}_x^T \mathbf{y} \pm 2 \times \hat{\sigma} ||\boldsymbol{\ell}_x||$$

Confidence bands (cont.)

- Application to Fossil data:

```
> library(locfit)
> fossil.locfit <- locfit(strontium.ratio~lp(age, h=6),
data=fossil)
> plot(fossil.locfit, band="global", col=3)
> points(age, strontium.ratio)
```



- **locfit** also offers the possibility to account for heteroscedasticity using the option `band ="local"`. The confidence bands for the fossil data are almost the same then.

Confidence bands (cont.)

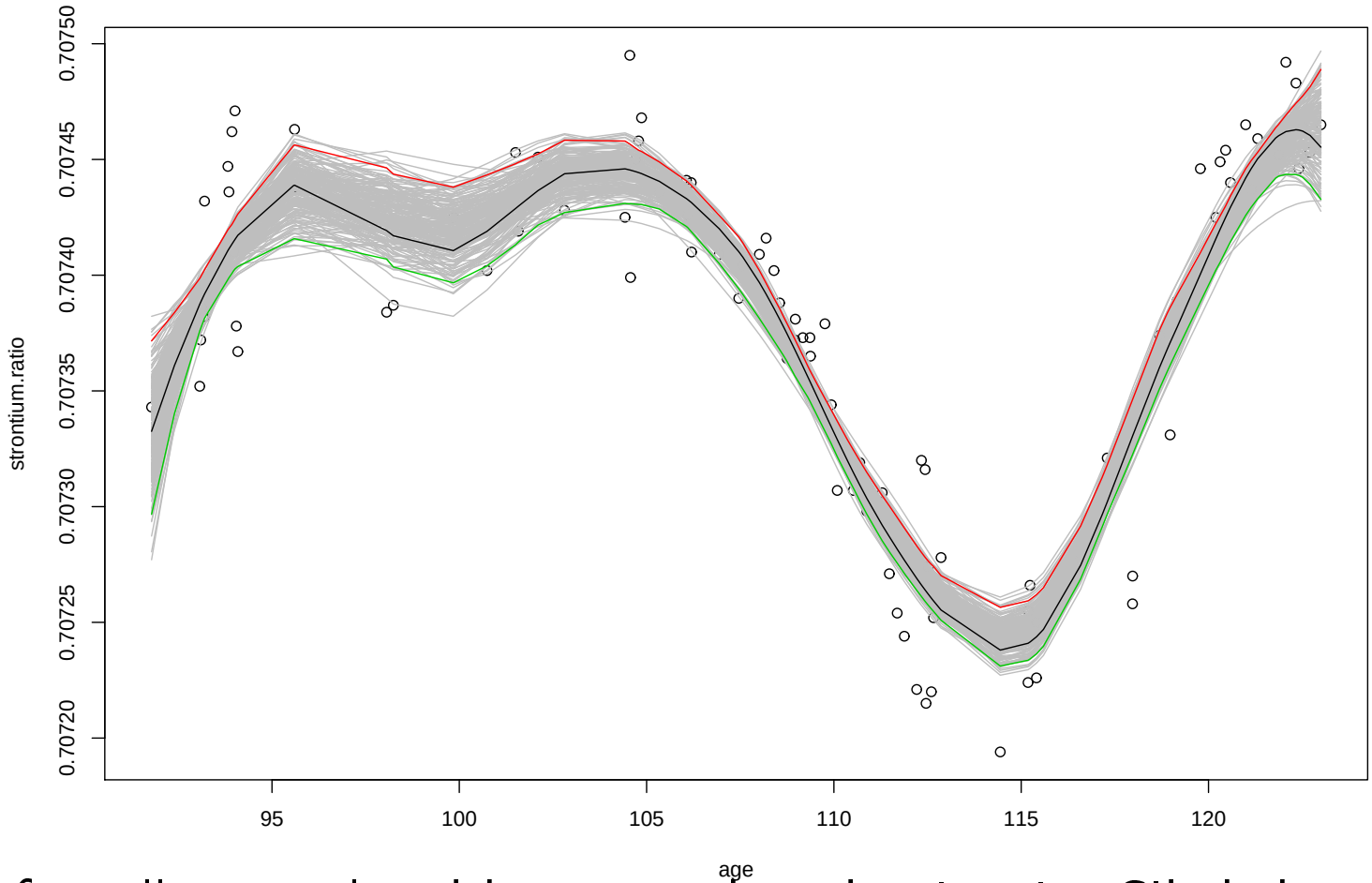
- Note that this approximation is very rough.... Confidence bands constructed in this manner
  - ignore the variability in estimating σ.
  - are incorrect for small sample sizes ($\hat{m}(x)$ is only asymptotically normal).
  - ignore the bias.
  - ignore additional variability due to smoothing parameter selection.
- .... so better only refer to them as variability bands.
- An alternative technique which tries to avoid (especially) the first two problems uses bootstrapping.

Confidence bands via bootstrapping

- Roughly, the strategy is as follows [BA]:
  - Fit a pilot smoother to get a set of residuals.
  - Resample the residuals with replacement, yielding "new" errors.
  - Add these errors to the pilot estimate.
  - Compute the smooth regression function using the new errors.
  - Repeat all this a number of times, say $b = 200$.
  - At each point along the curve, compute sample quantiles using the b estimates.
- Several variants of this do exist, including bias and boundary correction.

Confidence bands via bootstrapping (cont.)

- Pointwise 95% confidence bands for fossil data, using `locfit`, with $h = 6$ and $b = 200$:



- After all, normal and bootstrap-based pointwise CI's behave similarly.

Prediction intervals

- Pointwise confidence bands give us for every x a *confidence interval* for the "true" value of $m(x)$.
- When interested in a *prediction interval* for Y given a new observation at x, one can apply the formula

$$\boldsymbol{\ell}_x^T \mathbf{y} \pm 2 \times \hat{\sigma} \sqrt{1 + ||\boldsymbol{\ell}_x||^2}$$

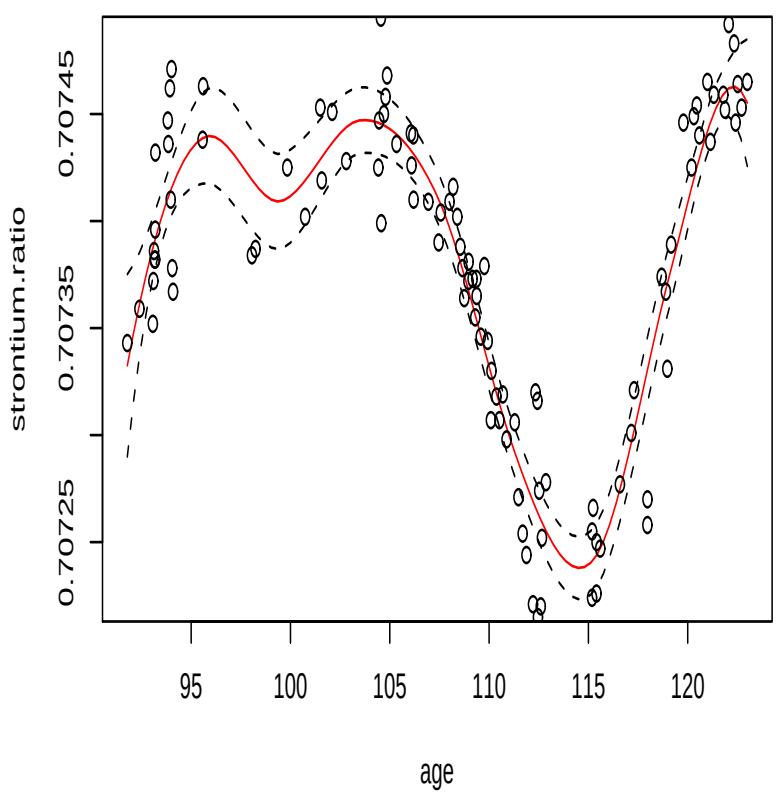
- The additional term "1" under the square root captures (in complete anlogy to parametric regression) the variability of the observations around the estimated curve (in addition to the variability of the curve fitting itself).
- A 95% prediction interval should contain about 95% of the observations.
- The same words of caution as for confidence bands apply, though they are less relevant here, as the prediction interval is larger and will often "swamp" the bias and other sources of inaccuracy.

Confidence and Prediction intervals

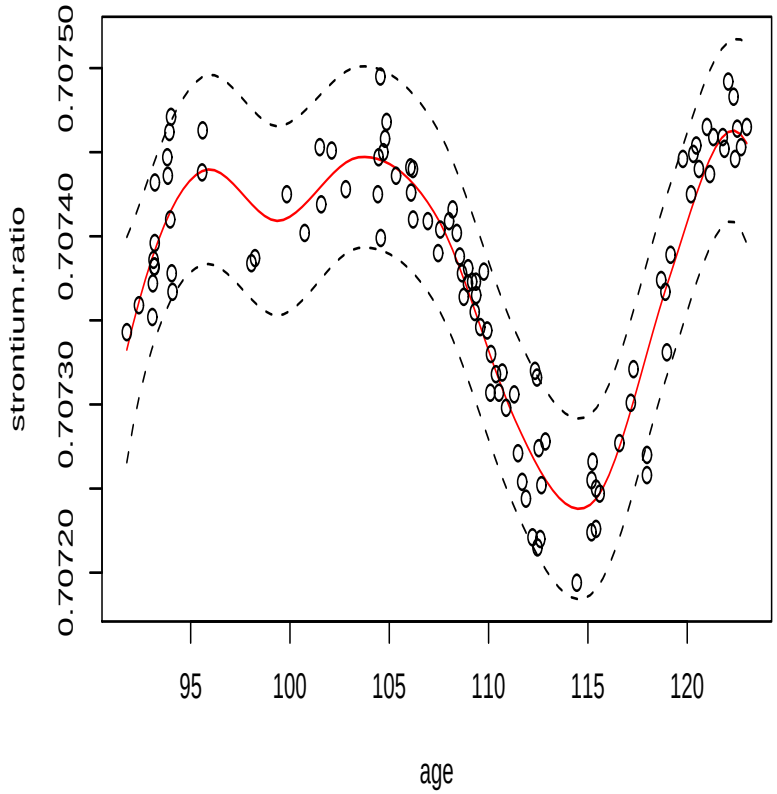
- Prediction intervals for fossil data:

```
> plot(fossil.locfit, band="pred",col=2)
```

Confidende bands



Prediction “bands”



Simultaneous Confidence bands

- Again, pointwise confidence bands give us for every x a *confidence interval* for the "true" value of $m(x)$.
- However, often would like to have a band $[L(x), U(x)]$ such that

$$P\{L(x) \leq m(x) \leq U(x) \text{ for all } x \in R\} \geq 1 - \alpha$$

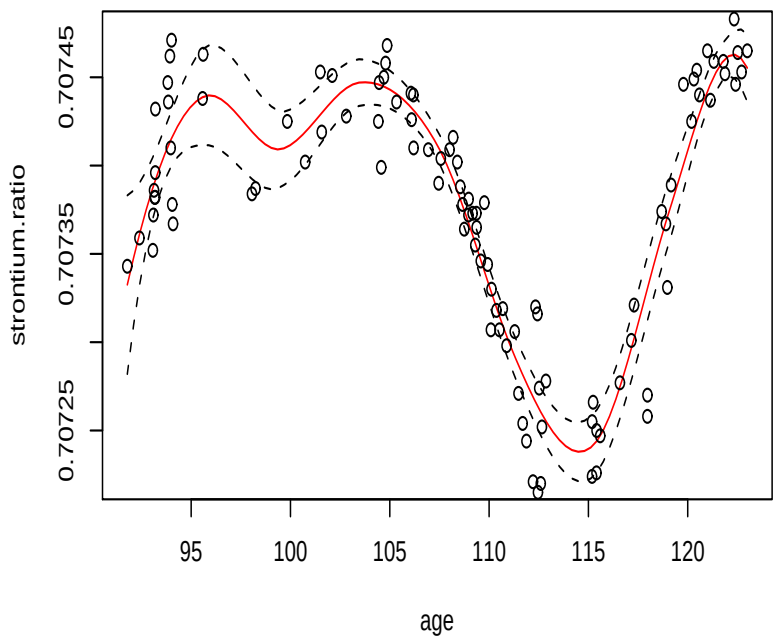
  where R is the range of the sample values $X_1, \ldots, X_n$.
- The theory to this is relatively complicated, though analytical solutions exist ([L] , Sec. 9.2, [RWC], Sec 6.5).
- implemented in function `kappa0` in R package **locfit**.
- Generally, for the width of prediction intervals (PI), simultaneous confidence bands (SCB), and pointwise confidence bands (PCB), one will have the relation

$$\text{PI} \gtrsim \text{SCB} \geq \text{PCB}$$

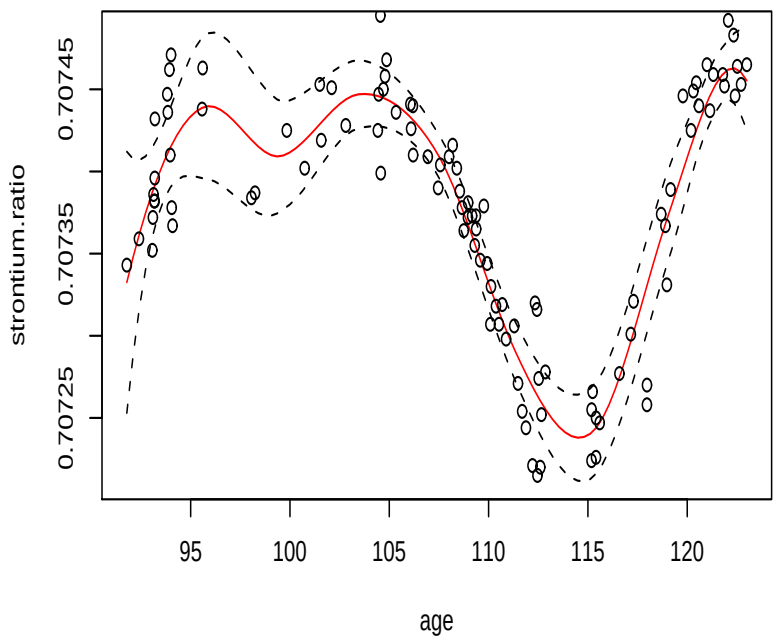
Pointwise and simultaneous confidence bands

Example: 95% confidence bands for Fossil data

Pointwise



Simultaneous




```
> foss.loc <- locfit(strontium.ratio~lp(age,h=6), data=fossil)
> crit(foss.loc) <- crit(foss.loc,cov=0.95)
> plot(foss.loc, band="local",col=2); points(age, strontium.ratio)
> crit(foss.loc) <- kappa0(foss.loc,data=fossil,cov=0.95)
> plot(foss.loc, band="local",col=2); points(age, strontium.ratio)
```

Error variance estimation

- The (non-bootstrapped) CI's and PI's require estimation of $\hat{\sigma}$.
- Model-based estimate: $\hat{\sigma} = \sqrt{\frac{1}{n-d}\sum \hat{\epsilon}_i^2}$, where $d > 0$ is a bias correction. Common choices:

  (1) $d = 0$ ("crude" estimate),

  (2) $d = \text{tr}(\mathbf{L})$ (by analogy to parametric regression),

  (3) $d = 2\text{tr}(\mathbf{L}) - \text{tr}(\mathbf{L}\mathbf{L}^T)$ (correcting the bias of the RSS),

  (4) $d = 1.25\text{tr}(\mathbf{L}) - 0.5$ (approximation of the latter).

```
> foss.loc
Fitted Degrees of freedom: 8.774 # this is tr(L)
Residual scale: 2.64e-05 # this is (3)
> sqrt(sum((residuals(foss.loc))^2)/106)
2.504392 e- 05 # this is (1)
> sqrt(sum((residuals(foss.loc))^2)/(106-8.774))
2.614954e-05 # this is (2)
> sqrt(sum((residuals(foss.loc))^2)/(106-(1.25*8.774-0.5)))
2.63803e-05 # this is (4)
```

Selection of smoothing parameters

- Smoothers generally involve some kind of smoothing parameter, which steers the degree of smoothing.
- In case of kernel smoothers, the smoothing parameter is the bandwidth h.
- For selection of the smoothing parameter, there are several possibilities:
  - An experienced data analyst may use his favorite, well proven smoothing parameter (or "rule of thumb" smoothing parameter selection tool) for a particular application.
  - "Trial and error": Try several smoothing parameters until optical inspection of the fitted curves indicates a reasonable fit.
  - Use automatic model selection tools, which normally try to maximize or minimize some numeric "optimality criterion"
  - Use Bayesian or mixed models which estimate the smoothing parameter as a by-product.

Automatic smoothing parameter selection

- Let us denote with $\hat{m}_\lambda(x)$ an estimate of x using the smoothing parameter λ.
- A possible optimality criterion that may come into mind is the average squared error

$$ASR(\lambda) = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{m}_\lambda(X_i))^2$$

- This criterion will be optimal (i.e. minimal), when $\hat{m}_\lambda(x_i) = Y_i$, which would mean interpolation (i.e., strong undersmoothing) of the data!
- What has gone wrong here is that the same data are used to construct the estimate and to validate it.

Cross-validation

- One possible solution would be to divide the data at random into a training set (e.g. 75% of the data) and a validation set. The training set is used for model fitting, and the validation set for model selection.
- This is often not feasible, as data are scarce.
- This leads to the idea of cross-validation: Divide the data set into $K = 5$ or 10 subsets $\kappa = 1, \ldots, K$. For each $i = 1, \ldots, n$, denote $\kappa(i)$ the subset to which it belongs. Then minimize

$$CV_K(\lambda) = \frac{1}{n} \sum_{i=1}^{n} \left(Y_i - \hat{m}_\lambda^{-\kappa(i)}(X_i) \right)^2$$

  where $\hat{m}_\lambda^{-\kappa(i)}(X_i)$ is the estimate of $m(X_i)$ using all data except subset $\kappa(i)$.
- An extreme, but the most popular, case of this is leave-one-out cross-validation, in which $K = n$.

Leave-one out cross-validation

- Again, the leave-one out cross-validation criterion takes the form

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{m}_\lambda^{-i}(X_i)\right)^2$$

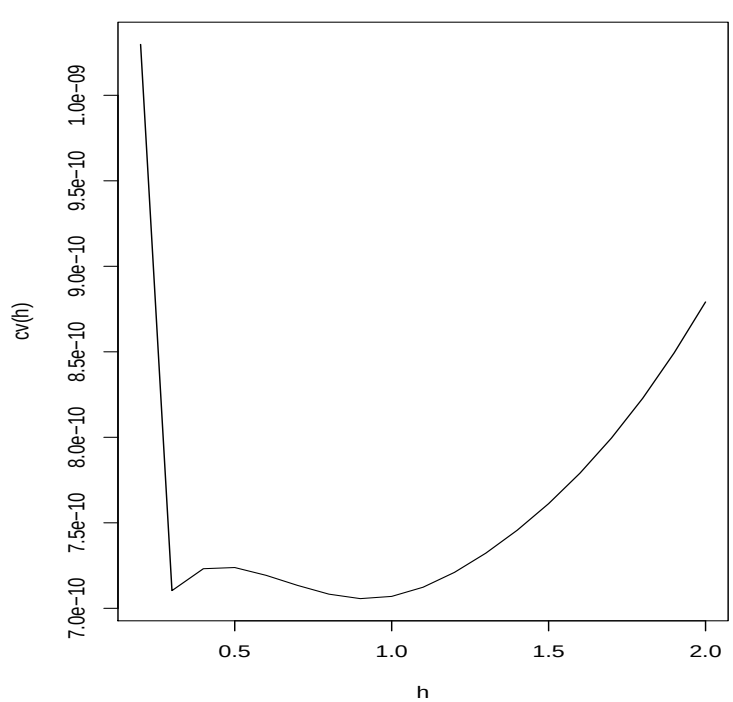
  where $\hat{m}_\lambda^{-i}(X_i)$ means that all data except X_i are used to estimate $m(X_i)$.

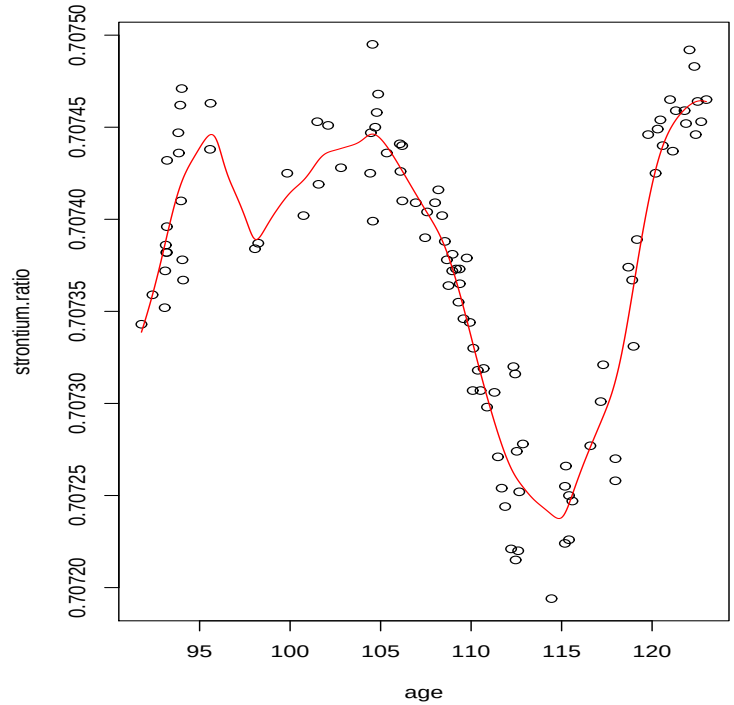
- For linear smoothers $\hat{m}_\lambda(X_i) = \sum_j \ell_{ij} Y_j$, with smoother matrix $\mathbf{L}_\lambda = (\ell_{ij})_{1\leq i\leq n, 1\leq j\leq n}$, this criterion can (after some simple steps [FT, p. 161]) be written as

$$CV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{m}_\lambda(X_i)}{1-\ell_{ii}}\right)^2 .$$

Leave-one out cross-validation

- Fossil data.
- $CV(\lambda)$ for a local linear fit (left), with bandwidth selected at $h = 0.9$, and the corresponding fitted curve (right):





- The fitted curve is partly undersmoothed, which is quite typical for cross-validation.

Generalized cross-validation

- The leave-one-out CV criterion can be computationally burdensome.
- A simplified version is obtained by substituting the leverage values ℓ_{ii} through their average, $\bar{\ell} = \frac{1}{n}\sum_{i=1}^{n} \ell_{ii} = \frac{1}{n}\text{tr}(\mathbf{L}_\lambda)$.
- This leads to the so-called generalized cross-validation criterion

$$GCV(\lambda) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{m}_\lambda(X_i)}{1-\bar{\ell}}\right)^2 = \frac{ASR(\lambda)}{(1-\bar{\ell})^2} \qquad (5)$$

  which is implemented in many smoothing software packages (e.g. **locfit**, **pspline**).
- Normally, the curves $CV(\lambda)$ and $GCV(\lambda)$ are very close.

Other model selection tools

- Another important criterion is Mallows' C_p

$$C_p(\lambda) = ASR(\lambda) + 2\hat{\sigma}^2 df_{\text{fit}}(\lambda)/n$$

  using *any* estimate of σ^2.

- It can be shown through an Taylor expansion of (5) [RWC, p. 120] that

$$GCV(\lambda) \approx ASR(\lambda) + 2\hat{\sigma}^2(\lambda) df_{\text{fit}}(\lambda)/n$$

  with $\hat{\sigma}^2(\lambda) = ASR(\lambda)$ [Recall that $df_{\text{fit}}(\lambda) = \text{tr}(\mathbf{L}_\lambda)$].

- Further, for Gaussian and *known* error σ^2, the AIC criterion

$$AIC(\lambda) = -2\log L + 2df_{\text{fit}}(\lambda)$$

  is exactly $\frac{n}{\sigma^2} C_p(\lambda)$.

- The three criteria differ essentially in the type of variance estimator used, and one should get similar results using either of them.

Plug-in bandwidth selection

- For kernel smoothers, one has convenient access to optimal bandwidths through asymptotic expressions.
- Basic idea: Find the bandwidth h minimizing the integrated asymptotic mean squared error $MISE(h) = \int MSE(\hat{m}(x)|\mathbb{X})\, dx$, with

$$\begin{aligned} MSE(\hat{m}(x)|\mathbb{X}) &= \text{Bias}^2(\hat{m}(x)|\mathbb{X}) + \text{Var}(\hat{m}(x)|\mathbb{X}) = \\ &= \frac{h^4 \mu_2^2}{4}(m''(x))^2 + \frac{\nu_0 \sigma^2(x)}{nhf(x)}. \end{aligned}$$

We set $MISE'(h) = 0$ and get

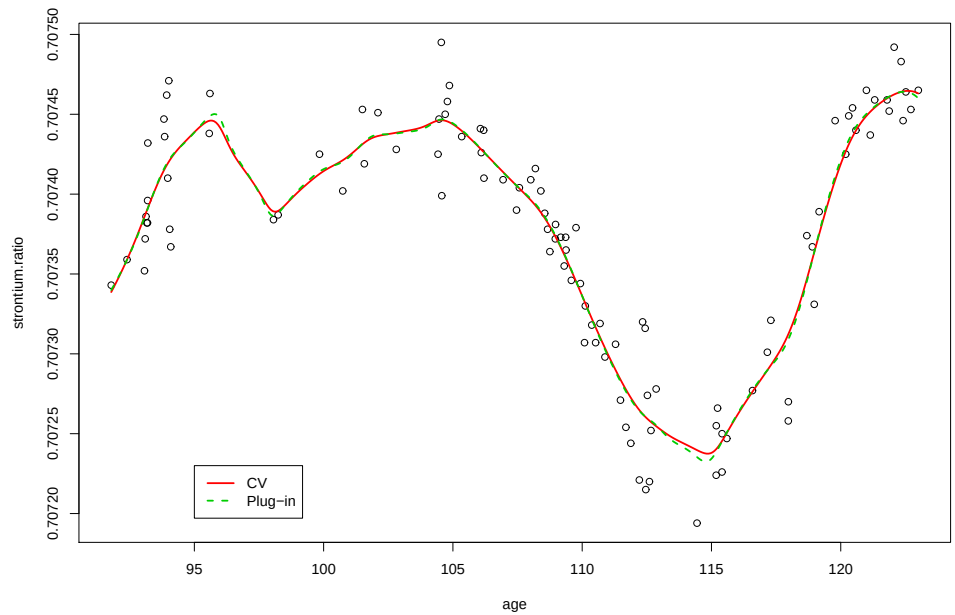
$$h = const \cdot \left(\frac{\int \sigma^2(x)/f(x)\, dx}{\int (m''(x))^2\, dx} \right)^{1/5} n^{-1/5}$$

- Here, $\int (m''(x))^2\, dx$ is unknown. It has to be estimated and "plugged in" to the formula above.

Plug-in bandwidth selection (cont.)

- A plug-in bandwidth selector for local linear regression is implemented in R function `dpill` (**KernSmooth**)
- The smoothness of the estimate of the (integrated) m'' is controlled through Mallows' C_p.
- Example: Fossil data:

```
> library(KernSmooth)
> dpill(age,strontium.ratio)
[1] 0.7875148
```



Variable bandwidth selection

- We have so far considered constant bandwiths h.
- In practice, the regression curve m might exhibit different degrees of smoothness in different parts of the predictor domain, so that a *variable* bandwidth $h(x)$ is more adequate.
- In `locpoly`, one can specify a variable bandwidth by providing a vector $h_1, \ldots, h_m$, where m is the size of the grid $x_1, \ldots, x_m$ on which the estimated regression function is estimated.
- In principle, a variable bandwidth $h(x)$ is immediately obtained by calculating $MSE'(h) = 0$, yielding

$$h(x) = \left(\frac{\nu_0 \sigma^2(x)}{f(x)(m''(x))^2 \mu_2^2} \right)^{1/5} n^{-1/5}$$

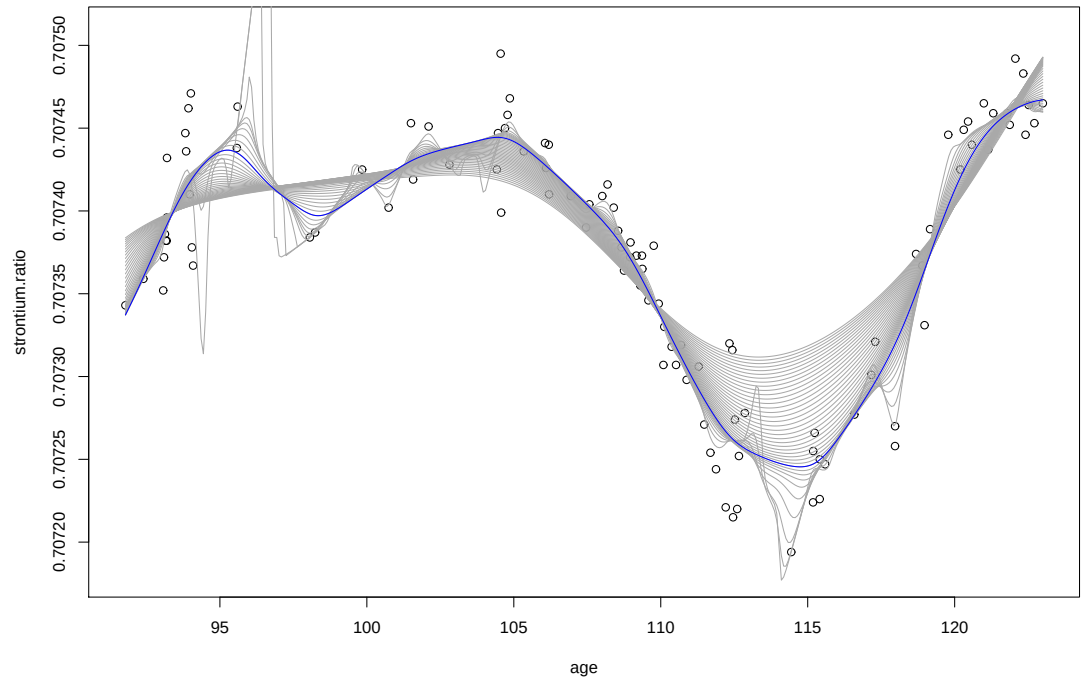
  for the local linear smoother – but no implementation known.
- The function `lokerns` in package **lokern** features variable plug-in bandwidth selection for local constant kernel smoothers.

Bandwidth selection - Classical vs. plugin

- In the nineties, there was a clear tendency to move away from "classical methods" as CV, as they *"exhibit very inferior asymptotic and practical performance"* compared to plug-in methods (e.g. Ruppert, Sheather & Wand, 1995)
- Among several objections against CV, it was claimed that CV bandwidth are ambiguous (several minima) or too variable, that they undersmooth, and do not allow for *variable* bandwidths.
- These views were questioned by Loader (1999), who argued that "*plug-in methods are heavily dependent on arbitrary specification of pilot bandwidths and fail when this specification is wrong. The often quoted variability and undersmoothing of CV reflects* **the uncertainty of bandwidth selection**".
- This insight leads to the *resolution* interpretation of the bandwidth.

Family plots

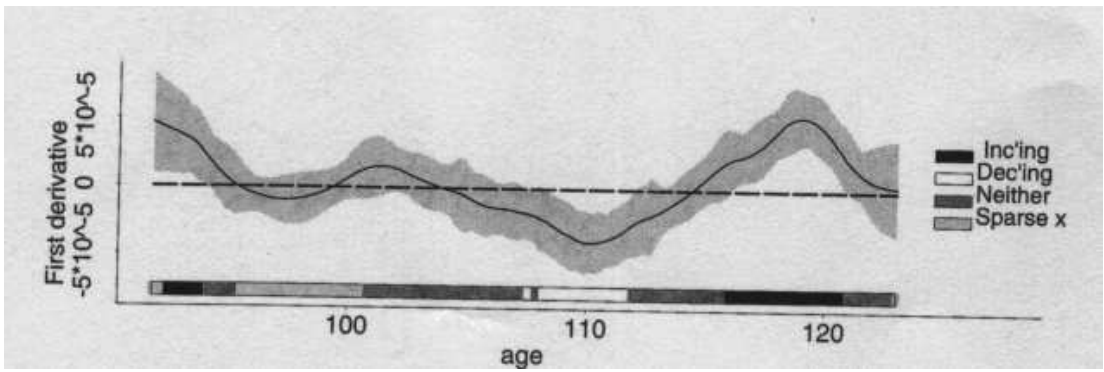
- Study simultaneously a wide range of bandwidths
- The philosophy behind this is that different useful information can be available at different degrees of smoothing.
- Family plot for fossil data with emphasized bandwith at $h = 1.2$:



- A question naturally arising from this is, for instance: Is the dip at age ≈ 98 "really there"?

Feature extraction with SiZer plots

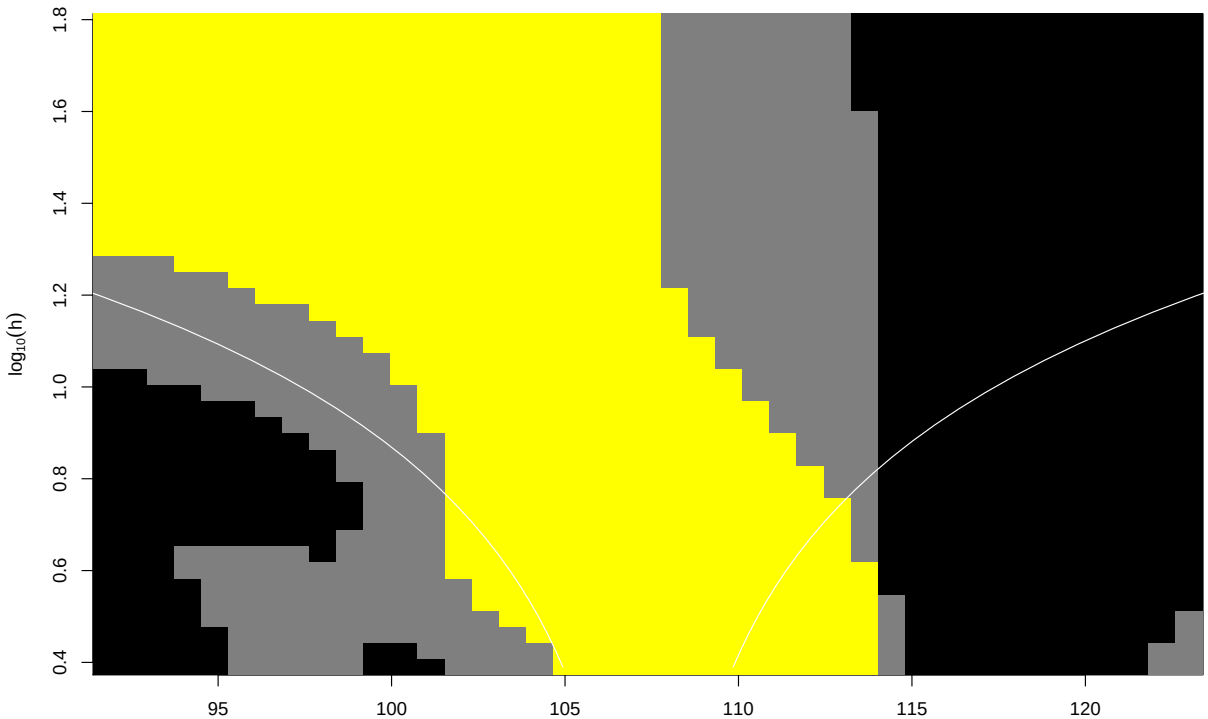
- A bump or a dip corresponds to a location where the first derivative crosses the zero line.
- Hence, look at the first derivative and assess if it "significantly crosses zero"
- This is the **SiZer** (**Si**gnificant **Zer**o crossings of derivatives) technique.
- Illustration for fossil data, with confidence bands $\hat{m}'_h(x) \pm q \cdot \hat{\mathrm{SD}}(\hat{m}'_h(x))$ [RWC, p. 157]



Feature extraction with SiZer plots (cont.)

- Full Sizer plot of Fossil data (black =↑, yellow=↓).

```
> require(SiZer); fossil.sizer <- SiZer(age, strontium.ratio)
> plot(fossil.sizer, colorlist=c("yellow", "grey50", "black"))
```



- The bump at age 115 is significant for all bandwidths considered.
- There is no level of smoothing at which the dip at age =98 is significant. Hence, it is "not really there".

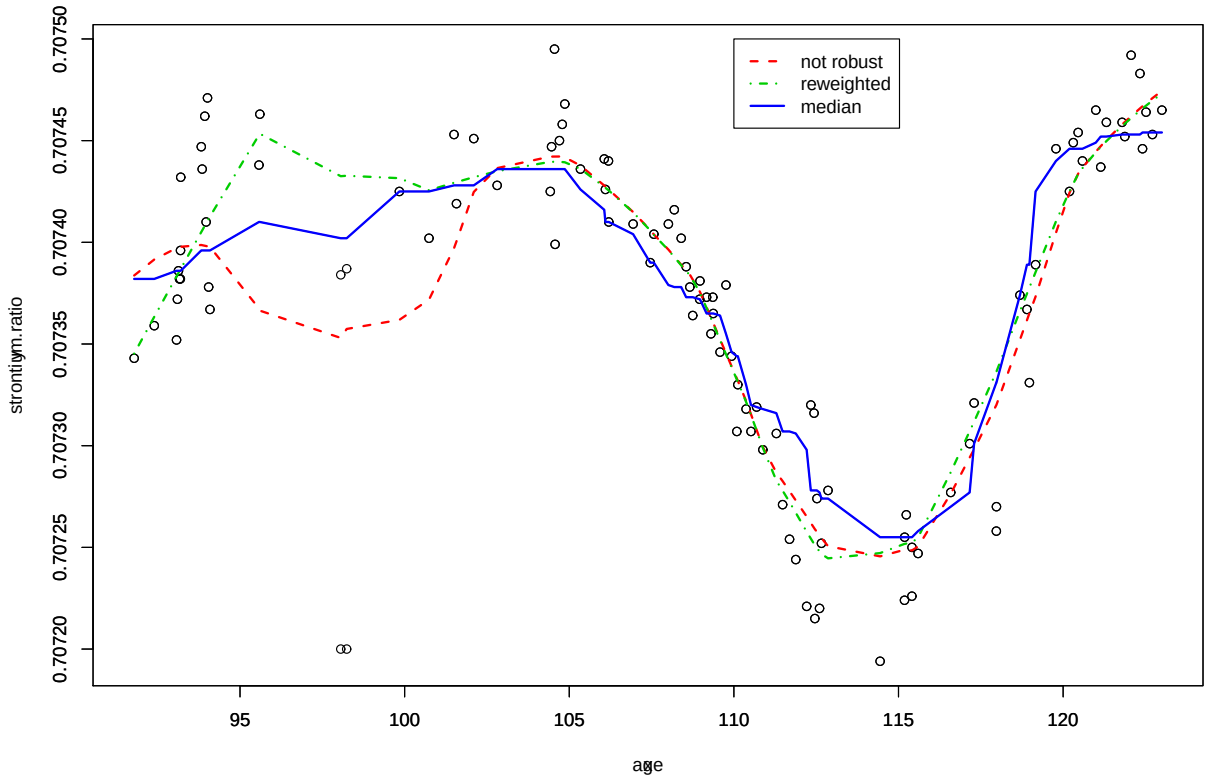
Robust smoothing

- Most smoothing methods are relatively sensitive to outliers due to the use of a *quadratic* error criterion.
- Among the approaches to *robust* smoothing, the following tools have been suggested
  - Iterative downweighting of observations associated to large residuals (Cleveland, 1979):
    - R functions `lowess` (scatterplot smoothing)
    - and `loess` (surface fitting).
  - Estimate Med$(Y|X = x)$ instead of $E(Y|X = x)$. This entails to minimize the sum of absolute instead of sum of squared residuals:
    - Implementation `wm` at the course home page.
- More by Roland Fried on Wednesday!

Robust smoothing (cont.)

- Fossil data with two artificial outliers:

```
> fossil.0 <- lowess(fossil2$age, fossil2$str, f=0.2,iter=0)
# not robust
> fossil.1 <- lowess(fossil2$age, fossil2$str, f=0.2) # reweighted
> fossil.2 <- wm(fossil2$age, fossil2$str, h=2) # median
```



Multivariate Smoothing

- Local regression can also be used if the space of explanatory variables is multivariate, i.e. if a smooth function $m : \mathbb{R}^d \longrightarrow \mathbb{R}$ is to be estimated from observations

$$\{(\boldsymbol{X}_i^T, Y_i), i = 1, \ldots, n\}, \quad \text{with } \boldsymbol{X}_i = (X_{i1}, \ldots, X_{id})^T.$$

- Let $\boldsymbol{x} = (x_1, \ldots, x_d)$ a point in $\mathbb{R}^d$. A multivariate version of (3) is given by

$$\sum_{i=1}^{n} \left\{ Y_i - \beta_0 - \sum_{j=1}^{d} \beta_j (X_{ij} - x_j) \right\}^2 K_{\mathbf{H}}(\mathbf{X}_i - \mathbf{x}), \qquad (6)$$

where $K_{\mathbf{H}}(\cdot)$ is now a multivariate kernel function as defined before for kernel density estimation.

Multivariate Smoothing (cont.)

- In matrix notation, $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_d)$, $\boldsymbol{W} = \text{diag}\{K_{\mathbf{H}}(\mathbf{X}_i - x)\}$,
  $\boldsymbol{X} = \begin{pmatrix} 1 & X_{11} - x_1 & \cdots & X_{1d} - x_d \\ \vdots & \vdots & \ddots & \vdots \\ 1 & X_{n1} - x_1 & \cdots & X_{nd} - x_d \end{pmatrix}$, and the solution takes the same form as (4).
- According to the multivariate Taylor expansion, the coefficents

$$\beta_j = \frac{\partial m}{\partial x_j}(\mathbf{x}),\ j = 1, \ldots, d$$

  play the role of partial derivatives, and $\hat{\beta}_j$ are their estimates.

Bivariate kernel smoothing

- R functions `npreg` (see Example 3) or `loess` (using "nearest neighbors" instead of bandwidths).
- Let's do the local linear fit ourselves:

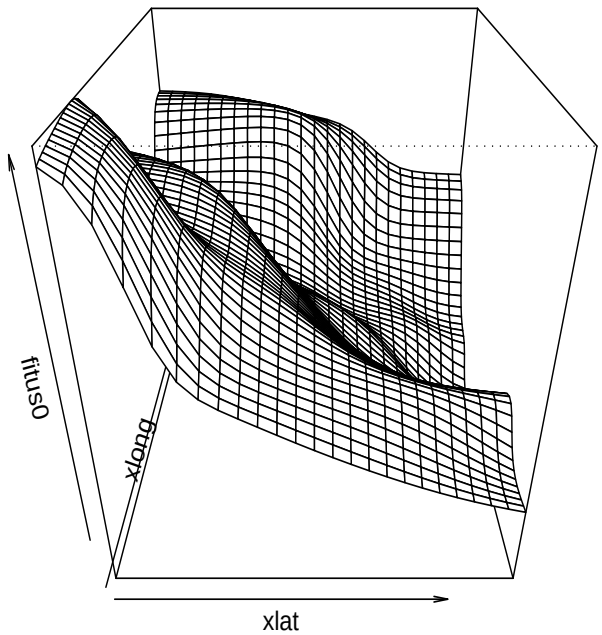
```
> fitus <- matrix(0,25,55)
> for (i in 1:25){
>  for (j in 1:55){
>  bi <- ustemp$latitude-xlat[i]
>  bj <- ustemp$longitude-xlong[j]
>  fitus.lm <- lm(min.temp~bi+bj, weights = dnorm(latitude,
xlat[i],h1) * dnorm(longitude, xlong[j],h2), data = ustemp)
>  fitus[i,j]<-fitus.lm[[1]][1]
> }
> }
```

- Remarks:
  - For the local *constant* fit, remove the lines for `bi` and `bj` and just write `1` after the $\sim$ symbol.
  - `dnorm(latitude,xlat[i],h1)`$= \frac{1}{h_1} K\left(\frac{\texttt{latitude}-\texttt{xlat[i]}}{h_1}\right)$.

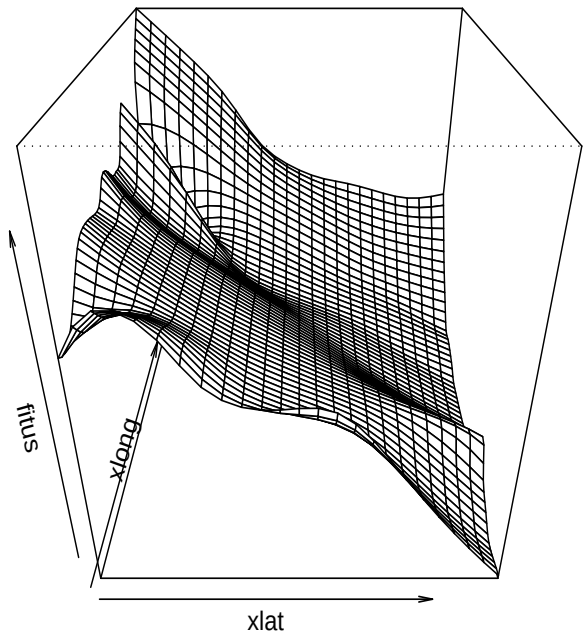
Bivariate kernel smoothing (cont.)

- Back to US temperatute data. Using $h_1 = h_2 = 3$:

Local constant



Local linear



The curse of dimensionality

- Attention: With localized multivariate smoothing happens what statisticians call the "curse of dimensionality".
- Thought experiment: Assume the bandwidth vector $\boldsymbol{h} = (h_1, \ldots, h_d)$ is constructed in way such that the h_j cover each a half of the observations of their associated variable. For some interior point $\mathbf{x}$, which fraction p of the data $X_i, i = 1, \ldots, n$ is actually contained in our local window $\boldsymbol{x} \pm \boldsymbol{h}$?
  - If $d = 1$, then $p = 1/2$ (clearly).
  - If $d = 2$, then $p = 1/4$.
  - If $d = 10$, then $p = (1/2)^{10} \approx 0.001$.
- Hence, in high dimensions local neighborhoods tend to be empty even for large bandwidths! [HTF, p. 22ff]
- This problem can in tendency already be observed in the local linear fit to the US Temp data.

The curse of dimensionality (cont.)

- This phenomenon makes kernel smoothing for higher dimensions difficult.
- Poor results can particular be expected in the boundary regions.
- For $d \geq 2$, local constant smoothing, i.e. $\beta_1 \equiv \ldots \equiv \beta_d \equiv 0$, may be the safer option than local linear smoothing.
- The usual way to circumvent the curse of dimensionality is to replace the full interaction model $m(x_1, x_2)$ by an additive representation $m(x_1) + m(x_2)$. This leads to additive models and we look at this later.
- Also, it useful to look at alternative smoothing methods which do not suffer from these problems (to this extent). One such family of methods are the spline based methods.

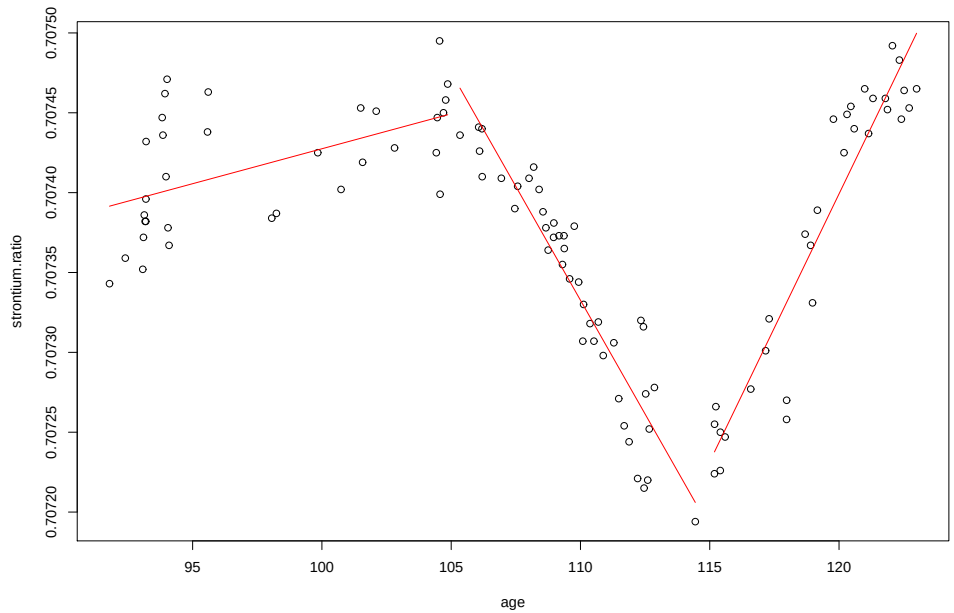
Section 2: Spline based methods

Scope of this section:

- The broken stick model;
- The linear spline basis;
- Knot selection;
- Penalization;
- Smoothing splines;
- Basis function systems;
- B- and P-Splines;
- Additive and semiparametric models;
- Generalized additive models;

An alternative approach

- Reconsider our piecewise linear model for the fossil data:



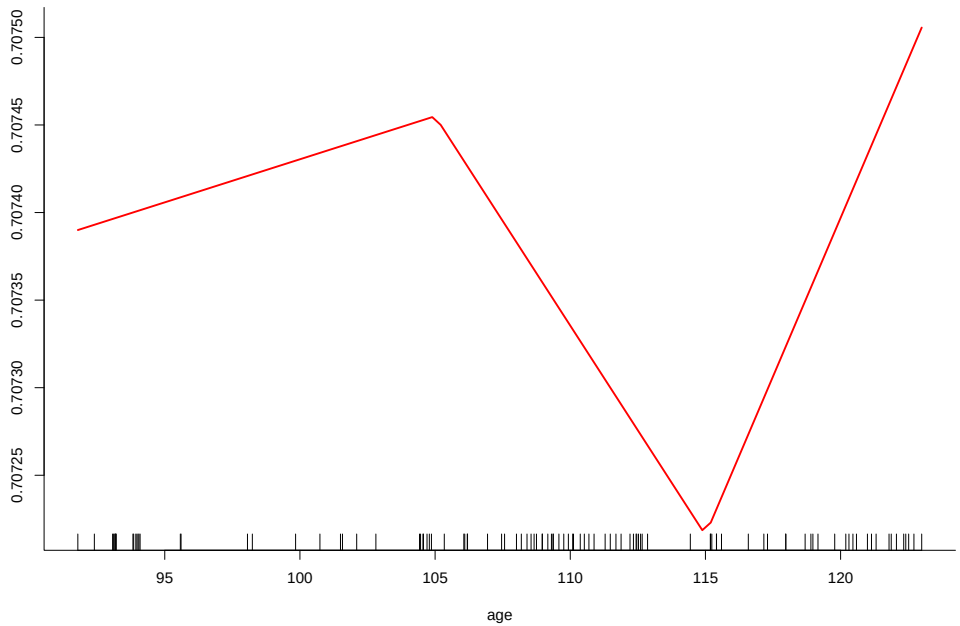
- not too bad, but discontinuous, and uses 6 degrees of freedom
- we can save 2df by "joining the sticks at the knots"
- this leads to a continuous curve

The broken stick model

- The "Broken Stick model" for the fossil data:

  $$\text{strontium.ratio} = \beta_0 + \beta_1 \text{age} + \beta_2(\text{age} - 105)_+ + \beta_3(\text{age} - 115)_+ + \epsilon$$

  where $a_+ = a$ if $a > 0$ and 0 otherwise.




```
> library(SemiPar)
> fossil.spmfit <- spm(strontium.ratio ~f(age,knots = c(105,115),
basis = "trunc.poly", spar=0.001, degree=1))
> plot(fossil.spmfit, se=FALSE, col=2)
```

The linear spline basis

- This idea can be further exploited. Assume we use a large number of such broken sticks, with split points (knots) at locations $\kappa_1, \ldots, \kappa_K$. This leads to *the truncated linear spline basis*

$$1, x, (x - \kappa_1)_+, \ldots, (x - \kappa_K)_+$$

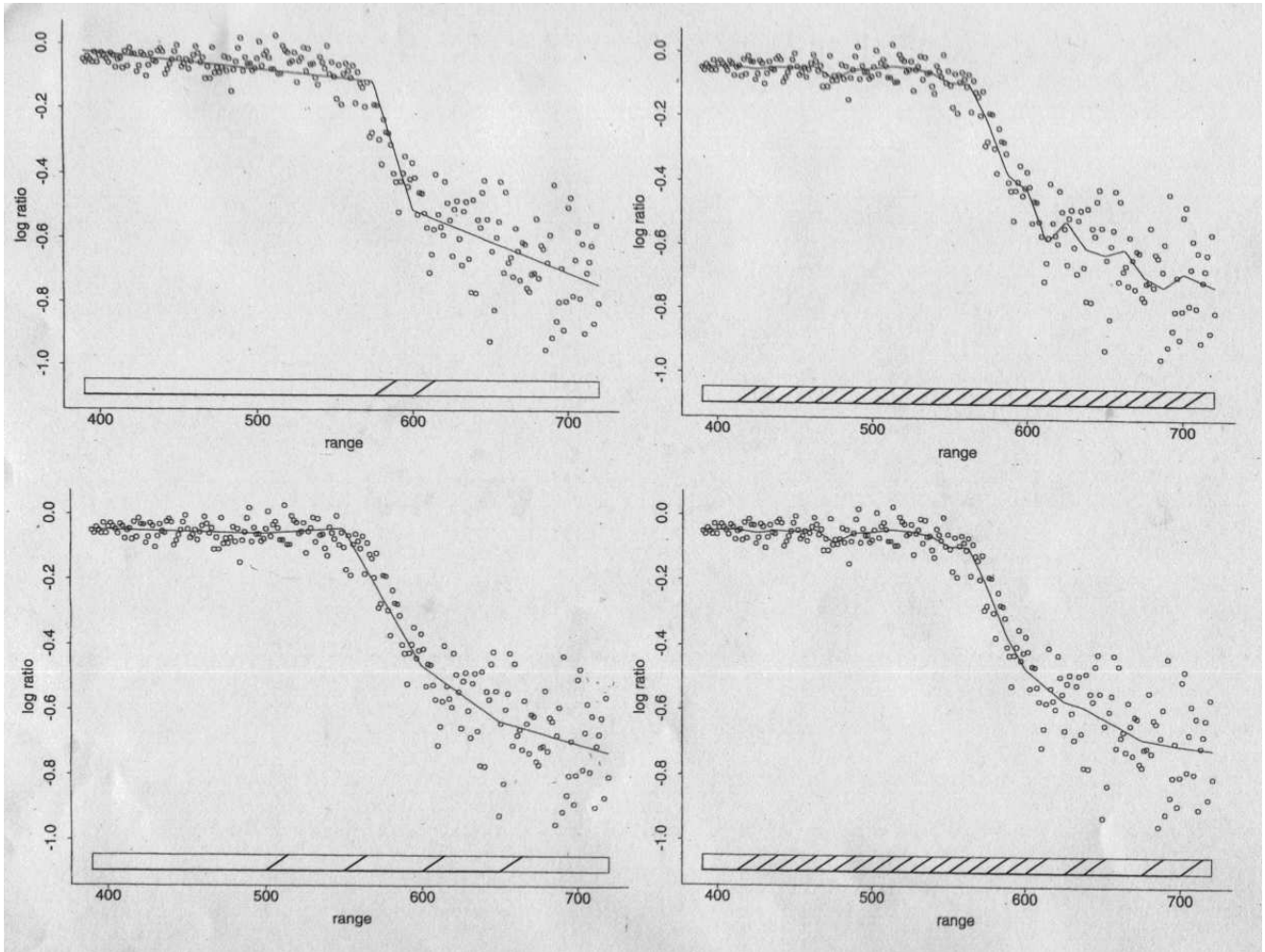
and the corresponding spline model for m:

$$m(x) = \beta_0 + \beta_1 x + \sum_{k=1}^{K} b_k (x - \kappa_k)_+ \tag{8}$$

- The crucial point is then the choice of the number and locations of the knots.

Knot selection for linear splines

- LIDAR data [RWC p. 62–64].
- The horizontal bar at the bottom indicates the knot positions.



Knot selection for linear splines (cont.)

- The more knots we use, the more flexible is the fitted function and the smaller is the bias.
- However, a large number of knots leads to an increased variance and to overfitting (meaning that the curve is following small, apparently random, fluctuations).
- A possible remedy: Pruning, i.e. selectively deleting the knots (not very pleasant as a time-consuming trial-and error work)
- Hence, there is need for automatic knot selection tools. Candidates are here classical model selection criteria, as for example:
  - Cross-validation
  - Mallow's C_p
- Note: To look for all possible submodels of a model using K knots, one has to compare $\sum_{k=0}^{K} \binom{K}{k} = 2^K$ submodels!
- Computationally very cumbersome, though some stepwise search algorithms exist.

Knot selection for linear splines (cont.)

We are left with the following options:

- We can painstakenly ("automatic" or by hand) try to find the right knot positions κ_k, and the right number of knots K.
- Or we do not care too much about the κ_k and K, but we try to control in some way the *influence* of the knots (through the parameters b_k).
- There are two basic approaches to implement the latter idea:
  - Through penalization;
  - Through a mixed model or Bayesian approach.
- Before we look closer at these approaches, we introduce matrix notation, which enables us to work in a somewhat more general framework.

Linear splines in matrix notation

- $$\mathbf{X} = \begin{pmatrix} 1 & X_1 & (X_1 - \kappa_1)_+ & \dots & (X_1 - \kappa_p)_+ \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & X_n & (X_n - \kappa_1)_+ & \dots & (X_n - \kappa_p)_+ \end{pmatrix},$$

  $\boldsymbol{\beta} = (\beta_0, \beta_1, b_1, \dots, b_k)^T$, $\mathbf{y} = (y_1, \dots, y_n)^T$.

- Fitting criterion is

$$\text{minimize} \quad ||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 = (\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^{\mathbf{T}}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta}) \tag{9}$$

- Hence, we have usual least squares theory. Taking the derivative with respect to $\boldsymbol{\beta}$ gives the normal equations

$$2\mathbf{X}\mathbf{X}^T\boldsymbol{\beta} - 2\mathbf{X}\mathbf{y} = 0$$

Linear splines in matrix notation (cont.)

- Parameter estimates are

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

- Fitted values are

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \equiv \mathbf{L}\mathbf{y}$$

- The hat matrix is

$$\mathbf{L} = \mathbf{X}(\mathbf{X}^\mathbf{T}\mathbf{X})^{-\mathbf{1}}\mathbf{X}^\mathbf{T}.$$

- Degrees of freedom used

$$df_{\text{fit}} = \text{tr}(\mathbf{L}) = K + 2$$

Penalization

- We want to control in some way the coefficients b_k, $k = 1 \ldots, K$ of the truncated polynomials, such that they are kept as small as possible (hence, virtually eleminating superfluous knots), but are still able resolve the complexity of the underlying function m.
- The idea how to do this is via a roughness penalty on the sum of squared coefficients $b_1^2 + \ldots b_k^2$: Instead of minimizing (9), we minimize for some $\lambda > 0$

$$||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda(b_1^2 + \ldots b_K^2)$$

which can be written as

$$||\mathbf{y} - \mathbf{X}\boldsymbol{\beta}||^2 + \lambda\boldsymbol{\beta}^T\mathbf{D}\boldsymbol{\beta}$$

with $\mathbf{D} = \text{diag}(\mathbf{0}, \mathbf{0}, \mathbf{1_K})$

Penalization (cont.)

- λ plays the role of a smoothing parameter similar to the bandwidth h in kernel regression:
  - For $\lambda = 0$, we have the usual unpenalized basis function approximation of m ("no smoothing")
  - For $\lambda \longrightarrow \infty$, the $b_1, \ldots, b_K$ will be shrunk to 0. Hence, we remain with β_0 and β_1 (which are not penalized), and the resulting fit will be a straight line ("maximal smoothing").
- Parameter estimates:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y} \tag{10}$$

- Fitted values:

$$\hat{\mathbf{y}} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D})^{-1}\mathbf{X}^T\mathbf{y} \tag{11}$$

- Smoother (Hat) matrix:

$$\mathbf{L} = \mathbf{X}(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{D})^{-1}\mathbf{X}^T \tag{12}$$

Smoothing splines

- A special (but important) case of penalization-based smoothing methods are smoothing splines.
- Here, a knot κ_k is positioned *at every observation* X_i, giving $K = n$ knots.
- One tries to find a function m that minimizes

$$\sum_{j=1}^{n}(Y_i - m(X_i))^2 + \lambda \int (m''(u))^2 \, du$$

- The penalization is clearly necessary here as otherwise $\hat{m}(x_i) = Y_i$, $i = 1, \ldots n$, which would mean that we had interpolated (most seriously overfitted!) the data.
- This form of penalization makes the interpretation as a roughness penalty evident: We penalize large second derivatives, i.e. large curvature, of the regression function.

Smoothing splines (cont.)

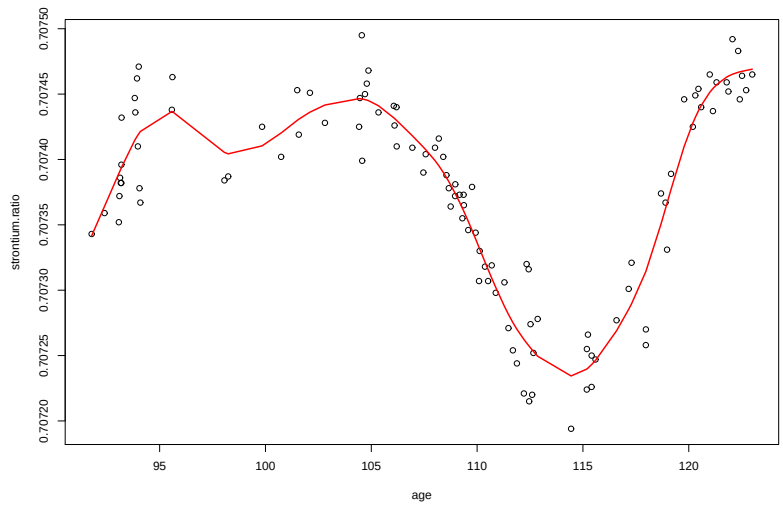
- One can show theoretically ([GS], p. 15ff) that the solution to this problem is a natural cubic spline, i.e. a string of polynomials of third degree such that the second derivatives are continuous at the knot locations, and the second and third derivatives are zero at the boundary knots.
- Then, in the notation used before,
  - $\boldsymbol{\beta} = (m(X_1), \ldots, m(X_n)) \equiv \mathbf{m}$,
  - $\mathbf{X} = \mathbf{I}_n$,
  - $\mathbf{D}$ is a rather complex penalty matrix that we do not display here (e.g., [FT, p. 154]).
- Hence, we have from (10), (11)

$$\hat{\mathbf{m}} = \hat{\mathbf{y}} = \hat{\boldsymbol{\beta}} = (\mathbf{I}_n + \lambda \mathbf{D})^{-1} \mathbf{y}$$

Smoothing splines (cont.)

- Fossil data with smoothing splines:

```
> library(pspline)
> fossil.ssp<- smooth.spline(age, strontium.ratio)
> fossil.ssp
Smoothing Parameter spar= 0.8190292 lambda= 5.905486e-05 (13
iterations)
Equivalent Degrees of Freedom (Df): 13.10510
Penalized Criterion: 5.782872e-08
GCV: 7.10339e-10
> plot(age, strontium.ratio)
> lines(fossil.ssp$x, fossil.ssp$y, col=2)
```



Basis function systems

- Smoothing splines become computationally infeasible if the sample size n is large (as $\boldsymbol{\beta} \in \mathbb{R}^n$).
- Hence, for large n, we really need to use a basis, with a relatively small number of knots.
- We have seen already the linear spline basis $1, x, (x - \kappa_1)_+, \ldots, (x - \kappa_K)_+$
- This can, in principle, be replaced by any general basis $\phi_1(x), \ldots, \phi_D(x)$ (often $\phi_1(x) \equiv 1$) such that we have the model

$$m(x) = \sum_{j=1}^{D} \beta_j \phi_j(x).$$

- The $n \times D$ design matrix $\mathbf{X}$ then consists of the entries $\phi_j(X_i)_{[1 \le i \le n, 1 \le j \le D]}$.

Basis function systems (cont.)

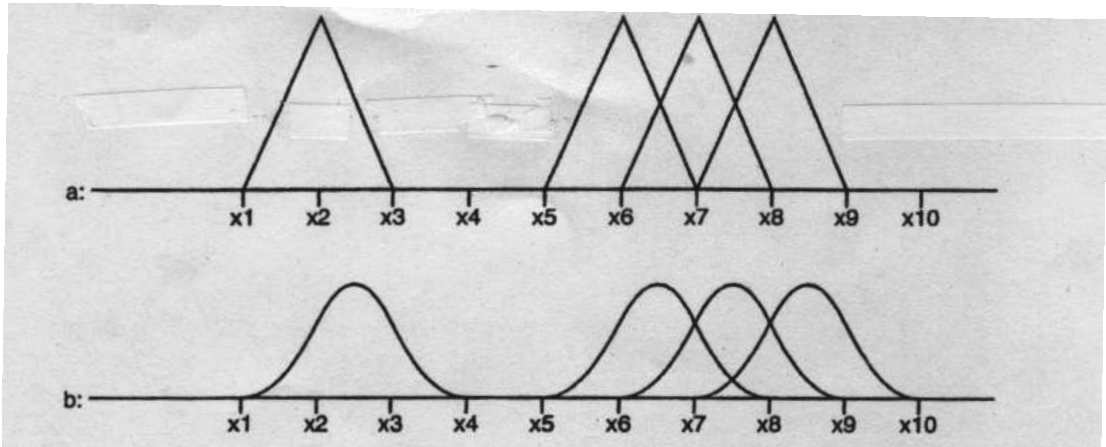
Examples for alternative basis function systems are

- Truncated power series
$$m(x) = \beta_0 + \beta_1 x + \ldots \beta_p x^p + \sum_{k=1}^{K} b_k (x - \kappa_k)_+^p$$
- Radial basis functions ("thin plate splines"):
$$m(x) = \sum_{j=0}^{m} \beta_j x^j + \sum_{k=1}^{K} u_k |x - \kappa_k|^{2m-1}, m = 1, 2, 3....$$
- Polynomial $\phi_j(x) = (x - \omega)^j$ and exponential $\phi_j(x) = e^{\lambda_j x}$ bases.
- The Fourier series $1, \sin(\omega x), \cos(\omega x), \sin(2\omega x), \cos(2\omega x), \ldots$.
- Wavelets
- Others . . . See [RS], p. 43ff.
- B-Splines.

B-Splines

A B-Spline of degree d is constructed as follows :

- It consists of $d+1$ polynomial pieces, each of degree d.
- The polynomial pieces join at d inner knots.
- At the joining points, derivatives up to order $d-1$ are continuous.
- The B-Spline is positive on a domain spanned by $d+2$ knots; elsewehere it is zero.
- Except at the boundaries, it overlaps with $2d$ polynomial pieces of its neighbors.
- At a given x, $d+1$ B-Splines are nonzero.



Illustrations of one isolated B-spline and several overlapping ones (a) *degree* 1; (b) *degree* 2.

B-Splines (cont.)

- An example for a B-Spline basis constructed of B-Splines of degree 3 is given below [RS, p. 50]

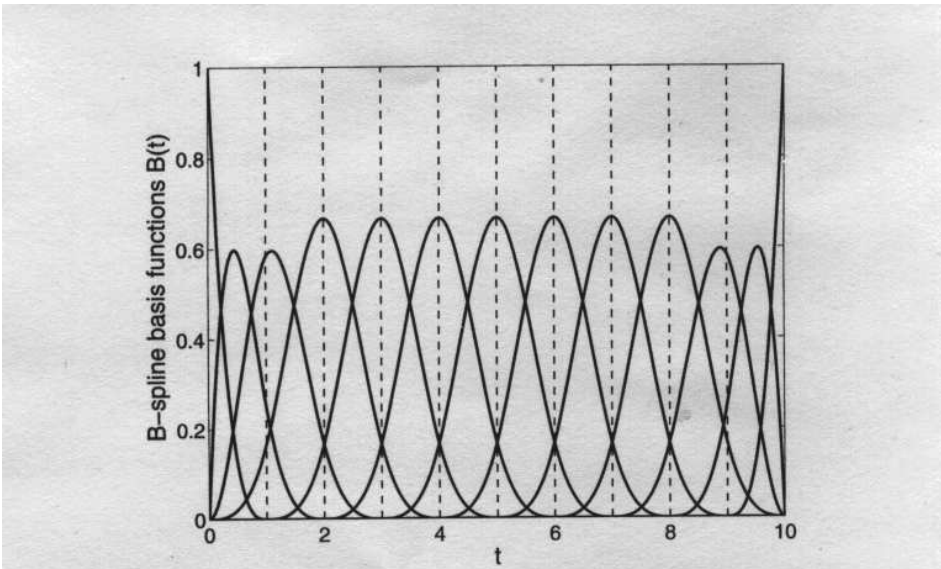


Figure 3.5. The thirteen basis functions defining an order four spline with nine interior knots, shown as vertical dashed lines.

- A B-spline of degree three is also called an order four spline - order is the number of parameters of one B-spline.

B-Splines (cont.)

Technically, a B-Spline basis is constructed as follows:

- Devide the domain $[x_{min}, x_{max}]$ into m' intervals using $m' + 1$ knots.
- Each interval will then be covered by $d + 1$ B-Splines of degree d.
- Add additional knots to the outermost left and right knots such that the condition above is also fulfilled near the boundaries, this implies that a total number of knots is $m' + 2d + 1$ (Remark: Any point where any B-Spline begins or end is a knot. If two B-Splines begin at the same x, then there are two knots at x.)
- The number of B-Splines in the regression is then $m = m' + d$.
- For a given d, denote $B_j(x)$ the value of the $j - th$ B-spline, $j = 1, \ldots, m + d$.
- A B-Spline basis can be generated automatically e.g. by using function `bs` in package **splines**.

B-Splines (cont.)

- Given a set of n data points $(X_1, Y_1), \ldots, (X_n, Y_n)$, the function $m(x)$ relating predictor and response is estimated as

$$\hat{m}(x) = \sum_{\ell=1}^{m} \hat{\alpha}_\ell B_\ell(x) \equiv \mathbf{B}(x)^T \hat{\boldsymbol{\alpha}}$$

where $\hat{\boldsymbol{\alpha}} = (\hat{\alpha}_1, \ldots, \hat{\alpha}_m)^T$ is obtained by minimizing the objective function

$$\begin{aligned} S = \sum_{i=1}^{n} \left\{ Y_i - \mathbf{B}(X_i)^T \boldsymbol{\alpha} \right\}^2 &\equiv (\mathbf{y} - \mathbf{B}\boldsymbol{\alpha})^T (\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}) \equiv \\ &\equiv ||\mathbf{y} - \mathbf{B}\boldsymbol{\alpha}||^2 \end{aligned}$$

with $\mathbf{B} = B_\ell(X_i)_{1 \le i \le n, 1 \le \ell \le m}$.

P-Splines

- Now again, let the number of knots be large (to capture all relevant features of the curve). Similarly as for smoothing splines, one could penalize the second derivative (O'Sullivan, 1986)

$$S = \sum_{i=1}^{n} \left\{ Y_i - \sum_{\ell=1}^{m} \alpha_\ell B_\ell(x_i) \right\}^2 + \lambda \int \left\{ \sum_{\ell=1}^{m} \alpha_\ell B_\ell''(x) \right\}^2 dx,$$

- Minimization of that leads to rather complex mathematics
- Eilers and Marx (1996) proposed a simple alternative penalization. Idea: The resulting curve will be smooth, if adjacent coefficents $\alpha_{\ell-1}$, α_ℓ do not differ greatly. This could be controlled by penalizing their differences $\triangle\alpha_\ell = \alpha_\ell - \alpha_{\ell-1}$, or – resembling second derivatives – second differences:

$$\triangle\alpha_\ell^2 = \triangle\alpha_\ell - \triangle\alpha_{\ell-1} = \alpha_\ell - 2\alpha_{\ell-1} + \alpha_{\ell-2}$$

P-Splines (cont.)

- This leads to the minimization problem

$$\begin{aligned} S &= \sum_{i=1}^{n} \left\{ Y_i - \sum_{\ell=1}^{m} \alpha_\ell B_\ell(x_i) \right\}^2 + \lambda \sum_{\ell=3}^{m} (\triangle^2 \alpha_\ell)^2 \\ &= ||\mathbf{y} - \boldsymbol{\alpha}\mathbf{B}(x)||^2 + \lambda \boldsymbol{\alpha}^T \mathbf{D}_2^T \mathbf{D}_2 \boldsymbol{\alpha} \end{aligned}$$

  with $\mathbf{D}_2 = \text{diag}(\triangle^2) \in \mathbb{R}^{m-2 \times m-2}$.

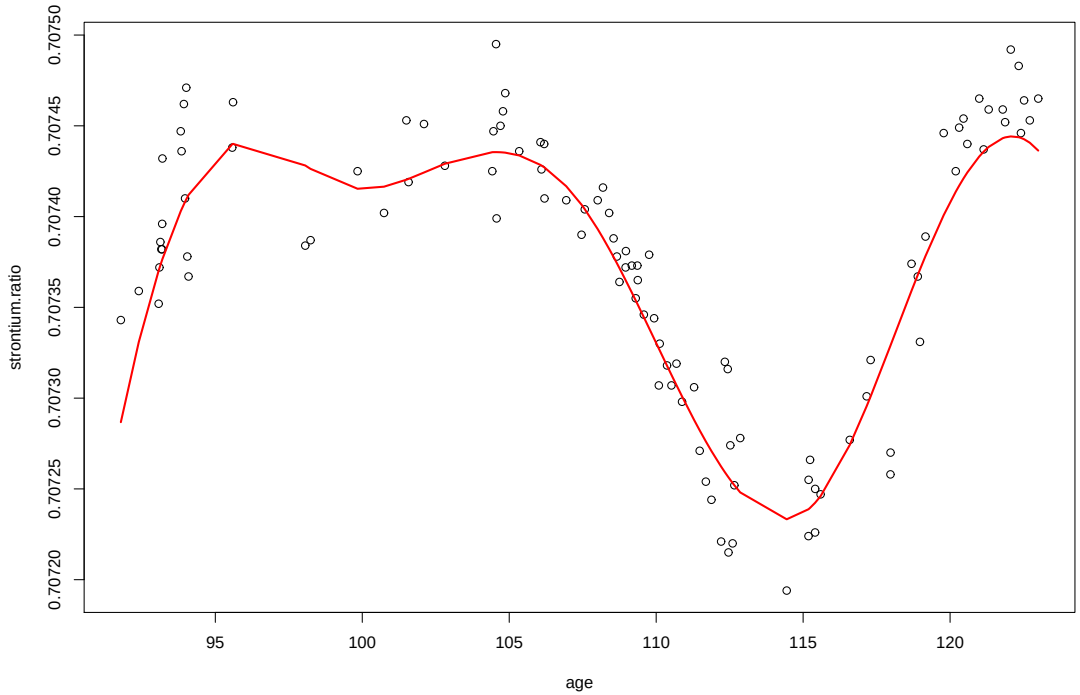
- According to (11), parameter estimates are then given by

$$\hat{\boldsymbol{\alpha}} = (\mathbf{B}^T\mathbf{B} + \lambda \mathbf{D}_2^T \mathbf{D}_2)^{-1} \mathbf{B}^T \mathbf{y}$$

- Other orders of differences are possible. Generally: Smaller difference orders lead to easy computations, but large difference orders give smoother fits. The order 2 is a good compromise.

P-Splines (cont.)

- P-spline fit for fossil data:




```
> source("http://www.stat.lsu.edu/faculty/marx/pspline.txt")
> fossil.pspline <- pspline.fit(strontium.ratio, age,
x.predicted=age, degree=3, order=2, lambda=0.15)
> plot(fossil); lines(age[order(age)],
fossil.pspline$summary.predicted[,("Predicted")][order(age)], col=2)
```

P-Spline Software

- Attention: R function names are here quite confusing:
  - The function `smooth.Pspline` in package **pspline** does actually *not* use B-Splines, but penalized smoothing splines.
  - The function `smooth.spline` in package **stats** *does* use B-Splines if the number of knots specified is smaller than n, and it also features penalization. However, it uses second derivatives instead of differences.
- P-Splines (in the Eilers/Marx sense) do not exist as an invokable function within an R package.
- However, they have been frequently employed as building blocks for other purposes (**mgcv, survival**)
- Further useful implementations of P-Splines do exist in conjunction with (partially) Bayesian methods, and we look at this later.

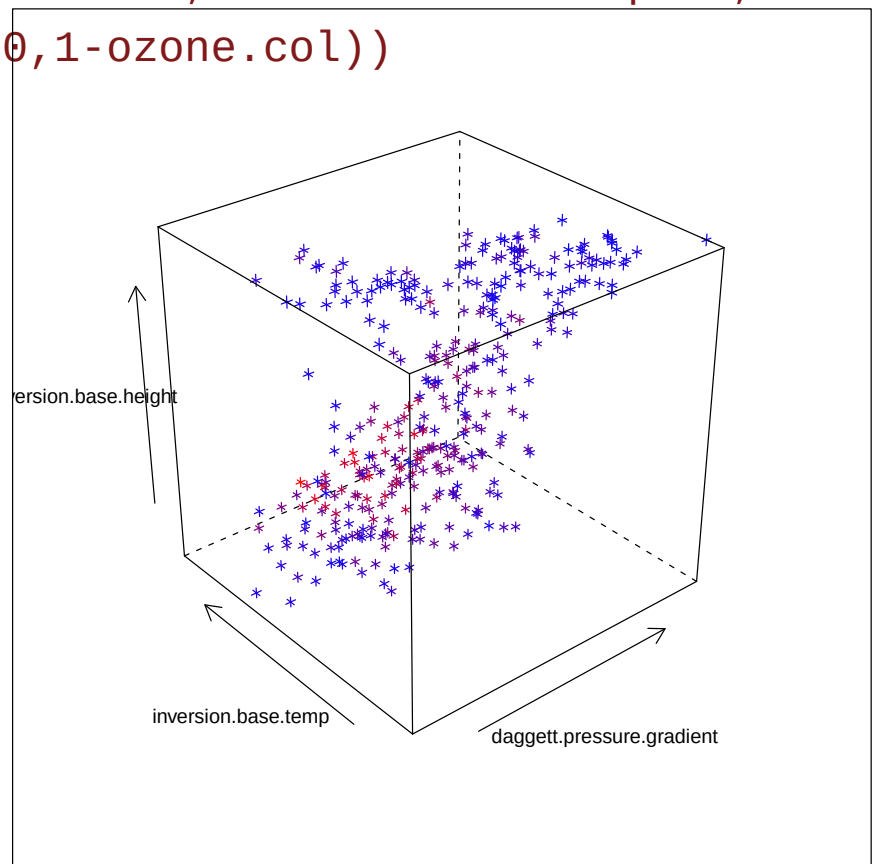
Multiple predictors

- Consider data on athmospheric ozone concentration in the Los Angeles basin (Breiman and Friedman, 1985)
- We are having $n = 345$ observations on the following four variables:
  - `daggett.pressure.gradient` $\equiv g$: pressure gradient at Daggett, California, in mmHg,
  - `inversion.base.height` $\equiv h$: inversion base height, in feet.
  - `inversion.base.temp` $\equiv t$: inversion base temperature, in degrees Fahrenheit.
  - `ozone.level` $\equiv y$: daily ozone concentration (response), in ppm.

Multiple predictors (cont.)

- We attempt to visualize the data (red= high ozone level):

```
> data(calif.air.poll, package="SemiPar"); require(lattice)
> attach(callf.air.poll)
> ozone.col<- ozone.level/max(ozone.level)
> cloud(inversion.base.temp~inversion.base.height +
daggett.pressure.gradient, data=calif.air.poll,
col=rgb(ozone.col,0,1-ozone.col))
```



Additive Models

- Due to the curse of dimensionality, fitting a full interaction model
$$y = m(g, h, t) + \varepsilon$$
is difficult!
- A useful simplification is the additive model
$$y = \alpha + m_1(g) + m_2(h) + m_3(t) + \varepsilon$$
(The intercept α could in principle be absorbed into any of the m_j's, but then they are not uniquely defined any more).
- We have to estimate the three functions m_1, m_2, and m_3 simultaneously, using smoothers with associated smoother matrices, say, $\boldsymbol{L}_1, \boldsymbol{L}_2$, and $\boldsymbol{L}_3$ (for instance, penalized smoothing splines).
- The idea is simple: Note that $y - \alpha - m_1(g) - m_2(h) = m_3(t) + \epsilon$, so given α, m_1 and m_2, we can fit the left hand side of this equation versus t, yielding an estimate of m_3. This process, called backfitting, is iterated until convergence.

Backfitting

- Formally, assume we are having p predictors X_j, $j = 1, \ldots, p$, and corresponding observations $X_{ij}, i = 1, \ldots, n$. Consider a a model of type $y = \alpha + \sum_{j=1}^{p} m_j(x_j)$, and let us denote $\mathbf{m}_j = (m_j(X_{1j}), \ldots, m_j(X_{nj}))^T$. Then the backfitting algorithm [HT, p. 91; FG, p. 266] proceeds as follows:

  (i) Initialize: $\alpha = \bar{y}, \mathbf{m}_j = \mathbf{m}_j^0$

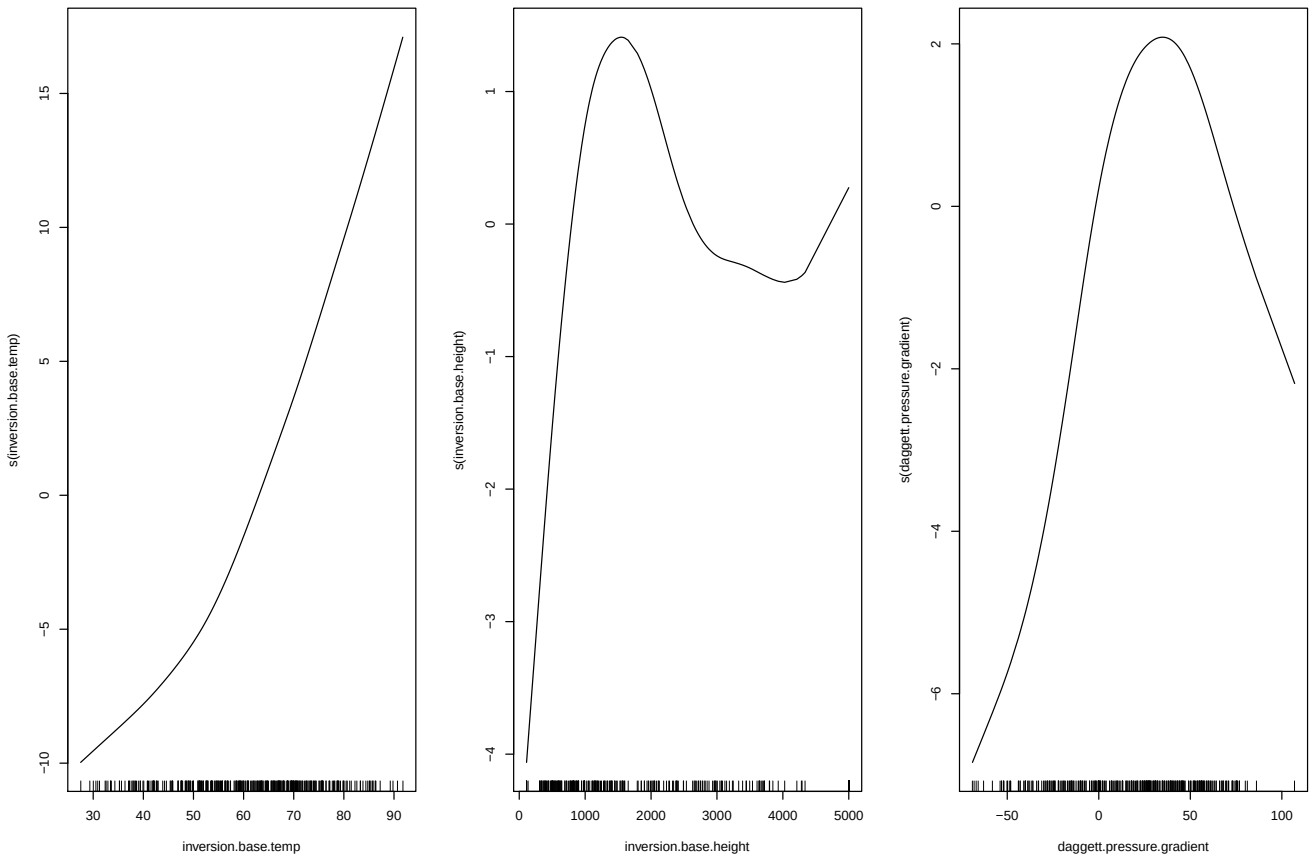
  (ii) Cycle: For $j = 1, \ldots, p, 1, \ldots, p, \ldots$

$$\mathbf{m}_j = \mathbf{L}_j(\mathbf{y} - \alpha - \sum_{k \neq j} \mathbf{m}_j)$$

  (iii) Continue (ii) until the individual functions don't change.

Additive model for ozone data

```
> library(gam)
> ozone.gam<- gam(ozone.level~s(inversion.base.temp)+
s(inversion.base.height) + s(daggett.pressure.gradient),
data=calif.air.poll)
> par(mfrow=c(1,3)); plot(ozone.gam)
```

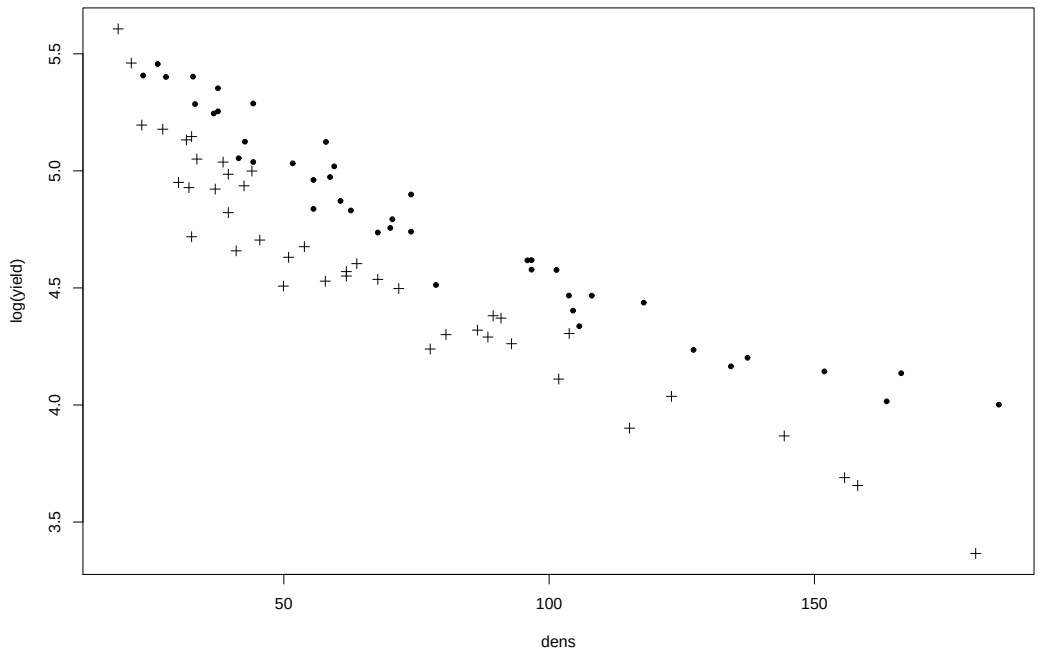


Additive model for ozone data (cont.)

```
>  summary(ozone.gam)

(Dispersion Parameter for gaussian family taken to be 18.6155)

    Null Deviance: 21854.51 on 344 degrees of freedom
Residual Deviance: 6180.345 on 332.0003 degrees of freedom
AIC: 2002.594

DF for Terms and F-values for Nonparametric Effects

                              Df Npar Df  Npar F     Pr(F)
(Intercept)                    1
s(inversion.base.temp)         1       3 10.8124 8.528e-07 ***
s(inversion.base.height)       1       3  9.0253 9.231e-06 ***
s(daggett.pressure.gradient)   1       3 16.6828 4.055e-10 ***
```

Simple semiparametric models

- Onions data: contains 84 observations from an experiment involving the production of white Spanish onions in two South Australian locations ("+" and "•").
- Plotted is log(onion `yield`) in grammes per plant vs. areal density of plants (plants per square metre):




```
> data(onions, package=''SemiPar''); attach(onions)
> plot(dens,log(yield), pch=ifelse(location==1, 3,20))
```

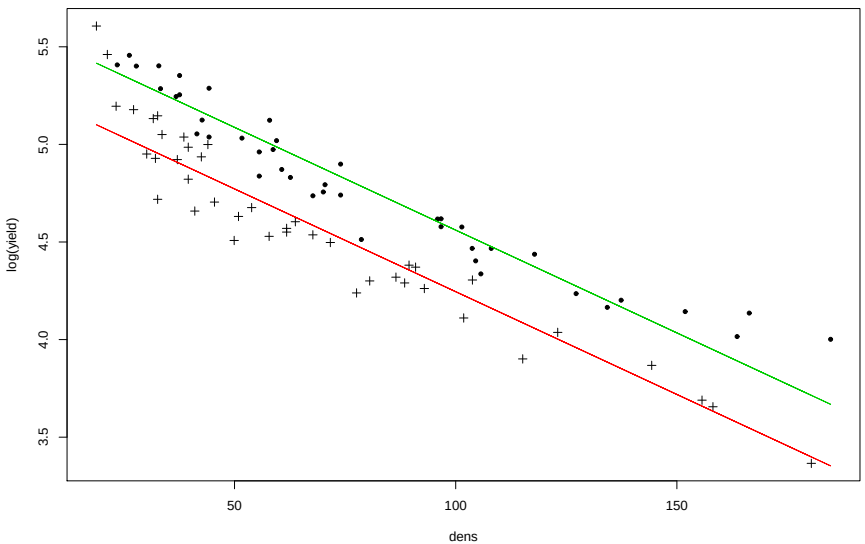
Simple semiparametric Models (cont.)

We try firstly a parametric additive model:

$$\log(\texttt{yield}_i) = \beta_0 + \beta_1 \texttt{location}_i + \beta_2 \texttt{dens}_i$$

where

$$\texttt{location}_i = \begin{cases} 0 & \text{if } i\text{th measuremeant from Virginia} \\ 1 & \text{if } i\text{th measuremeant from Purnong Landing} \end{cases}$$



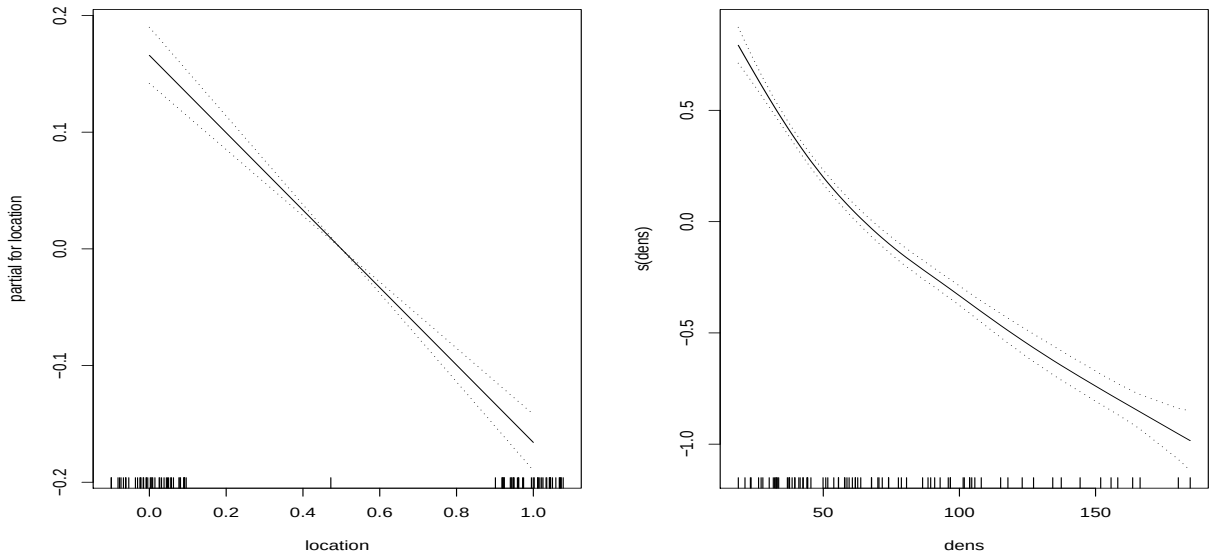

```
> onions.lm<- lm(log(yield)~
dens+location)
> print(onions.lm)
Coefficients:
  (Intercept)   dens       location
  5.61383       -0.01053   -0.31543
```

Simple semiparametric Models (cont.)

- At the first glance, a good fit, but close inspection reveals that there seems to be some curvature.
- This suggests to fit a semiparametric model

$$\log(\texttt{yield}_i) = \beta_0 + \beta_1 \texttt{location}_i + m(\texttt{dens}_i)$$

- Standard `gam` output with pointwise confidence bands:




```
> onions1.gam<- gam(log(yield)~location +m(dens))
> plot(onions1.gam, se=TRUE)
```

Fitting Semiparametric Models

- Model: $Y_i = \mathbf{t}_i^T \boldsymbol{\beta} + m(X_i) + \epsilon_i$, $\mathbf{t}_i \in \mathbb{R}^p$, $X_i \in \mathbb{R}$, $i = 1, \ldots, n$.
- With design matrix $\mathbf{T} = (\mathbf{t}_1, \ldots, \mathbf{t}_n)^T$ for the parametric part, this can be seen as a model with two "smoothers"
  - $\mathbf{L}_1 = \mathbf{T}(\mathbf{T}^T\mathbf{T})^{-1}\mathbf{T}^T$ (this is just the usual hat matrix for the linear model!)
  - $\mathbf{L}_2$ (smoother matrix for $m(X)$)
- Iterative estimation:

$$\begin{aligned} \boldsymbol{\beta} &= (\mathbf{T}^T\mathbf{T})^{-1}\mathbf{X}^T(\mathbf{y} - \mathbf{m}) \\ \mathbf{m} &= \mathbf{L}_2(\mathbf{y} - \mathbf{T}\boldsymbol{\beta}) \end{aligned}$$

Plugging the latter into the former, this has an explicit solution

$$\hat{\boldsymbol{\beta}} = \{\mathbf{T}^T(\mathbf{I} - \mathbf{L}_2)\mathbf{T}\}^{-1}\mathbf{T}^T(\mathbf{I} - \mathbf{L}_2)\mathbf{y}.$$

implying that no backfitting is needed here! (Green, Jennison, & Seheult, 1985)

Generalized additive models

- A generalized additive model is a model of type

$$\mu \equiv E(Y) = h(\eta) \equiv h\left(\beta_0 + \mathbf{t}^T\boldsymbol{\beta} + \sum_{j=1}^{q} m_j(X_j)\right)$$

  where the density $f(Y)$ is a member of the exponential family (the expectation and the density have to be seen conditional on $\mathbf{t}, X_1, \ldots, X_q$).
- $h(\cdot)$ is called the response function and $h^{-1}(\cdot)$ the link function.
- Model fitting happens iteratively using weighting least squares (Fisher-Scoring), either in conjunction with backfitting or using a mixed model approach (later today).
- In the important special case $q = 0$ we have a generalized linear model.

Generalized additive models (cont.)

- For the choice of the exponential family and link function, the following general (but not obligatory) rules apply:

Response type	Exp. family	$h(\eta)$
continuous	Normal	η
cont., positive	Gamma	η^{-1} or $\exp(\eta)$
count data	Poisson	$\exp(\eta)$
0-1 data	Bernoulli	$\exp(\eta)/(1+\exp(\eta))$
proportions	Binomial	$\exp(\eta)/(1+\exp(\eta))$

- The link function printed in red corresponds to the so-called natural link, which leads to models with convenient mathematical and statistical properties [FT, p. 20].

(Generalized) additive model software

- A wide range of alternative response distributions are supported by the R function `gam` in package **gam**. This function uses backfitting, and either local polynomials (of degree 1 or 2) or smoothing splines as smoothers.
- There exists an alternative `gam` implementation in R package **mgcv** using a penalized basis function approach, which features automatic smoothing parameter selection via generalized cross-validation.
- **SemiPar** supports Gaussian, Poisson, and Binomial response distributions.

Example: Respiratory deaths

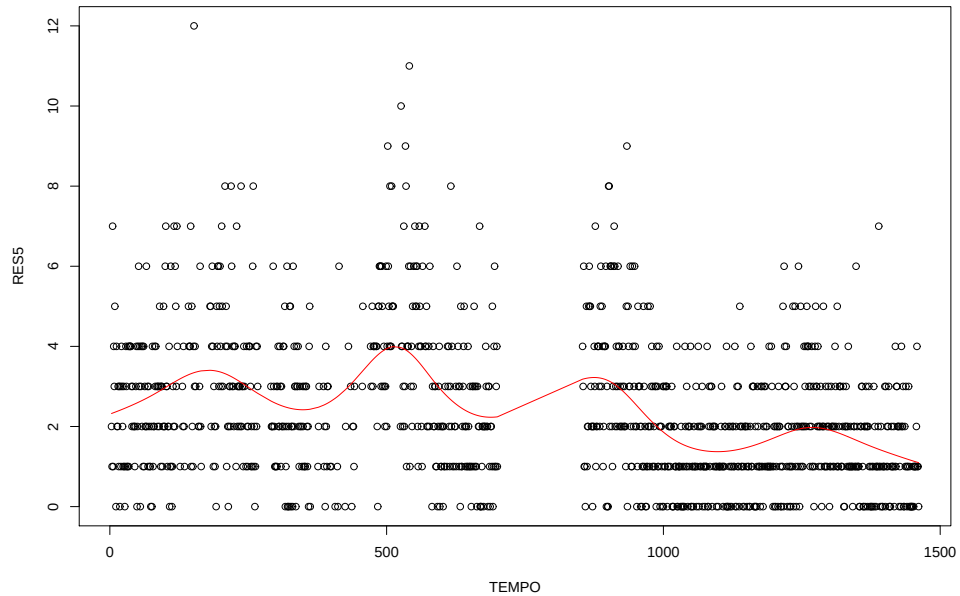
- Respiratory Deaths in São Paulo, Brazil, 1994-1997:
- The response variable is the number of daily respiratory deaths of children under five in the city of São Paulo.
- As explanatory variables we have daily measurements of humidity, temperature, number of deaths due to other reasons, and a variety of pollutant concentrations.
- Sample size excluding observations with missing values: $n = 1128$.

Example: Respiratory deaths (cont.)

TEMPO	Enumeration of days	
SEGUNDA	Indicator for Monday	1 : Monday
		0 : not Monday
TERCA	Analogous indicators for	Tuesday,
QUARTA		Wednesday,
QUINTA		Thursday,
SEXTA		Friday,
SABADO		Saturday
OTHRES5	Number of other death causes than	respiratory.
TMIN.2	Two-day-lag of minimum temperature	in $^{\circ}C$
UMID	Relative humidity	in %.
PMME.2	Two-day-lag of concentration of PM_{10}	in $\mu g/m^3$
SO2ME.2	Two-day-lag of of SO_2	in $\mu g/m^3$.
COME.2	Two-day-lag of concentration of CO	in ppm.
O3ME.2	Two-day-lag of concentration of O_3	in $\mu g/m^3$.
RES5	Number of respiratory deaths.	

Example: Respiratory deaths (cont.)

- The response is count data — usually this is modelled by a Poisson model.
- For explanatory purposes we look closer at the number of deaths over time:




```
> library(gam)
> gam.resp2<- gam(RES5~s(TEMPO,df=10),
family="poisson"(link=log),data=spdata)
> plot(TEMPO,RES5); lines(TEMPO,gam.resp2$fitted,col=2)
```

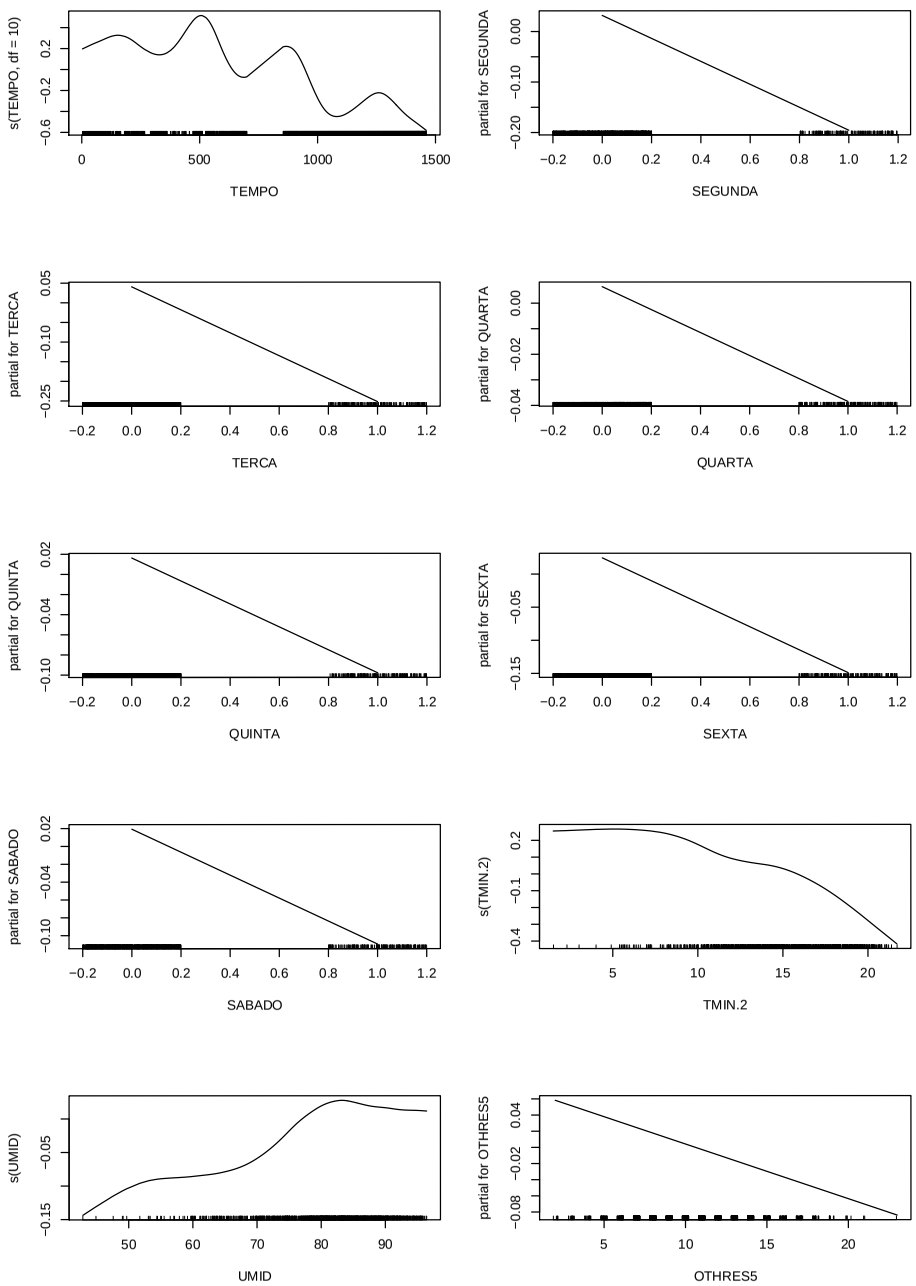
Example: Respiratory deaths (cont.)

- There is a clear seasonal trend which we model nonparametrically
- We are fitting now the so-called 'Core-Model", as e.g. in Singer et al. (2002):

$$\begin{aligned}\eta &= \beta_0 + f_1(\text{TEMPO}) + f_2(\text{TMIN.2}) + f_3(\text{UMID}) + \\ &+ \beta_1 \cdot \text{SEGUNDA} + \ldots + \beta_6 \cdot \text{SABADO} + \beta_7 \cdot \text{OTHRES5}\end{aligned}$$

```
> gam.resp3<- gam(RES5~ s(TEMPO,df=10)+ SEGUNDA + TERCA + QUARTA +
QUINTA + SEXTA + SABADO + s(TMIN.2)+ s(UMID) + OTHRES5,
family="poisson"(link=log))
> plot(gam.resp3)
```

Example: Respiratory deaths (cont.)



Example: Respiratory deaths (cont.)

- The usual stategy is then to add subsequently the pollutants to this model and see if they lead to a substantial decrease in deviance ($= -2 \log L + \text{const}$).
- For model comparion, one way is to look at AIC $= -2 \log L + 2 df_{\text{fit}}$ (the smaller, the better!)
- For example, adding a smooth term for `SO2ME.2` decreases the deviance and AIC from

```
Residual Deviance: 1325.020 on 1099.000 degrees of freedom
AIC: 4053.545
```

to

```
Residual Deviance: 1303.354 on 1095.000 degrees of freedom
AIC: 4039.879
```

which can be checked using the `summary` function.

Example: Respiratory deaths (cont.)

Assume we have decided for the following model:

```
> gam.resp5 <- gam(RES5~s(TEMPO) + SEGUNDA + TERCA + QUARTA + QUINTA
+ SEXTA + SABADO + s(TMIN.2) + UMID + OTHRES5 + s(SO2ME.2) +
s(O3ME.2), family=poisson(link=log))
```

We predict the number of respiratory deaths on Monday, the 3rd Jan 1994, with 16 deaths from other sources, min. temperature of 16.7 C, humidity 86.78%, and two-day-lag SO_2 and O_3 concentrations of 5.9 and 101.9 $\mu g/m^3$, respectively, through

```
> predict(gam.resp5, newdata=data.frame(TEMPO=3, SEGUNDA=1, TERCA=0,
QUARTA = 0, QUINTA=0, SEXTA=0, SABADO=0, OTHRES5=16, TMIN.2=16.7,
UMID=86.78, SO2ME.2= 5.9, O3ME.2= 101.9))
0.9532295
```

That is, the predicted value is $\exp(0.9532295) = 2.594074$ (compared to the observed value 2).

Section 3: (Partially) Bayesian smoothing methods

Scope of this section:

- Mixed model approach to smoothing;
- Geoadditive models,
- Random effect models;
- Parametric and nonparametric Bayesian inference;
- Bayesian nonparametric density estimation;
- Bayesian P-Splines;
- Geoadditive models based on maps.

Mixed model representation of linear splines

- Consider the model formula

$$Y_i = \beta_0 + \beta_1 X_i + \sum_{k=0}^{K} u_k (X_i - \kappa_k)_+ + \epsilon_i \tag{21}$$

- The idea is to make a distributional assumption about $\mathbf{u} = (u_1, \ldots, u_K)^T$, which is often taken to be

$$\mathbf{u} \sim N(0, \sigma_u^2 \mathbf{I})$$

- Thus, the coefficients are hampered from taking arbitrary values, which will ensure an increased smoothness compared to the unrestricted model.

Mixed model representation of linear splines (cont.)

- Denoting $\boldsymbol{\beta} = (\beta_0, \beta_1)^T$,

$$\mathbf{X} = \begin{pmatrix} 1 & X_1 \\ \vdots & \vdots \\ 1 & X_n \end{pmatrix}, \quad \mathbf{Z} = \begin{pmatrix} (X_1 - \kappa_1)_+ & \cdots & (X_1 - \kappa_K)_+ \\ \vdots & \ddots & \vdots \\ (X_n - \kappa_1)_+ & \cdots & (X_n - \kappa_K)_+ \end{pmatrix}$$

the model (21) can be written as a mixed model

$$\boldsymbol{y} = \mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\mathbf{u} + \boldsymbol{\epsilon}, \quad \text{Cov} \begin{bmatrix} \mathbf{u} \\ \boldsymbol{\epsilon} \end{bmatrix} = \begin{bmatrix} \sigma_u^2 \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \sigma_\varepsilon^2 \mathbf{I} \end{bmatrix} \tag{22}$$

- A mixed model can be seen as a *partially Bayesian model* - one parameter vector ($\mathbf{u}$) carries a distributional assumption, but the other one ($\boldsymbol{\beta}$) doesn't.

Parameter estimation in mixed models

- Rewrite (22) as

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}^* \text{ , where } \quad \boldsymbol{\epsilon}^* = \mathbf{Z}\mathbf{u} + \boldsymbol{\epsilon}$$

- This is just a linear model with correlated errors, since

$$\text{Cov}(\boldsymbol{\epsilon}^*) \equiv \mathbf{V} = \sigma_u^2 \mathbf{Z}\mathbf{Z}^T + \sigma_\varepsilon^2 \mathbf{I}$$

- The estimator $\tilde{\boldsymbol{\beta}}$ for $\boldsymbol{\beta}$ is then obtained by *weighted least squares*. We obtain the so-called BLUPs (best linear unbiased predictors) for $\boldsymbol{\beta}$

$$\tilde{\boldsymbol{\beta}} = (\mathbf{X}^T \mathbf{V}^{-1} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{V}^{-1} \mathbf{y} \tag{23}$$

- and $\mathbf{u}$ (via *best prediction*, [RWC], p. 98ff),

$$\tilde{\mathbf{u}} = E(\mathbf{u}|\mathbf{y}) = \sigma_u^2 \mathbf{Z}^T \mathbf{V}^{-1} (\mathbf{y} - \mathbf{X}\tilde{\boldsymbol{\beta}}). \tag{24}$$

Mixed models and penalized smoothing

- With $\boldsymbol{y}|\boldsymbol{u} \sim N(\mathbf{X}\boldsymbol{\beta} + \mathbf{Z}\boldsymbol{u}, \sigma_\varepsilon^2\mathbf{I})$ and $\boldsymbol{u} \sim N(\mathbf{0}, \sigma_u^2\mathbf{I})$, the log-likelihood of $(\boldsymbol{y}, \boldsymbol{u})$ is given by

$$\log(f(\boldsymbol{y}, \boldsymbol{u})) = \log f(\boldsymbol{y}|\boldsymbol{u}) + \log f(\boldsymbol{u}) =$$

$$= \quad \mathsf{c} - n\log\sigma_\varepsilon - K\log\sigma_u - \frac{1}{2\sigma_\varepsilon^2}||\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{u}||^2 - \frac{1}{2\sigma_u^2}||\boldsymbol{u}||^2$$

- This shows how the concepts of random effects relates to that of penalization: The actual smoothing effect is achieved through penalizing high values of the u_k, $k = 1, \ldots, K$.
- Set $\lambda^2 = \sigma_\varepsilon^2/\sigma_u^2$. We arrive at the simple minimization problem (after multiplication with $2\sigma_\varepsilon^2$)

$$||\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta} - \mathbf{Z}\boldsymbol{u}||^2 + \lambda^2||\boldsymbol{u}||^2. \tag{25}$$

On-the-fly estimation of smoothing parameters

- The higher lambda, the more the coefficients of the linear spline basis are shrunk, and the smoother the fit will be.
- $\lambda = 0$ corresponds to the case "no smoothing", i.e. to the fixed effect model (8).
- In contrast to "classical" smoothing, where the smoothing parameter has to be *selected*, it can be *estimated* here as

$$\hat{\lambda} = \sqrt{\frac{\hat{\sigma}_\varepsilon^2}{\hat{\sigma}_u^2}}$$

within the mixed model framework – no need for smoothing parameter selection routines! This is a crucial advantage over the – otherwise equivalent – formulation as penalized splines.

Estimating the variance components

- We remain with the task of estimating the variance components σ_ϵ^2 and σ_u^2.
- This is possible via Maximum Likelihood (ML). As $\boldsymbol{y} \sim N(\mathbf{X}\boldsymbol{\beta}, \mathbf{V})$, the log-likelihood of $\boldsymbol{y}$ under this model is given by

$$\ell(\boldsymbol{\beta}, \mathbf{V}) = -\frac{1}{2}\left\{n\log(2\pi) + \log|\mathbf{V}| + (\boldsymbol{y} - \mathbf{X}\boldsymbol{\beta})^T\mathbf{V}^{-1}(\boldsymbol{y} - \boldsymbol{X}\beta)\right\}$$

- Substituting the formula (23) for $\boldsymbol{\beta}$ into the log-likelihood, one obtains the profile likelihood $\ell_P(\mathbf{V})$, which is a function only of $\mathbf{V}$, which in turn only depends on σ_u^2 and σ_ε^2. Maximization of the profile-likelihood with respect to σ_u^2 and σ_ε^2 gives the ML-estimates $\hat{\sigma}_u^2$ and $\hat{\sigma}_\varepsilon^2$ [RWC, p 111].
- Alternative: Bias-corrected version REML (REstricted Maximum Likelihood [RWC, p. 101]) $\ell_R = \ell - \frac{1}{2}\log|\mathbf{X}^T\mathbf{V}^{-1}\mathbf{X}|$.

Knot selection in mixed models

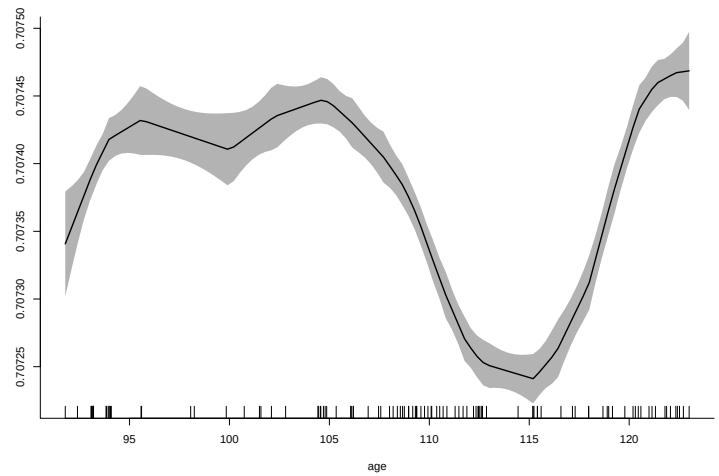
- Note again that this methodology does *not* do the choice of the number of knots K and the choice of the knot locations κ_k for us.
- However, due to the implicit penalization of knot parameters, the problem of knot choice/selection is now less relevant. The point is that one needs enough knots to resolve the underlying structure. Coefficients corresponding to superfluous knots will automatically be shrunk to small values. In fact, **SemiPar** uses

$$K = \max\left(\frac{n}{4}, 20\right)$$

with
$\kappa_k = \left(\frac{k+1}{K+2}\right)$ th sample quantile of the unique X_i, $k = 1 \ldots, K$.

Smoothing with SemiPar

- Fitting Fossil data using a linear spline basis with **SemiPar:**

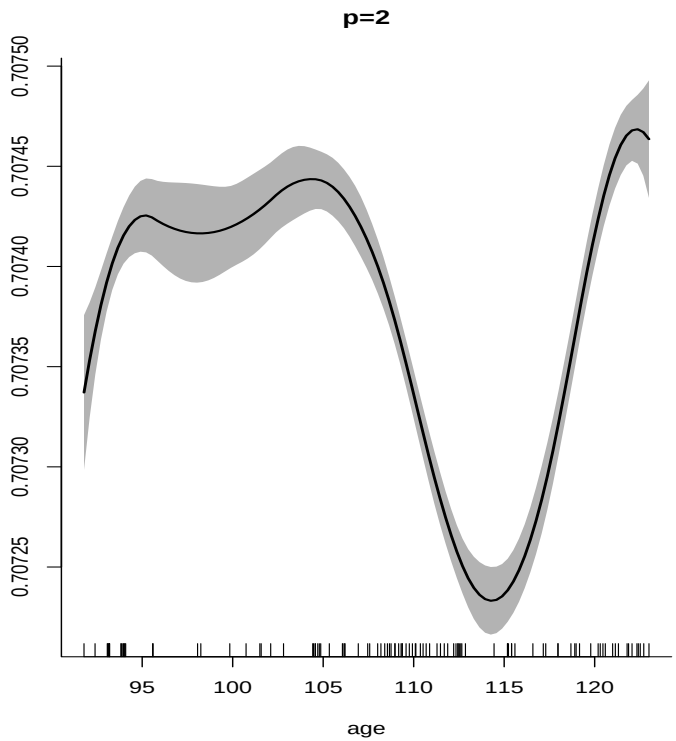


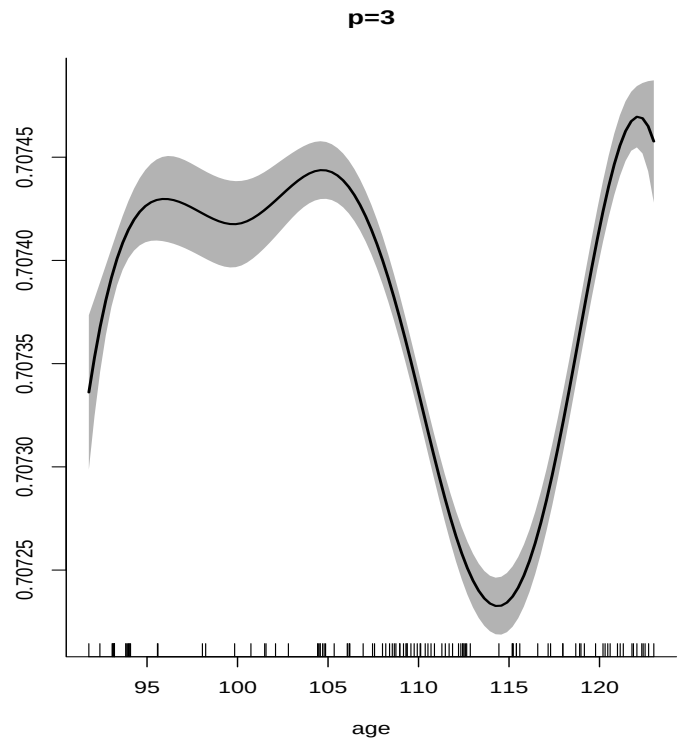

```
> library(SemiPar)
> fossil.fit1<- spm(strontium.ratio~ f(age,basis="trunc.poly",
degree=1))
> summary(fossil.fit1)
Summary for non-linear components:
           df     spar   knots
 f(age)  12.76   1.324     25
```

- looks okay, but still quite wiggly!

Smoothing with SemiPar (cont.)

- Truncated polynomial splines: Normally a degree $p = 2$ is sufficient to get a smooth fit.
- Attention: The default setting in **SemiPar** is $p = 1$.
- Fossil data:





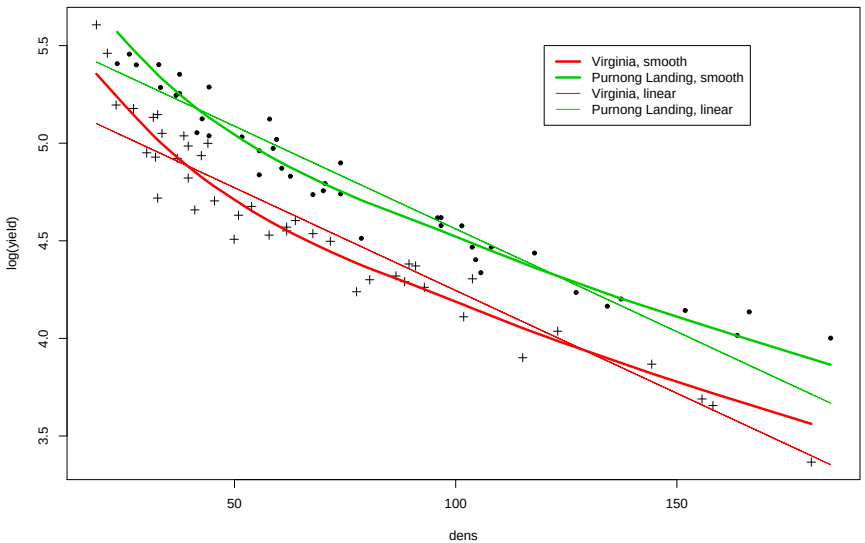
	df	spar	knots
f(age)	10.06	2.243	25

	df	spar	knots
f(age)	8.867	3.419	25

Multiple, additive, and semiparametric regression

- We have learned how to fit models of type $Y_i = m(X_i) + \epsilon_i$
- All such extensions can be conveniently handled in a mixed model framework by simply adding columns to $\mathbf{X}$ and $\mathbf{Z}$.
- For instance, for a semiparametric model
$$Y_i = \beta_1 T_i + m(X_i) + e_i$$
in linear spline representation for m, one has
$$\mathbf{X} = \begin{pmatrix} 1 & T_1 & X_1 \\ \vdots & \vdots & \vdots \\ 1 & T_n & X_n \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix},$$
$$\mathbf{Z} = \begin{pmatrix} (X_1 - \kappa_1)_+ & \cdots & (X_1 - \kappa_K)_+ \\ \vdots & \ddots & \vdots \\ (X_n - \kappa_1)_+ & \cdots & (X_n - \kappa_K)_+ \end{pmatrix}, \mathbf{u} = \begin{pmatrix} u_1 \\ \vdots \\ u_k \end{pmatrix}$$
- Model fitting as before — no backfitting necessary!

Model comparison




```
> onions1.spm <-
spm(log(yield) ~location+
f(dens, basis="trunc.poly"))
> summary(onions1.spm)
Summary for non-linear
components:
                df     spar   knots
 f(dens)   4.463   37.92     17
```

- We spend 4.463df instead of 1df to reveal a rather small curvature. Is this worth the effort?
- This would require a statistical test

$$H_0 : \beta_2 \text{dens} \quad \text{versus} \quad H_1 : m(\text{dens})$$

which means in a mixed model context to test

$$H_0 : \sigma_u^2 = 0 \quad \text{versus} \quad H_1 : \sigma_u^2 > 0$$

Model comparison (cont.)

- Ruppert et al. [RWC, p. 147] recommend to use a (restricted) likehood ratio test with test statistics

$$T = -2\{\ell_R(0, \hat{\sigma}_\varepsilon^2; \mathbf{y}) - \ell_R(\hat{\sigma}_u^2; \hat{\sigma}_\varepsilon^2; \mathbf{y})\} = \ldots = 35.90$$

- The asymptotic distribution of T under H_0 is rather complicated (roughly, it results in a mixture of χ^2 distributions [RWC, p. 106, 168].)
- $p-$ values are obtained by simulating from this mixture and counting the number (i.e. proportion) of times that the value $T = 35.90$ is exceeded. For these data, this gives $p = 7 \times 10^{-10}$, which is a surprisingly strong result. Hence, the nonparametric term is clearly significant!
- not implemented in **SemiPar**.

Geoadditive models

- Geoadditive offer the possibility to include spatial information into an additive model.
- There are two types of geoadditive models:
  1. Models using geographical information in terms of coordinates, e.g. longitude and latitude.
  2. Models using maps, wich are divided into certain districts.
- The extension of additive to geoadditive models of type 1. is straightforward in a mixed model framework and is implemented in **SemiPar**
- Geoadditive models of type 2 are supported in **BayesX**.
- We have already seen a very primitive form of a geoadditive model of type 1.: The bivariate local constant/ local linear fit to the model

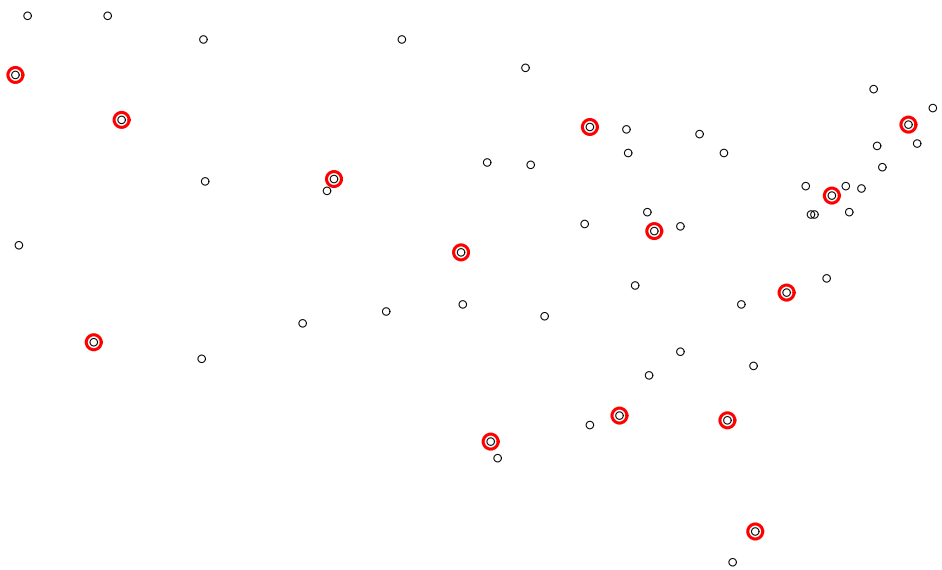
$$\texttt{min.temp} = \texttt{m}(\texttt{longitude}, \texttt{latitude})$$

for the US temperature data.

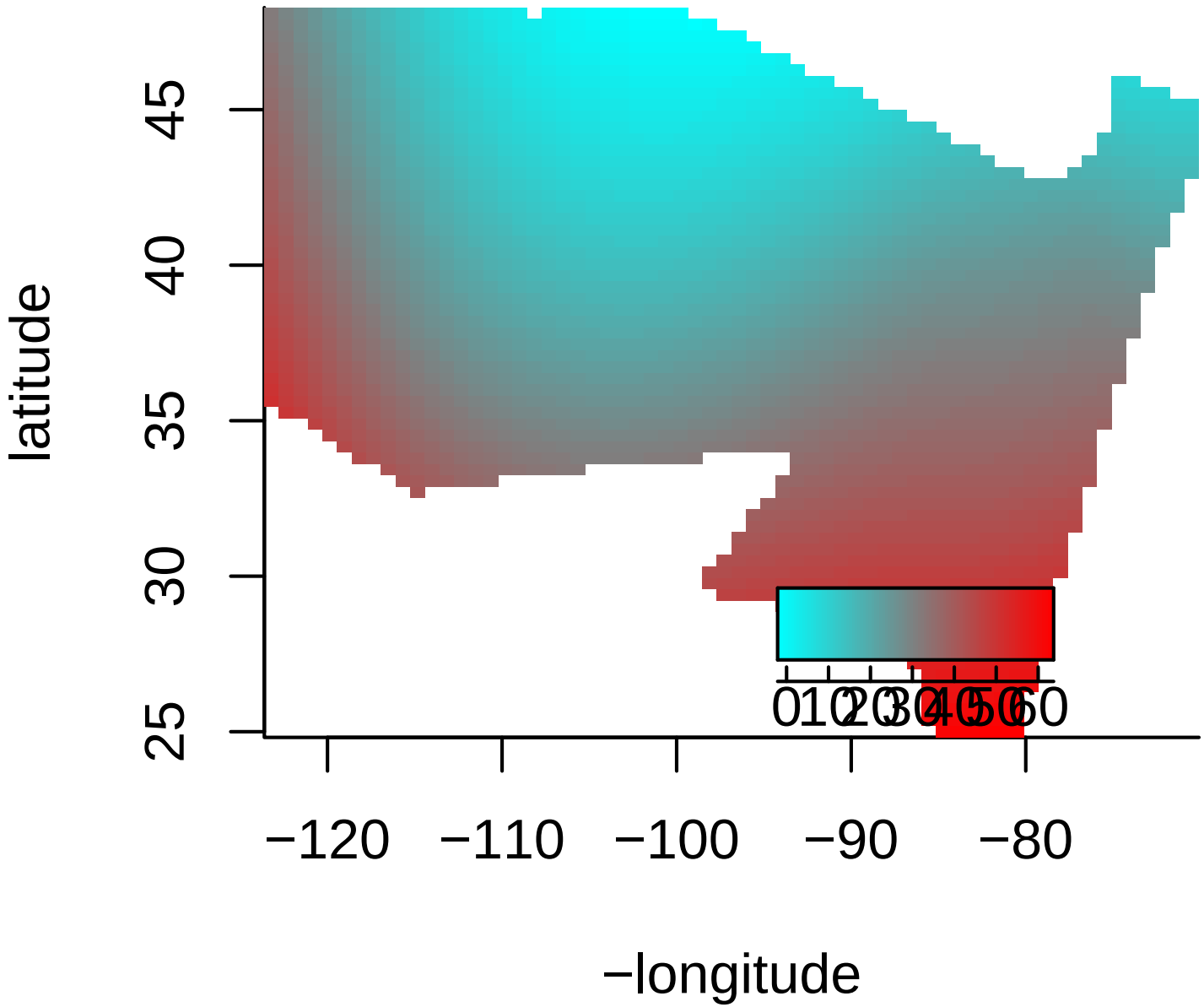
Geoadditive models (cont.)

- The essential idea is to use a bivariate grid of knots $\boldsymbol{\kappa}_k \in \mathbb{R}^2$, $k = 1, \ldots, K$ "covering" the space of the $\mathbf{X}_i$, $i = 1, \ldots, n$.
- Application on US temperature data:

```
> library(SemiPar)
> spm.us2 <- spm(min.temp~f(-longitude, latitude))
```

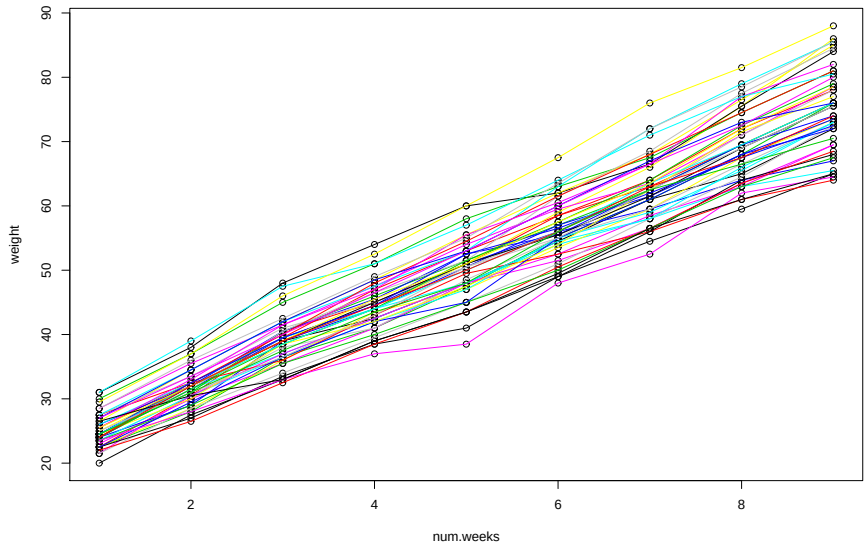


```
> plot(spm.us2)
```


latitude
25
30
35
40
45
−120
−110
−100
−90
−80
−longitude
0
1
2
3
4
5
6

Random Effect models

- Look at pig weight data: Growth curves for 48 pigs.

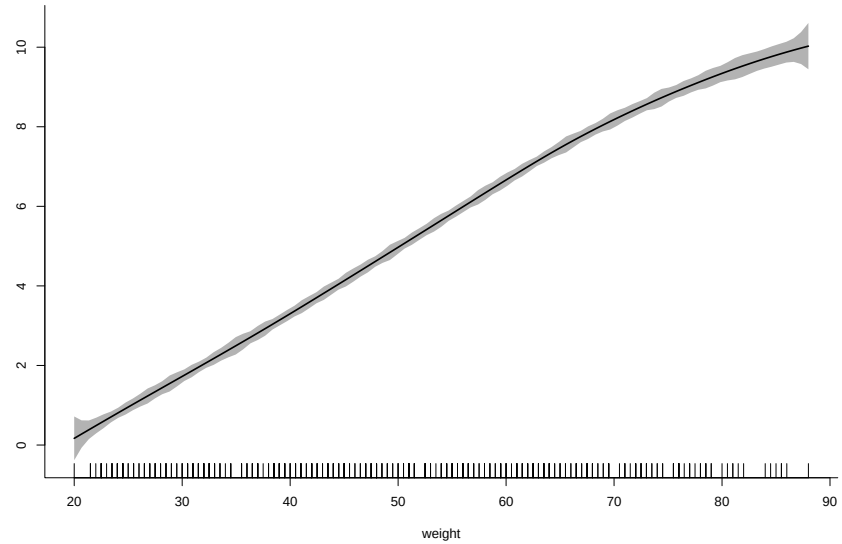


- For each pig, we have 9 *repeated measurements*. Hence, we have "within-pig-correlation" and the data are not iid.
- A possible solution would be to install an intercept for each pig, which would add 47 parameters!

Random Effect models (cont.)

- The better idea is to use random effects $U_i \sim N(0, \sigma_U^2)$ (which need only one parameter), and to model
$$\texttt{weight}_{ij} = \beta_0 + U_i + m(\texttt{week}_j) + \varepsilon_{ij}$$
for pig i in week j.
- Again, easily done by including the U_i into $\mathbf{u}$ and adding the corresponding columns to $\mathbf{Z}$.

```
> data(pig.weights)
> names(pig.weights)
[1] "id.num" "num.weeks" "weight"
> plot(spm(num.weeks~f(weight),
random= 1, group=id.num))
```



- Only a tiny trend to non-linearity here.

Bayesian Smoothing

- Back to our original classification of smoothing methods:
  1. Nonparametric (kernel) density estimation,
  2. Nonparametric regression, including additive, semiparametric, and spatial models,
  3. Principal curves (and the like).
- Bayesian versions exist to concepts 1. and 2. There are no Bayesian principal curves yet.
- Generally, one has to distinguish between Bayesian parametric and Bayesian nonparametric inference.

Parametric and nonparametric Bayesian inference

- Given:
  - Data $D = X_1, \ldots, X_n$ sampled from a population P, the distribution of which is modelled depending on a parameter vector $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.
  - Some prior belief $p(\boldsymbol{\theta})$ about $\boldsymbol{\theta}$.
- Then Bayesian inference is based on analyzing the posterior

$$p(\boldsymbol{\theta}|D) = \frac{p(D|\boldsymbol{\theta})p(\boldsymbol{\theta})}{p(D)}$$

- If $\dim(\boldsymbol{\Theta}) < c < \infty$ then one speaks of parametric Bayesian inference.
- If $\dim(\boldsymbol{\Theta}) = \infty$ or $\dim(\boldsymbol{\Theta}) = O(n)$, then we are in the world of nonparametric Bayesian inference. $\boldsymbol{\Theta}$ is then a function space.

Bayesian nonparametric density estimation

- Quite well studied, using methods from Bayesian nonparametrics.
- Suppose we have iid data $X_i \sim F(x_i), i = 1, \ldots, n$ with unknown distribution function F.
- Under a Bayesian nonparametric perspective we need a prior probability model $p(F)$ for F in some infinite dimensional function space. This requires to define probability measures on collections of distribution functions, so-called random probability measures (RPM).
- Ferguson (1973) showed that the Dirichlet process is a possible RPM.

Bayesian nonparametric density estimation (cont.)

- The Dirichlet process has a simple update rule [Ghosh & Ramamoorthi, p. 96], and one can show that the posterior distribution is given by

$$\hat{F}(t|x_1,\ldots,x_n) = p_n F_0(t) + (1-p_n)F_n(t|X_1,\ldots,X_n),$$

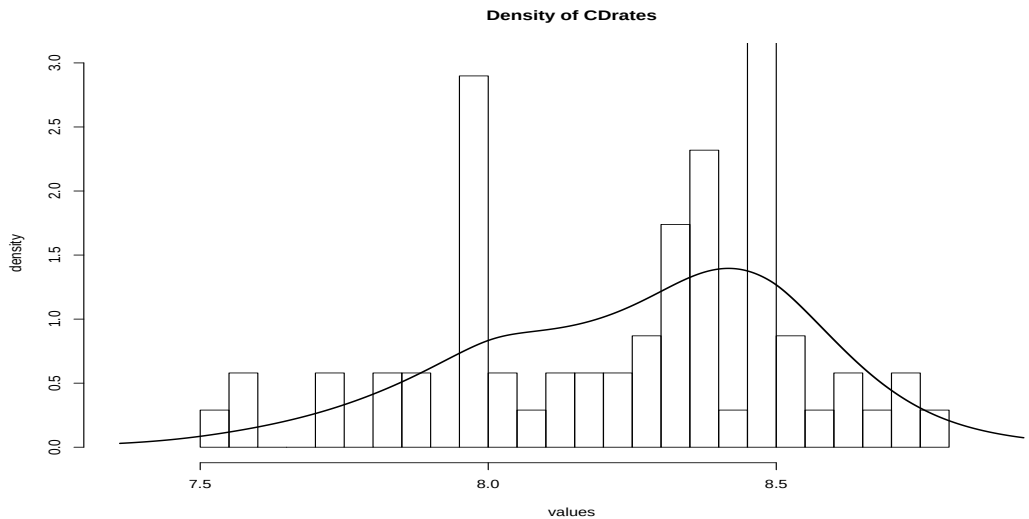
  where $F_0(t)$ is a "prior' distribution function, $F_n(t|\ldots)$ the empirical distribution function, and $p_n \in [0,1]$ a sequence of form $\frac{c}{c+n}$, for some $c > 0$.
- Such estimates are not "smooth". A smoothing effect enters solely through the prior, which is then "roughened" through the data.
- Bayesian nonparametric density estimation based on *mixtures of* Dirichlet Process priors is implemented in the new R package DPpackage

Bayesian nonparametric density estimation (cont.)

- Attempt for the CD rates:

```
> library(DPpackage)
> saveDP1 <- DPdensity(CDrates, prior=prior1, mcmc=mcmc,
status=TRUE)
```

- Note that in `prior1` and `mcmc` about 10 tuning- and hyper-parameters have to be specified, and computation takes ~ 20 sec.



Bayesian nonparametric regression

- This can be read in two ways:
  - Bayesian nonparametric regression
  - Bayesian nonparametric regression

  What's the difference?
- Bayesian nonparametric regression uses the theory of Bayesian nonparametrics, constructing probability measures on density or function spaces, which then act as prior distributions for a Bayesian analysis.
  - object of intensive current research
  - not object of this course!

Bayesian nonparametric regression (cont.)

- Within this course, we constrain ourselves to

  Bayesian nonparametric regression,

  meaning parametric Bayesian versions of usual nonparametric regression methods.
- Also, this is quite new and no "textbook material".
- Current approaches:
  - Bayesian P-Splines (Lang & Brezger, JCGS, 2005)
  - Bayesian Regression Splines (Smith & Kohn, Journal of Econometrics, 1996)
  - Bayesian Smoothing Splines (Hastie & Tibshirani, Statistical Science, 2000)
  - Bayesian piecewise polynomials (Denison, Mallick, & Smith, JRSSB, 1998)
  - ⋮ ?

Towards Bayesian P-Splines

- P-Splines work generally quite well.
- The problem of smoothing parameter selection remains. Typically, it is estimated via cross-validation or by minimizing the AIC criterion.
- This often tends to fail, especially when several smoothing parameters have to be selected simultaneously within an additive model.
- A Bayesian version overcomes these problems, as it estimates the smoothing parameter(s) as a by-product, similar as in the mixed model approach.
- The Bayesian approach is very powerful (thanks to MCMC) and adopts to very complex models, including (generalized) additive, semiparametric, and spatial, models.

Towards Bayesian P-Splines (cont.)

- We work directly within the framwork of the additive model, which we write in short form as

$$Y_i = \mathbf{t}_i^T \boldsymbol{\beta} + m_1(X_{i1}) + \ldots + m_q(X_{iq}) + \varepsilon_i \quad , i = 1, \ldots, n. \quad (26)$$

  where $\mathbf{t}_i$ is a vector containing all components which are modelled parametrically, $\boldsymbol{\beta}$ contains the corresonding parameters, and $m_j, j = 1, \ldots, q$ are unknown functions of metric covariates. ε_i is iid error with $\text{Var}(\varepsilon_i) = \sigma^2$.

- Modelling each function $m_j(x_j)$ through an appropriate B-Spline basis such that $m_j(x) = \sum_{\ell=1}^{m} \alpha_{j\ell} B_{j\ell}(x_j) \equiv \mathbf{B}_j(x)^T \boldsymbol{\alpha}_j$, model (26) can be written as

$$\mathbf{y} = \mathbf{B}_1 \boldsymbol{\alpha}_1 + \ldots \mathbf{B}_q \boldsymbol{\alpha}_q + \mathbf{T}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

  with, as before, $\mathbf{B}_j = B_{j\ell}(X_i)_{1 \leq i \leq n, 1 \leq \ell \leq m}$.

Bayesian P-Splines

- The penalized likelihood can then be written as

$$L(\mathbf{y};\boldsymbol{\beta},\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_q) - \lambda_1 \sum_{\ell=3}^{m} (\triangle^2 \alpha_{1\ell})^2 - \ldots - \lambda_q \sum_{\ell=3}^{m} (\triangle^2 \alpha_{q\ell})^2$$

  which has to be maximized with respect to $\boldsymbol{\beta},\boldsymbol{\alpha}_1,\ldots,\boldsymbol{\alpha}_q$.

- In a Bayesian approach these parameter vectors are random and have to be supplemented with appropriate prior distributions.

- For the fixed effect $\boldsymbol{\beta} = (\beta_1,\ldots,\beta_p)^T$, one simply uses diffuse priors, i.e.

$$\beta_j \propto const, \quad j = 1,\ldots,p.$$

Bayesian P-Spines (cont.)

- The prior for $\boldsymbol{\alpha}$ has to account for the penalization
- To this end, note that from $\triangle\alpha_{j\ell}^2 = \alpha_{j\ell} - 2\alpha_{j,\ell-1} + \alpha_{j,\ell-2}$ follows

$$\alpha_{j\ell} = 2\alpha_{j,\ell-1} - \alpha_{j,\ell-2} + \triangle^2\alpha_{j\ell}$$

- The idea is to construct from this a second order random walk prior for $\alpha_{j\ell}$, by setting

$$\alpha_{j\ell} = 2\alpha_{j,\ell-1} - \alpha_{j,\ell-2} + u_{j\ell},$$

with

$$u_{j\ell} \sim N(0, \tau_j^2).$$

- The variance parameter τ_j^2 controls the amount of smoothing, corresponding to the λ_j in the classical approach. Note the analogy to the mixed model approach!

Bayesian P-Spines (cont.)

- Summarizing, we work with the hierarchical model

[1] $\beta_j \propto const, \quad j = 1, \ldots, p.$
$\alpha_{j\ell} \propto const, \quad j = 1, 2.$
$\alpha_{j\ell} = 2\alpha_{j,\ell-1} - \alpha_{j,\ell-2} + u_{j\ell}, \quad j = 3, \ldots, m,$

[2] $u_{j\ell} \sim N(0, \tau_j^2),$

[3] $\tau_j^2 \sim IG(a_j, b_j);\ \sigma^2 \sim IG(a_0, b_0)$

[4] $a_j = 1,\ b_j = 0.005$

- The choice of hyperparameters in [4] makes the hyperpriors in [3] almost diffuse. It must not be completely diffuse, as otherwise the posterior for the $\boldsymbol{\alpha}_j$ is improper.
- The actual parameters of interest are $(\beta_j)_{1\leq j\leq p}$, $(\alpha_{j\ell})_{1\leq j\leq q, 1\leq l\leq m}$, $(\tau_j^2)_{j=1,\ldots,q}$, and σ^2. We sumarize all these parameters in one vector $\boldsymbol{\theta}$.

Bayesian P-Splines (cont.)

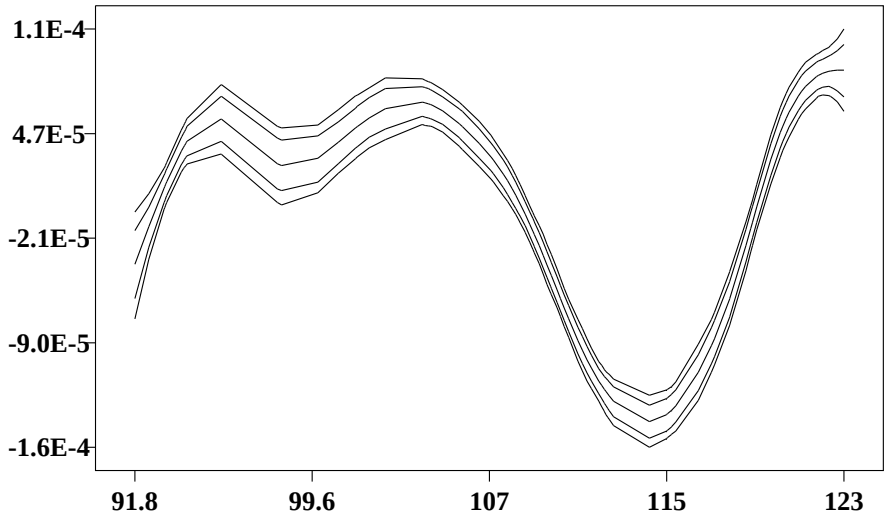
- Then the posterior of the model is given by

$$p(\boldsymbol{\theta}|\mathbf{y}) \propto L(\mathbf{y}; \boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_q, \boldsymbol{\beta}, \sigma^2) \prod_{j=1}^{q} \left[p(\boldsymbol{\alpha}_j|\tau_j^2)p(\tau_j^2)\right] p(\boldsymbol{\beta})p(\sigma^2)$$

- This posterior is is analytically intractable.
- Therefore, inference is carried out by Markov Chain Monte Carlo simulation techniques (i.e., algorithms for sampling from probability distributions based on constructing a Markov chain that has the desired distribution as its stationary distribution. The state of the chain after a large number of steps is then used as a sample from the desired distribution.)
- The drawings happen for blocks of parameters given the other parameters and the data.
- No more datails here! See Lang & Brezger, JCGS, 2005.

Example

- We fit the Fossil data using **BayesX**:



- The central line is is the actual fitted curve.
- The outermost variability bands are 95% pointwise credible intervals.
- The inner variability bands are 80% pointwise credible intervals.

Example (cont.)

- How to arrive at this plot?
- First, need to get the data set from R into BayesX:
  - in R:

```
> library(SemiPar); data(fossil)
> write(t(as.matrix(fossil)), file='J:/Data/fossil.dat')
```

  - in **BayesX**:

```
> dataset fossil
> fossil.infile age strontium using J:\Data\fossil.dat
```

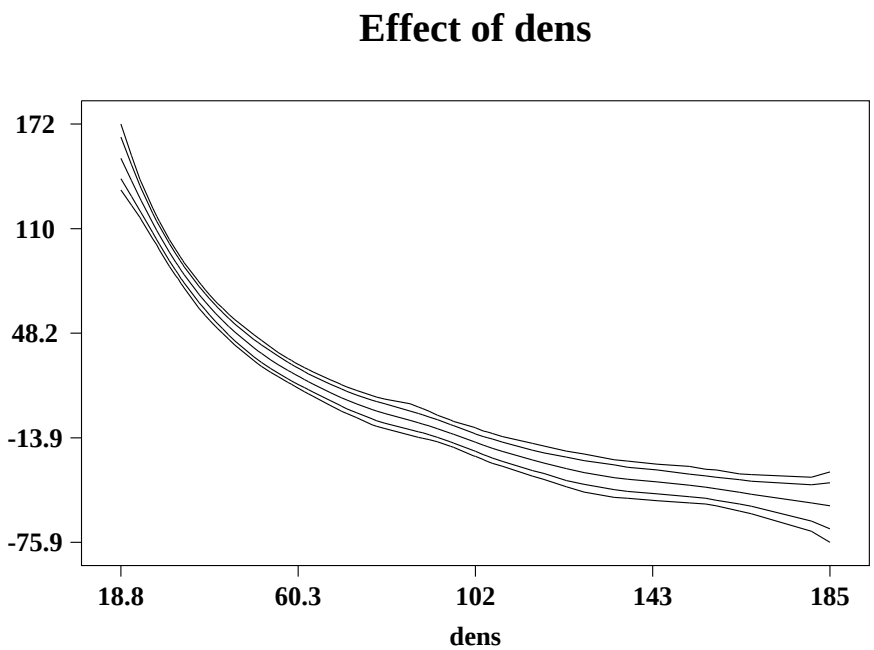
- Then, fit the model $\texttt{strontium}_\texttt{i} = m(\texttt{age}_\texttt{i}) + \varepsilon_i$:

```
> bayesreg b
> b.regress strontium=age(psplinerw2), family= gaussian using
fossil
> b.plotnonp 1
```

Example for additive (semiparametric) model

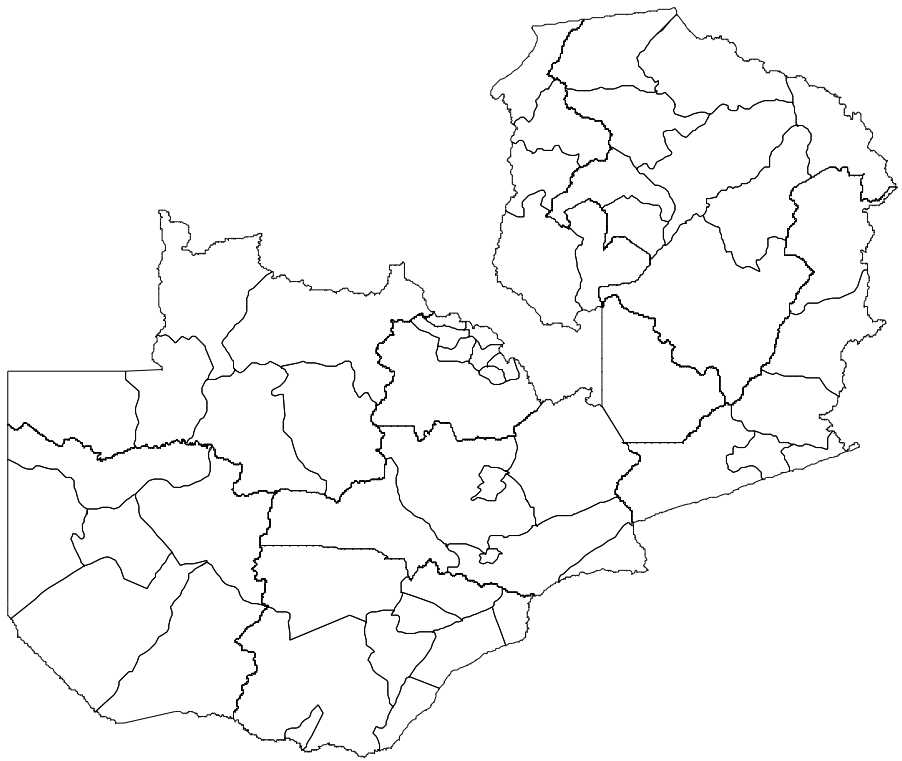
- Reconsider Onions data.

```
> data(onions, package='SemiPar')
> write(t(as.matrix(onions)), file='J:/Data/onions.dat')
> dataset onions
> onions.infile dens yield location using J:\Data\onions.dat
> bayesreg c
> c.regress yield= location + dens(psplinerw2), family=gaussian
using onions
> c.plotnonp 1
```



Geoadditive models based on maps

- **BayesX** also offers the possibility to model spatial effects directly, when one has a categorization of the observations into districts.
- For instance, Zambian map of districts stored in a boundary file (*.bnd) :



Geoadditive models based on maps (cont.)

- Given a map with, say, S districts, it is reasonable to assume that responses depend on the spatial location where they have been observed (for instance, due to similar geographical, regional, or political conditions).
- This can be addressed through a spatial effect $f_{sp} = \mathbf{X}_{sp}\boldsymbol{\gamma}$, which is added as an extra term to the linear predictor.
  - $\mathbf{X}_{sp}$ is a $n \times S$ incidence matrix whose (i, s)-th entry is one if observation i corresponds to district s
  - $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_S)^T$ is equipped with a Markov random field prior which injects the spatial correlation into the model:

$$\gamma_s | \gamma_{u, u \neq s} \sim N\left(\sum_{u|\text{neighbors of } s} c\gamma_u,\ c\tau^2\right)$$

- This prior suggests that similar regions should have similar parameters. Effectively, this means to do spatial smoothing!

Example: Undernutrition in Zambia

- Data on undernutrition of children in Zambia.
- Undernutrition on children is measured through a stunting score ("Z-score") $Z-$ score which is defined as

$$Z_i = \frac{AI_i - MAI}{\sigma}$$

  where AI refers to the childs anthropologic indicator (here, height at a certain age), MAI refers to the median of the reference population and σ refers to its standard deviation.
- The main interest is on modelling the dependence of undernutrition on covariates including the age of the child, the body mass index of the child's mother, the district the child lives in, among others.

Example: Undernutrition in Zambia (cont.)

List and description of variables:

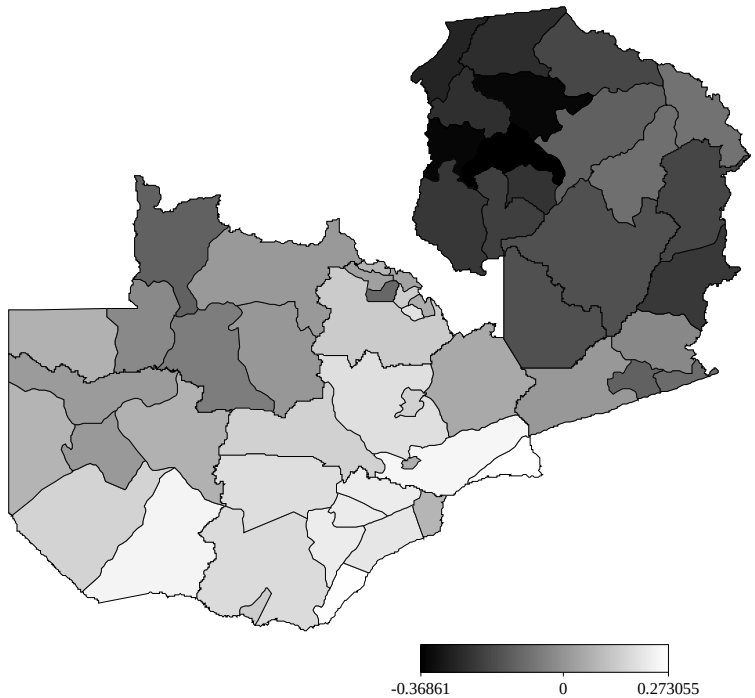
Variable	Description
`hazstd`	standardized stunting Z-score (Z_i)
`bmi`	mother's body mass index
`district`	district where the child lives
`rcw`	mother's employment status with categories "working" ($= 1$) and "not working" ($= -1$)
`edu1/2`	mother's educational status with $edu1 = 1$ and $edu2 = 1$ for complete primary and secondary education, respectively (and -1 otherwise).
`tpr`	locality of domicile with categories "urban" ($= 1$) and "rural" ($= -1$)
`sex`	child's gender: male $= 1$, female $= -1$.

Example: Undernutrition in Zambia (cont.)

- A model of tupe

  $$\texttt{hazstd} = \texttt{rcw} + \texttt{edu1} + \texttt{edu2} + \texttt{tpr} + \texttt{sex} + m(\texttt{bmi}) + m(\texttt{agc}) + f_{sp}$$

  is fitted, leading to the following posterior mean for the spatial effect:



- You will create this map in the practical!

Section 4: Principal curves

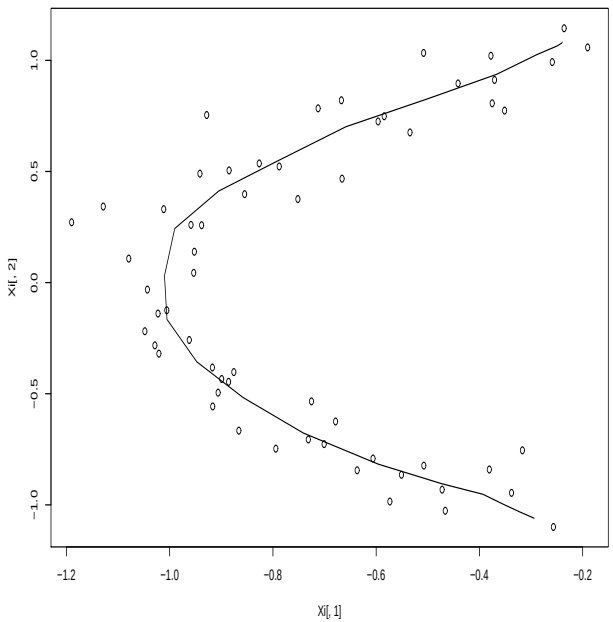
Scope of this section:

- Principal component analysis;
- Nonlinear PCA;
- Hastie & Stuetzle principal curves;
- Self-consistency;
- 3D principal curves;
- Alternative approaches;
- Principal manifolds.

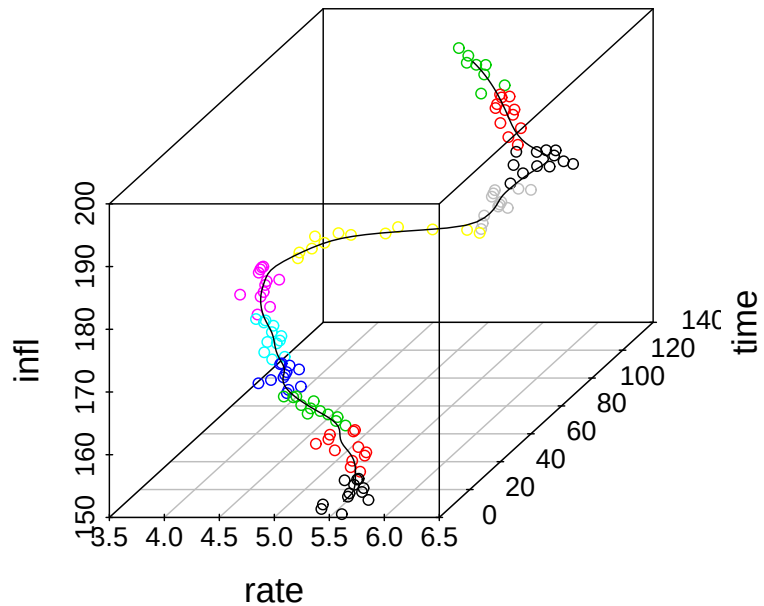
Principal curves - Introduction

- Principal Curves are smooth curves passing through the 'middle' of a multidimensional data cloud $X = (X_1, \ldots, X_n)$, where $X_i \in \mathbb{R}^d$.
- Examples:

d=2
Simulated "C"

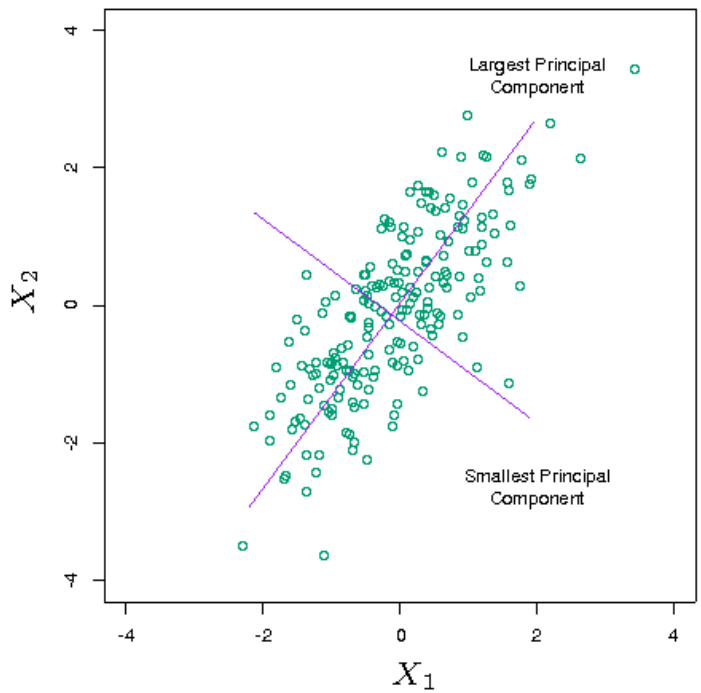


d=3
Price index/ employment
in the US 95-05



Principal components

- Principal curves are nonparametric extensions of principal components.
- Principal components provide a sequence of best linear orthogonal approximations to a data cloud $\mathbf{X} = (\mathbf{X}_1, \ldots, \mathbf{X}_n)$, where $\mathbf{X}_i \in \mathbb{R}^d$.
- For example, first and second principal component for a simulated data set [HTF]:



Principal component analysis (PCA)

- Given a random variable $\mathbf{X}$ in $\mathbb{R}^d$ with covariance matrix $\mathbf{\Sigma}$, the eigen decomposition is given by

$$\mathbf{\Sigma} = \mathbf{\Gamma}\mathbf{\Lambda}\mathbf{\Gamma}^T, \tag{19}$$

  where $\mathbf{\Lambda}$ is a diagonal matrix containing the ordered eigenvalues $\lambda_j = \mathrm{Var}(\boldsymbol{\gamma}_j^T\mathbf{X})$ of $\mathbf{\Sigma}$, $j = 1, \ldots, d$.

- The columns of

$$\mathbf{\Gamma} = (\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_d)$$

  are the eigenvectors of $\mathbf{\Sigma}$.

- The first eigenvector $\boldsymbol{\gamma}_1$ maximizes the variance of $\boldsymbol{\gamma}^T\mathbf{X}$ (among all $\boldsymbol{\gamma} \in \mathbb{R}^d$ with $||\boldsymbol{\gamma}|| = 1$); the second eigenvector $\boldsymbol{\gamma}_2$ maximizes the variance of $\boldsymbol{\gamma}^T\mathbf{X}$ over all $\boldsymbol{\gamma} \in \mathbb{R}^d$ with $||\boldsymbol{\gamma}|| = 1$ which are orthogonal to $\boldsymbol{\gamma}_1$, and so on.

Linear and Nonlinear PCA

- The process of finding the first principal component line can be dissected into two steps (w.l.o.g. $\boldsymbol{\mu} = 0$):

  Projection: Find a linear mapping

  $$f(\mathbf{x}) : \mathbb{R}^d \longrightarrow \mathbb{R}, \mathbf{x} \mapsto \boldsymbol{\gamma}_1^T \mathbf{x} \, (\equiv t).$$

  Reconstruction: Map back to the data space

  $$\mathbf{g}(t) : \mathbb{R} \longrightarrow \mathbb{R}^d, t \mapsto \boldsymbol{\gamma}_1 t.$$

  ... such that

  $$\sum_{i=1}^{n} ||\mathbf{X}_i - \mathbf{g}(t_i)||^2 = \sum_{i=1}^{n} ||\mathbf{X}_i - (\mathbf{g} \circ f)(\mathbf{X}_i)||^2 \qquad (20)$$

  is minimized.

- Substituting f and/or $\mathbf{g}$ by nonlinear functions leads to nonlinear principal component analysis (NLPCA) [G, Chapter 2]

Principal curves

- Principal curves go one step further: They aim for a fully nonparametric estimation of the curve.
- Formally, a one-dimensional curve in a $d-$dimensional space is a vector-valued function

$$\mathbf{g}(t) : \mathbb{R} \mapsto \mathbb{R}^d.$$

  Its $j-$th component, $j = 1, \ldots, d$, are called *coordinate functions.*
- Defining a model of type

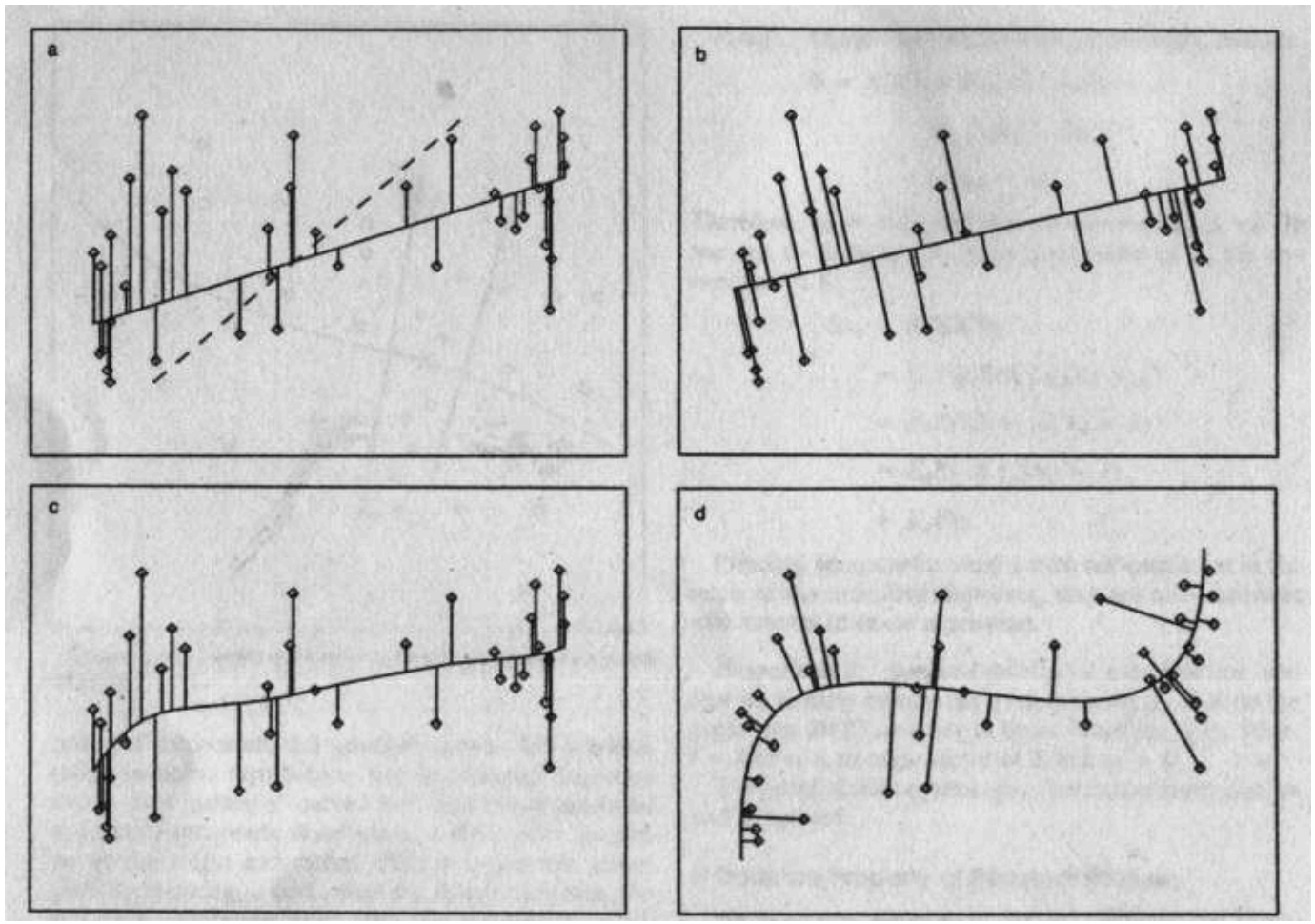
$$\mathbf{X}_i = \mathbf{g}(t_i) + \mathbf{e}_i,$$

  the task is to find "the best" $\mathbf{g}$.

Principal curves (cont.)

Parametric and nonparametric regesssion

Principal components and curves



Principal curve parametrization

- A common parametrization t for (principal) curves $\mathbf{g}(t) : \mathbb{R} \mapsto \mathbb{R}^d$ is the unid-speed-parametrization:
  - Recall that the arc length of a curve $\mathbf{g}$ from a parameter t_0 to t_1 is given by $\ell = \int_{t_0}^{t_1} ||\mathbf{g}'(t)|| \, dt$.
  - That is, if $||\mathbf{g}'(t)|| = 1$, then $\ell = t_1 - t_0$. This means that distances in parameter space correspond to the arc length in data space, which is intuitively desirable.
  - Every curve with $||\mathbf{g}'|| > 0$ can be reparametrized to make it unit-speed.
  - The vector $\mathbf{g}'(t)$ is called the *velocity* at t and $\mathbf{g}''(t)$ the acceleration. For a unit-speed curve, the acceleration is orthogonal to the velocity.
- Other choices of the parametrization are possible as long as it is monotone, i.e. maintains the order of the data points projected onto it.

HS Principal curves

- The first and groundbreaking work on principal curves is from Hastie & Stuetzle (HS), JASA **84**, 1989.
- HS principal curves generalize linear principal components in a very natural and direct way: The idea is to minimize the distance property (20) over all "smooth" functions $\mathbf{g} : \mathbb{R} \longrightarrow \mathbb{R}^d$.
- Let $\mathbf{X} \in \mathbb{R}^d$ be a random vector with density f, $E(\mathbf{X}) = 0$ and Var$(\mathbf{X})$ finite.
- Let $\mathbf{g} : T \longrightarrow \mathbb{R}^d$, $T \subset \mathbb{R}$ be a non-intersecting unit-speed curve $\mathbf{g} : T \longrightarrow \mathbb{R}^d$, $T \subset \mathbb{R}$.
- The projection index $t_{\mathbf{g}} : \mathbb{R}^d \longrightarrow \mathbb{R}$ is defined as

$$t_{\mathbf{g}}(\mathbf{x}) = \sup_t \{t : ||\mathbf{x} - \mathbf{g}(t)|| = \inf_{\ell} ||\mathbf{x} - \mathbf{g}(\ell)||\}$$

In words, $t_{\mathbf{g}}(\mathbf{x})$ is the value of t for which $\mathbf{g}(t)$ is closest to $\mathbf{x}$.

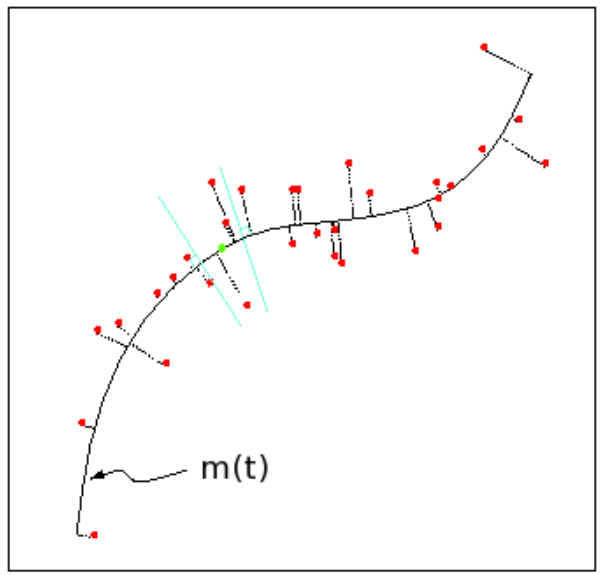
HS Principal curves (cont.)

- Definition: The curve $\mathbf{g}$ is a principal curve of f when it is self-consistent, i.e.

$$E(\mathbf{X}|t_{\mathbf{g}}(\mathbf{X}) = t) = \mathbf{g}(t)$$

for all $t \in T$.

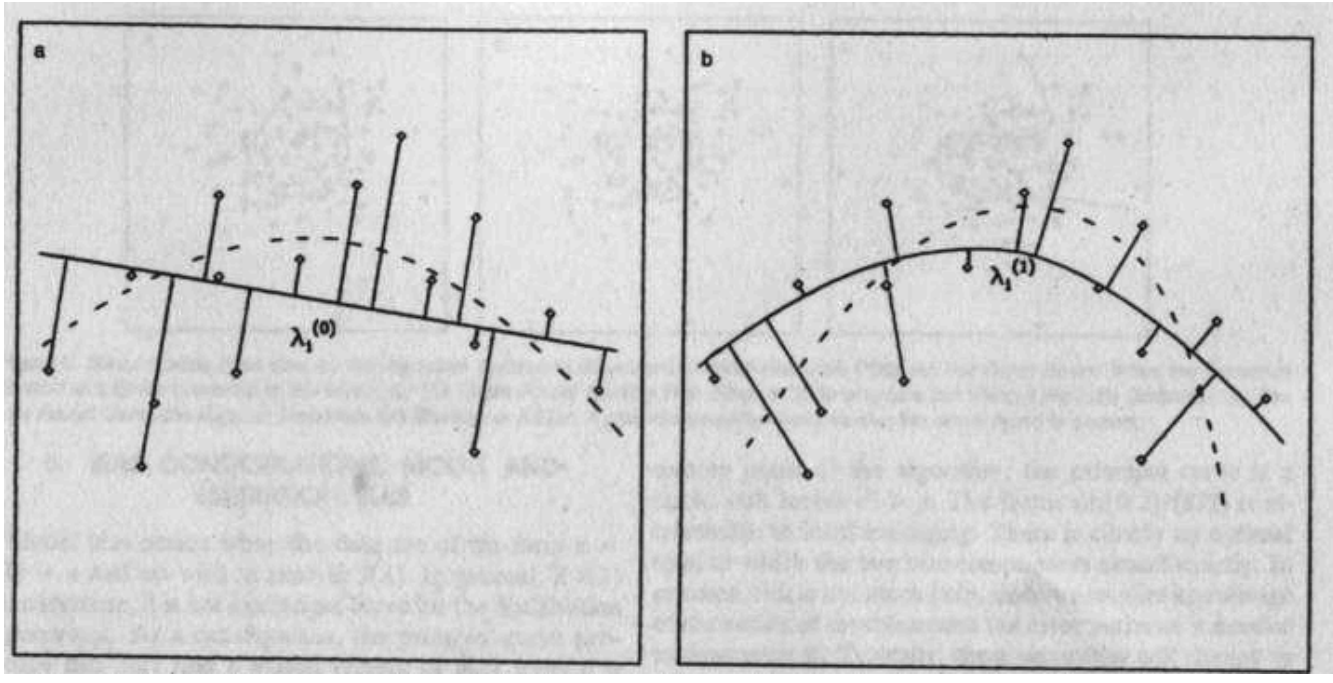
- Self-consistency means that each point on the principal curve is the average of all points which project there [HTF].
- If a straight line is self-consistent, then it is a principal component.



HS Principal curve estimation

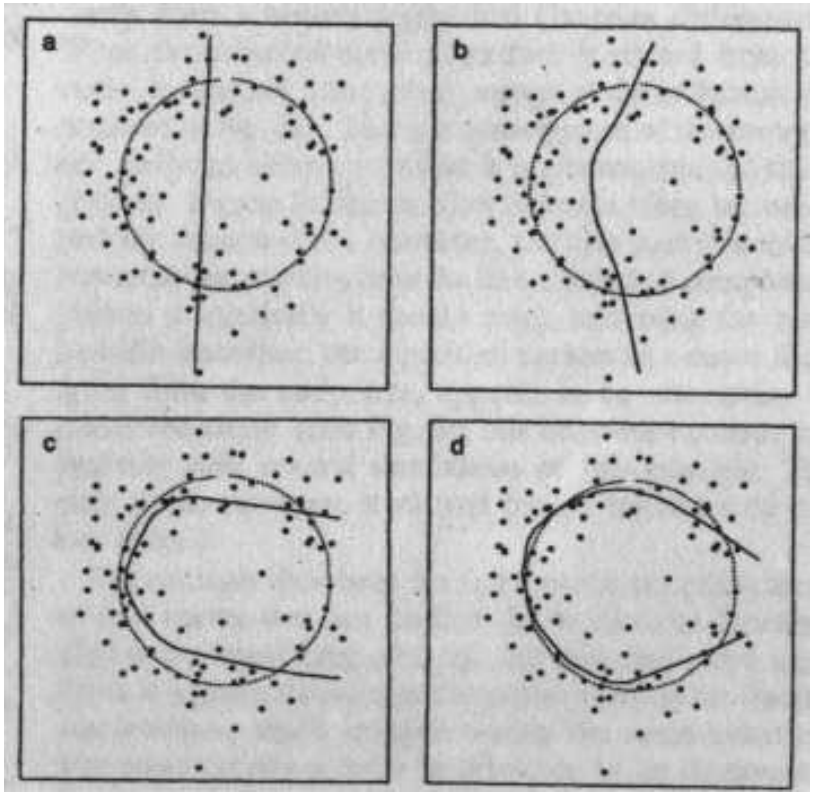
- Start with the first linear principal component line.
- Then, iterate between
  - Projection: Project all data points onto this line.
  - Reconstruction: Use a nonparametric scatterplot smoother (like kernels, splines) to fit each component of $\mathbf{X}$ against the projection indices t.

.... until the change in the distance function gets negligible.



HS Principal curve estimation (cont.)

- Ilustration: Circle in 4 steps.



HS Principal curve software

- HS principal curves are implemented in the R functions
  - `principal.curve` in R package **princurve** offering
    - Splines
    - Robust local linear smoothers ("lowess")

    in the reconstruction step.
  - `pcurve` in R package **pcurve** offering
    - B-Splines

    in the reconstruction step.
- **pcurve** is more powerful and it seems to have a somewhat better smoothing parameter selection, while **princurve** is somewhat more compact to use.

HS Principal curves in practice

Example: Letter recognition

- Simulate "C":

```
> t <- seq(pi/2+0.2,3*pi/2-0.2, length=60) # Parametriziation
> cx <- cos(t); cy <-sin(t) # gives a circular arg segment
> cx2 <- cx + rnorm(60,0,0.1); cy2 <- cy + rnorm(60,0,0.1)
# adds noise
> c2 <- cbind(cx2,cy2) # creates a data matrix
```

- Fit principal curve
  - using smoothing splines in the reconstruction step

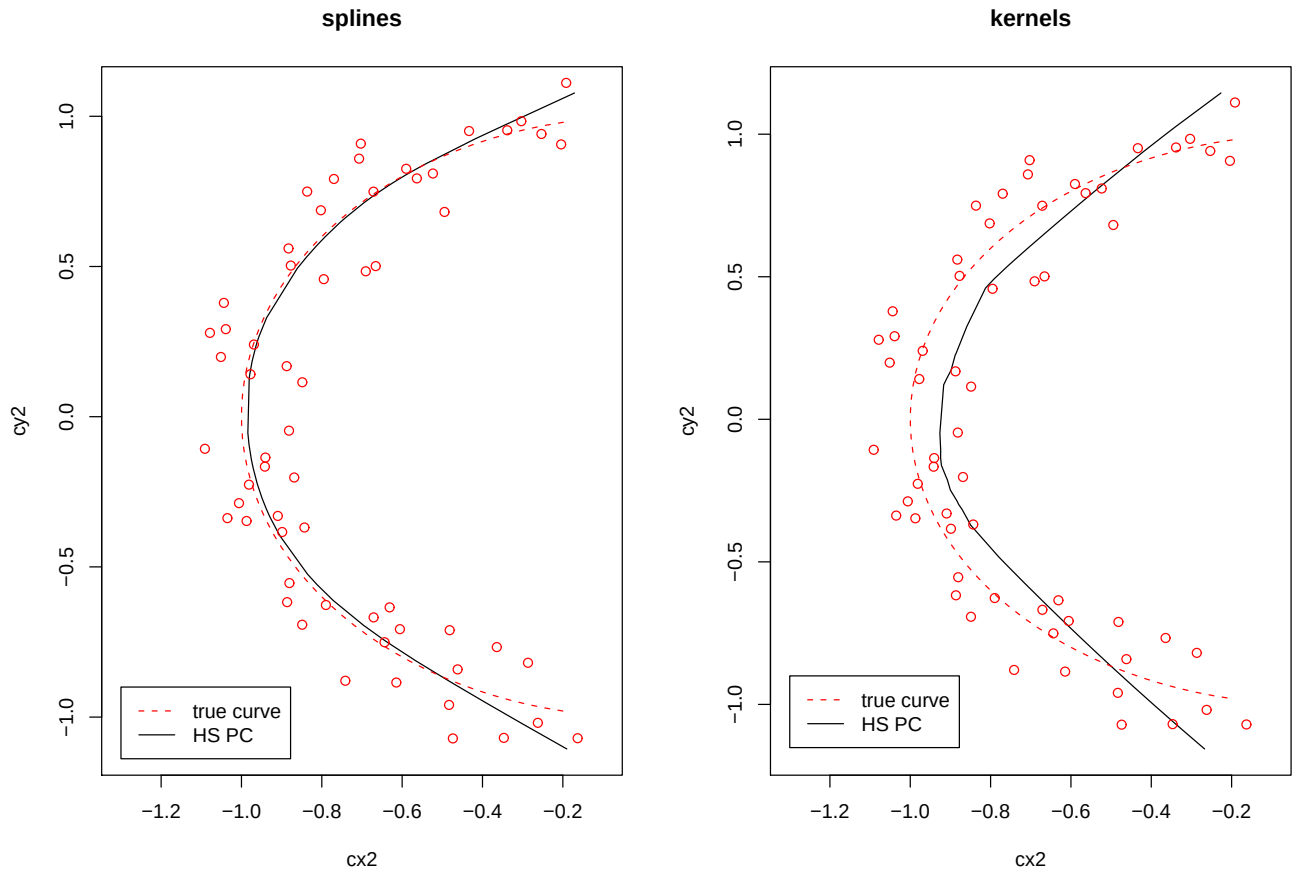
```
> library(princurve)
> c2prin <- principal.curve(c2)
```

  - using kernels in the reconstruction step

```
> c2prinl <- principal.curve(c2,smoother="lowess")
```

HS Principal curves in practice (cont.)

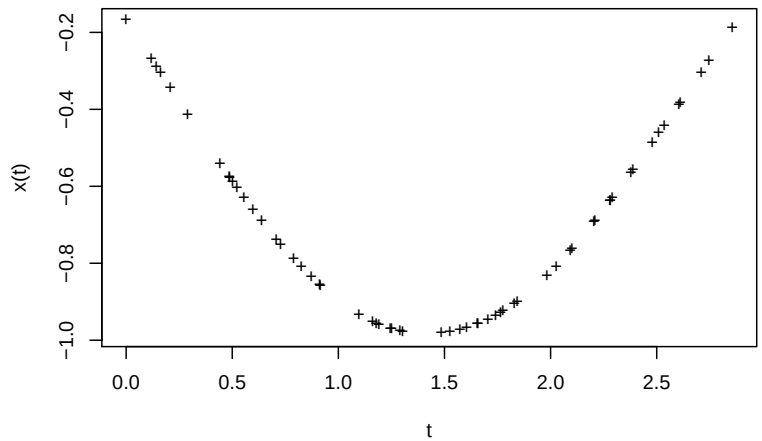
- Fitted curves:

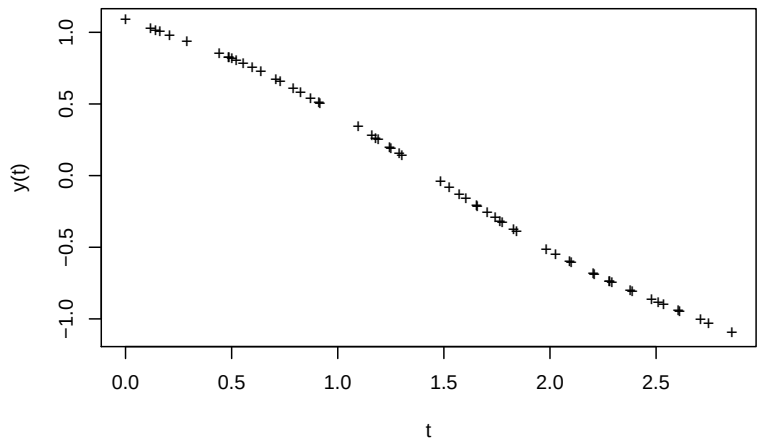


HS Principal curves in practice (cont.)

- One can also extract directly the coordinate functions $x(t), y(t)$, e.g. for the spline based version:

```
> plot(c2prin$lambda, c2prin$s[,1], xlab="t", ylab="x(t)", pch="+")
> plot(c2prin$lambda, c2prin$s[,2], xlab="t", ylab="y(t)", pch="+")
```

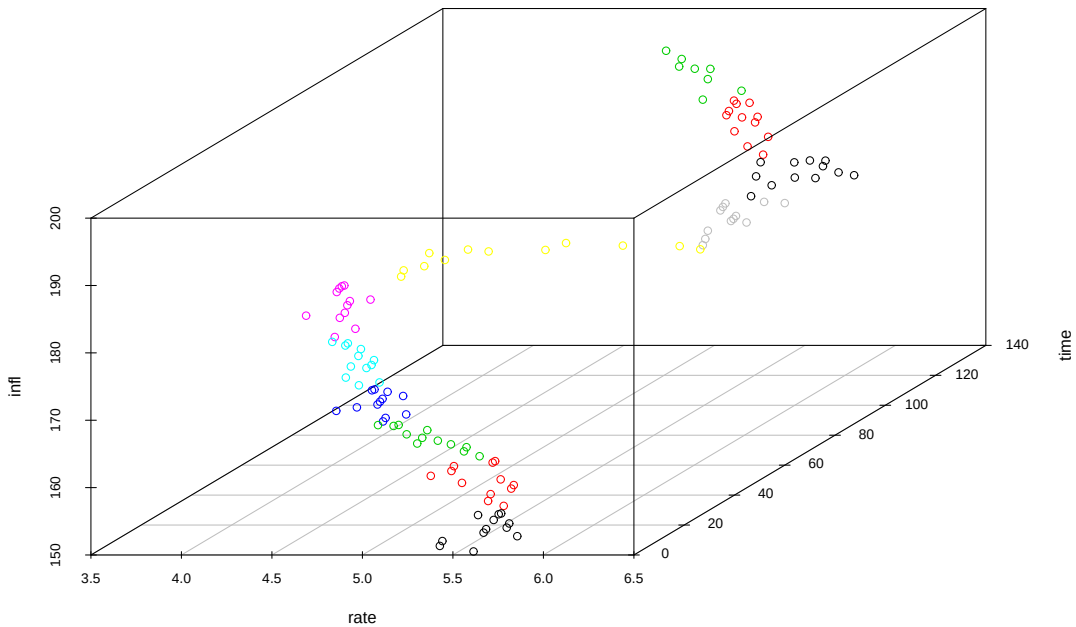




HS Principal curve estimation (cont.)

3D-Example: Phillips curves.

- Data on Inflation and Unemployment in the US 1995-2005
- We have three variables:
  - Price index: `infl`
  - Unemployment: `rate`
  - Time (in months): `time`.
- One colour corresponds to one year:



HS Principal curves in practice (cont.)

- Read data:

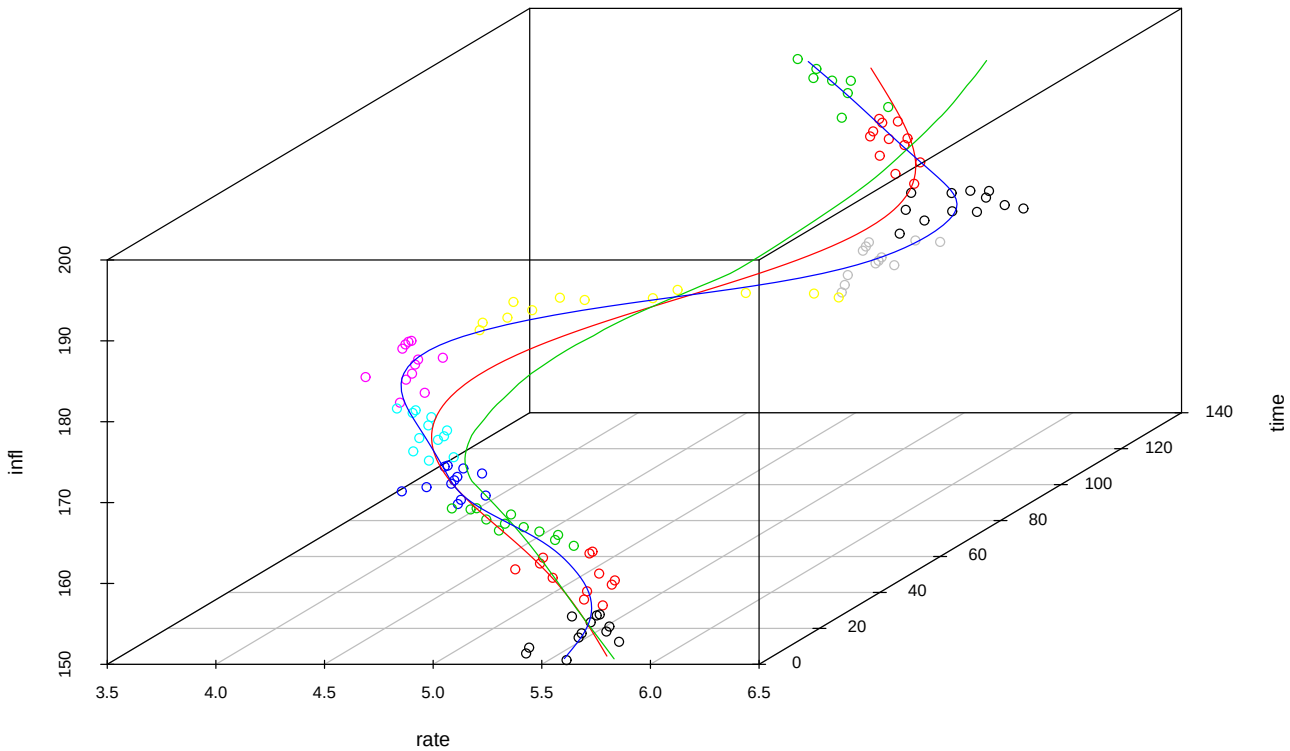
```
> phil.rate <- read.table("UNRATE.txt", header=T) # 1948-2005
> phil.infl <- read.table("CPIAUCNS.txt", header=T) # 1921-2005
> rate <- phil.rate[565:692,2] # 01/1995-08/2005
> infl <- phil.infl[889:1016,2] # 01/1995-08/2005
```

- Fit principal curves:

```
> library(pcurve); library(princurve)
> phil.hs <- principal.curve(cbind(rate,time,infl)) # Smoothing
Splines
> phil.hs1 <- principal.curve(cbind(rate,time,infl),
smoother="lowess") # LOWESS
> phil.hs2 <- pcurve(cbind(rate,time,infl)) # B-Splines
```

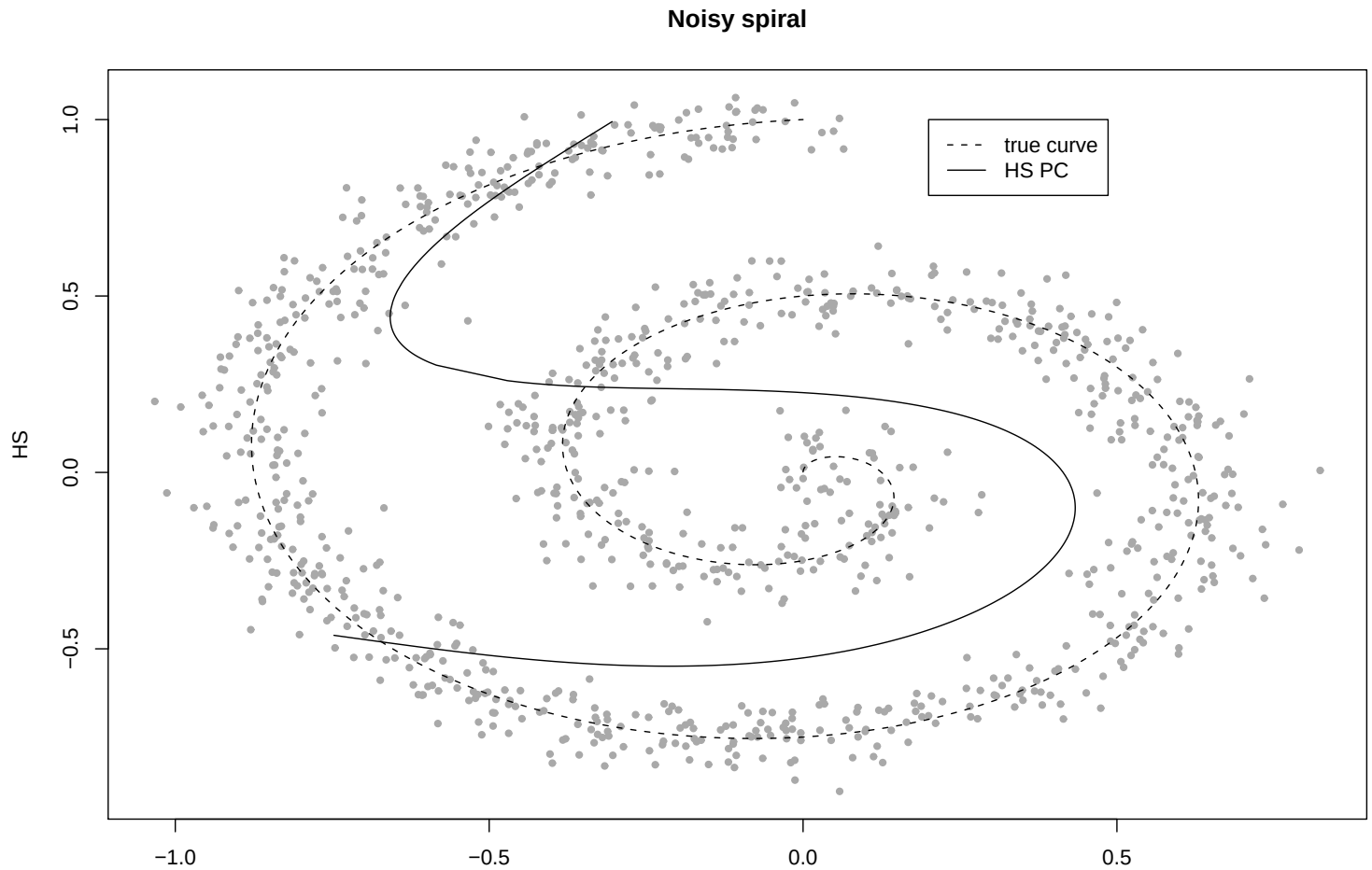
HS Principal curves in practice (cont.)

```
> phil.scat <- scatterplot3d(rate, time, infl, color=...,ylab="time")
> hil.scat$points3d(phil.hs$s[,1],phil.hs$s[,2],phil.hs$s[,3],
type="l",col=2) # Smoothing Splines
> phil.scat$points3d(phil.hs1$s[,1],phil.hs1$s[,2],phil.hs2$s[,3],
type="l",col=3) # LOWESS
> phil.scat$points3d(phil.hs2$s[,1],phil.hs2$s[,2],phil.hs2$s[,3],
type="l",col=4) # B-Splines
```



Limits of HS curves

- Noisy spiral:



- Most of the curve passes through regions with no data at all!

Limits of HS curves (cont.)

- HS principal curves work quite nice for simple data structures. For more complex structures they often fail, for several reasons:
  - The dependance on an initial line leads to a lack of flexibility, as an initial unsuitable assignment of projection indices can often not be correct in the further run of the algorithm.
  - It has been shown that that HS principal curves as defined above find only *saddle points*, and not minimizers, of the distance function (Duchamps and Stuetzle, 1996) (though their originial motivation was to minimize it!).
- When HS fails, one can resort to other principal curve algorithms.
- They all attempt to identify "the middle of the data cloud", but differ in what they understand of this "middle".

Alternative principal curve algorithms

Polygonal line algorithms. Kégl et al. (2000) define a principal curve as the curve minimizing the average squared distance over all curves with bounded length L, and construct a polygon to estimate it.
Software (JAVA Applet) at

`http://www.iro.umontreal.ca/~kegl/research/pcurves/`

Generative model. Tibshirani (1992) defines principal curves such that for data generated as

$$\mathbf{X} = \mathbf{g}(\lambda) + \epsilon \qquad \text{with} \quad E(\boldsymbol{\epsilon}) = 0$$

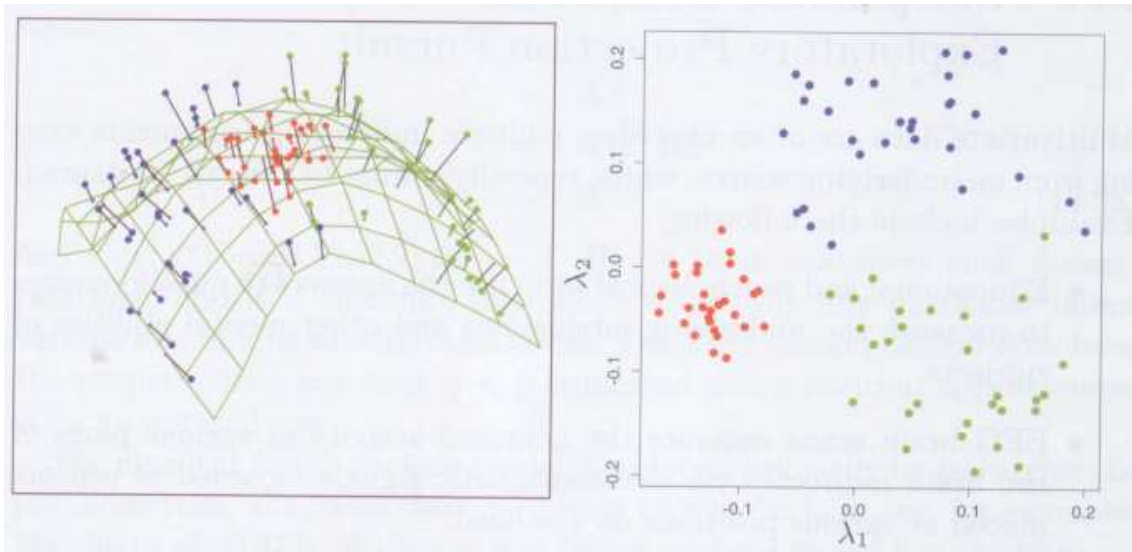
curve $\mathbf{g}$ is also principal curve of the data cloud $\mathbf{X}$.

Local approaches. Instead of starting with a straight line, the curve is successively built up while proceeding through the data cloud (see talk by M. Zayed).
R source code for local principal curves at

`http://www.maths.dur.ac.uk/~dma0je/lpc/lpc.htm`

Principal manifolds

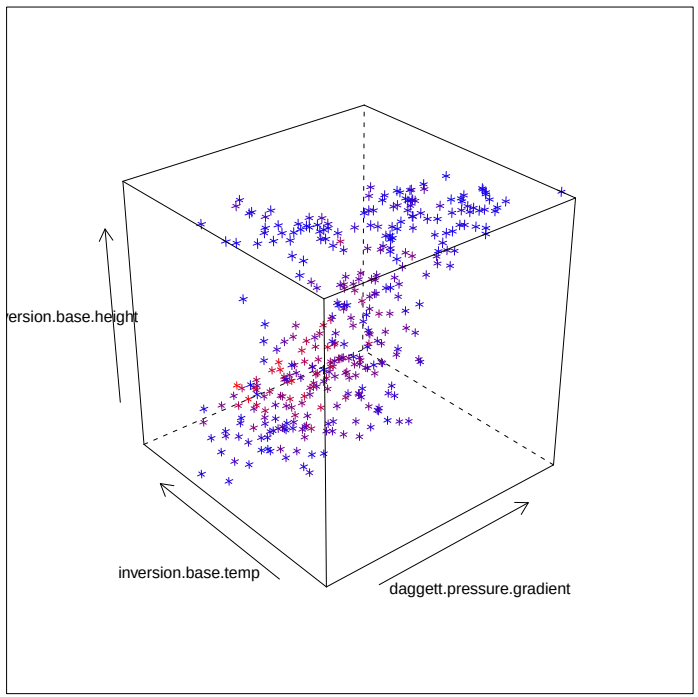
- The idea of principal curves can be extended towards higher-dimensional smooth objects, known as principal surfaces or principal manifolds.
- Already (conceptually) introduced by HS, though a public implementation was never provided:

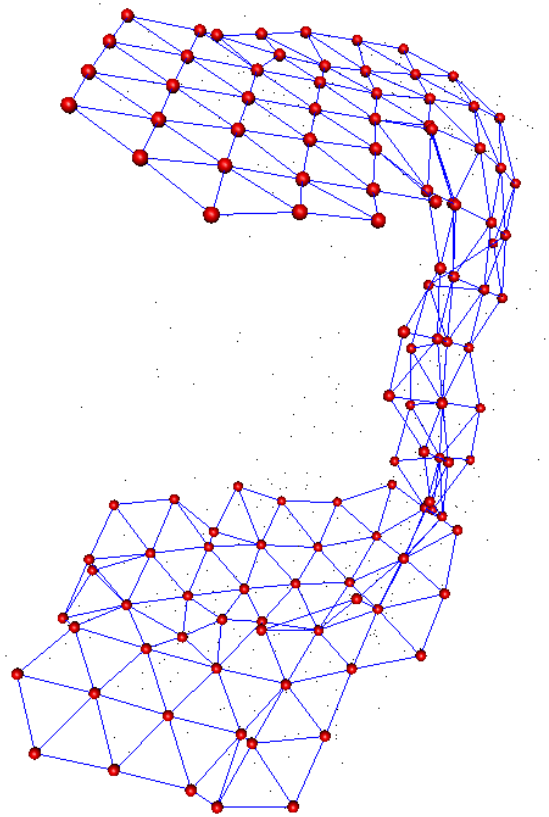


- The image [HTF] suggests a possible application: Classification.
  - has been used for gene classification [G, Chapter 4].

Principal manifolds (cont.)

- Alternative to HS approach: *Local principal surfaces* are constructed through a grid of triangles.
- LPS for Calif. Air Pollution data:





- suggests a second application: Regression.
  - work in progress!