Lecture 2a

block 2

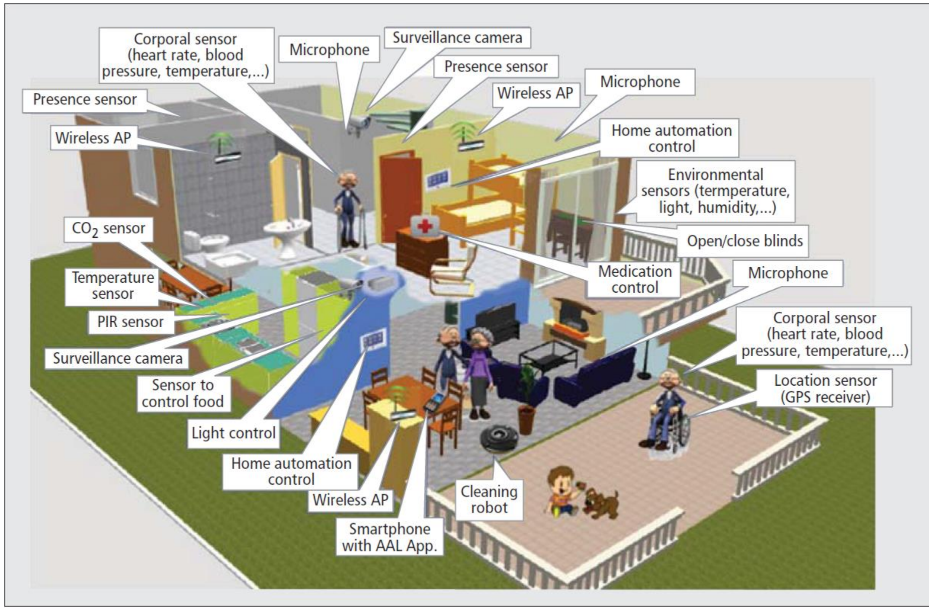
Splines

Generalized additive models

Moving beyond simple models

- Sometimes using simple models (linear regression) is not enough
  - Too simple → ***more flexible models are needed***

Example: Ambient Assisted Living



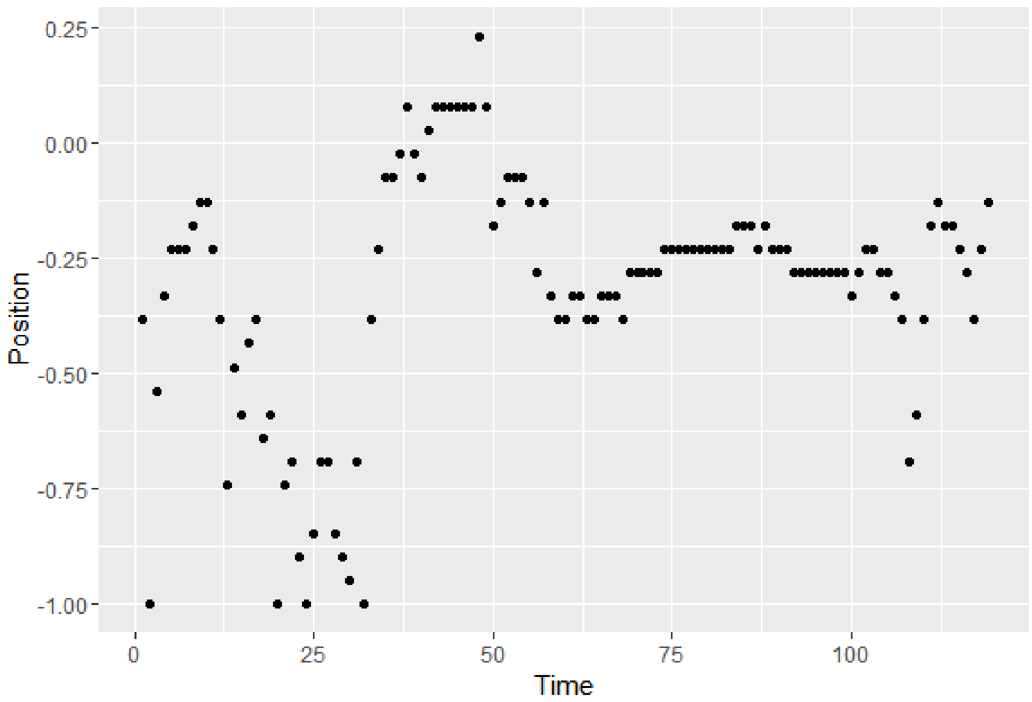
https://bstassen.files.wordpress.com/2015/01/aal.png

- digitally connected and controlled devices for support of people with special needs

- emergency buttons, pressure emergency services with connection to a broader smart home

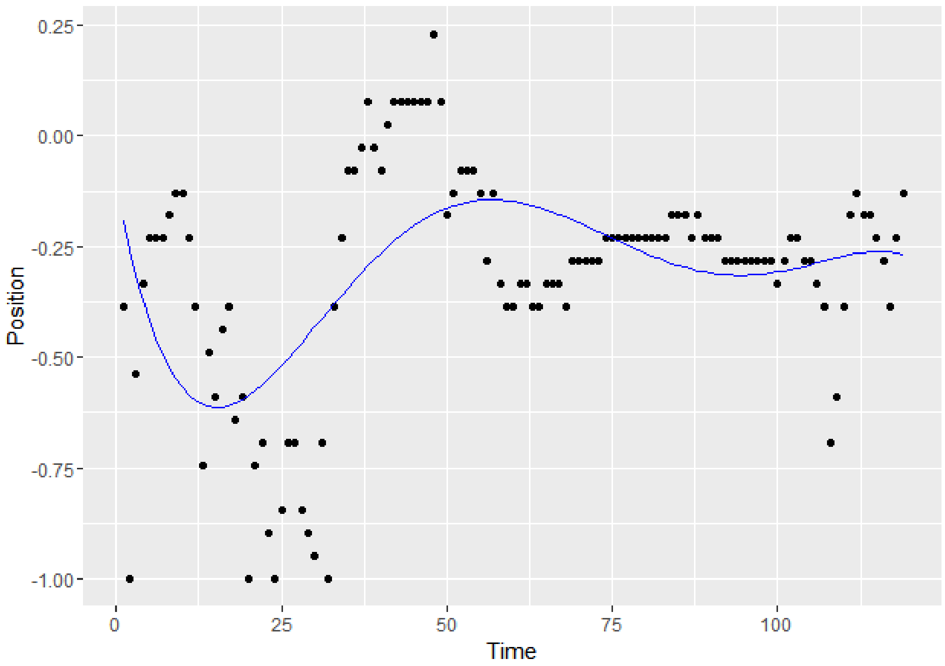
Moving beyond simple models

- Ambient Assisted Living
  - Person’s movement is detected by Radio Signal Strength (RSS) measurements→how to remove noise?



Moving beyond simple models

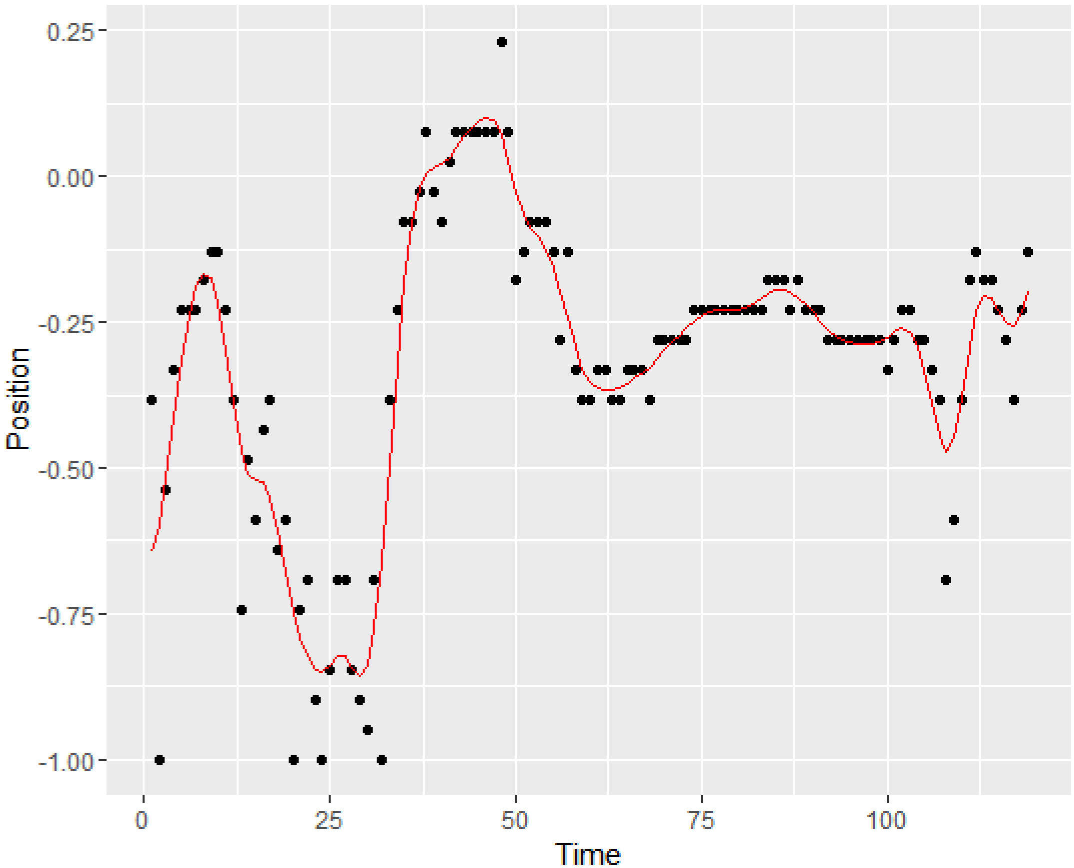
- Attempt 1: 5th degree polynomial



Model underfits data→need more flexible models

Moving beyond simple models

- Using smoothing splines



Basis function expansion

If $y = w_0 + w_1 x_1 + w_2 x_1^2 + w_3 e^{-x_2} + \epsilon$,

Model becomes linear if to recompute:

$$\phi_1(x_1) = x_1$$
$$\phi_2(x_1) = x_1^2$$
$$\phi_3(x_1) = e^{-x_2}$$

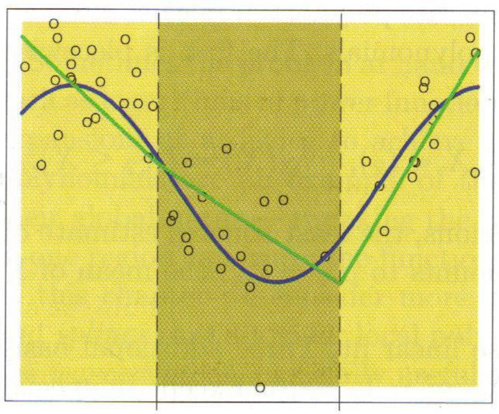
- Any model of the type $Ey = \sum_i w_i \phi_i(\boldsymbol{x})$ can be fit by linear regression!

Constructing a piecewise linear function

Method A. Introduce linear functions on each interval and a set of constraints

(4 free parameters)
$$\begin{cases} y_1 = \alpha_1 x + \beta_1 \\ y_2 = \alpha_2 x + \beta_2 \\ y_3 = \alpha_3 x + \beta_3 \end{cases}$$

$$\begin{cases} y_1(\xi_1) = y_2(\xi_1) \\ y_2(\xi_2) = y_3(\xi_2) \end{cases}$$



Method B. Use a basis expansion (4 free parameters)

$$h_1(X) = 1,\ h_2(X) = X,\ h_3(X) = (X - \xi_1)_+,\ h_4(X) = (X - \xi_2)_+$$

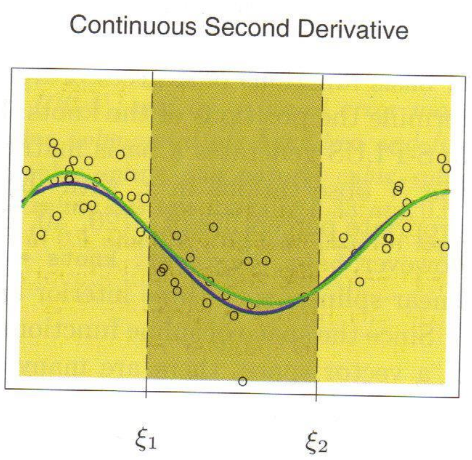
Theorem. The two methods are equivalent.

Splines

- A piecewise polynomial is called an **order-M** (or degree M-1) **spline** if it is continuous and has continuous derivatives up to order M-2 at the knots.
- Equivalent: An order-M spline with K knots:

$$h_j(X) = X^{j-1},\ j = 1, \dots, M$$

$$h_{M+l}(X) = (X - \xi_l)_+^{M-1},\ l = 1, \dots, K$$



- An order-4 (degree-3) spline is called a **cubic spline**

In cubic splines, knot discontinuity is not visible

Natural cubic spline

- A cubic spline f is called **natural cubic spline** if its 2nd and 3rd derivatives are zero at a and b

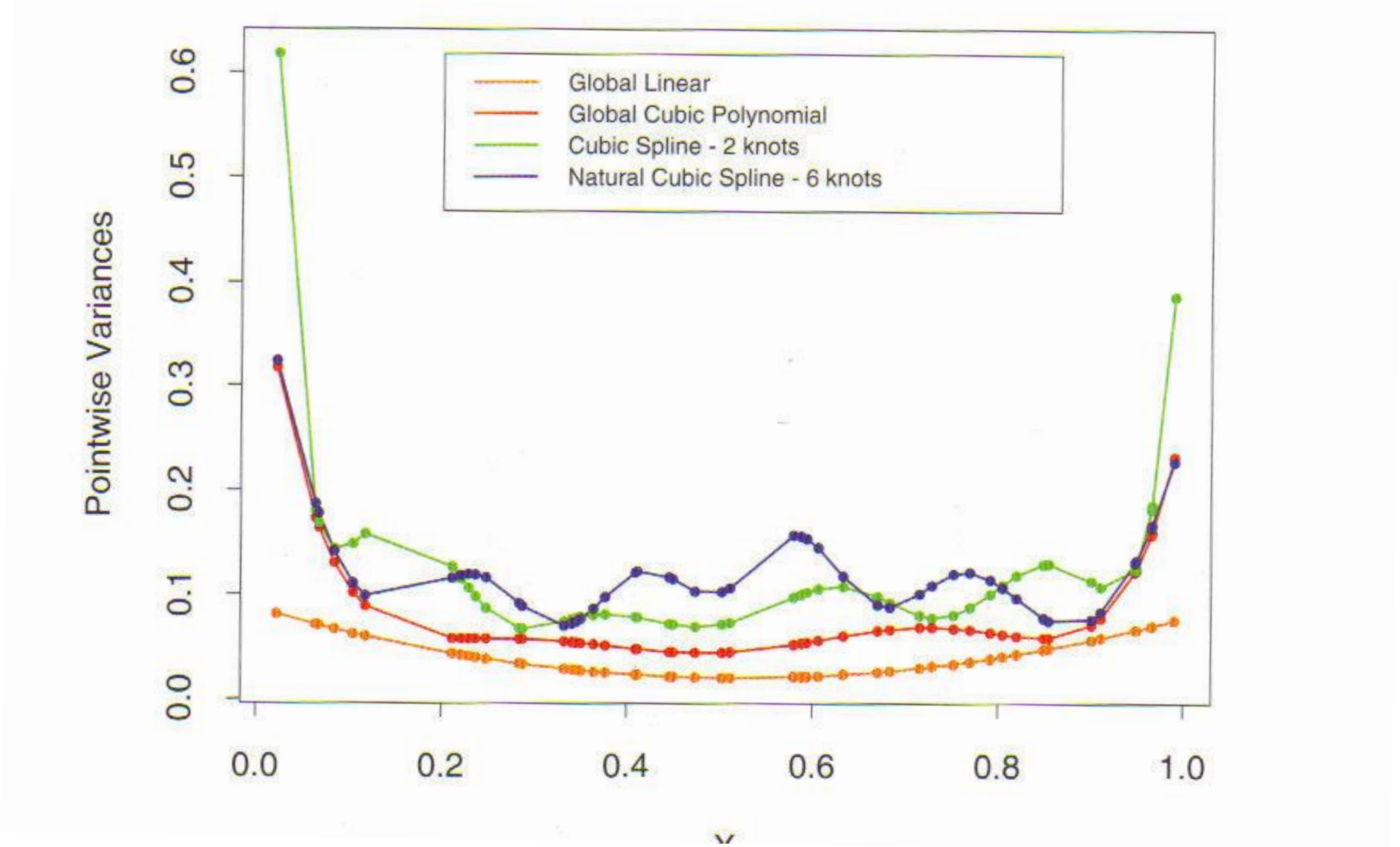
Note that f is linear on extreme intervals

Basis functions of natural cubic splines

$$N_1(X)=1,\ N_2(X)=X,\ N_{k+2}=d_k(X)-d_{K-1}(X),\quad k=1,\ldots,K-2$$

$$\text{where}\quad d_k(X)=\frac{(X-\xi_k)_+^3-(X-\xi_K)_+^3}{\xi_K-\xi_k}$$

Variance of spline estimators – boundary effects



Fitting smooth functions to data

- Minimize

$$RSS(f,\lambda)=\sum_{i=1}^{N}\left(y_i - f(x_i)\right)^2 + \lambda\int\{f''(t)\}^2 dt$$

where λ is **smoothing parameter**.

$\lambda=0$: any function interpolating data

$\lambda=+\infty$: least squares line fit

Optimality of smoothing splines

- The function f minimizing RSS for a given λ is a natural cubic spline with knots at all unique values of x_i (NOTE: N knots!)

- Minimizing sum of squares:

$$f(x) = \sum_{j=1}^{N} N_j(x)\theta_j = N(x)^T \Theta$$

$$RSS(\Theta,\lambda) = (\mathbf{y} - \mathbf{N}\Theta)^T(\mathbf{y} - \mathbf{N}\Theta) + \lambda\Theta^T\Omega_N\Theta$$

$$\{\mathbf{N}\}_{ij} = N_j(x_i) \quad \{\Omega_N\}_{ij} = \int N_i''(t)N_j''(t)dt$$

$$\hat{\Theta} = (\mathbf{N}^T\mathbf{N} + \lambda\Omega_N)^{-1}\mathbf{N}^T\mathbf{y}$$

A smoothing spline is a linear smoother

- Smoothing spline

$$\hat{f} = \mathbf{N}\left(\mathbf{N}^T\mathbf{N} + \lambda\Omega_N\right)^{-1}\mathbf{N}^T\mathbf{y} = \mathbf{S}_\lambda\mathbf{y}$$

  is a **linear smoother**.

- Compare with other smoothers, such as linear regression.

Degrees of freedom

- It can be shown that

$$\mathbf{S}_\lambda = \left(\mathbf{I} + \lambda\mathbf{K}\right)^{-1}$$

where $\mathbf{K}$ is **penalty matrix**

- Eigenvalue decomposition of $\mathbf{K}$:

$$\mathbf{S}_\lambda = \sum_{k=1}^{N} \rho_k(\lambda)\mathbf{u}_k\mathbf{u}_k^T$$

$$\rho_k(\lambda) = \frac{1}{1+\lambda d_k}$$

- d_k and $\mathbf{u}_\mathrm{k}$ are eigenvalues and eigenvectors

Smoothing splines and shrinkage

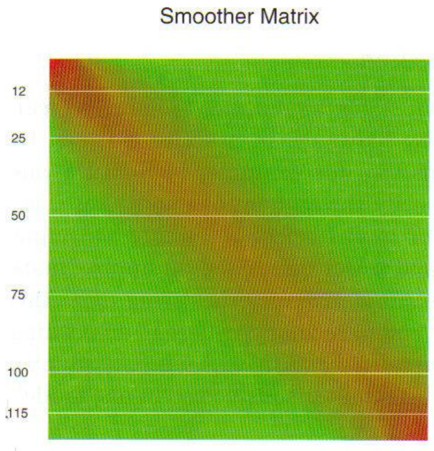
$$\mathbf{S}_{\lambda}\mathbf{y} = \sum_{k=1}^{N} \mathbf{u}_k \rho_k(\lambda) \mathbf{u}_{\mathbf{k}}^{\mathbf{T}} \mathbf{y}$$

- Smoothing spline decomposes vector $\mathbf{y}$ with respect to basis of eigenvectors and shrinks respective contributions

- The eigenvectors ordered by ρ increase in complexity. The higher the complexity, the more the contribution is shrunk.

Penalty and degrees of freedom

- $df_\lambda = trace(\mathbf{S}_\lambda)$ → $df_\lambda = \sum_{k=1}^{N} \frac{1}{1+\lambda d_k}$
- λ increase → df_λ decrease
- higher λ → higher penalization.
- Smoother matrix is has banded nature → local fitting method



Automated selection of smoothing parameters

What can be selected:

Regression splines

- Degree of spline
- Placement of knots

Smoothing spline

- Penalization parameter

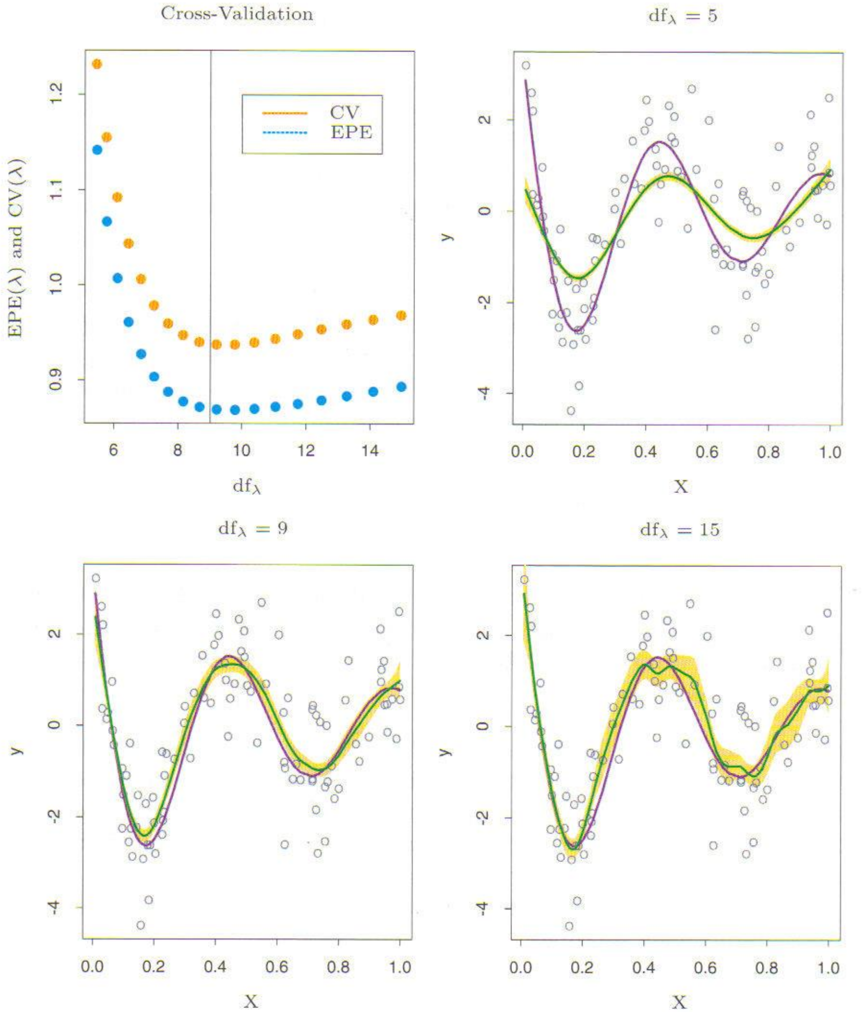
Automated selection of smoothing parameters

$$df_{\lambda} = trace(\mathbf{S}_{\lambda}) = \sum_{k=1}^{N} \frac{1}{1+\lambda d_k}$$

- Use either df_{λ} or λ
  - Given df_{λ} →solve equation→find λ
- Use holdout principle or cross validation for parameter tuning

Automated selection of smoothing parameters

- Bias-variance tradeoff



Multidimensional splines

How to fit data smoothly in higher dimensions?

- Formulate a new problem

$$\min \sum_i \left(y_i - f(x_i)\right)^2 + \lambda J[f]$$

- The solution is **thin-plate splines**

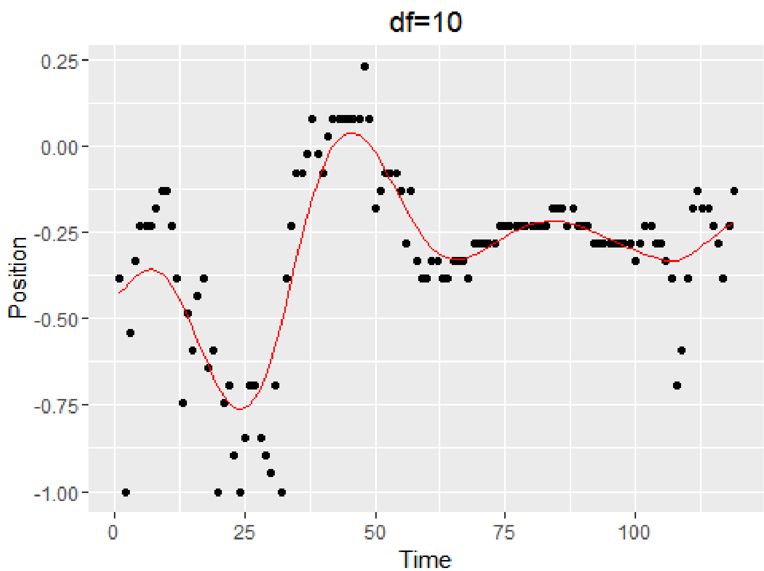
- The solution in 2 dimensions is essentially sum of radial basis functions

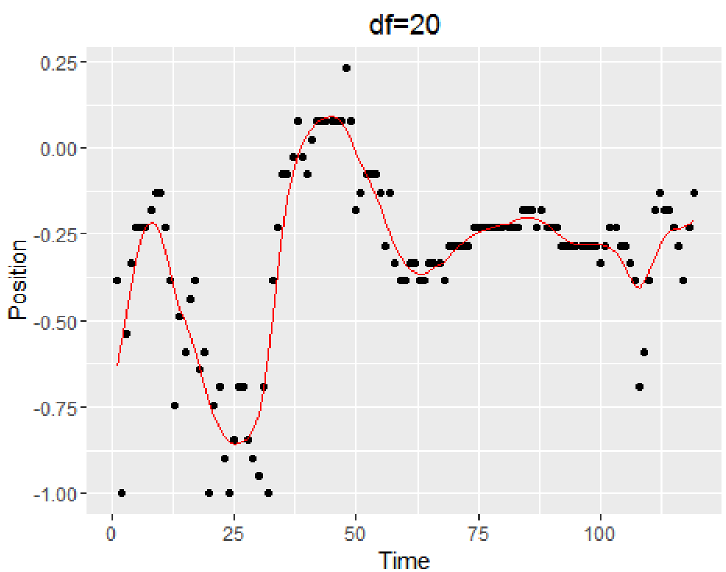
$$f(x) = \beta_0 + \beta^T x + \sum \alpha_j \eta\left(\left\| x - x_j \right\|\right)$$

Splines: R code

- Smoothing splines : smooth.spline()
- Natural clubic splines: ns() in **splines**
- Thin plate splines: Tps() in **fields**

```
res1=smooth.spline(data$Time,data$RSS_anchor2,df=10)
predict(res1,x=data$Time)$y
```





Generalized additive models

- Model

$$Y \sim EF(\mu, \ldots)$$

where

  - $g(\mu) = \alpha + s_1(X_1) + s_2(X_2) + s_p(X_p)$
  - $s_i(X)$ - smoothers, normally splines
  - EF – distribution from exponential family
  - g – Link function

- Often linear terms are often included separately

$$EY = \alpha + s_1(X_1) + \ldots + s_p(X_p) + \sum_{j=1}^{q} \beta_j X_{p+j}$$

Example: EF= normal, EF=Bernoulli (logistic)

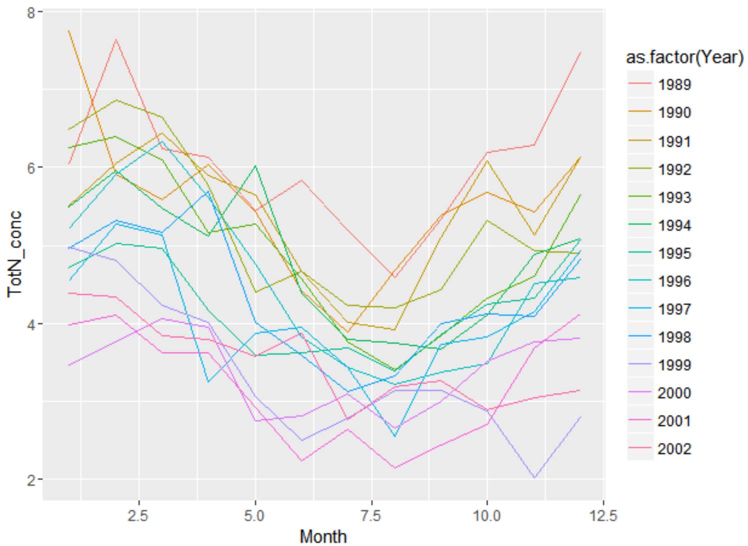
Generalized additive models

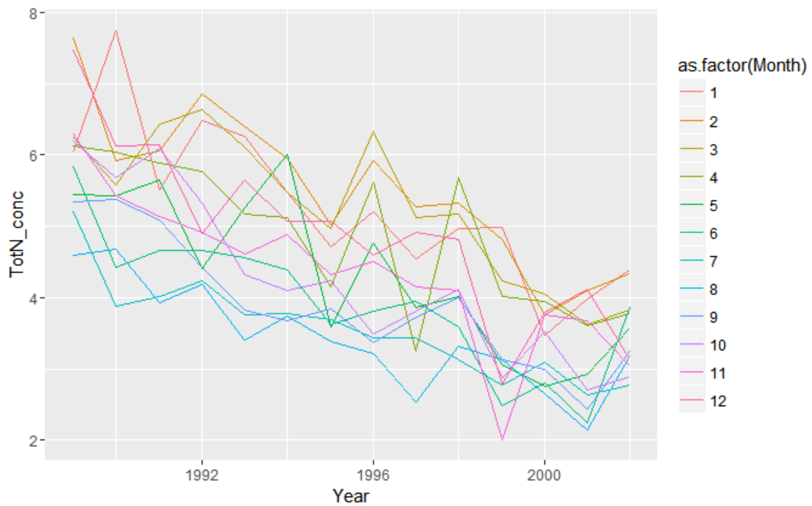
- Sometimes even higher orders are included (thin-plate splines)

$$g(\mu) = \alpha + s_1(X_1) + \ldots + s_p(X_p) + \sum_{j=1}^{q} \beta_j X_{p+j} + s_{12}(X_1, X_2)$$

- Method is reasonable to apply when additivity is observed or admissble

Example: Total Nitrogen level in Rhine river





Estimation of additive models

Estimation by MLE

$$g(\mu) = \alpha + f_1(x_1) + \ldots + f_p(x_p)$$

The backfitting algorithm for Normal model

$$1.\text{Initialize}: \hat{\alpha} = \frac{1}{N}\sum_{i=1}^{N} y_i, \quad \hat{f}_j \equiv 0,\ j = 1,\ldots,p$$

$$2.\text{Cycle}:\ j = 1,\ldots,p,1,\ldots,p,\ldots,1,\ldots,p$$

$$\hat{f}_j \leftarrow s_j\left[\left\{(y_i - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\right\}\right]$$

$$\hat{f}_j \leftarrow \hat{f}_j - \frac{1}{N}\sum_{i=1}^{N} \hat{f}_j(x_{ij})$$

λ in each term can be estimated by CV

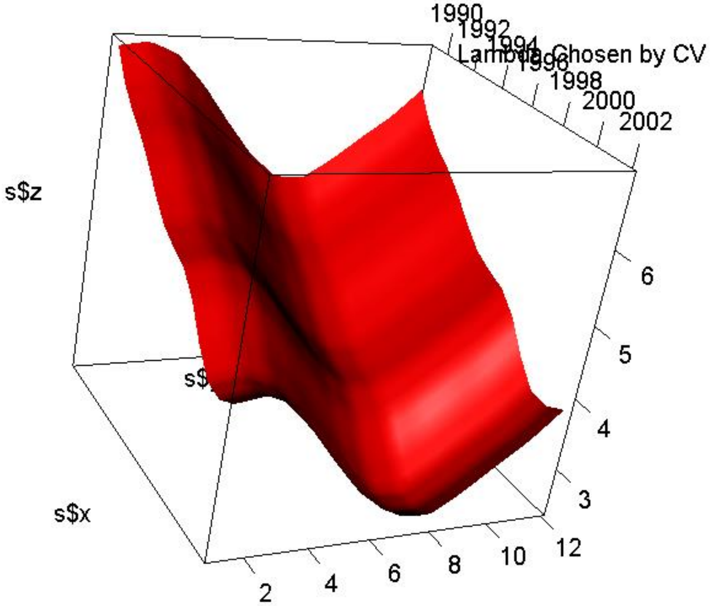
Generalized additive models

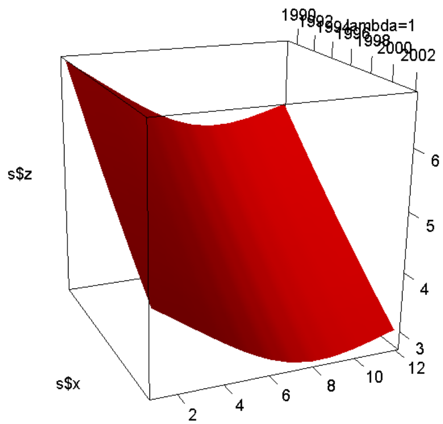
- Example: Modelling the concentration of total nitrogen at Lobith on the Rhine
  - There are seasonal trends (GAM reasonable)
  - Variables
    - Nitrogen level
    - Year
    - Month
- R: package **mgcv** (also package **gam**)
  - gam(formula, family,data,select, method)
    - Select allows for term (variable) selection
  - predict(), plot(), summary()…
  - s(k, sp)
    - k should be the same as the amount of **unique values** of this variable in **smoothing splines**
    - sp - smoothing penalty.

Generalized additive models

- R code

```
river=read.csv2("Rhine.csv")
res=gam(TotN_conc~Year+Month+s(Year)
          +s(Month), data=river)
library(rgl)
library(akima)
s=interp(river$Year,river$Month,
fitted(res))
persp3d(s$x, s$y, s$z, col="red")
```





Generalized additive models

```
> summary(res)

Family: gaussian
Link function: identity

Formula:
TotN_conc ~ Year + Month + s(Year) + s(Month)

Parametric coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 0.0008852  0.0009512   0.931   0.3535
Year        0.0014169  0.0003421   4.142 5.63e-05 ***
Month       0.2517641  0.1048467   2.401   0.0175 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Approximate significance of smooth terms:
           edf Ref.df     F p-value
s(Year)  6.049  7.206 66.72  <2e-16 ***
s(Month) 4.476  5.611 35.45  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Rank: 19/21
R-sq.(adj) =  0.819   Deviance explained = 83.2%
GCV = 0.27689  Scale est. = 0.25638   n = 168
> res$sp
    s(Year)    s(Month)
0.003342167 0.007087835
```

Generalized additive models

- Seeing trend and seasonal pattern

plot(res)