

STK-IN4300

Statistical Learning Methods in Data Science

Riccardo De Bin

debin@math.uio.no

Piecewise polynomials and splines: beyond linear regression

For regression problems:

- usually $f(X) = E[Y|X]$ is considered **linear in X** :
 - easy and convenient **approximation**;
 - **first Taylor** expansion;
 - model easy to **interpret**;
 - **smaller variance** (fewer parameter to be estimated);
- often in reality $f(X)$ is **not linear** in X ;
- IDEA: use **transformations** of X to capture **non-linearity** and fit a linear model in the new derived input space.

Outline of the lecture

- Basis Expansions and Regularization
 - Piecewise polynomials and splines
 - Smoothing splines
 - Selection of the smoothing parameters
 - Multidimensional splines

Piecewise polynomials and splines: linear basis expansion

Consider the following model (**linear basis expansion** in X),

$$f(X) = \sum_{m=1}^M \beta_m h_m(X),$$

where $h_m(X) : \mathbb{R}^p \rightarrow \mathbb{R}$ s denote the m -th transformation of X .

Note:

- the new variables $h_m(X)$ **replace** X in the regression;
- the new model is **linear** in the new variables;
- **usual** fitting procedures are used.

Piecewise polynomials and splines: choices of $h_m(X)$

Typical choices of $h_m(X)$:

- $h_m(X) = X_m$: **original** linear model;
- $h_m(X) = X_j^2$ or $h_m(X) = X_j X_k$: **polynomial** terms,
 - augmented space to achieve **higher-order Taylor** expansions;
 - the number of variables **grow exponentially** ($O(p^d)$, where d is the order of the polynomial, p the number of variables);
- $h_m(X) = \log(X_j), \sqrt{X_j}, \dots$: **non-linear** transformations;
- $h_m(X) = \mathbb{1}(L_m \leq X_k < U_m)$: **indicator** for a region of X_k ,
 - **breaks** the range of X_k into M_k **non-overlapping regions**;
 - **piecewise constant** contribution of X_k .

Piecewise polynomials and splines: introduction

Remarks:

- particular functional forms (e.g., logarithm) are useful in **specific** situations;
- polynomial forms are more **flexible** but **limited** by their global nature;
- **piecewise-polynomials** and **splines** allow for **local** polynomials;
- the class of functions is **limited**,

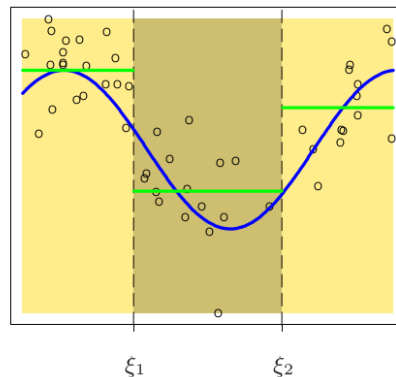
$$f(X) = \sum_{j=1}^p f_j(X_j) = \sum_{j=1}^p \sum_{m=1}^M \beta_{jm} h_{jm}(X_j),$$

by the number of basis M_j used for each component f_j .

Piecewise polynomials and splines: piecewise constant

The **piecewise constant** function:

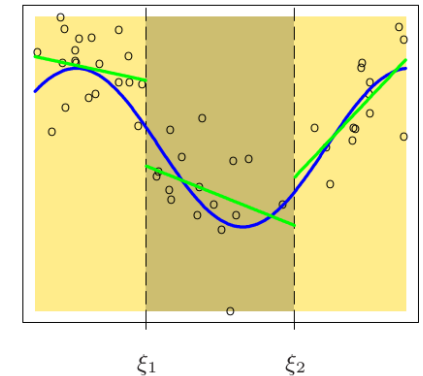
- **simplest** solution;
- three basis functions:
 - $h_1(X) = \mathbb{1}(X < \xi_1)$
 - $h_2(X) = \mathbb{1}(\xi_1 \leq X < \xi_2)$
 - $h_3(X) = \mathbb{1}(X \geq \xi_2)$
- disjoint regions;
- $f(X) = \sum_{m=1}^3 \beta_m h_m(X)$;
- $\hat{\beta}_m = \bar{Y}_m$, the **mean** of Y in the region m .



Piecewise polynomials and splines: piecewise linear

A **piecewise linear** fit:

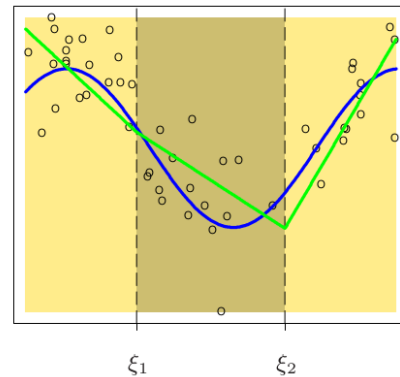
- a **linear fit** instead of a constant fit in each region;
- three **additional** basis functions:
 - $h_4(X) = h_1(X)X$
 - $h_5(X) = h_2(X)X$
 - $h_6(X) = h_3(X)X$
- $\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3$ are the intercepts;
- $\hat{\beta}_4, \hat{\beta}_5, \hat{\beta}_6$ are the slopes;



Piecewise polynomials and splines: piecewise linear

A **continuous piecewise linear** fit:

- **force continuity** at knots;
- generally **preferred to the non-continuous** version;
- add constraints,
 - $\hat{\beta}_1 + \xi_1 \hat{\beta}_4 = \hat{\beta}_2 + \xi_1 \hat{\beta}_5$;
 - $\hat{\beta}_2 + \xi_2 \hat{\beta}_5 = \hat{\beta}_3 + \xi_2 \hat{\beta}_6$;
- 2 restrictions \rightarrow 4 free parameters;

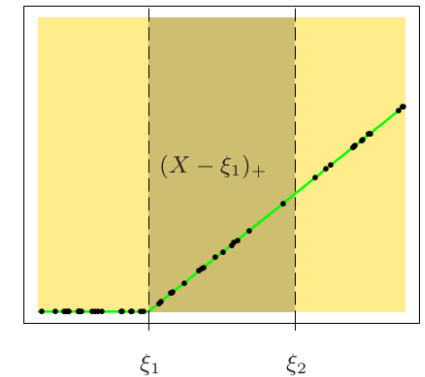


Piecewise polynomials and splines: piecewise linear

The constrain can be **directly incorporated** into the basis functions,

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

where $(\cdot)_+$ denotes the **positive part**.



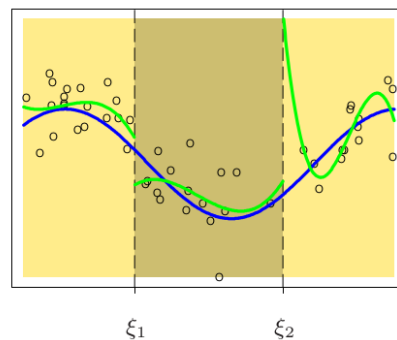
Piecewise polynomials and splines: piecewise cubic polynomials

Further “improvements”:

- **smoother** functions;
- increase the **order** of the polynomials;
- e.g., a **cubic polynomial** in each disjoint region;



discontinuous piecewise cubic polynomials.



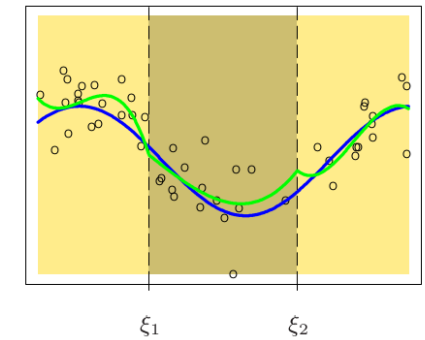
Piecewise polynomials and splines: piecewise cubic polynomials

Also in this case:

- we can **force** the function to be **continuous** at the nodes;
- by **adding constraints**;



continuous piecewise cubic polynomials.



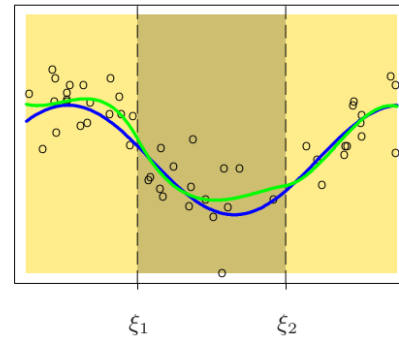
Piecewise polynomials and splines: piecewise cubic polynomials

Since we have third order polynomials:

- we can increase the **order of continuity** at knots;
- not only $f(\xi_k^-) = f(\xi_k^+)$;
- **additionally**, $f'(\xi_k^-) = f'(\xi_k^+)$.



first derivative continuous piecewise cubic polynomials.

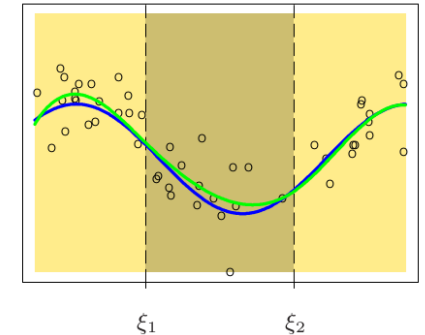


Piecewise polynomials and splines: piecewise cubic polynomials

Finally,

- **further** increase the order of continuity;
- constrain $f''(\xi_k^-) = f''(\xi_k^+)$

↓
cubic splines.



Basis for cubic splines with two knots ξ_1 and ξ_2 :

$$\begin{aligned} h_1(X) &= 1, & h_3(X) &= X^2, & h_5(X) &= (X - \xi_1)_+^3 \\ h_2(X) &= X, & h_4(X) &= X^3, & h_6(X) &= (X - \xi_2)_+^3 \end{aligned}$$

Piecewise polynomials and splines: general order-M splines

In general, an **order-M spline** with knots $\xi_j, j = 1, \dots, K$:

- is a **piecewise-polynomial** of degree $M - 1$;
- has **continuous derivatives** up to order $M - 2$;
- the **general form** of the basis is:

$$\begin{aligned} h_j(X) &= X^{j-1}, j = 1, \dots, M; \\ h_{M+\ell}(X) &= (X - \xi_\ell)_+^{M-1}, \ell = 1, \dots, K; \end{aligned}$$

- e.g., cubic spline $\rightarrow M = 4$;
- cubic splines are the **lowest-order** spline for which the discontinuity at the knots **cannot be seen** by a human eye



no reason to use higher-order splines

Piecewise polynomials and splines: specifications

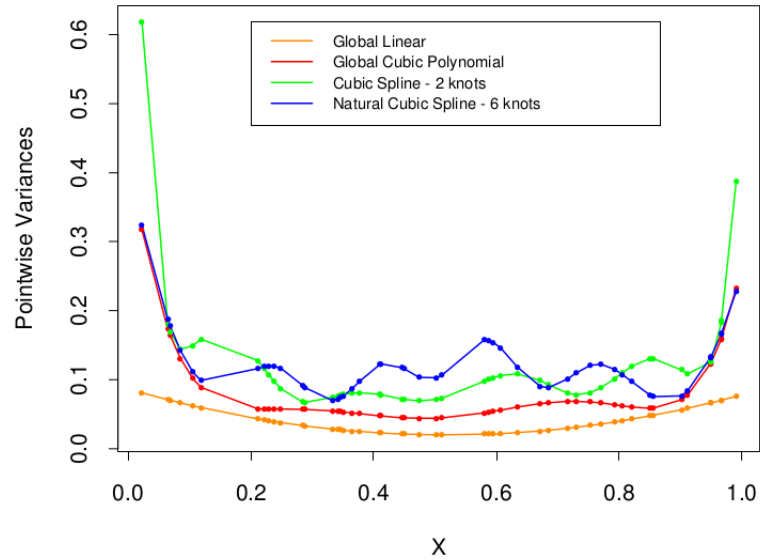
For this kind of splines (a.k.a. regression splines), one needs:

- specify the **order** of the **spline**;
- select the **number** of the **knots**;
- choose their **placement**.

Often:

- use cubic splines ($M = 4$);
- use the **degrees of freedom** to choose the number of knots;
- e.g., for cubic splines,
 - 4 degrees of freedom for the first cubic polynomial;
 - 1 degree of freedom for each knots ($4 - 1 - 1 - 1$);
 - **number of basis = number of knots + 4**;
- use the x_i to **place** the knots;
 - e.g., with 4 knots, 20th, 40th, 60th, 80th percentiles of x .

Piecewise polynomials and splines: natural cubic splines



STK-IN4300: lecture 7

17/ 43

Piecewise polynomials and splines: natural cubic splines

At the boundaries:

- same issues saw for kernel density;
- high variance.

Solution:

- force the function to be linear beyond the boundary knots;
- by adding additional constraints;
- it frees up 4 (2 for each boundary) degrees of freedom.

Basis (derived from those of the cubic splines):

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

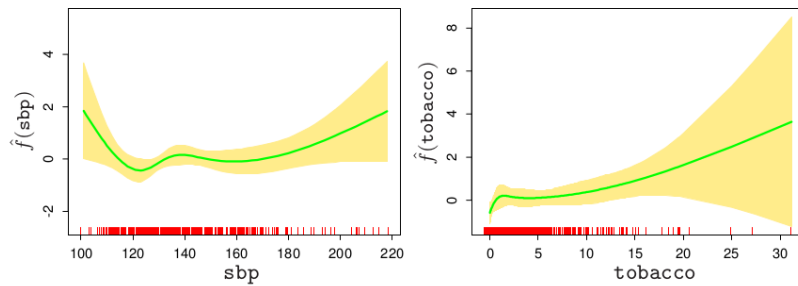
where

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

STK-IN4300: lecture 7

18/ 43

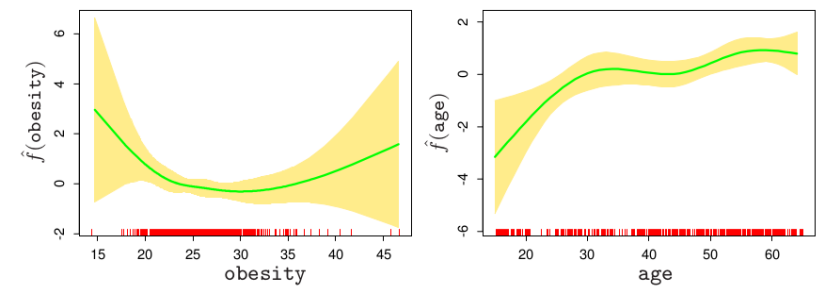
Piecewise polynomials and splines: example



STK-IN4300: lecture 7

19/ 43

Piecewise polynomials and splines: example



STK-IN4300: lecture 7

20/ 43

Smoothing splines: introduction

To avoid choosing the number of knots and their placement:

- use the maximal number (one for each observation);
- control the complexity with a penalty term.

Smoothing splines: minimizer

Consider the minimization problem,

$$\hat{f}(x) = \operatorname{argmin}_{f(x)} \left\{ \sum_{i=1}^N (y_i - f(x_i))^2 + \lambda \int \{f''(t)\}^2 dt \right\}$$

such that $f(x)$ has two continuous derivatives.

Here λ is the smoothing parameter:

- $\lambda = 0 \rightarrow$ no constrain ($f(x)$ can be any function interpolating the data);
- $\lambda = \infty \rightarrow$ least squares line fit (no curvature tolerated).

It can be shown that the unique minimizer is a natural cubic spline with knots at the unique values $x_i, i = 1, \dots, N$.

Smoothing splines: solution

If we consider the natural spline

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

then

$$\hat{\theta} = \operatorname{argmin}_{\theta} \{ (y - N\theta)^T (y - N\theta) + \lambda \theta^T \Omega_N \theta \},$$

where:

- $\{N\}_{ij} = N_j(x_i)$, and $N_j(\cdot)$ are the basis functions;
- $\{\Omega\}_{jk} = \int N_j''(t) N_k''(t) dt$.

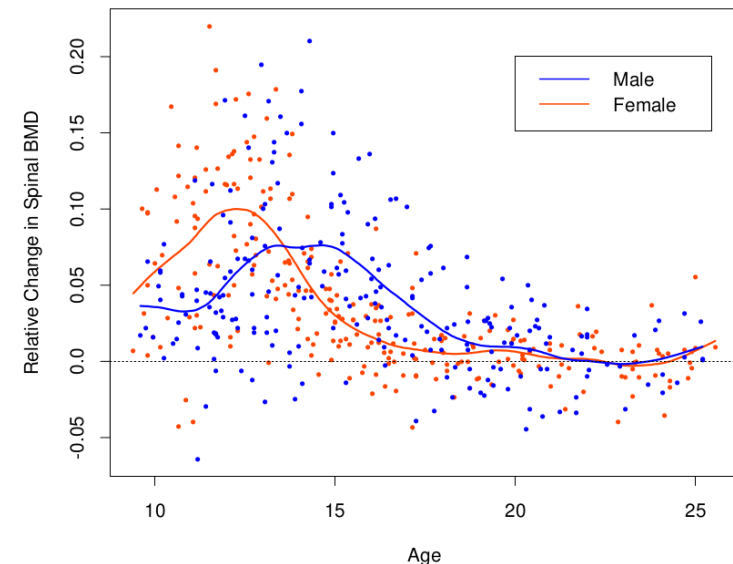
Therefore,

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

and

$$\hat{f}(x) = \sum_{j=1}^N N_j(x) \hat{\theta}_j.$$

Smoothing splines: example



Selection of the smoothing parameters: linear operators

Polynomial splines and smoothing splines are **linear operators**:

- cubic splines: $\hat{f}(x) = \underbrace{B_{\xi}^T (B_{\xi}^T B_{\xi})^{-1} B_{\xi}^T}_{H_{\xi}} y$;
- smoothing splines: $\hat{f}(x) = \underbrace{N(N^T N + \lambda \Omega_N)^{-1} N^T}_{S_{\lambda}} y$.

H_{ξ} is called **hat matrix**, S_{λ} **smoothing matrix**:

- they do **not depend** on y (linear [operator / smoother]);
- are **symmetric** and **semidefinite positive**;
- $H_{\xi} H_{\xi} = H_{\xi}$ (**idempotent**), $S_{\lambda} S_{\lambda} \leq S_{\lambda}$ (**shrinking**);
- H_{ξ} has rank M , S_{λ} has rank N .

Selection of the smoothing parameters: degrees of freedom

The expression $M = \text{trace}(H_{\xi})$ gives:

- the **dimension** of the projection space;
- number** of basis function;
- number** of parameters involved in the fit.

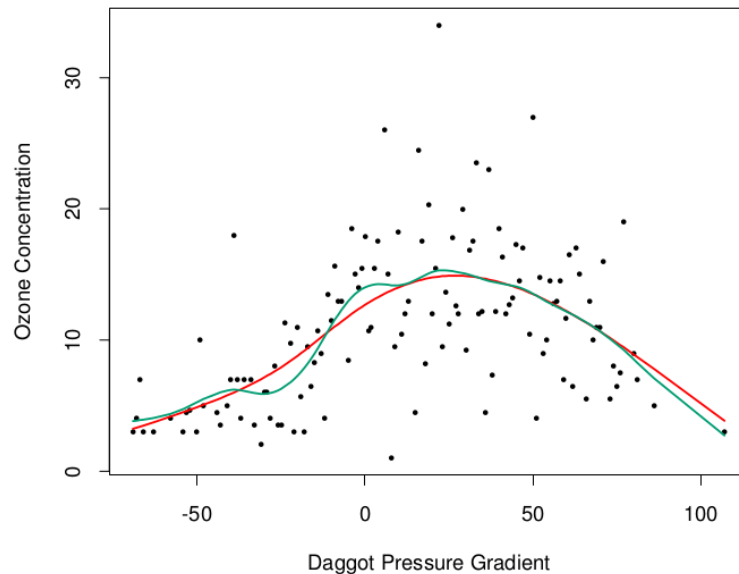
Similarly, we define the **effective degrees of freedom** as

$$\text{df}_{\lambda} = \text{trace}(S_{\lambda}).$$

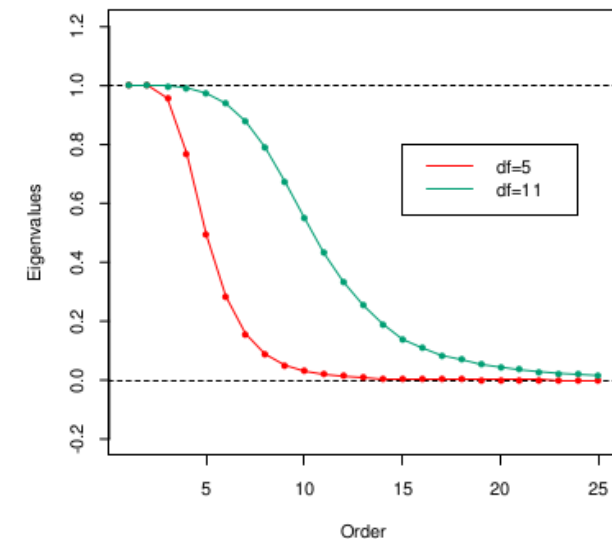
We can **fix** the degrees of freedom and **find** the value of λ :

- e.g., in the last example, $\text{df}_{\lambda} = 12 \rightarrow \lambda = 2.2 \times 10^{-4}$.

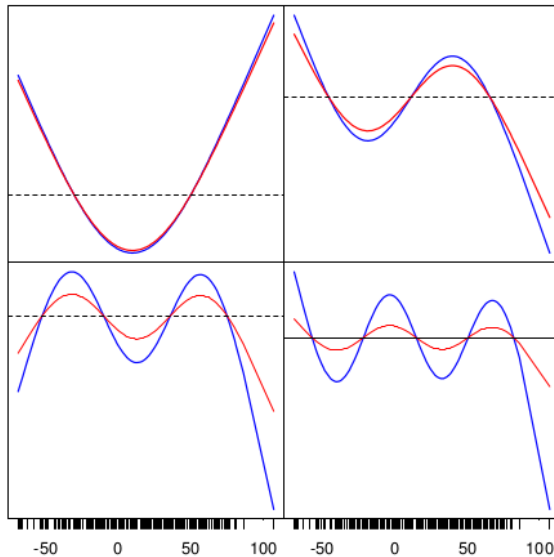
Selection of the smoothing parameters: example



Selection of the smoothing parameters: example



Selection of the smoothing parameters: example



STK-IN4300: lecture 7

29/ 43

Selection of the smoothing parameters: smoother matrices

Let us rewrite S_λ in his Reinsch form,

$$S_\lambda = (I + \lambda K)^{-1},$$

where K (penalty matrix) does not depend on λ .

The eigen-decomposition of S_λ is

$$S_\lambda = \sum_{k=1}^N \rho_k(\lambda) u_k u_k^T$$

with $\rho_k(\lambda) = \frac{1}{1 + \lambda_k}$.

STK-IN4300: lecture 7

30/ 43

Selection of the smoothing parameters: smoother matrices

Note that:

- the eigenvectors are **not affected** by changes in λ ,
 - the **whole family** of smoothing splines indexed by λ has the **same** eigenvectors;
- $S_\lambda y = \sum_{k=1}^N u_k \rho_k(\lambda) \langle u_k, y \rangle$,
 - smoothing splines **decompose** y w.r.t. the basis u_k ;
 - differentially **shrink** the contribution using $\rho_k(\lambda)$;
- the eigenvalues $\rho_k(\lambda) = 1/(1 + \lambda_k)$ are **inverse function** of the eigenvalues d_k of the penalty matrix K , **moderated by λ** ,
 - λ **controls the rate** at which $\rho_k(\lambda)$ decreases to 0.

STK-IN4300: lecture 7

31/ 43

Selection of the smoothing parameters: bias variance trade-off

Consider the following example:

- $Y = f(X) + \epsilon$;
- $f(X) = \frac{\sin(12(X+0.2))}{X+0.2}$;
- $\epsilon \sim N(0, 1)$;
- $X \sim \text{Unif}[0, 1]$;
- $N = 100$.

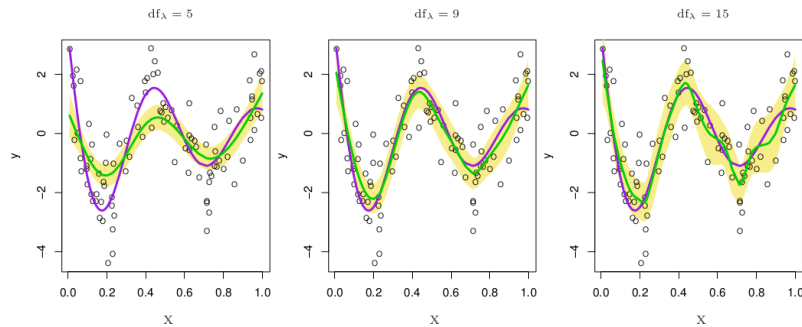
We fit smoothing splines with **three different values** of df_λ :

- $\text{df}_\lambda = 5$;
- $\text{df}_\lambda = 9$;
- $\text{df}_\lambda = 15$.

STK-IN4300: lecture 7

32/ 43

Selection of the smoothing parameters: bias variance trade-off



Selection of the smoothing parameters: bias variance trade-off

In yellow it is shown the area $\hat{f}_\lambda(x) \pm 2 \cdot se(\hat{f}_\lambda(x))$.

Since $\hat{f}(x) = S_\lambda(x)y$,

$$\text{Cov}(\hat{f}(x)) = S_\lambda \text{Cov}(y) S_\lambda^T = S_\lambda S_\lambda^T$$

The diagonal contains the pointwise **variances** at the points x_i .

About the **bias**,

$$\text{Bias}(\hat{f}(x)) = f(x) - E(\hat{f}_\lambda(x)) = f - S_\lambda^T f.$$

We can estimate bias and variance via Monte Carlo methods.

Selection of the smoothing parameters: bias variance trade-off

Note from the last figure:

- $df_\lambda = 5$: **strong bias, low variance**;
 - *trim down the hills and fill the valleys* behaviour;
- $df_\lambda = 9$: the bias is **strongly reduces**, paying a **relatively low** price in terms of variance;
- $df_\lambda = 15$: close to the true function (i.e., low bias), but somehow **wiggly** → **high variance**.

Here the term “bias” is used loosely, in the picture it is actually shown $\hat{f}(x)$, not $E[\hat{f}(x)]$.

Selection of the smoothing parameters: bias variance trade-off

We want to minimize the **expected prediction error**,

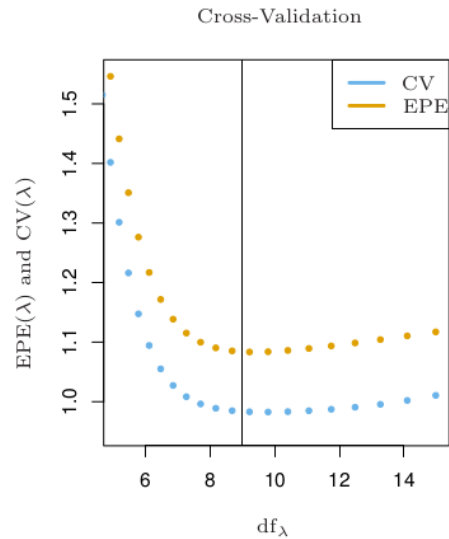
$$EPE(\hat{f}_\lambda(x)) = \text{Var}(Y) + E[\text{bias}^2(\hat{f}_\lambda(x)) + \text{Var}(\hat{f}_\lambda(x))]$$

We do not know the true function → **cross-validation**:

- **K-fold** cross-validation;
- **leave-one-out** cross validation,

$$\begin{aligned} \text{CV}(\hat{f}_\lambda(x)) &= \frac{1}{N} \sum_{i=1}^N (y_i - \hat{f}_\lambda^{(-i)}(x_i))^2 \\ &= \frac{1}{N} \sum_{i=1}^N \left(\frac{y_i - \hat{f}_\lambda(x_i)}{1 - S_{\lambda[i,i]}} \right)^2 \end{aligned}$$

Selection of the smoothing parameters: bias variance trade-off



Multidimensional splines: multidimensional splines

All spline models generalize to **multidimensional** cases.

Consider $X \in \mathbb{R}^2$, then

$$g(X) = \sum_{j=1}^{M_1} \sum_{k=1}^{M_2} \theta_{jk} g_{jk}(X),$$

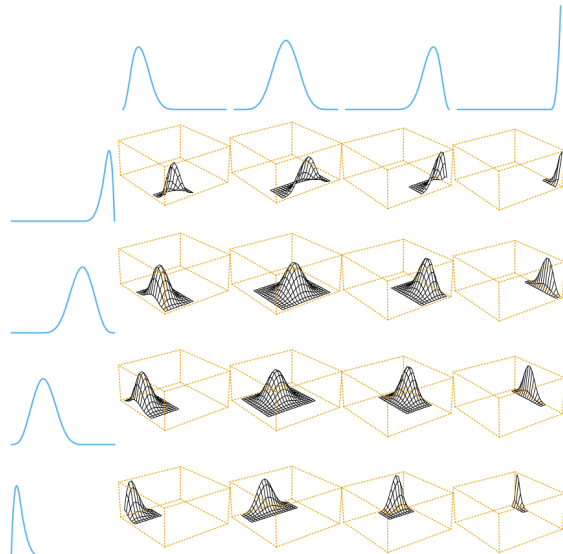
where:

- $g_{jk}(X)$ is an element of the $M_1 \times M_2$ **tensor product basis**

$$g_{jk}(X) = h_{1j}(X_1)h_{2k}(X_2), j = 1, \dots, M_1, k = 1, \dots, M_2;$$

- $h_{1j}(X_1)$ is a **set of M_1 basis** for the coordinate X_1 ;
- $h_{2j}(X_2)$ is a **set of M_2 basis** for the coordinate X_2 ;
- $\theta = \theta_{jk}$ is the $M_1 \times M_2$ -dimensional vector of **coefficients**.

Multidimensional splines: multidimensional splines



Multidimensional splines: multidimensional smoothing splines

Smoothing splines can be extended to **more than one** dimension as well, by **generalizing**

$$\hat{f}(x) = \operatorname{argmin}_{f(x)} \sum_{i=1}^N \{y_i - f(x_i)\}^2 + \lambda J[f(x)],$$

where $J[f(x)]$ takes care of the **"smoothness"** in \mathbb{R}^d .

For example, in the case $d = 2$,

$$J[f(x)] = \int_{\mathbb{R}} \int_{\mathbb{R}} \left[\left(\frac{\partial^2 f(x)}{\partial x_1^2} \right)^2 + 2 \left(\frac{\partial^2 f(x)}{\partial x_1 \partial x_2} \right)^2 + \left(\frac{\partial^2 f(x)}{\partial x_2^2} \right)^2 \right] dx_1 dx_2.$$

Multidimensional splines: multidimensional smoothing splines

The minimizer $\hat{f}(x)$ (in \mathbb{R}^2) is known as **thin-plate spline**:

- for $\lambda \rightarrow 0$, $\hat{f}(x) \rightarrow$ **interpolation** function;
- for $\lambda \rightarrow \infty$, $\hat{f}(x) \rightarrow$ **least square** hyperplane;
- for intermediate values of λ , linear expansion of basis with their coefficients computed by a form of **generalized ridge**.

The solution has the form

$$f(x) = \beta_0 + \beta^T x + \sum_{j=1}^N \alpha_j h_j(x),$$

where $h_j(x) = ||x - x_j|| \log ||x - x_j||$ (**radial basis** function).

Multidimensional splines: multidimensional smoothing splines

Remarks:

- the computational **complexity** is $O(N^3)$;
- often the thin-plate splines are only computed on a **grid of K knots** distributed on the domain (see figure);
- the computational complexity **reduces** to $O(NK^2 + K^3)$;

Simplification:

- by imposing a **specific structure**;
- e.g., **additivity**:
 - $f(x) = \alpha + f_1(x_1) + \dots + f_d(x_d)$ (**GAM**, see next lecture);
 - then

$$J[f(x)] = J(f_1(x_1) + \dots + f_d(x_d)) \\ = \sum_{j=1}^d \int f_j''(t_j) dt_j.$$

Multidimensional splines: multidimensional smoothing splines

