Nonparametric Methods

Wei Li

Syracuse University

Spring 2021

Nonparametric regression

Kernel smoothting (Locally weighted averages)

Local linear regression

Basis expansion

Smoothing splines

Nonparametric regression

Nonparametric regression

Given a random pair $(X, Y) \in \mathbb{R}^p \times \mathbb{R}$, the function

$$f(x) = E(Y \mid X = x)$$

is called the **regression function** (of Y on X) or **conditional expectation function**.

- The basic goal is to estimate f from some i.i.d. sample $(X_i, Y_i)_{i=1}^n$.

For an i.i.d. sample $(X_i, Y_i) \in \mathbb{R}^p \times \mathbb{R}, i = 1, \ldots, n,,$

$$Y_i = f(X_i) + \epsilon_i, \quad i = 1, \ldots, n$$

where $\epsilon_i, i = 1, \ldots, n$ are i.i.d. random errors, with mean zero and satisfy $E(\epsilon_i | X_i) = 0$.

k-nearest-neighbors regression

$$\hat{f}(x) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i$$

where $\mathcal{N}_k(x)$ contains the indices of the k closest points of $x_1, \ldots, x_n$ to x.

One can write the regression prediction as a linear smoother (in $\mathbf{y}$).

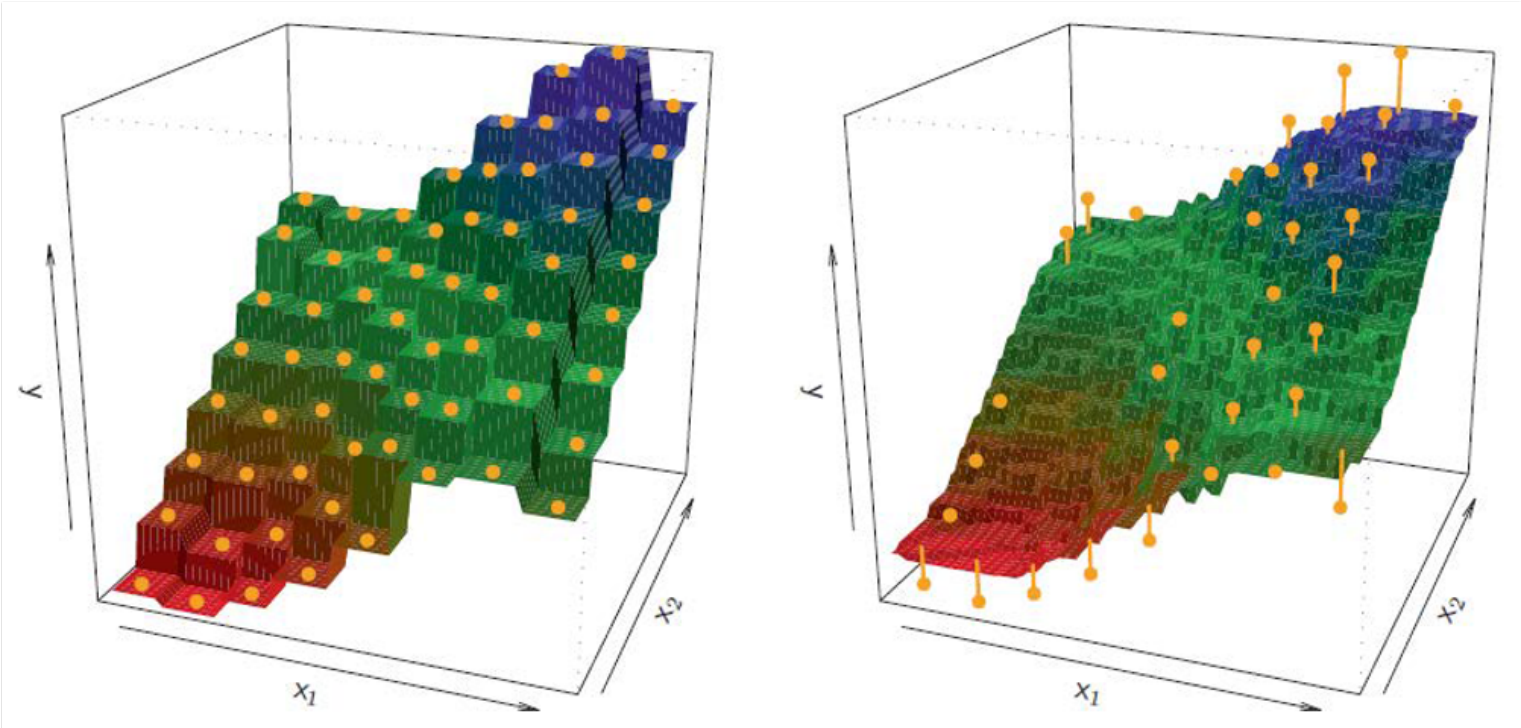
$$\hat{f}(x) = \sum_{i=1}^{n} w_i(x) y_i$$

where the weights $w_i(x), i = 1, \ldots, n$ are defined as

$$w_i(x) = \begin{cases} 1/k & \text{if } x_i \text{ is one of the } k \text{ nearest points to } x \\ 0 & \text{else} \end{cases}$$

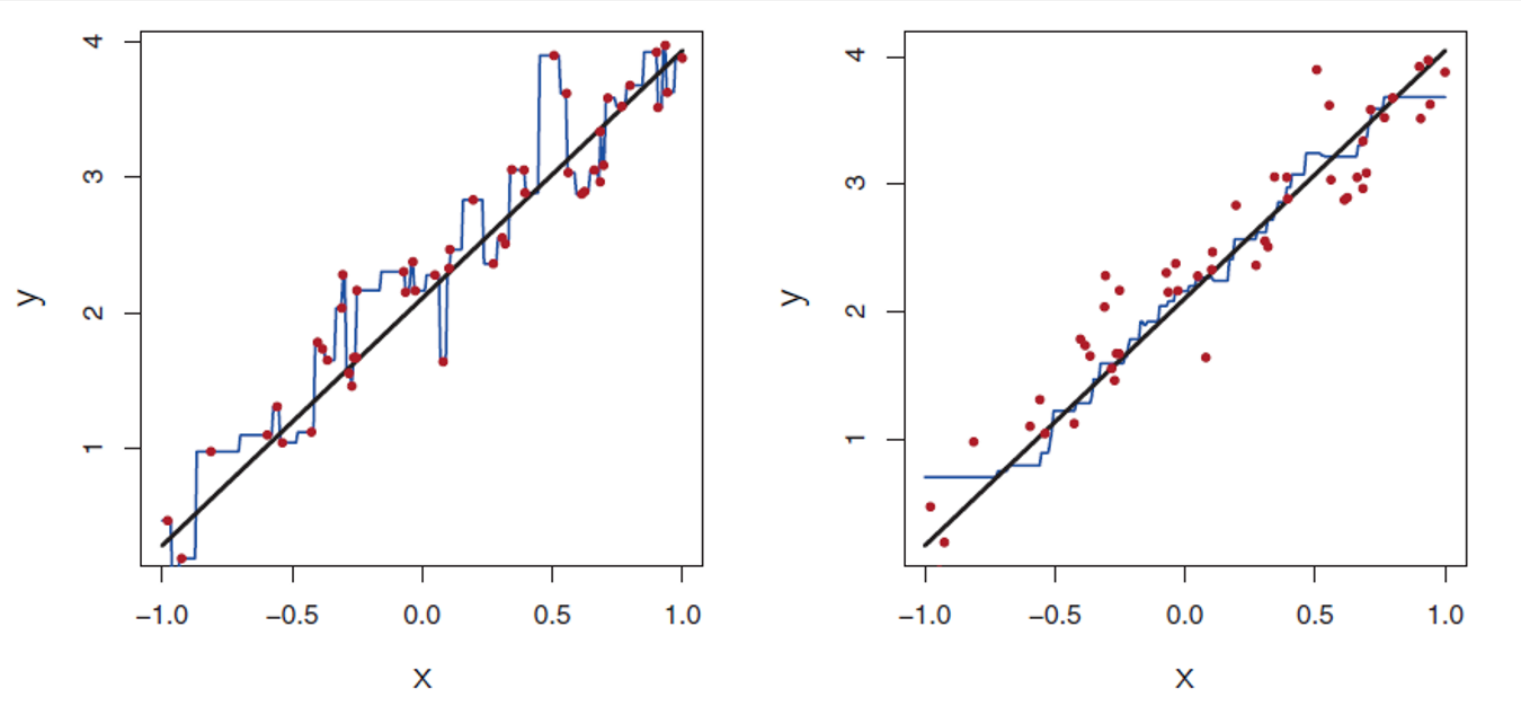
Curse of dimensionality

Suppose $x \in \mathbb{R}^p$.



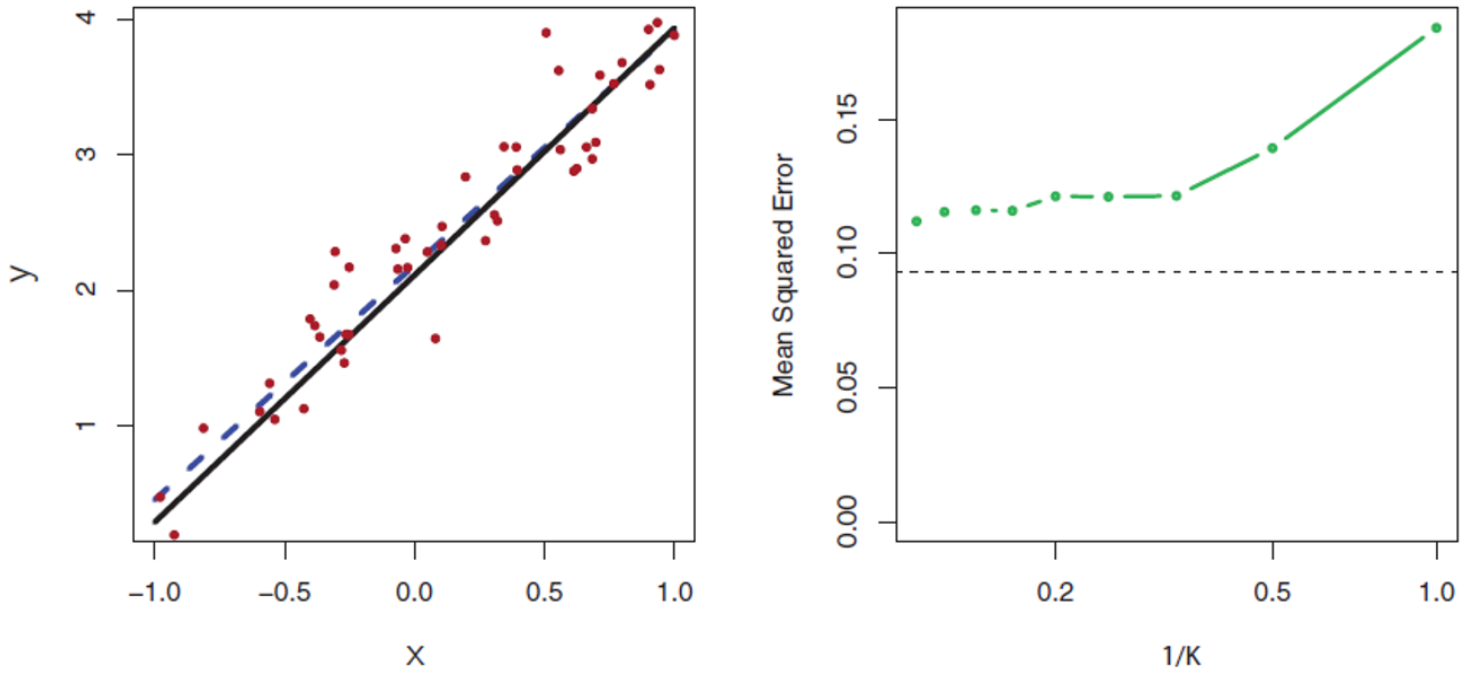
p=2. Left: k=1, most flexible fit, high variation; Right: k=9, smoother fit and less variation

Image source: ISL Chapter 3.



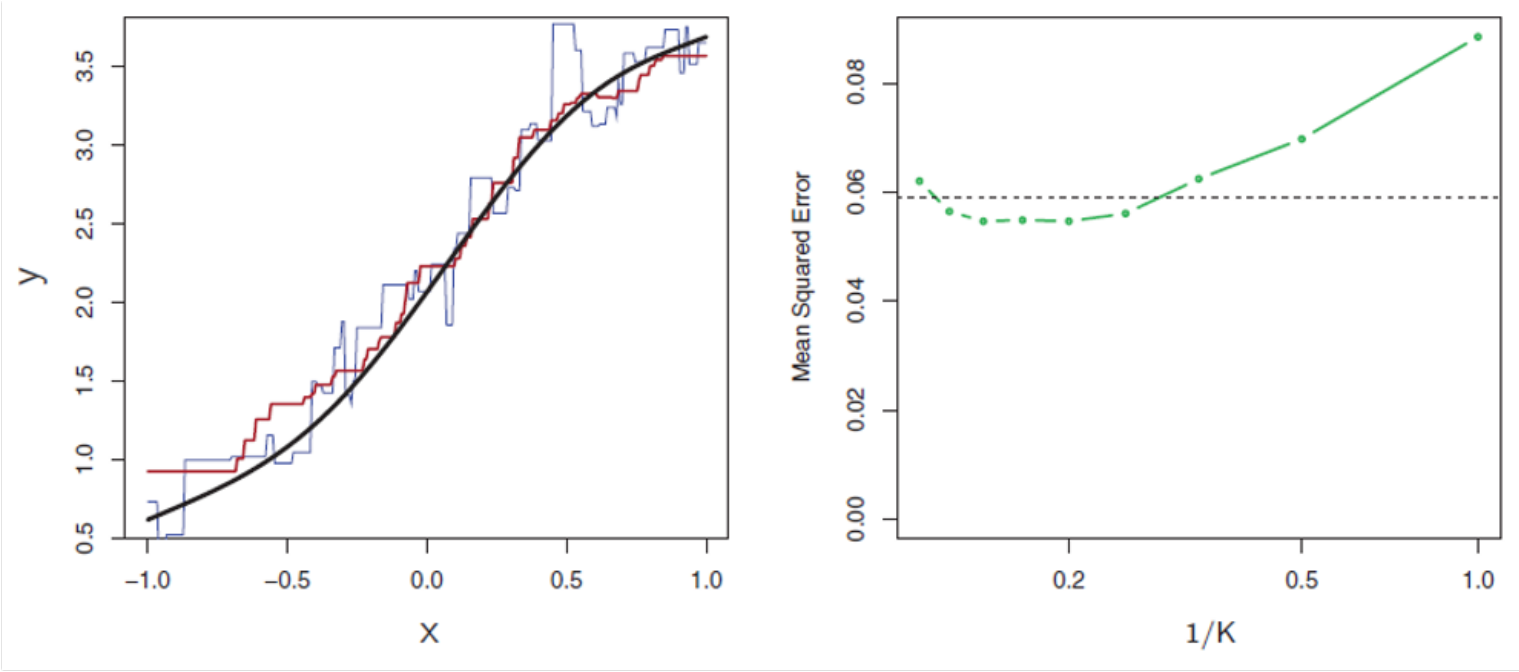
p=1, truth=linear. k=1, k=9

Image source: ISL Chapter 3.



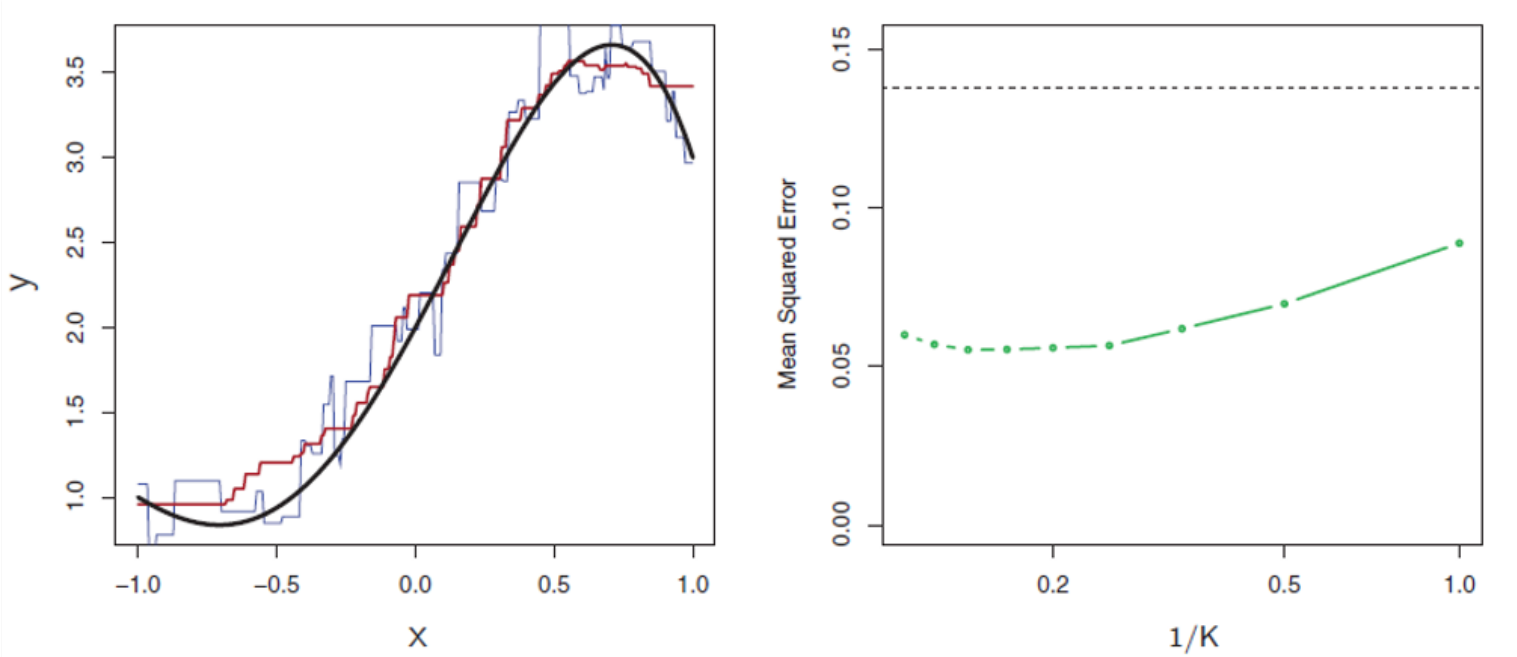
p=1, truth=linear. Left: black(truth), dashed(LS fit); Right: dashed black (LS fit test MSE), dashed green (KNN test MSE)

Image source: ISL Chapter 3.



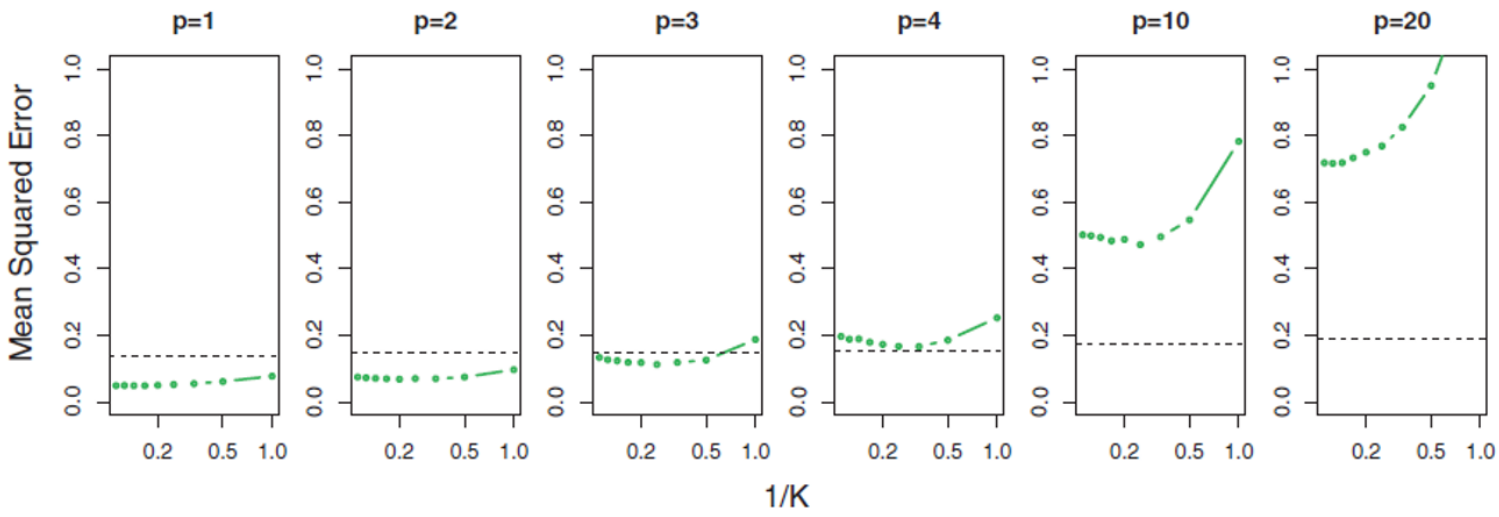
p=1, truth=nonlinear. Left: black(truth), blue(k=1), red(k=9); Right: Test errors

Image source: ISL Chapter 3.



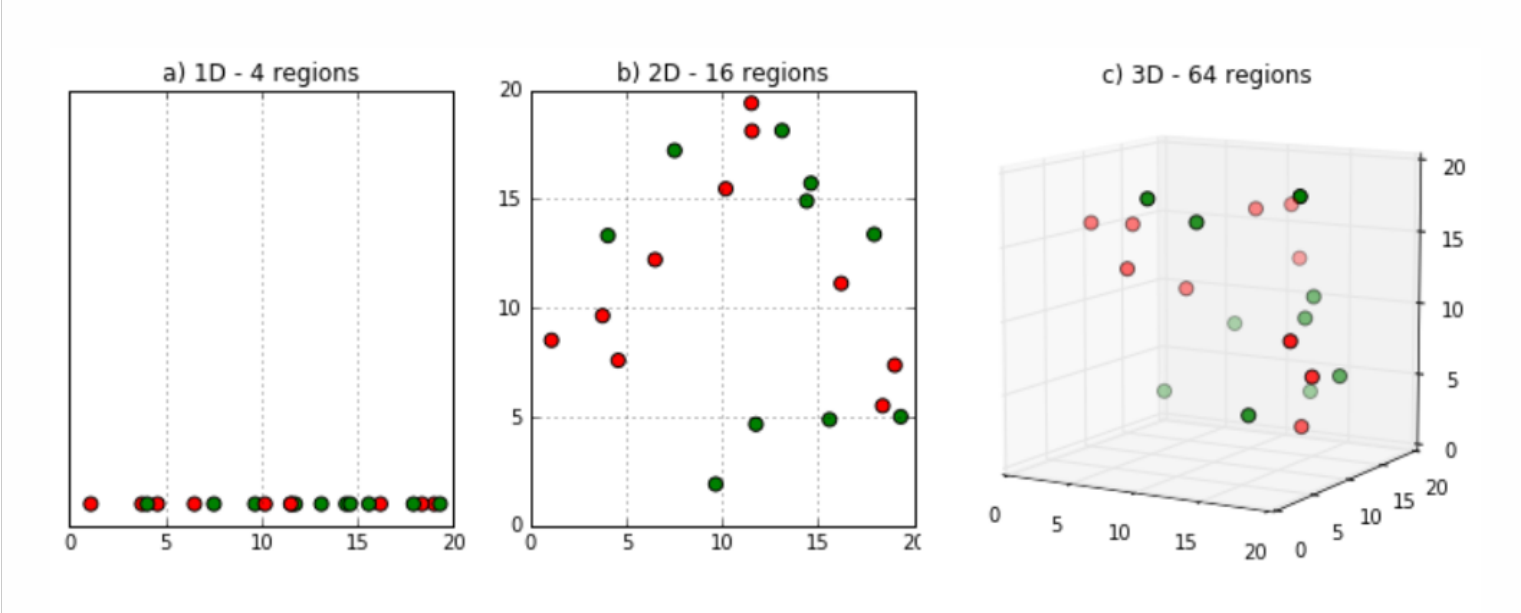
p=1, truth=nonlinear. Left: black(truth), blue(k=1), red(k=9); Right: Test errors

Image source: ISL Chapter 3.



If p is greater than 4, linear regression is superior to KNN even for nonlinear truth (sample size fixed)

Image source: ISL Chapter 3.



Curse of Dimensionality

Image source:
https://deepai.org/machine-learning-glossary-and-terms/curse-of-dimensionality

Convergence rates

The k -nearest-neighbors estimator is universally consistent, which means the MSE $E[(\hat{f}(x) - f(x))^2]$ go to zero as $n \to \infty$, with no assumptions other than $E\left(Y^2\right) \leq \infty$, provided that $k = k_n$ such that $k_n \to \infty$ and $k_n/n \to 0$; e.g., $k = \sqrt{n}$ will do.

Assuming f is Lipschitz continuous, with $k \asymp n^{2/(2+p)}$, it satisfies the MSE $\lesssim n^{-2/(2+p)}$, which is optimal.

However the above error rate $n^{-2/(2+p)}$ exhibits a very poor dependence on the dimension p.

Kernel smoothting (Locally weighted averages)

Kernel smoothting (Locally weighted averages)

Given a bandwidth $\lambda > 0$, the **Nadaraya-Watson** kernel regression estimate is defined as

$$\hat{f}(x_0) = \frac{\sum_{i=1}^{n} K_\lambda(x_0, x_i)\, y_i}{\sum_{i=1}^{n} K_\lambda(x_0, x_i)}$$

Kernel smoothing is also a linear in y with weights

$$w(x_0, x_i) = K_\lambda(x_0, x_i) / \sum_{j=1}^{n} K_\lambda(x_0, x_j).$$

That is $\hat{f}(x_0) = \sum_{i=1}^{n} w(x_0, x_i) y_i$.

$$\hat{f} = (\hat{f}(x_1), \ldots, \hat{f}(x_n))^T = Sy$$

where S is $n \times n$ matrix and $S_{i,j} = w(x_i, x_j)$.

In general we let

$$K_\lambda(x_0, x) = \bar{K}\left(\frac{\|x - x_0\|}{\lambda}\right)$$

Define a kernel function $\bar{K} : \mathbb{R} \to \mathbb{R}$ satisfying

$$\int \bar{K}(t)dt = 1, \quad \int t\bar{K}(t)dt = 0, \quad 0 < \int t^2\bar{K}(t)dt < \infty$$

Three common examples are the box-car kernel:

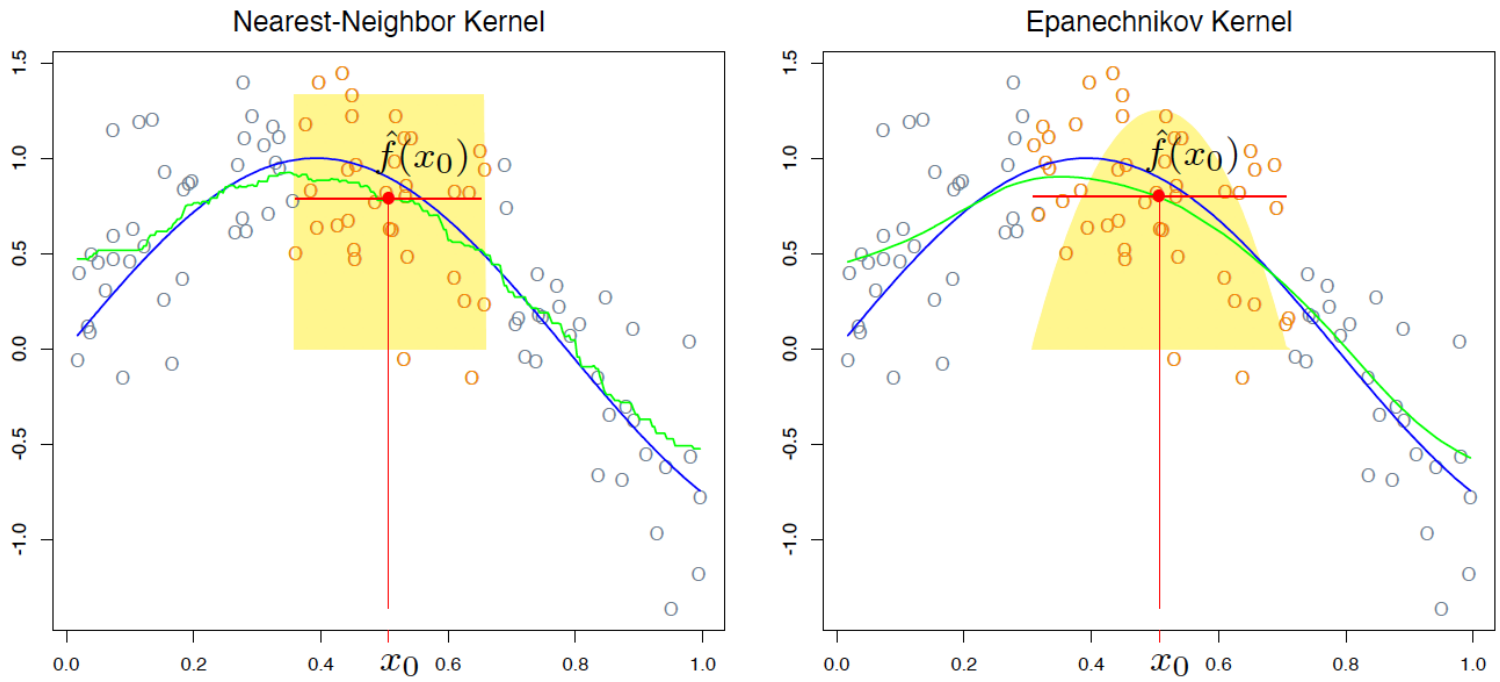
$$\bar{K}(t) = \begin{cases} 1 & |t| \leq 1/2 \\ 0 & \text{otherwise} \end{cases}$$

the Gaussian kernel:

$$\bar{K}(t) = \frac{1}{\sqrt{2\pi}} \exp\left(-t^2/2\right)$$

and the Epanechnikov kernel:

$$\bar{K}(t) = \begin{cases} 3/4\left(1 - t^2\right) & \text{if } |t| \leq 1 \\ 0 & \text{else} \end{cases}$$



The nearest-neighbor kernel compared with Epanechnikov kernel.

ESL Figure 6.1

Convergence rates

In theory, suppose $X \in \mathbb{R}^p$ and f is s-times differentiable ($s \geq 1$), the minimal MSE is $n^{-2s/(2s+p)}$.

Suppose f is at least two-times differentiable, using the Gaussian kernel (second order kernel), one can show

$$\text{Bias}(\hat{f}(x)) \leq \tilde{C}_1 h^2$$
$$\text{Var}(\hat{f}(x)) \leq \frac{\tilde{C}_2}{nh^p}$$

As long as $h \to 0$ and $nh^p \to \infty$, the MSE goes to 0.

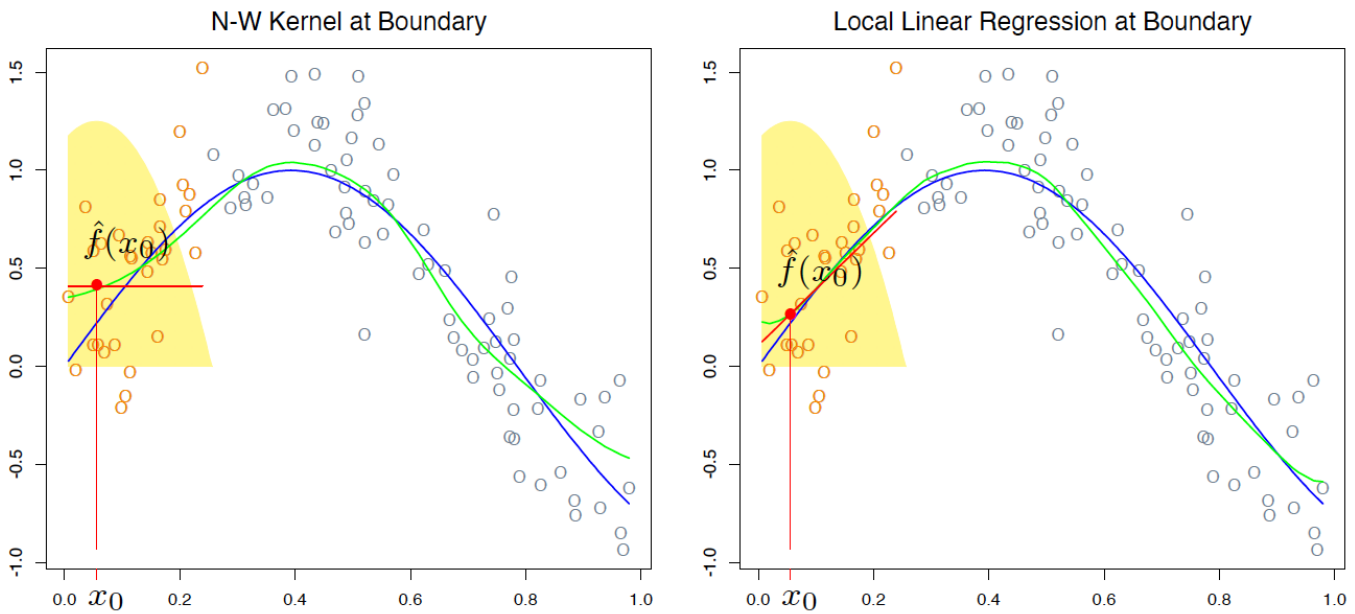
Balancing both errors, $h = n^{-1/(p+4)}$ and the MSE is $n^{-4/(p+4)}$.

- the rate is "saturated" (not improve with $s \geq 2$)
- One may use higher-oder kernel or boundary-corrected kernel to achieve the optimal rate.
- or use the local polynomial regression, or series function based regression.

Local linear regression

Local linear regression

The kernel regression suffers from poor bias at the boundaries of the domain of the inputs $x_1, \ldots x_n$.



NW estimator compared with local linear regression.

ESL Figure 6.1

Use linear regression in local neighborhoods...

At each point, we predict by using linear regression weighting only nearby points.

Let's assume that $p = 1$. At each target point x_0, solve

$$\min_{\alpha(x_0),\beta(x_0)} \sum_{i=1}^{n} K_\lambda\left(x_0, x_i\right)\left[y_i - \alpha\left(x_0\right) - \beta\left(x_0\right) x_i\right]^2$$

Define the vector-valued function $b(x)^T = (1, x)$. Let $\mathbf{B}$ be the $n \times 2$ regression matrix with ith row $b\left(x_i\right)^T$, and $\mathbf{W}\left(x_0\right)$ the $n \times n$ diagonal matrix with i th diagonal element $K_\lambda\left(x_0, x_i\right)$. Then

$$\begin{aligned}\hat{f}\left(x_0\right) &= b\left(x_0\right)^T\left(\mathbf{B}^T\mathbf{W}\left(x_0\right)\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{W}\left(x_0\right)\mathbf{y} \\ &= \sum_{i=1}^{n} l_i\left(x_0\right) y_i = l(x_0)^T y\end{aligned}$$

The weights $l_i\left(x_0\right)$ combine the weighting kernel $K_\lambda\left(x_0, \cdot\right)$ and the least squares operations (the equivalent kernel).

A local polynomial of any degree d:

$$\min_{\alpha(x_0),\beta_j(x_0),j=1,\ldots,d} \sum_{i=1}^{n} K_\lambda\left(x_0, x_i\right) \left[y_i - \alpha\left(x_0\right) - \sum_{j=1}^{d} \beta_j\left(x_0\right) x_i^j \right]^2$$

with solution $\hat{f}\left(x_0\right) = \hat{\alpha}\left(x_0\right) + \sum_{j=1}^{d} \hat{\beta}_j\left(x_0\right) x_0^j$.

$$\hat{f}(x_0) = b(x_0)\left(\mathbf{B}^T \mathbf{W}\left(x_0\right) \mathbf{B}\right)^{-1} \mathbf{B}^T \mathbf{W}(x_0) y = l(x_0)^T y$$

where $b(x) = \left(1, x, \ldots, x^d\right)$, $\mathbf{B}$ is an $n \times (d+1)$ matrix with i-th row $b(x_i) = \left(1, x_i, \ldots, x_i^d\right)$, and $\mathbf{W}(x_0)$ is as before. Note local polynomial regression is a linear smoother, so $\hat{f} = S_\lambda y$ where $[S_\lambda]_{i,j} = l_j(x_i)$.

p>1

let $b(X)$ be a vector of polynomial terms in X of maximum degree d.

For example, with $d = 1$ and $p = 2$ we get $b(X) = (1, X_1, X_2)$; with $d = 2$ we get $b(X) = \left(1, X_1, X_2, X_1^2, X_2^2, X_1 X_2\right)$; and trivially with $d = 0$ we get $b(X) = 1$. At each $x_0 \in \mathbb{R}^p$ solve

$$\min_{\beta(x_0)} \sum_{i=1}^{N} K_\lambda\left(x_0, x_i\right)\left(y_i - b\left(x_i\right)^T \beta\left(x_0\right)\right)^2$$

to produce the fit $\hat{f}\left(x_0\right) = b\left(x_0\right)^T \hat{\beta}\left(x_0\right)$.

Inference about kernel estimation

For the kernel density estimator $\hat{f}$

$$\hat{f}(x) - f(x) = (E\hat{f}(x) - f(x)) + (\hat{f}(x) - E\hat{f}(x))$$

where $B_n(x) = E\hat{f}(x) - f(x) = O\left(h^2\right)$ is the bias and
$\xi_n(x) = \hat{f}(x) - E\hat{f}(x) = O_p\left(\sqrt{1/nh^p}\right)$ is the stochastic variation and
$\sqrt{nh^p}\xi_n(x) \to N\left(0, V_n(x)\right)$ for some variance $V_n(x)$.

A valid pointwise confidence interval for $f(x)$ is

$$\left[\hat{f}(x) - B_n(x) - z_{1-\alpha/2}\sqrt{\frac{V_n(x)}{nh^p}}, \hat{f}(x) - B_n(x) + z_{1-\alpha/2}\sqrt{\frac{V_n(x)}{nh^p}}\right]$$

where $z_{1-\alpha/2}$ is the $1-\alpha/2$ quantile of a standard normal distribution.

To eliminate bias B_n:

- *undersmooth* the kernel estimate, say $h = o\left(\left(nh^p\right)^{-1/4}\right)$, or
- estimate B_n

Selection of the width of window

- If the window is narrow, large variance, small bias.
- If the window is wide, small variance, large bias.
- λ can be chosen using cross-validation.

Structural regression functionns

To fit a regression function $E(Y \mid X) = f(X_1, X_2, \ldots, X_p)$ in $\mathbb{R}^p$, consider analysis-of-variance (ANOVA) decompositions of the form

$$f(X_1, X_2, \ldots, X_p) = \alpha + \sum_j g_j(X_j) + \sum_{k<\ell} g_{k\ell}(X_k, X_\ell) + \cdots$$

introducing structure by eliminating some of the higher-order terms.

An iterative backfitting algorithms can be employed to fit models of above type.

Varying coefficients models

Divide the p predictors in X into a set $(X_1, X_2, \ldots, X_q)$ with $q < p$, and the remainder of the variables we collect in the vector Z.

Assume the conditionally linear model

$$f(X) = \alpha(Z) + \beta_1(Z)X_1 + \cdots + \beta_q(Z)X_q$$

For given Z, this is a linear model, but each of the coefficients can vary with Z. Fit by locally weighted least squares:

$$\min_{\alpha(z_0),\beta(z_0)} \sum_{i=1}^{n} K_\lambda(z_0, z_i)\left(y_i - \alpha(z_0) - x_{1i}\beta_1(z_0) - \cdots - x_{qi}\beta_q(z_0)\right)^2$$

Local likelihood

Associated with each observation y_i is a parameter $\theta_i = \theta(x_i) = x_i^T \beta$ linear in the covariate(s) x_i, and inference for β is based on the loglikelihood $l(\beta) = \sum_{i=1}^n l\left(y_i, x_i^T \beta\right)$.

Model $\theta(x)$ by using the likelihood local to x_0 for inference of $\theta(x_0) = x_0^T \beta(x_0)$

$$l(\beta(x_0)) = \sum_{i=1}^n K_\lambda(x_0, x_i)\, l\left(y_i, x_i^T \beta(x_0)\right)$$

Local version of multi-class linear logistic regression model

The data consist of features x_i and an associated categorical response $y_i \in \{1, 2, \ldots, J\}$, and the linear model has the form

$$\Pr(Y = j \mid X = x) = \frac{e^{\beta_{j0} + \beta_j^T x}}{1 + \sum_{k=1}^{J-1} e^{\beta_{k0} + \beta_k^T x}}$$

where $\beta_{J0} := 0$ and $\beta_J := 0$ by the definition of the model.

The local log-likelihood for this J class model can be written

$$\begin{aligned} &\sum_{i=1}^{n} K_\lambda(x_0, x_i) \Big\{ \beta_{y_i 0}(x_0) + \beta_{y_i}(x_0)^T (x_i - x_0) \\ &- \log \Big[1 + \sum_{k=1}^{J-1} \exp\Big(\beta_{k0}(x_0) + \beta_k(x_0)^T (x_i - x_0) \Big) \Big] \Big\} \end{aligned}$$

Basis expansion

Polynomial regression

Suppose $p = 1$.

Replace simple linear regression

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

With polynomial regression

$$y_i = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \ldots + \beta_d x_i^d + \epsilon_i$$

Higher degree polynomials give more flexible fit.

Polynomial basis functions have limitations: impose global structure on the non-linear functions of predictors and leads to global fitting.

One obvious local method would be to fit polynomials piecewise. For example,

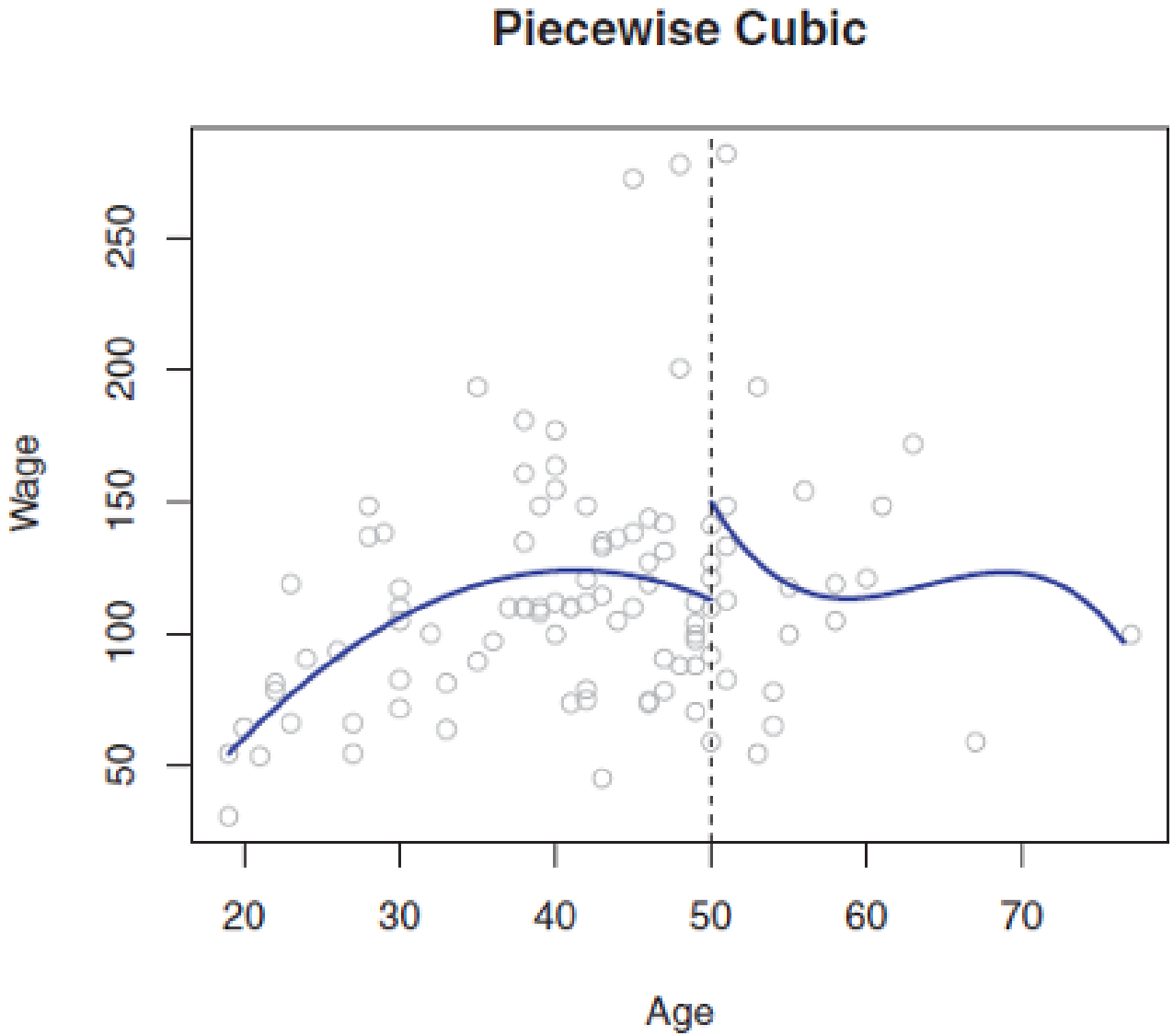
Piecewise cubic (degree=3) with a single cutpoint (knot) at $c = 50$.

$$y_i = \begin{cases} \beta_{01} + \beta_{11} x_i + \beta_{21} x_i^2 + \beta_{31} x_i^3 + \epsilon_i & \text{if } x_i < c \\ \beta_{02} + \beta_{12} x_i + \beta_{22} x_i^2 + \beta_{32} x_i^3 + \epsilon_i & \text{if } x_i \geq c \end{cases}$$

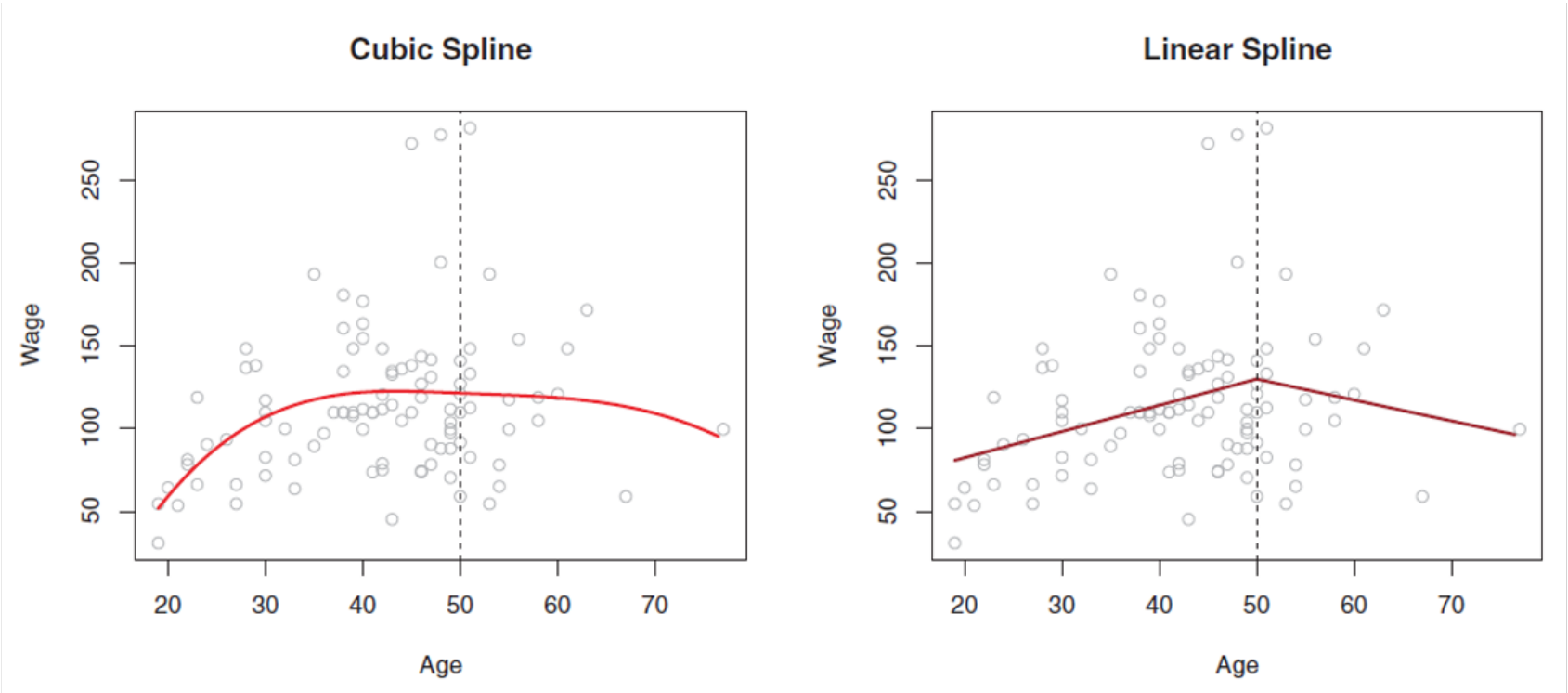
In general, if we place K different knots throughout the range of X, then we will end up fitting K + 1 different cubic polynomials.

Polynomial splines

Note the piecewise polynomials lack of continuity at the knots.



ISL: Fig 7.3


Cubic Spline
Linear Spline
Wage
Age
50
100
150
200
250
20
30
40
50
60
70

ISL: Fig 7.3.

This motivate the polynomial splines:

A d th-degree spline f is a piecewise polynomial function of degree d that is continuous and has continuous derivatives of degrees $1, \ldots, d-1$, at its knot points. Specifically, there are $t_1 < \ldots < t_K$ such that f is a polynomial of degree d on each of the intervals

$$(-\infty, t_1], [t_1, t_2], \ldots, [t_K, \infty)$$

and $f^{(j)}$ is continuous at $t_1, \ldots, t_K$, for each $j = 0, 1, \ldots, d-1$

e.g.: cubic spline, d=3, linear, d=1

A degree-d polynomial spline has the basis given by the "truncated power basis":

$$h_j(x) = x^{j-1}, j = 1, \ldots, d+1$$
$$h_{d+1+l}(x) = (x - t_l)_+^d, \quad l = 1, \ldots, K$$

Unconstrained piecewise polynomial would have d.f.
$(K+1)(d+1) = Kd + d + K + 1$. But, the polynomial splines have d constraints for each of K knots: left with $df = d + K + 1$.

- Cubic: $d = 3, df = K + 4$
- Linear: $d = 1, df = K + 2$

If we use splines as basis funciton for regression fit, the regression function is then called the **regression splines**.

Write a set of $d + \mathrm{K} + 1$ basis functions, known as a degree-d spline basis with K knots.

$$b_1(x_i), b_2(x_i), \ldots, b_{K+d+1}(x_i)$$

Then fit, for example using cubic splines (d = 3):

$$y_i = \beta_1 1 + \beta_2 b_2(x_i) + \beta_3 b_3(x_i) + \cdots + \beta_{K+4} b_{K+4}(x_i) + \epsilon_i$$

Natural splines

Regression splines often give superior results to polynomial regressions.

A problem with regression splines is that the estimates tend to display erratic behavior, i.e., they have high variance, at the boundaries of the input domain. This only gets worse as the polynomial order k gets larger.

natural (cubic) splines is invented which are regression splines with additional boundary constraints: the function is required to be linear at the boundary (in the region where X is smaller than the smallest knot, or larger than the largest knots).

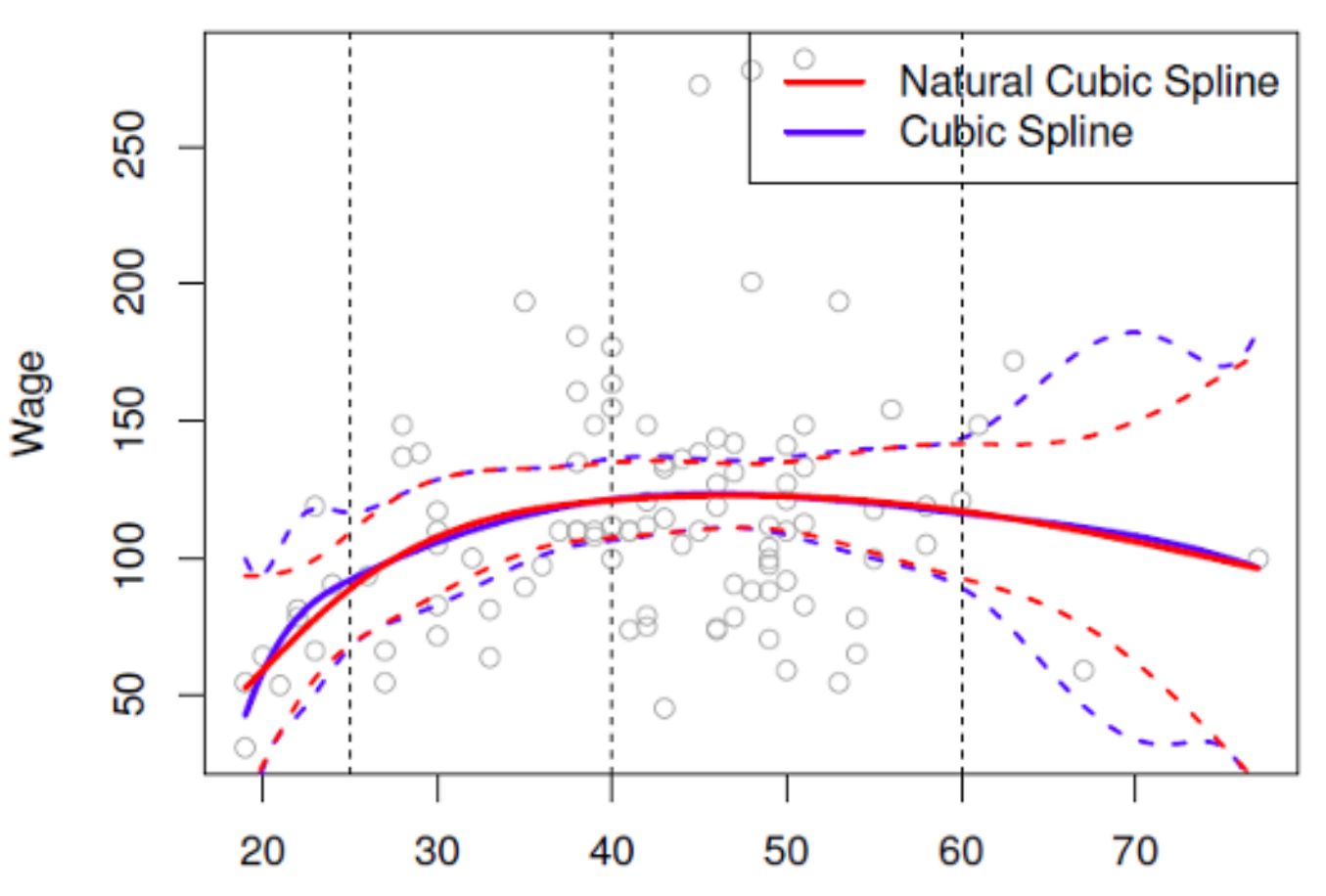
A natural spline of degree d, with knots at $t_1 < \ldots < t_K$, is a piecewise polynomial function f such that

- f is a polynomial of degree d on each of $[t_1, t_2], \ldots, [t_{K-1}, t_K]$
- f is a polynomial of degree $(d-1)/2$ on $(-\infty, t_1]$ and $[t_K, \infty)$
- f is continuous and has continuous derivatives of orders $1, \ldots, d-1$ at $t_1, \ldots, t_K$

The dimension of the span of d-th degree natural splines with knost $t_1, \ldots, t_K$ is given by K itself, so independent of the degree.

$$\underbrace{(d+1)\cdot(K-1)}_{a} + \underbrace{\left(\frac{(d-1)}{2}+1\right)\cdot 2}_{b} - \underbrace{d\cdot K}_{c} = K$$

- a is the number of free parameters in the interior intervals $[t_1, t_2], \ldots, [t_{K-1}, t_K]$ - b is the number of free parameters in the exterior intervals $(-\infty, t_1], [t_p, \infty)$, - c is the number of constraints at the knots $t_1, \ldots, t_p$. - the total dimension is p



A natural cubic spline with 15 d.f. compared to a degree-15 polynomials (using up to X^{15}).

Better yet, the most popular class of splines are called **B-splines**.

ISL: Fig 7.4

B-Splines

- Assume the X is supported on $[0,1]$. Let a sequence of knots $\{t_i : 0 = t_0 < \cdots < t_{K+1} = 1\}$ be a partition of the interval $[0,1]$ into $K+1$ subintervals. (t_0, t_{K+1} called boundary knots, the rest K interior knots).
- To construct B-spline basis on $[0,1]$, extended knots are required and the actual values of these additional knots beyond the boundary are arbitrary, and it is customary to make them all the same and equal to t_0 and t_{K+1}. That is,
  $0 = t_{-(q-1)} = \cdots = t_0 < t_1 < \ldots < t_K < t_{K+1} = \cdots = t_{K+q} = 1$
- We denote the B-spline basis functions of order q by $\{B_{1,q}, \ldots, B_{K+q,q}\}$. These functions can be using the following convenient recursive formula:

$$\text{For } q \geq 2, B_{i,q}(x) = \frac{x - t_{i-q}}{t_{i-1} - t_{i-q}} B_{i-1,q-1}(x) + \frac{t_i - x}{t_i - t_{i+1-q}} B_{i,q-1}(x),$$

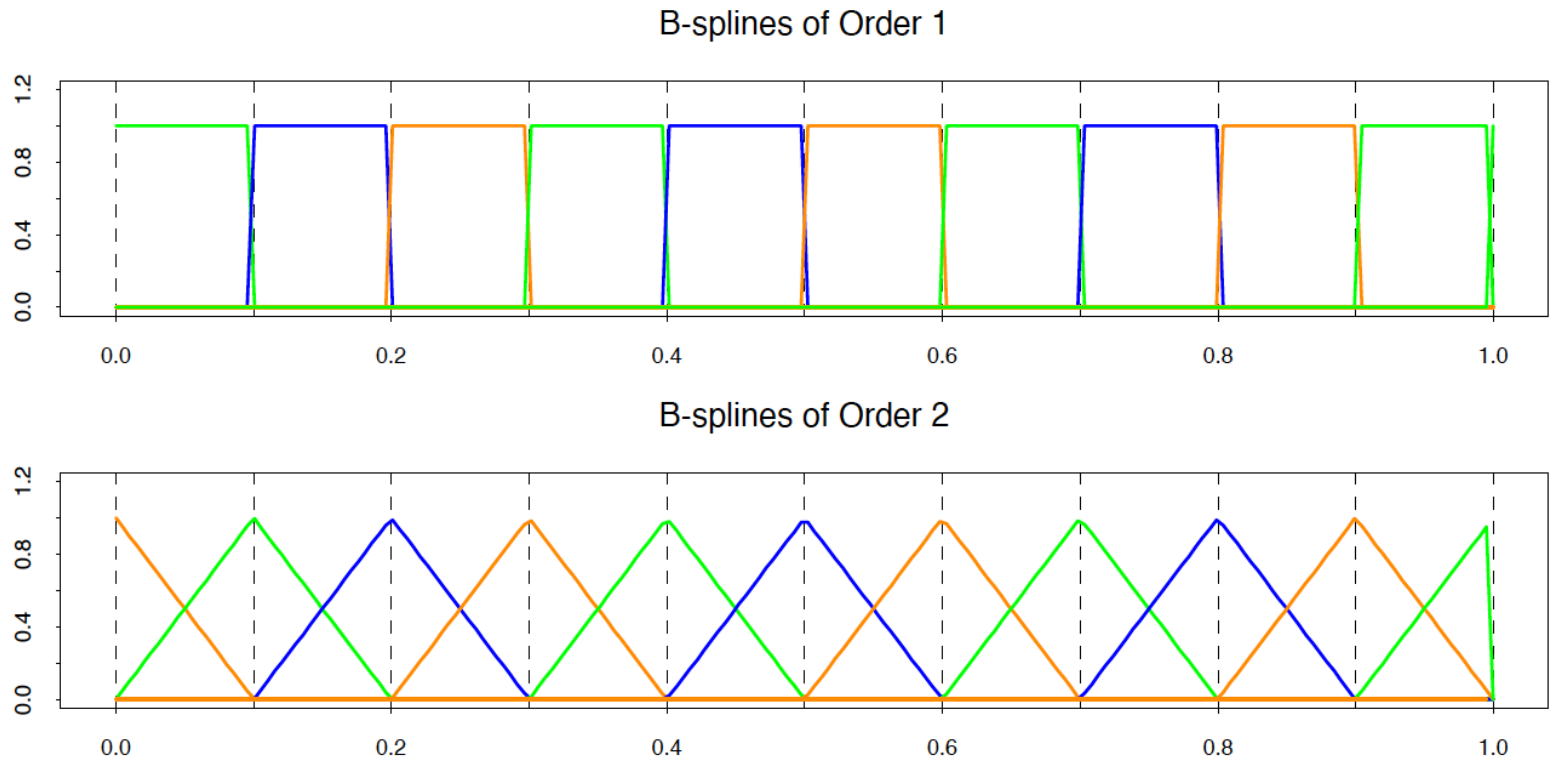
$$i = 1, \ldots, q + K$$

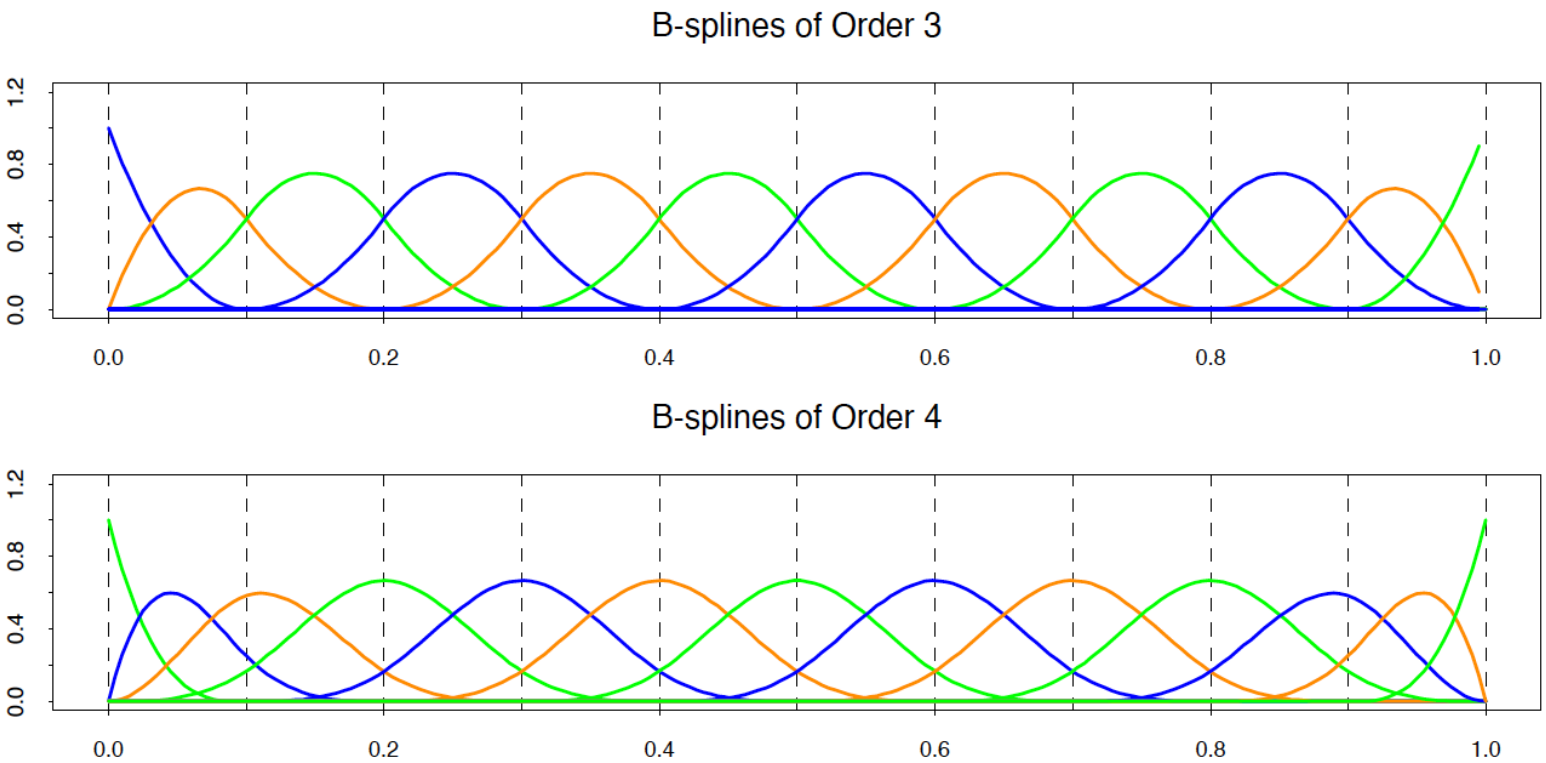
$$\text{For } q = 1, B_{i,q}(x) = 1_{[t_{i-1}, t_i)}(x), i = 1, \ldots, K+1$$

Some useful properties:

- $B_{i,q} \geq 0, i = 1, \ldots, q + K$; and $B_{i,q} > 0$ on (t_{i-q}, t_i)
- $\sum_{i=1}^{q+K} B_{i,q} = 1$
- All most q adjacent B-splines functions are nonzero at any give $x \in [0, 1]$.

The follow give the B-splines up to order four (i.e., cubic degree) with 11 knots ($K = 9$ interior knots) even spaced from 0 to 1.




B-splines of Order 3
0.0
0.4
0.8
1.2
0.0
0.2
0.4
0.6
0.8
1.0
B-splines of Order 4
0.0
0.4
0.8
1.2
0.0
0.2
0.4
0.6
0.8
1.0

Choosing number and location of knots

Where to place knots?

- Ideally, place more knots in regions that function varies more, and less in regions of lower variation.
- In practice, placing knots at uniform quantiles of data.

How many knots?

- number of knots and order of the basis functions count as together
- Try out different number of knots, then use CV to decide order needed
- Try out different order, then use CV to decide number of knots

Inference about basic expansion estimation

Let $f(x) = E(Y|X = x) = \sum_j \beta_j b_j(x)$, where $\{b_j : j = 1, \ldots, M\}$ is some basis functions ($M < n$). Let $\mathbf{B}$ denote the matrix whose (i,j)-th element is given by $b_j(x_i)$. Then the estimate for β is given by

$$\hat{\beta} = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{y}$$

The estimated covariance matrix of $\hat{\beta}$ is

$$\widehat{\mathrm{Var}}(\hat{\beta}) = \left(\mathbf{B}^T\mathbf{B}\right)^{-1}\hat{\sigma}^2$$

where we have estimated the noise variance by $\hat{\sigma}^2 = \sum_{i=1}^{N}\left(y_i - \hat{f}\left(x_i\right)\right)^2/n$. Let $b(x) = \{b_1(x), \ldots, b_M(x)\}$. The pointwise standard error for f at x is given by

$$\widehat{\mathrm{se}}[\hat{f}(x)] = \left[b^T\left(\mathbf{B}^T\mathbf{B}\right)^{-1}b\right]^{\frac{1}{2}}\hat{\sigma}$$

Confidence band

Suppose that $\hat{f}(x) = \sum_i w_i(x) y_i$. The conditional variance is $\sum_i w_i^2(x)\sigma^2(x)$ which can be estimated by $\sum_i w_i^2(x)\hat{\sigma}^2(x)$.

An asymptotic, pointwise (*biased*) confidence band is

$$\widehat{f}(x) \pm z_{\alpha/2}\sqrt{\sum_i w_i^2(x)\widehat{\sigma}^2(x)}.$$

To eliminate bias, one may want to **undersmooth** the fit $\hat{f}$, that is M should increase with n at a sufficiently fast rate.

Smoothing splines

Smoothing splines

With inputs $x_1, \ldots, x_n$ lying in an interval $[0, 1]$, the **smoothing spline** estimate f, of a given odd integer degree $d \geq 0$, is defined as

$$\hat{f} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y_i - f\left(x_i\right)\right)^2 + \lambda \int_0^1 \left(f^{(m)}(x)\right)^2 dx, \qquad \text{where } m = (d+1)/2$$

The **cubic smoothing splines** are given by

$$\hat{f} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \left(y_i - f\left(x_i\right)\right)^2 + \lambda \int_0^1 f''(x)^2 dx.$$

The larger the λ, the smoother f will be.

The solution to above problem is a **shrunken version of a natural cubic spline** with knots at the unique values of the data $x_1, \ldots, x_n$.

The solution is $\hat{f}(x) = \sum_{j=1}^{n} \hat{\beta}_j N_j(x)$, where N_j is the set of d-th degree natural splines with knots at $x_1, \ldots, x_n$.

So the problem becomes

$$\hat{\boldsymbol{\beta}} = \underset{\boldsymbol{\beta} \in \mathbb{R}^n}{\operatorname{argmin}} \|y - \mathbf{N}\boldsymbol{\beta}\|_2^2 + \lambda \boldsymbol{\beta}^T \Omega \boldsymbol{\beta}$$

where

$$\mathbf{N}_{ij} = N_j(x_i) \quad \text{and} \quad \Omega_{ij} = \int_0^1 N_i^{(m)}(x) N_j^{(m)}(x) dx \text{ for } i, j = 1, \ldots, n$$

The solution is a ridge-type estimator:

$$\hat{\boldsymbol{\beta}} = \left(\mathbf{N}^T\mathbf{N} + \lambda\Omega\right)^{-1}\mathbf{N}^T\mathbf{y}$$

and therefore the fitted values $\hat{f} = \left(\hat{f}\left(x_1\right), \ldots, \hat{f}\left(x_n\right)\right)$ are

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

The $\hat{f}$ can be rewritten as

$$\begin{aligned}\hat{f} &= \mathbf{N}\left(\mathbf{N}^T\mathbf{N} + \lambda\Omega\right)^{-1}\mathbf{N}^T\mathbf{y} = \mathbf{S}_\lambda\mathbf{y} \\ &= \mathbf{N}\left(\mathbf{N}^T\left(I + \lambda\left(\mathbf{N}^T\right)^{-1}\Omega\mathbf{N}^{-1}\right)\mathbf{N}\right)^{-1}\mathbf{N}^T\mathbf{y} \\ &= (I + \lambda\mathbf{Q})^{-1}\mathbf{y}\end{aligned}$$

where $Q = \left(\mathbf{N}^T\right)^{-1}\Omega\mathbf{N}^{-1}$.

- $S_\lambda = (I + \lambda\mathbf{Q})^{-1}$ (called **Reinsch form**)
- The eigen-decomposition $Q = UDU^T$, $D = diag(d_1, \ldots, d_n)$
- U is orthogonal whose columns are basis $\{u\}_{i=1}^n$ (**Memmler-Reinsch basis**)
- $0 = d_1 = d_2 \leq d_3 \leq \ldots \leq d_n$

the **eigendecomposition** of S_λ is

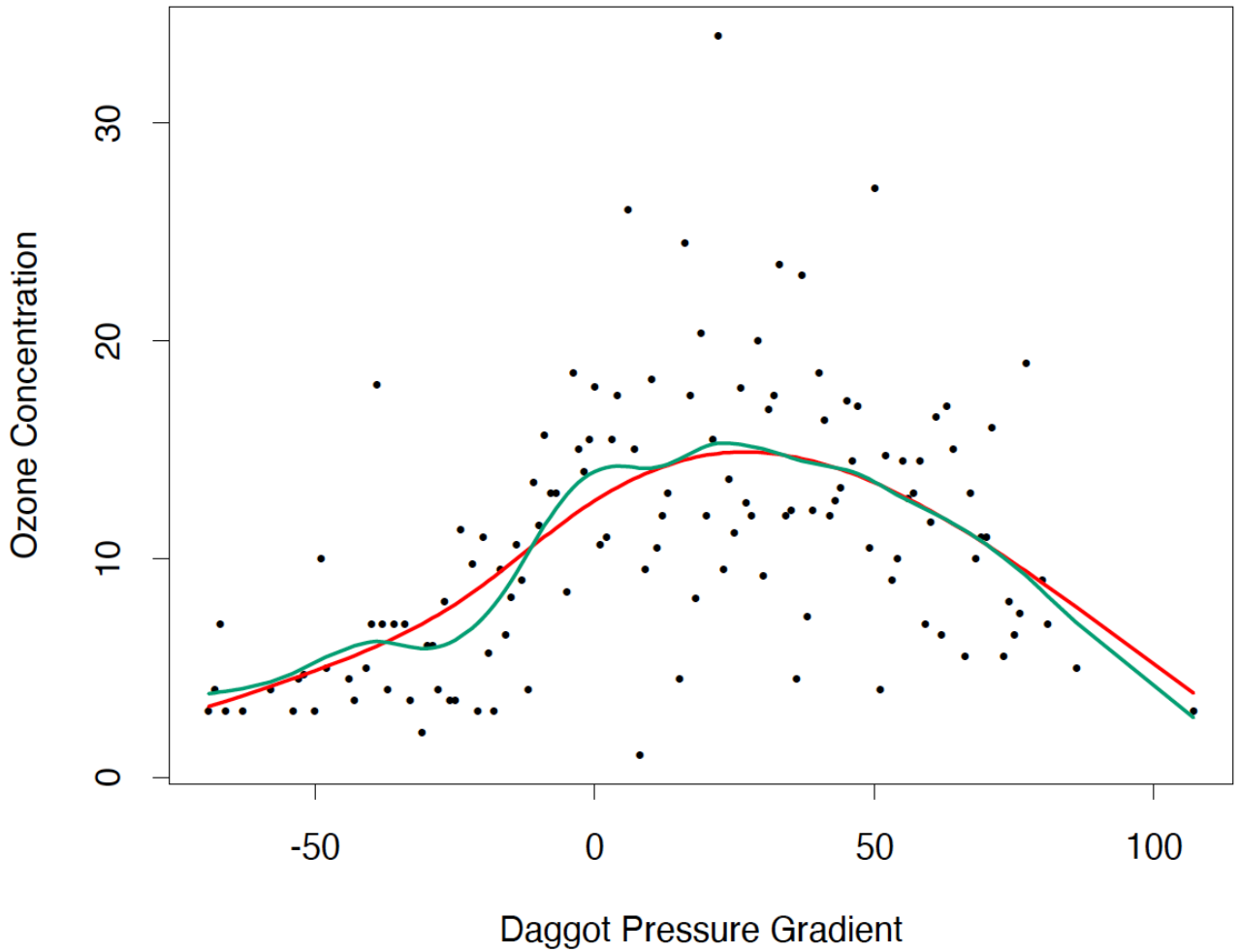
$$S_\lambda = (I + \lambda Q)^{-1} = \left(I + \lambda U D U^{-1}\right)^{-1}$$
$$= \sum_{j=1}^{n} \rho_j(\lambda) u_j u_j^T$$

The eigenvalues of S_λ are given by $\{\rho_j(\lambda) := \frac{1}{1+\lambda d_j} : j = 1, \ldots, n\}$.

$$\hat{f} = S_\lambda \mathbf{y} = \sum_{j=1}^{n} \frac{u_j^T y}{1 + \lambda d_j} u_j$$

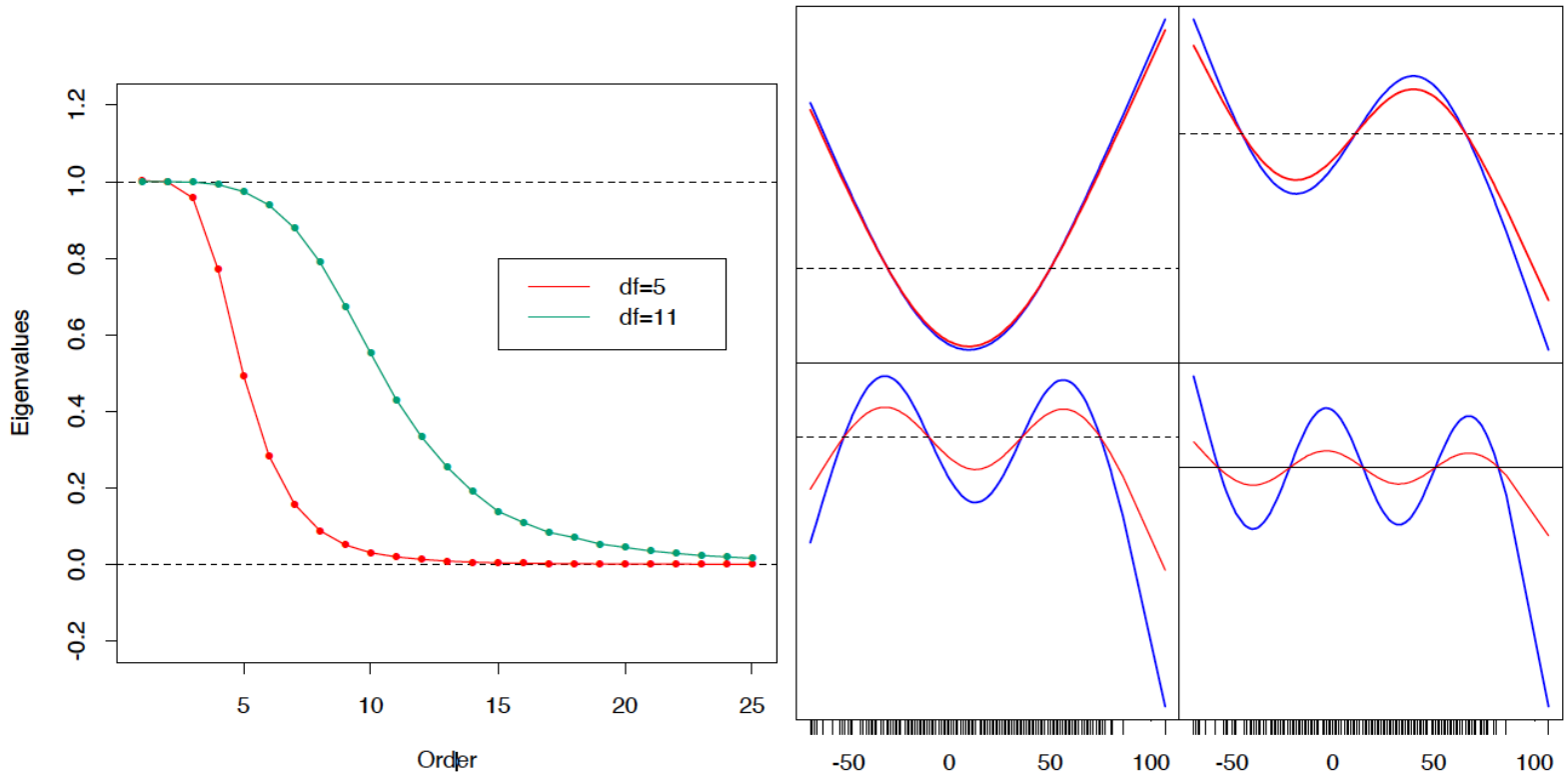
The smoothing spline operates by decomposing y w.r.t. the orthonormal basis $\{u\}_{i=1}^n$ and differentially shrinking the contributions using $1/(1 + \lambda d_j)$.

- The sequence of u_j, ordered by decreasing $1/(1 + \lambda d_j)$, appear to increase in complexity.
- Each of the eigenvectors themselves are shrunk by the smoothing spline: the higher the complexity (higher d_j), the more they are shrunk
- Increasing λ in the smoothing spline estimator tunes out the more wiggly components.



The two fits correspond to different values of the smoothing parameter, chosen to achieve five and eleven effective degrees of freedom.

ESL: Figure 5.7



ESL: Figure 5.7. (Lower left) First 25 eigenvalues for the two smoothing-spline matrices. The first two are exactly 1, and all are ≥ 0. (Lower right) Third to sixth eigenvectors of the spline smoother matrices.

A comparison between smoothing spline and the regression splines

For regression splines,

$$\hat{f}_P = B_P(B_P^T B_P)^{-1} B_P^T y$$

- B_P is $n \times M$ matrix of basic functions evaluated at data points. B_P is full column rank.
- $H_P = B_P(B_P^T B_P)^{-1} B_P^T$, which is symmetric, p.s.d., idempotent and rank(H_P)=M.
- SVD: $B_P = \tilde{U}\tilde{D}\tilde{V}^T$, and $H_P = \tilde{U}\tilde{U}^T$, where $\tilde{U}$ is $n \times M$.

Columns in $\tilde{U}$ are eigenvectors of H_P corresponding to eigenvalue 1.

$$\hat{f}_P = H_P y = \sum_{i=1}^{M} \tilde{u}_j \langle \tilde{u}_j, y \rangle$$

$$\text{Regression Splines } : \hat{f}_P = H_P \mathbf{y} = \sum_{j=1}^{M} \tilde{u}_j \langle \tilde{u}_j, \mathbf{y} \rangle$$

$$\text{Smoothing Splines } : \hat{f} = S_\lambda \mathbf{y} = \sum_{j=1}^{n} \frac{1}{1+\lambda d_j} u_j \langle u_j, \mathbf{y} \rangle$$

- regression splines smoother are called **projection smoothers**
- smoothing splines are called **shrinking smoothers**

(effective) degrees of freedom

For the projection linear smoother, $\hat{f}_p = H_P \mathbf{y}$, $M = trace(H_P)$ gives the dimension of the projection space.

The n parameters in a smoothing spline are heavily constrained, λ controls the effective degrees of freedom.

Define the **effective degrees of freedom** of a smoothing spline to be

$$\mathrm{df}_\lambda = \mathrm{trace}\left(\mathbf{S}_\lambda\right) = \sum_{k=1}^{n} \frac{1}{1+\lambda d_k}$$

- as $\lambda \to 0$, $df_\lambda \to n$ and $S_\lambda \to I$
- as $\lambda \to \infty$, $df_\lambda \to 2$ and $S_\lambda \to H$ the hat matrix for linear regression on X.

Choosing the smoothing parameter Unlike the regression splines, natural splines or B-splines, do not need to choose number of knots or location of knots. Instead, choose λ.

Tensor-product basis expansion

Suppose $X = (X_1, X_2) \in \mathbb{R}^2$, a basis of functions $h_{1k}(X_1), k = 1, \ldots, M_1$ for representing functions of coordinate X_1, and likewise a set of M_2 functions $h_{2k}(X_2)$ for coordinate X_2.

The $M_1 \times M_2$ dimensional **tensor product basis** defined by

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

can be used for representing a two-dimensional function:

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

A generalization on $\mathbb{R}^p$ is straightforward,

$$g(X) = \sum_{j_1=1}^{M_1} \sum_{j_2=1}^{M_2} \cdots \sum_{j_p=1}^{M_p} \theta_{j_1 \ldots j_p} h_{1j_1}(X_1) \cdots h_{pj_p}(X_p)$$

For smoothing splines, suppose $p = 2$,

$$\min_f \sum_{i=1}^{n} \{y_i - f(x_i)\}^2 + \lambda J[f]$$

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

The solution is the so-called **thin-plate spline**.

on $\mathbb{R}^p$, one can expand f using the tensor product of a collectionn of basis functions, say B-splines, Wavelets, and then apply some regularizer on f.