Lec 15: Spline Smoothing

Weiping Zhang

2019.11.18

Linear Spaces (Parametric vs. Nonparametric)

Local polynomial

Splines

Consider the following regression function:

$$m(x) = E(Y|X = x), x \in I \subset \mathbb{R}. \tag{1}$$

- our approach to estimating m involves the use of finite dimensional linear spaces.
- Why use linear spaces?
    - Makes estimation and statistical computations easy.
    - Has nice geometrical interpretation.
    - It actually can specify a broad range of models given we have discrete data.

- Using linear spaces we can define many families of function m: straight lines, polynomials, splines, functions with two continuous derivatives, and many other spaces (these are examples for the case where **x** is a scalar). The point is: we have many options.
- Notice that in most practical situation we will have observations $(\mathbf{X}_i, Y_i), i = 1, \dots, n$. In some situations we are only interested in estimating $m(\mathbf{X}_i), i = 1, \dots, n$.
- Let's say we are interested in estimating m. A common practice in statistics is to assume that m lies in some *linear space*, or is well approximated by a g that lies in some *linear space*. For example for simple linear regression we assume that m lies in the linear space of lines:

$$\alpha + \beta \mathbf{x}, (\alpha, \beta)' \in \mathbb{R}^2.$$

- A linear model of order p for the regression function (1) consists of a p-dimensional linear space $\mathcal{G}$, having as a basis the function

$$B_j(\mathbf{x}), j = 1, \ldots, p$$

defined for $\mathbf{x} \in I$.

- Each member $g \in \mathcal{G}$ can be written uniquely as a linear combination

$$g(\mathbf{x}) = g(\mathbf{x}; \mathbf{g}\theta) = \theta_1 B_1(\mathbf{x}) + \cdots + \theta_p B_p(\mathbf{x})$$

for some value of the coefficient vector $\mathbf{g}\theta = (\theta_1, \ldots, \theta_p)' \in \mathbb{R}^p$.

- Notice that $\mathbf{g}\theta$ specifies the point $g \in \mathcal{G}$.

How would you write this out for linear regression?

- Given observations $(\mathbf{X}_i, Y_i), i = 1, \dots, n$ the least squares estimate (LSE) of m or equivalently $m(\mathbf{x})$ is defined by $\hat{m}(\mathbf{x}) = g(\mathbf{x}; \widehat{\mathbf{g}\theta})$, where

$$\widehat{\mathbf{g}\theta} = \arg \min_{\mathbf{g}\theta \in \mathbb{R}^p} \sum_{i=1}^{n} \{Y_i - g(\mathbf{X}_i, \mathbf{g}\theta)\}^2.$$

- Define the vector $g = \{g(x_1), \dots, g(x_n)\}'$. Then the distribution of the observations of $Y|X = x$ are in the family

$$\{N(g, \sigma^2 \mathbf{I}_n); g = [g(x_1), \dots, g(x_n)]', \; g \in \mathcal{G}\} \qquad (2)$$

- and if we assume the errors ε are IID normal and that $m \in \mathcal{G}$ we have that $\hat{m} = [g(x_1; \widehat{\mathbf{g}\theta}), \dots, g(x_n; \widehat{\mathbf{g}\theta})]$ is the maximum likelihood estimate. The estimand m is an $n \times 1$ vector. But how many parameters are we really estimating?

- Equivalently we can think of the distribution is in the family

$$\{N(\mathbf{B}\mathbf{g}\theta, \sigma^2); \mathbf{g}\theta \in \mathbb{R}^p\} \tag{3}$$

  and the maximum likelihood estimate for $\mathbf{g}\theta$ is $\widehat{\mathbf{g}\theta}$. Here $\mathbf{B}$ is a matrix of basis elements defined soon...

- Here we start seeing where the name **non-parametric** comes from. How are the approaches (2) and (3) different?
- Notice that obtaining $\widehat{\mathbf{g}\theta}$ is easy because of the linear model set-up. The ordinary least square estimate is

$$(\mathbf{B}'\mathbf{B})\widehat{\mathbf{g}\theta} = \mathbf{B}'\mathbf{Y}$$

  where $\mathbf{B}$ is is the $n \times p$ design matrix with elements $[\mathbf{B}]_{ij} = B_j(\mathbf{X}_i)$. When this solution is unique we refer to $g(x; \widehat{\mathbf{g}\theta})$ as the OLS projection of $\mathbf{Y}$ into $\mathcal{G}$.

Parametric versus non-parametric

- In some cases, we have reason to believe that the function m is actually a member of some linear space $\mathcal{G}$.
- Traditionally, inference for regression models depends on m being representable as some combination of known predictors. Under this assumption, m can be written as a combination of basis elements for some value of the coefficient vector $\mathbf{g}\theta$.
- This provides a **parametric** specification for m. No matter how many observations we collect, there is no need to look outside the fixed, finite-dimensional, linear space $\mathcal{G}$ when estimating m.

- In practical situations, however, we would rarely believe such relationship to be exactly true.
- Model spaces $\mathcal{G}$ are understood to provide (at best) approximations to m; and as we collect more and more samples, we have the freedom to audition richer and richer classes of models.
- In such cases, all we might be willing to say about m is that it is **smooth** in some sense, a common assumption being that m have two bounded derivatives.
- Far from the assumption that m belong to a fixed, finite-dimensional linear space, we instead posit a **nonparametric** specification for m.

- In this context, model spaces are employed mainly in our approach to inference; For example, we are less interested in the actual values of the coefficient $\mathbf{g}\theta$, e.g. whether or not an element of $\mathbf{g}\theta$ is significantly different from zero to the 0.05 level. Instead we concern ourselves with functional properties of $g(\mathbf{x}; \widehat{\mathbf{g}\theta})$, the estimated curve or surface, e.g. whether or not a peak is real.

To ascertain the local behavior of OLS projections onto approximation spaces $\mathcal{G}$, define the pointwise, mean squared error (MSE) of $\hat{g}(\mathbf{x}) = g(\mathbf{x}; \widehat{\mathbf{g}\theta})$ as

$$E\{m(\mathbf{x}) - \hat{g}(\mathbf{x})\}^2 = bias^2\{\hat{g}(\mathbf{x})\} + var\{\hat{g}(\mathbf{x})\}$$

where

$$bias\{\hat{g}(\mathbf{x})\} = m(x) - E\{\hat{g}(\mathbf{x})\} \tag{4}$$

and

$$var\{\hat{g}(\mathbf{x})\} = E\{\hat{g}(\mathbf{x}) - E[\hat{g}(\mathbf{x})]\}^2$$

- When the input values $\{\mathbf{X}_i\}$ are deterministic the expectations above are with respect to the noisy observation Y_i. In practice, MSE is defined in this way even in the random design case, so we look at expectations conditioned on $\mathbf{X}$.
- When we do this, standard results in regression theory can be applied to derive an expression for the variance term

  $$Var\{\hat{g}(\mathbf{x})\} = \sigma^2 \mathbf{B}(\mathbf{x})'(\mathbf{B}'\mathbf{B})^{-1}\mathbf{B}(\mathbf{x})$$

  where $\mathbf{B}(\mathbf{x}) = (B_1(\mathbf{x}), \ldots, B_p(\mathbf{x}))'$, and the error variance is assumed constant.
- Under the parametric specification that $m \in \mathcal{G}$, what is the bias?

- This leads to classical $t-$ and $F-$hypothesis tests and associated parametric confidence intervals for $\mathbf{g}\theta$.
- Suppose on the other hand, that m is not a member of $\mathcal{G}$, but rather can be reasonably approximated by an element in $\mathcal{G}$. The bias (4) now reflects the ability of functions in $\mathcal{G}$ to capture the essential features of m.

Local polynomial

- In practical situations, a statistician is rarely blessed with simple linear relationship between the predictor X and the observed output Y.
- To overcome this deficiency, we might consider a more flexible polynomial model. Let $\mathcal{P}_k$ denote the linear space of polynomials in x of order at most k defined as

  $$g(x; \mathbf{g}\theta) = \theta_1 + \theta_2 x + \cdots + \theta_k x^{k-1}, x \in I$$

  for the parameter vector $\mathbf{g}\theta = (\theta_1, \ldots, \theta_k) \in \mathbb{R}^k$. Note that the space $\mathcal{P}_k$ consists of polynomials having degree at most $k - 1$.

- In exceptional cases, we have reasons to believe that the regression function m is in fact a high-order polynomial. This parametric assumption could be based on physical or physiological models describing how the data were generated.
- Recall Taylor's theorem: polynomials are good at approximating well-behaved functions in reasonably tight neighborhoods. If all we can say about m is that it is smooth in some sense, then either implicitly or explicitly we consider high-order polynomials because of their favorable approximation properties.
- If m is not in $\mathcal{P}_k$ then our estimates will be biased by an amount that reflects the approximation error incurred by a polynomial model.

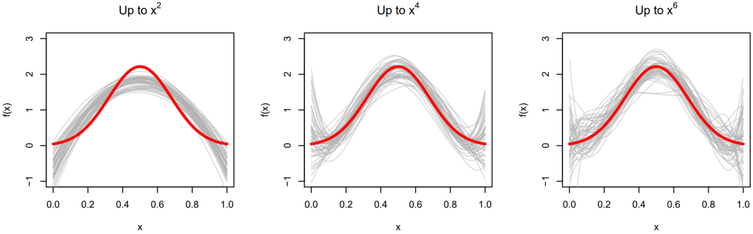
- **Computational Issue**: The basis of monomials

$$B_j(x) = x^{j-1} \text{ for } j = 1, \ldots, k$$

  is not well suited for numerical calculations (x^8 can be VERY BIG compared to x).

- While convenient for analytical manipulations (differentiation, integration), this basis is **ill-conditioned** for k larger than 8 or 9. Most statistical packages use the orthogonal Chebyshev polynomials (used by the R command `poly()`).
- Besides, polynomial terms introduce undesirable side effects: each observation affects the entire curve, even for x values far from the observation
- Not only does this introduce bias, but it also results in extremely high variance near the edges of the range of x

To illustrate this, consider the following simulated example (gray lines are models fit to 100 observations arising from the true f, colored red):



- An alternative to polynomials is to consider the space $\mathcal{PP}_k(\mathbf{t})$ of piecewise polynomials with break points $\mathbf{t} = (t_0, \ldots, t_{p+1})'$.
- Given a knots sequence $-\infty = t_0 < t_1 < \cdots < t_p < t_{p+1} = \infty$, construct $p+1$ (disjoint) intervals

$$I_l = [t_{l-1}, t_l), 1 \le l \le p \text{ and } I_{p+1} = [t_p, t_{p+1}],$$

whose union is $I = (-\infty, \infty)$. Define the piecewise polynomials of order k

$$g(x) = \begin{cases} g_1(x) = \theta_{1,1} + \theta_{1,2}x + \cdots + \theta_{1,k}x^{k-1}, & x \in I_1 \\ \vdots & \vdots \\ g_{p+1}(x) = \theta_{p+1,1} + \theta_{p+1,2}x + \cdots + \theta_{p+1,k}x^{k-1}, & x \in I_{k+1}. \end{cases}$$

- In many situations, breakpoints in the regression function do not make sense. Would forcing the piecewise polynomials to be continuous suffice? What about continuous first derivatives?
- We start by consider the subspaces of the piecewise polynomial space. We will denote it with $\mathcal{PP}_k(\mathbf{t})$ with $\mathbf{t} = (t_1, \ldots, t_p)'$ the break-points or interior knots. Different break points define different spaces.
- We can put constrains on the behavior of the functions g at the break points. (We can construct tests to see if these constrains are suggested by the data but, will not go into this here)

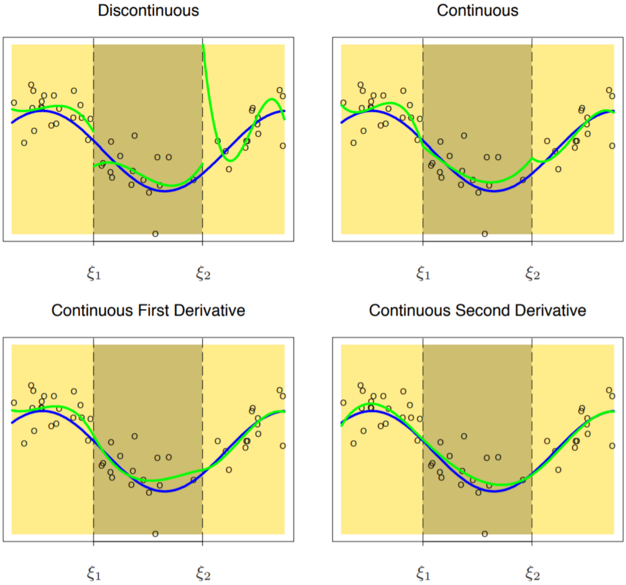


Figure 1: Illustration of the effects of enforcing continuity at the knots, across various orders of the derivative, for a cubic piecewise polynomial.

Splines

- A **kth-order spline** g is a piecewise polynomial function of degree k that is continuous and has continuous derivatives of orders $1, \ldots, k-1$, at its knot points.
- Splines have some special (some might say: amazing!) properties, and they have been a topic of interest among statisticians and mathematicians for a very long time. See de Boor (1978) for an in-depth coverage. Informally, a spline is a lot smoother than a piecewise polynomial, and so modeling with splines can serve as a way of reducing the variance of fitted estimators.
- A bit of statistical folklore: it is said that a cubic spline is so smooth, that one cannot detect the locations of its knots by eye!

- How can we parametrize the set of a splines with knots at $t_1, \ldots, t_p$? The most natural way is to use the **truncated power basis**, $g_1, \ldots, g_{p+k+1}$, defined as

$$g_1(x) = 1, g_2(x) = x, \cdots, g_{k+1}(x) = x^k,$$
$$g_{k+1+j}(x) = (x - t_j)_+^k, j = 1, \ldots, p.$$

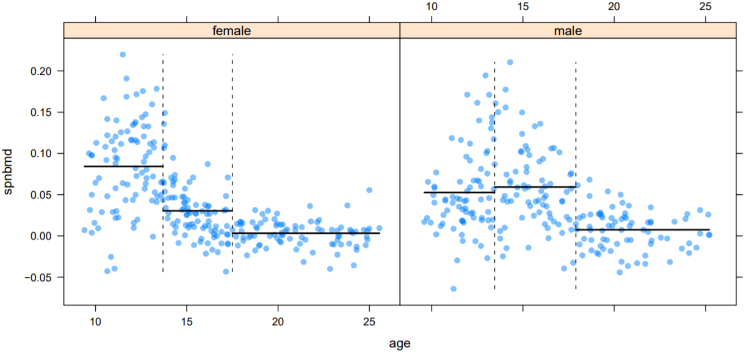
That is, we can write any function $g \in \mathcal{PP}_k(\mathbf{t})$ as:

$$\begin{aligned} g(x) &= \theta_{0,1} + \theta_{0,2}x + \cdots + \theta_{0,k}x^{k-1} + \\ &\quad \theta_{1,k}(x - t_1)_+^{k-1} + \cdots + \theta_{p,k}(x - t_p)_+^{k-1} \end{aligned}$$

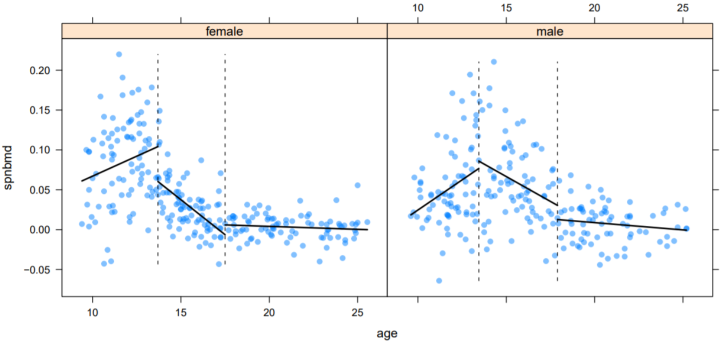
where $(\cdot)_+ = \max(\cdot, 0)$.

- Written in this way the coefficients $\theta_{1,1}, \ldots, \theta_{1,k}$ record the jumps in the different derivative from the first piece to the second.
- Notice that the constrains reduce the number of parameters. This is in agreement with the fact that we are forcing more smoothness.
- To understand splines, we will gradually build up a piecewise model, starting at the simplest one: the piecewise constant model
- First, we partition the range of x into $p+1$ intervals by knots $t_1 < t_2 < \cdots < t_p$

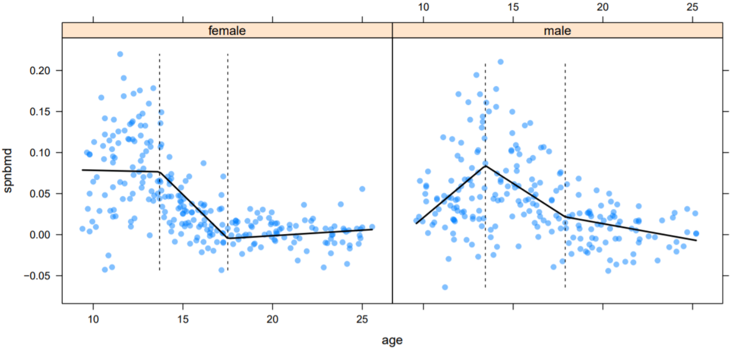
The piecewise constant model for bone mineral density data, with three knots:



The piecewise linear model for bone mineral density data, with three knots:



The continuous piecewise linear model for bone mineral density data, with three knots:



Basis functions for piecewise continuous models

- These constraints can be incorporated directly into the basis functions:

$$B_1(x) = 1, B_2(x) = x, B_3(x) = (x - t_1)_+, B_4(x) = (x - t_2)_+$$

- It can be easily checked that these basis functions lead to a composite function $f(x)$ that:
  - Is everywhere continuous
  - Is linear everywhere except the knots
  - Has a different slope for each region
- Also, note that the degrees of freedom add up: 3 regions × 2 degrees of freedom in each region - 2 constraints $=$ 4 basis functions

- The preceding is an example of a spline: a piecewise k degree polynomial that is continuous up to its first $k-1$ derivatives
- By requiring continuous derivatives, we ensure that the resulting function is as smooth as possible
- We can obtain more flexible curves by increasing the degree of the spline and/or by adding knots
- However, there is a tradeoff:
    - Few knots/low degree: Resulting class of functions may be too restrictive (bias)
    - Many knots/high degree: We run the risk of overfitting (variance)

- We will concentrate on the cubic splines which are continuous and have continuous first and second derivatives. In this case we can write:

$$\begin{aligned} g(x) &= \theta_{0,1} + \theta_{0,2} x + \theta_{0,3} x^2 + \theta_{0,4} x^3 \\ &\quad + \theta_{1,3}(x - t_1)_+^3 + \cdots + \theta_{p,3}(x - t_p)_+^3 \end{aligned}$$

  How many “parameters” in this space?

- We will concentrate on the cubic splines which are continuous and have continuous first and second derivatives. In this case we can write:

$$\begin{aligned} g(x) &= \theta_{0,1} + \theta_{0,2}x + \theta_{0,3}x^2 + \theta_{0,4}x^3 \\ &\quad + \theta_{1,3}(x - t_1)_+^3 + \cdots + \theta_{p,3}(x - t_p)_+^3 \end{aligned}$$

  How many "parameters" in this space?

- The cubic splines contain 4 + p degrees of freedom: p + 1 regions × 4 parameters per region - p knots × 3 constraints per knot

- Note: It is always possible to have less restrictions at knots where we believe the behavior is “less smooth”.
- While these basis functions are natural, a much better computational choice, both for speed and numerical accuracy, is the **B-spline basis**.
- There is asymptotic theory that goes along with all this but we will not go into the details. We will just notice that

$$E[m(x) - g(x)] = O(h_l^{2k} + 1/n_l)$$

where h_l is the size of the interval where x is in and n_l is the number of points in it. What does this say?

B-spline in R

- Fortunately, one can use B-splines without knowing the details behind their complicated construction
- In the splines package (which by default is installed but not loaded), the bs() function will implement a B-spline basis for you

  ```
  X <- bs(x,knots=quantile(x,p=c(1/3,2/3)))
  X <- bs(x,df=5)
  X <- bs(x,degree=2,df=10)
  Xp <- predict(X,newdata=x)
  ```

- By default, bs uses degree=3, knots at evenly spaced quantiles, and does not return a column for the intercept

Regression Spline

- A first idea: let's perform regression on a spline basis. In other words, given inputs $x_1, \dots, x_n$ and responses $y_1, \dots, y_n$, we consider fitting functions m that are kth-order splines with knots at some chosen locations $t_1, \dots, t_p$. This means expressing m as

$$m(x) = \sum_{j=1}^{p+k+1} \beta_j g_j(x),$$

where $\beta_1, \dots, \beta_{m+p+1}$ are coefficients and $g_1, \dots, g_{m+p+1}$ are basis functions for order k splines over the knots $t_1, \dots, t_p$(e.g., the truncated power basis or B-spline basis)

- Letting $y = (y_1, \ldots, y_n)$ and defining the basis matrix $G \in \mathcal{R}^{n \times (p+k+1)}$ by

$$G_{ij} = g_j(x_i), i = 1, \ldots, n, j = 1, \ldots, p+k+1,$$

we can just use least squares to determine the optimal coefficients $\hat{\beta} = (\hat{\beta}_1, \ldots, \hat{\beta}_{p+k+1})$:

$$\hat{\beta} = \arg\min_{\beta} \|y - G\beta\|_2^2$$

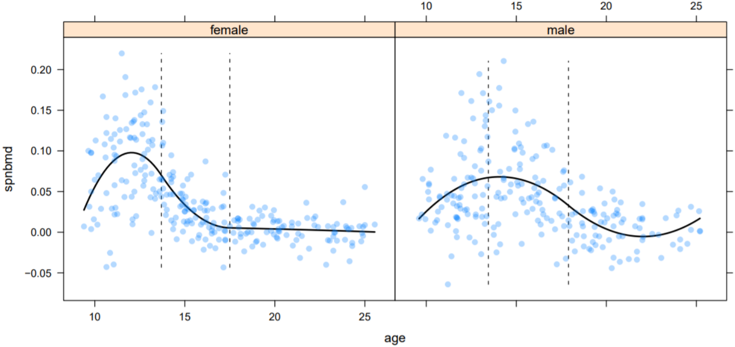
which then leaves us with the fitted **regression spline** $\hat{m}(x) = \sum_{j=1}^{p+k+1} \hat{\beta}_j g_j(x)$.

- Of course we know that $\hat{\beta} = (G'G)^{-1}G'y$, so the fitted values $\hat{\mu} = (\hat{m}(x_1), \ldots, \hat{m}(x_n))$ are

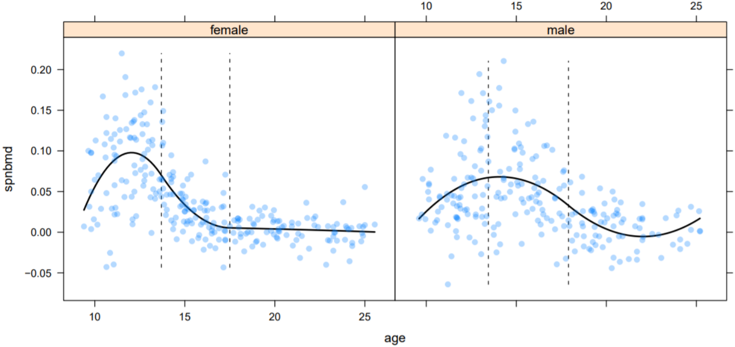
$$\hat{\mu} = G(G'G)^{-1}G'y$$

  and regression splines are linear smoothers.
- This is a classic method, and can work well provided we choose good knots $t_1, \ldots, t_p$, but in general choosing knots is a tricky business. There is a large literature on knot selection for regression splines via greedy methods like recursive partitioning.

Quadratic splines:



Cubic splines:



Natural splines

- A problem with regression splines is that the estimates tend to display erractic behavior, i.e., they have high variance, at the boundaries of the input domain. (This is the opposite problem to that with kernel smoothing, which had poor bias at the boundaries.) This only gets worse as the polynomial order k gets larger.
- A way to remedy this problem is to force the piecewise polynomial function to have a lower degree to the left of the leftmost knot, and to the right of the rightmost knot—this is exactly what **natural splines** do. A natural spline of order k, with knots at $t_1 < \cdots < t_p$, is a piecewise polynomial function g such that
  - g is a polynomial of degree k on each of $[t_1, t_2], \ldots, [t_{p-1}, t_p]$.
  - g is a polynomial of degree $(k-1)/2$ on $(-\infty, t_1]$ and $[t_p, \infty)$.
  - g is s continuous and has continuous derivatives of orders $1, \ldots, k-1$ at $t_1, \ldots, t_p$.

  It is implicit here that natural splines are only defined for odd orders k

- What is the dimension of the span of kth order natural splines with knots at $t_1, \ldots, t_p$? Recall for splines, this was $p + k + 1$ (the number of truncated power basis functions). For natural splines, we can compute this dimension by counting:

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

Above, a is the number of free parameters in the interior intervals $[t_1, t_2], \ldots, [t_{p-1}, t_p]$; b is the number of free parameters in the exterior intervals $(-\infty, t_1]$, $[t_p, \infty)$, and c is the number of constraints at the knots $t_1, \ldots, t_p$. The fact that the total dimension is p is amazing; this is independent of k!.

- Note that there is a variant of the truncated power basis for natural splines, and a variant of the B-spline basis for natural splines. Again, B-splines are the preferred parametrization for computational speed and stability
- Natural splines of cubic order is the most common special case: these are smooth piecewise cubic functions, that are simply linear beyond the leftmost and rightmost knots
- Note, then, that a natural cubic spline basis function with p knots has p degrees of freedom

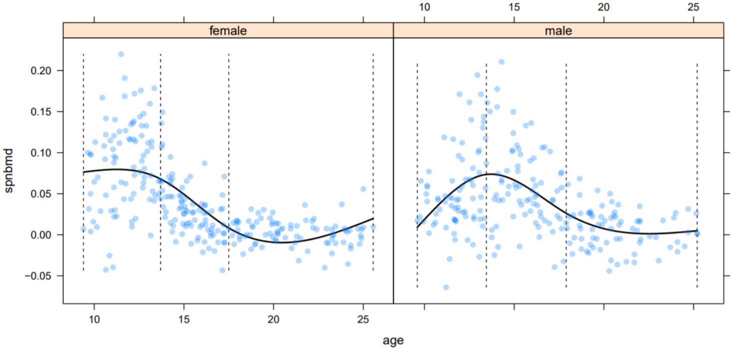
Understand how R counts the degree-of-freedom

- Recall that the linear functions in the two extreme intervals are totally determined by the other cubic splines. So data points in the two extreme intervals (i.e., outside the two boundary knots) are wasted since they do not affect the fitting. Therefore, by default, R puts the two boundary knots as the min and max of $x_i's$.
- You can tell R the location of knots, which are the interior knots. Recall that a natural cubic spline with p knots has p df. So the df is equal to the number of (interior) knots plus 2, where 2 means the two boundary knots.
- Or you can tell R the df. If intercept = TRUE (default is false), then we need $p = df - 2$ knots, otherwise we need $p = df - 1$ knots. Again, by default, R puts knots at the $1/(p+1), \ldots, p/(p+1)$ quantiles of $x_1, \ldots, x_n$.

- The following three design matrices (the first two are of n × 3 and the last one is of n × 4) correspond to the same regression model with natural cubic spline of df 4.

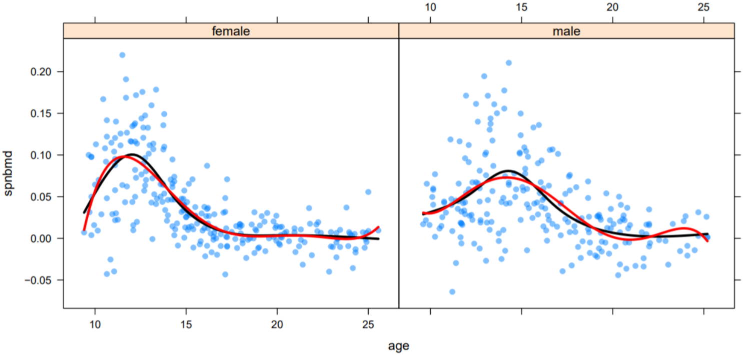
```
> ns(x, knots=quantile(x, c(1/3, 2/3)));
> ns(x, df=3);
> ns(x, df=4, intercept=TRUE);
```

Natural cubic splines:



Natural cubic splines:

Black line: 6 df natural cubic spline; red line: 6 df polynomial



- A natural cubic spline with n knots is represented by n basis functions. One can start from a basis for cubic splines, and derive the reduced basis by imposing the boundary constraints. For example, starting from the truncated power series basis, we arrive at

$$N_1(x) = 1, N_2(x) = x, N_{k+2}(x) = d_k(x) - d_{n-1}(x) \qquad (5)$$

where

$$d_k(x) = \frac{(x - \xi_k)^3_+ - (x - \xi_n)^3_+}{\xi_n - \xi_k}$$

Smoothing Splines

- So where do we put the knots? How many do we use? There are some data-driven procedures for doing this. Natural Smoothing Splines provide another approach.
- What happens if the knots coincide with the dependent variables $\{X_i\}$. Then there is a function $g \in \mathcal{G}$, the space of cubic splines with knots at $(x_1, \ldots, x_n)$, with $g(x_i) = y_i, i, \ldots, n$, i.e. we haven't smoothed at all.
- Smoothing splines are given by a regularized regression over the natural spline basis, placing knots at all inputs $x_1, \ldots, x_n$.

Consider the following problem: among all functions g with two continuous first two derivatives, find one that minimizes the penalized residual sum of squares

$$\sum_{i=1}^{n}\{y_i - g(x_i)\}^2 + \lambda \int_a^b \{g''(t)\}^2\, dt \tag{6}$$

where λ is a fixed constant, and $a \le x_1 \le \cdots \le x_n \le b$.

- In general, the smoothing spline estimate $\hat{m}$, of a given odd integer order $k \ge 0$, is defined as

  $$\sum_{i=1}^{n}\{y_i - g(x_i)\}^2 + \lambda \int_a^b \{g^{(m)}(t)\}^2 dt, \tag{7}$$

  where $m = (k+1)/2$. ($m \le n$). Schoenberg (1964a,b).

Remarkably, it so happens that the minimizer in the general smoothing spline problem is unique, and is a natural kth-order spline with knots at the input points $x_1, \ldots, x_n$! Here we give a proof for the cubic case, $k = 3$.

Theorem

The key result can be stated as follows: s: if $\tilde{g}$ is any twice differentiable function on $[a, b]$, and $x_1, \ldots, x_n \in [a, b]$, then there exists a natural cubic spline g with knots at $x_1, \ldots, x_n$ such that $g(x_i) = \tilde{g}(x_i), i = 1, \ldots, n$ and

$$\int_a^b (g''(x))^2 dx \leq \int_a^b (\tilde{g}''(x))^2 dx.$$

Proof. the natural spline basis with knots at $x_1, \dots, x_n$ is n-dimensional, so given any n points $z_i = \tilde{g}(x_i), i = 1, \dots, n$, we can always find a natural spline g with knots at $x_1, \dots, x_n$ that satisfies $g(x_i) = z_i, i = 1, \dots, n$. Now define

$$h(x) = \tilde{g}(x) - g(x).$$

Consider

$$\begin{aligned}
\int_a^b g''(x)h''(x)dx &= g''(x)h'(x)|_a^b - \int_a^b g'''(x)h'(x)dx \\
&= -\int_{x_1}^{x_n} g'''(x)h'(x)dx \\
&= -\sum_{j=1}^{n-1} g'''(x)h(x)|_{x_j}^{x_{j+1}} + \int_{x_1}^{x_n} g^{(4)}(x)h'(x)dx \\
&= -\sum_{j=1}^{n-1} g'''(x_j^+)(h(x_{j+1}) - h(x_j)),
\end{aligned}$$

where in the first line we used integration by parts; in the second we used the that $g''(a) = g''(b) = 0$, and $g'''(x) = 0$ for $x \leq x_1$ and $x \geq x_n$, as g is a natrual spline; in the third we used integration by parts again; in the fourth line we used the fact that g''' is constant on any open interval $(x_j, x_{j+1}), j = 1, \ldots, n-1$, and that $g^{(4)} = 0$, again because g is a natural spline. (In the above, we use $g'''(u^+)$ to denote $\lim_{x \downarrow u} g'''(x)$.) Finally, since $h(x_j) = 0$ for all $j = 1, \ldots, n$, we have

$$\int_a^b g''(x)h''(x)dx = 0$$

From this, it follows that

$$\begin{aligned}\int_a^b (\tilde{g}''(x))^2 dx &= \int_a^b (g''(x) + h''(x))^2 dx \\ &= \int_a^b (g''(x))^2 dx + \int_a^b (h''(x))^2 dx + 2\int_a^b g''(x)h''(x)dx \\ &= \int_a^b (g''(x))^2 dx + \int_a^b (h''(x))^2 dx\end{aligned}$$

and therefore,

$$\int_a^b (g''(x))^2 dx \leq \int_a^b (\tilde{g}''(x))^2 dx,$$

with equality if and only if $h''(x) = 0$ for all $x \in [a, b]$. Note that $h''(x) = 0$ implies that h must be linear, and since we already know that $h(x_j) = 0$ for all $j = 1, \ldots, n$, this is equivalent to $h = 0$. In other words, the above inequality) holds strictly except when $\tilde{g} = g$, so the solution in (6) is uniquely a natural spline with knots at the inputs.

Finite-dimensional form

- The key result presented above tells us that we can choose a basis $\eta_1, \ldots, \eta_n$ for the set of kth-order natural splines with knots over $x_1, \ldots, x_n$, and reparametrize the problem (7) as

$$\hat{\beta} = \arg \min_{\beta \in \mathcal{R}^n} \sum_{i=1}^n \Big(y_i - \sum_{j=1}^n \beta_j \eta_j(x_i) \Big)^2 + \lambda \int_a^b \Big(\sum_{j=1}^n \beta_j \eta_j^{(m)}(x) \Big)^2 dx, \tag{8}$$

  This is a finite-dimensional problem, and after we compute the coefficients $\hat{\beta} \in \mathcal{R}^n$, we know that the smoothing spline estimate is simply $\hat{g} = \sum_{j=1}^n \hat{\beta}_j \eta_j(x)$.

- Defining the basis matrix and penalty matrices $N, \Omega \in \mathcal{R}^{n \times n}$ by

$$N_{ij} = \eta_j(x_i), \text{and}, \Omega_{ij} = \int_a^b \eta_i^{(m)}(x) \eta_j^{(m)}(x) dx$$

  for $i, j = 1, \ldots, n$.

- The problem in (8) can be written more succintly as

  $$\hat{\beta} = \arg \min_{\beta \in \mathcal{R}^n} \|y - N\beta\|_2^2 + \lambda \beta' \Omega \beta,$$

  showing the smoothing spline problem to be a type of generalized ridge regression problem. In fact, the solution has the explicit form

  $$\hat{\beta} = (N'N + \lambda\Omega)^{-1} N'y,$$

  and therefore the fitted values $\hat{\mu} = (\hat{g}(x_1), \ldots, \hat{g}(x_n))$ are

  $$\hat{\mu} = N(N'N + \lambda\Omega)^{-1} N'y = S_\lambda y$$

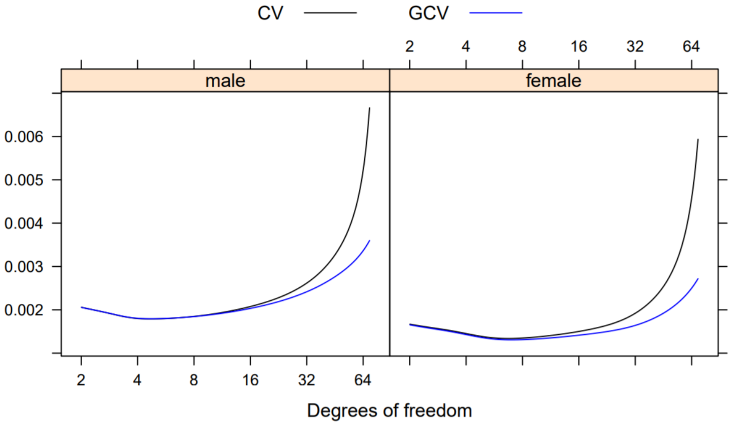
- A special property of smoothing splines: the fitted values $\hat{\mu}$ can be computed in $O(n)$ operations. This is achieved by forming N from the B-spline basis (for natural splines), and in this case the matrix $N'N + \Omega I$ ends up being banded (with a bandwidth that only depends on the polynomial order k). In practice, smoothing spline computations are extremely fast.

- As with ridge regression, this property provides us with a convenient way to calculate (or approximate) the leave-one-out cross-validation score as well as define the degrees of freedom of the estimate:

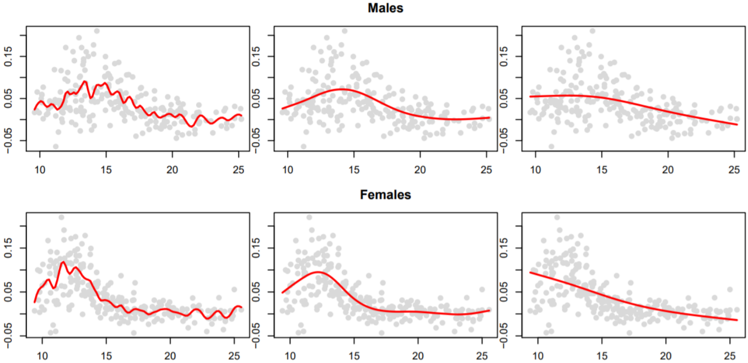
$$GCV = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - tr(S_\lambda)/n}\right)^2$$

$$df_\lambda = tr(S_\lambda)$$

CV, GCV for BMD example



Undersmoothing and oversmoothing of BMD data



Reinsch form

- It is informative to rewrite the fitted values $\hat{\mu}$ is what is called **Reinsch form**

$$\begin{aligned}\hat{\mu} &= N(N'N + \lambda\Omega)^{-1}N'y \\ &= N(N'(I + \lambda(N')^{-1}\Omega N^{-1})N)^{-1}N'y \\ &= (I + \lambda Q)^{-1}y,\end{aligned}$$

  where $Q = (N')^{-1}\Omega N^{-1}$.

- Note that this matrix Q does not depend on λ. If we compute an eigendecomposition $Q = UDU'$, then the eigen decomposition of $S_\lambda = N(N'N + \lambda\Omega)^{-1} = (I + \lambda Q)^{-1}$ is

$$S_\lambda = \sum_{j=1}^{n} \frac{1}{1 + \lambda d_j} u_j u_j'$$

  where $D = diag(d_1, \ldots, d_n)$.

- Therefore the smoothing spline fitted values are $\hat{\mu} = Sy$, i.e.,

$$\hat{\mu} = \sum_{j=1}^{n} \frac{u_j' y}{1 + \lambda d_j} u_j. \tag{9}$$

  Interpretation: smoothing splines perform a regression on the orthonormal basis $u_1, \ldots, u_n \in \mathcal{R}^n$, yet they shrink the coefficients in this regression, with more shrinkage assigned to eigenvectors u_j that correspond to large eigenvalues d_j

- So what exactly are these basis vectors $u_1, \ldots, u_n$? These are known as **the Demmler-Reinsch basis**, and a lot of their properties can be worked out analytically (Demmler & Reinsch 1975). Basically: the eigenvectors u_j that correspond to smaller eigenvalues d_j are smoother, and so with smoothing splines, we shrink less in their direction. Said differently, by increasing λ in the smoothing spline estimator, we are tuning out the more wiggly components.

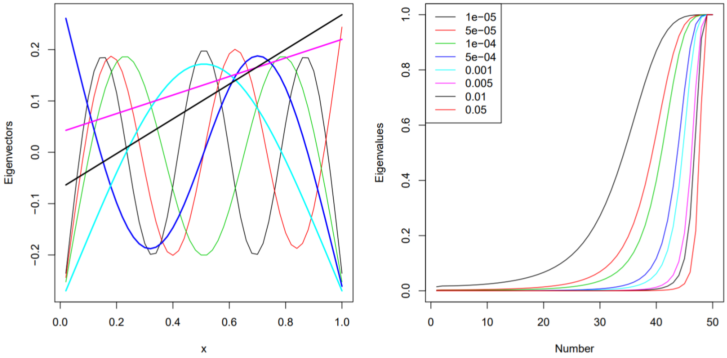


Figure 2: Eigenvectors and eigenvalues for the Reinsch form of the cubic smoothing spline operator, defined over $n = 50$ evenly spaced inputs on $[0, 1]$. The left plot shows the bottom 7 eigenvectors of the Reinsch matrix Q. We can see that the smaller the eigenvalue, the "smoother" the eigenvector. The right plot shows the weights $w_j = 1/(1 + \lambda d_j)$, $j = 1, \ldots, n$ implicitly used by the smoothing spline estimator (9), over 8 values of λ. We can see that when λ is larger, the weights decay faster, so the smoothing spline estimator places less weight on the "nonsmooth" eigenvectors

Multivariate spline

- Splines can be extended to multiple dimensions, in two different ways: **thin-plate splines** and **tensor-product splines**. The former construction is more computationally efficient but more in some sense more limiting; the penalty for a thin-plate spline, of polynomial order $k = 2m - 1$, is

$$\sum_{\alpha_1 + \cdots + \alpha_d = m} \int \left| \frac{\partial^m g(x)}{\partial x_a^{\alpha_1} \cdots \partial x_d^{\alpha_d}} \right|^2 dx$$

which is rotationally invariant. Both of these concepts are discussed in Chapter 7 of Green & Silverman (1994) (see also Chapters 15 and 20.4 of Gyorfi et al. (2002))

Green, P. & Silverman, B. (1994), Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach, Chapman & Hall/CRC Press.
Gyorfi, L., Kohler, M., Krzyzak, A. & Walk, H. (2002), A Distribution-Free Theory of Nonparametric Regression, Springer.

- The multivariate extensions (thin-plate and tensor-product) of splines are highly nontrivial, especially when we compare them to the (conceptually) simple extension of kernel smoothing to higher dimensions.
- In multiple dimensions, if one wants to study penalized nonparametric estimation, it's (argurably) easier to study reproducing kernel Hilbert space estimators. We'll see, in fact, that this covers smoothing splines (and thin-plate splines) as a special case.

Mercer kernels, RKHS

- Smoothing splines are just one example of an estimator of the form

$$\hat{g} = \arg\min_{g \in \mathcal{H}} \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda J(g),$$

  where $\mathcal{H}$ is a space of functions, and J is a penalty functional.
- Another important subclass of this problem form: we choose the function space $\mathcal{H} = \mathcal{H}_K$ to be what is called a **reproducing kernel Hilbert space**, or RKHS, associated with a particular kernel function $K : \mathcal{R}^d \times \mathcal{R}^d \mapsto \mathcal{R}$. **To avoid confusion: this is not the same thing as a smoothing kernel**! We'll adopt the convention of calling this second kind of kernel, i.e., the kind used in RKHS theory, a **Mercer kernel**, to differentiate the two.

- There is an immense literature on the RKHS framework; here we follow the RKHS treatment in Chapter 5 of Hastie et al. (2009). Suppose that K is a positive definite kernel; examples include the polynomial kernel

$$K(x,z) = (x'z+1)^k$$

and the Gaussian radial basis kernel:

$$K(x,z) = exp(-\delta \|x-z\|_2^2).$$

Mercer's theorem tells us that for any positive definite kernel function K, we have an eigenexpansion of the form

$$K(x,z) = \sum_{i=1}^{\infty} \gamma_i \phi_i(x) \phi_i(z),$$

for eigenfunction $\phi_i(x)$, $i = 1, 2, \ldots$ and eigenvalues $\gamma_i \geq 0, i = 1, 2, \ldots$, satisfying $\sum_{i=1}^{\infty} \gamma_i^2 < \infty$.

- We then define $\mathcal{H}_K$, the RKHS, as the space of functions generated by $K(\cdot, z), z \in \mathcal{R}^d$, i.e., elements in HK are of the form

$$g(x) = \sum_{m \in M} \alpha_m K(x, z_m),$$

for a (possibly infinite) set M.

- The above eigenexpansion of K implies that elements $g \in \mathcal{H}_K$ can be represented as

$$g(x) = \sum_{i=1}^{\infty} c_i \phi_i(x),$$

subject to the constraint that we must have $\sum_{i=1}^{\infty} c_i^2/\gamma_i < \infty$. In fact, this representation is used to define a norm $\| \cdot \|$ on $\mathcal{H}_K$: we define

$$\|g\|_{\mathcal{H}_K}^2 = \sum_{i=1}^{\infty} c_i^2/\gamma_i.$$

- The natural choice now is to take the penalty functional J as this squared RKHS norm, $J(g) = \|g\|^2_{\mathcal{H}_K}$. This yields the RKHS problem

$$\hat{g} = \arg\min_{g \in \mathcal{H}} \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \|g\|^2_{\mathcal{H}_K},$$

  A remarkable achievement of RKHS theory is that the above infinite-dimensional problem can be reduced to a finite-dimensional one (as was the case with smoothing splines). This is called the representer theorem and is attributed to Kimeldorf & Wahba (1970). In particular, this result tells us that the minimum in above problem is uniquely attained by a function of the form

$$g(x) = \sum_{i=1}^{n} \alpha_i K(x, x_i),$$

- or in other words, a function g lying in the span of the functions $K(\cdot, x_i)$, $i = 1, \ldots, n$. Furthermore, we can rewrite the above problem in finite-dimensional form, as

$$\hat{\alpha} = \arg \min_{\alpha \in \mathcal{R}^n} \|y - K\alpha\|_2^2 + \lambda \alpha' K \alpha \tag{10}$$

  where $K \in \mathcal{R}^{n \times n}$ is a symmetric matrix defined by $K_{ij} = K(x_i, x_j)$ for $i, j = 1, \ldots, n$. Once we have computed the optimal coefficients $\hat{\alpha}$, the estimated function $\hat{g}$ is give

$$\hat{g}(x) = \sum_{i=1}^{n} \hat{\alpha}_i K(x, x_i)$$

- Clearly, the solution in (10) is

$$\hat{\alpha} = (K + \lambda I)^{-1} y,$$

so the fitted values $\hat{\mu} = (\hat{g}(x_1), \ldots, \hat{g}(x_n))$ are

$$\hat{\mu} = K(K + \lambda I)^{-1} y = (I + \lambda K^{-1}) y,$$

showing that the RKHS estimator is yet again a linear smoother.

- In fact, it can be shown that thin-plate splines are themselves an example of smoothing via Mercel kernels, using the kernel $K(x,z) = \|x - z\|_2 log \|x - z\|_2$. See Chapter 7 of Green & Silverman (1994).

- Seen from a distance, there is something kind of subtle but extremely important about the problem in (10): to define a flexible nonparametric function, in multiple dimensions, note that we need not write down an explicit basis, but need only to define a "kernelized" inner product between any two input points, i.e., define the entries of the kernel matrix $K_{ij} = K(x_i, x_j)$. This encodes a notion of similarity between x_i, x_j, or equivalently,

  $$K(x_i, x_j) + K(x_j, x_j) - 2K(x_i, x_j)$$

  encodes a notion of distance between x_i and x_j.

- It can sometimes be much easier to define an appropriate kernel than to define explicit basis functions. Think about, e.g., the case when the input points are images, or strings, or some other weird objects-the kernel measure is defined entirely in terms of pairwise relationships between input objects, which can be done even in exotic input spaces.
- Given the kernel matrix K, the kernel regression problem (10) is completely specified, and the solution is implicitly fit to lie in the span of the (infinite-dimensional) RKHS generated by the chosen kernel. This is a pretty unique way of fitting flexible nonparametric regression estimates. Note: this idea isn't specific to regression: kernel classification, kernel PCA, etc., are built in the analogous way.