Nonparametric Regression (and Classification)

Statistical Machine Learning, Spring 2017
Ryan Tibshirani (with Larry Wasserman)

1 Introduction, and k-nearest-neighbors

1.1 Basic setup

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

$$f_0(x) = \mathbb{E}(Y|X = x)$$

is called the regression function (of Y on X). The basic goal in nonparametric regression: to construct an estimate $\hat{f}$ of f_0, from i.i.d. samples $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, n$ that have the same joint distribution as (X, Y). We often call X the input, predictor, feature, etc., and Y the output, outcome, response, etc.

- Note for i.i.d. samples $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, n$, we can always write

$$y_i = f_0(x_i) + \epsilon_i, \quad i = 1, \ldots, n,$$

where ϵ_i, $i = 1, \ldots, n$ are i.i.d. random errors, with mean zero. Therefore we can think about the sampling distribution as follows: (x_i, ϵ_i), $i = 1, \ldots, n$ are i.i.d. draws from some common joint distribution, where $\mathbb{E}(\epsilon_i) = 0$, and y_i, $i = 1, \ldots, n$ are generated from the above model

- It is typical to assume that each ϵ_i is independent of x_i. This is a pretty strong assumption, and you should think about it skeptically. We too will make this assumption, for simplicity. It should be noted that a good portion of theoretical results that we cover (or at least, similar theory) also holds without this assumption

1.2 Fixed or random inputs?

- Another common setup in nonparametric regression is to directly assume a model

$$y_i = f_0(x_i) + \epsilon_i, \quad i = 1, \ldots, n,$$

where now x_i, $i = 1, \ldots, n$ are *fixed* inputs, and ϵ_i, $i = 1, \ldots, n$ are i.i.d. with $\mathbb{E}(\epsilon_i) = 0$

- For arbitrary x_i, $i = 1, \ldots, n$, this is really just the same as starting with the random input model, and conditioning on the particular values of x_i, $i = 1, \ldots, n$. (But note: after conditioning on the inputs, the errors are only i.i.d. if we assumed that the errors and inputs were independent in the first place!)

- Generally speaking, nonparametric regression estimators are not defined with the random or fixed setups specifically in mind, i.e., there is no real distinction made here. A caveat: some estimators (like wavelets) do in fact assume evenly spaced fixed inputs, as in

$$x_i = i/n, \quad i = 1, \ldots, n,$$

for evenly spaced inputs in the univariate case

- Theory is not completely the same between the random and fixed input worlds (some theory is sharper when we assume fixed input points, especially evenly spaced input points), but for the most part the theory is quite similar

- Therefore, in what follows, we won't be very precise about which setup we assume—random or fixed inputs—because it mostly doesn't matter when introducing nonparametric regression estimators and discussing basic properties

1.3 Notation

- We will define an empirical norm $\|\cdot\|_n$ in terms of the training points x_i, $i = 1, \ldots, n$, acting on functions $f : \mathbb{R}^d \to \mathbb{R}$, by

$$\|f\|_n^2 = \frac{1}{n} \sum_{i=1}^n f^2(x_i).$$

  This makes sense no matter if the inputs are fixed or random (but in the latter case, it is a random norm)

- When the inputs are considered random, we will write P_X for the distribution of X, and we will define the L_2 norm $\|\cdot\|_2$ in terms of P_X, acting on functions $f : \mathbb{R}^d \to \mathbb{R}$, by

$$\|f\|_2^2 = \mathbb{E}[f^2(X)] = \int f^2(x)\, dP_X(x).$$

  So when you see $\|\cdot\|_2$ in use, it is a hint that the inputs are being treated as random

- A quantity of interest will be the (squared) error associated with an estimator $\hat{f}$ of f_0, which can be measured in either norm:

$$\|\hat{f} - f_0\|_n^2 \quad \text{or} \quad \|\hat{f} - f_0\|_2^2.$$

  In either case, this is a random quantity (since $\hat{f}$ is itself random). We will study bounds in probability or in expectation. The expectation of the errors defined above, in terms of either norm (but more typically the L_2 norm) is most properly called the risk; but we will often be a bit loose in terms of our terminology and just call this the error

1.4 What does "nonparametric" mean?

- Importantly, in nonparametric regression we don't assume a particular parametric form for f_0. This doesn't mean, however, that we can't estimate f_0 using (say) a linear combination of spline basis functions, written as $\hat{f}(x) = \sum_{j=1}^p \hat{\beta}_j g_j(x)$. A common question: the coefficients on the spline basis functions $\beta_1, \ldots, \beta_p$ are parameters, so how can this be nonparametric? Again, the point is that *we don't assume a parametric form for* f_0, i.e., we don't assume that f_0 itself is an exact linear combination of splines basis functions $g_1, \ldots, g_p$

- With (say) splines as a modeler, we can still derive rigorous results about the error incurred in estimating arbitrary functions, because "splines are nearly everywhere". In other words, for any appropriately smooth function, there's a spline that's very close to it. Classic results in approximation theory (de Boor 1978) make this precise; e.g., for any twice differentiable function f_0 on $[0, 1]$, and points $t_1, \ldots, t_N \in [0, 1]$, there is a cubic spline $\bar{f}_0^{\text{spl}}$ with knots at $t_1, \ldots, t_N$ such that

$$\sup_{x \in [0,1]} |\bar{f}_0^{\text{spl}}(x) - f_0(x)| \leq \frac{C}{N} \sqrt{\int_0^1 f_0''(x)^2\, dx},$$

for a constant $C > 0$. Note $\int_0^1 f_0''(x)^2\,dx$ is a measure of the smoothness of f_0. If this remains constant, and we choose $N = \sqrt{n}$ knots, then the approximation error is on the order of $1/n$, which is smaller than the final statistical estimation error that we should expect

1.5 What we cover here

- The goal is to expose you to a variety of methods, and give you a flavor of some interesting results, under different assumptions. A few topics we will cover into more depth than others, but overall, this will be far from a complete treatement of nonparametric regression. Below are some excellent texts out there that you can consult for more details, proofs, etc.
  - Kernel smoothing, local polynomials: Tsybakov (2009)
  - Regression splines, smoothing splines: de Boor (1978), Green & Silverman (1994), Wahba (1990)
  - Reproducing kernel Hilbert spaces: Scholkopf & Smola (2002), Wahba (1990)
  - Wavelets: Johnstone (2011), Mallat (2008).
  - General references, more theoretical: Gyorfi, Kohler, Krzyzak & Walk (2002), Wasserman (2006)
  - General references, more methodological: Hastie & Tibshirani (1990), Hastie, Tibshirani & Friedman (2009), Simonoff (1996)
- Throughout, our discussion will bounce back and forth between the multivariate case ($d > 1$) and univariate case ($d = 1$). Some methods have obvious (natural) multivariate extensions; some don't. In any case, we can always use low-dimensional (even just univariate) nonparametric regression methods as building blocks for a high-dimensional nonparametric method. We'll study this near the end, when we talk about additive models
- Lastly, a lot of what we cover for nonparametric regression also carries over to nonparametric classification, which we'll cover (in much less detail) at the end

1.6 k-nearest-neighbors regression

- Here's a basic method to start us off: *k-nearest-neighbors* regression. We fix an integer $k \geq 1$ and define

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

  where $\mathcal{N}_k(x)$ contains the indices of the k closest points of $x_1, \ldots, x_n$ to x
- This is not at all a bad estimator, and you will find it used in lots of applications, in many cases probably because of its simplicity. By varying the number of neighbors k, we can achieve a wide range of flexibility in the estimated function $\hat{f}$, with small k corresponding to a more flexible fit, and large k less flexible
- But it does have its limitations, an apparent one being that the fitted function $\hat{f}$ essentially always looks jagged, especially for small or moderate k. Why is this? It helps to write

$$\hat{f}(x) = \sum_{i=1}^n w_i(x) y_i, \tag{2}$$

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

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

Note that $w_i(x)$ is discontinuous as a function of x, and therefore so is $\hat{f}(x)$

- The representation (2) also reveals that the k-nearest-neighbors estimate is in a class of estimates we call *linear smoothers*, i.e., writing $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$, the vector of fitted values

$$\hat{\mu} = (\hat{f}(x_1), \ldots, \hat{f}(x_n)) \in \mathbb{R}^n$$

can simply be expressed as $\hat{\mu} = Sy$. (To be clear, this means that for fixed inputs $x_1, \ldots, x_n$, the vector of fitted values $\hat{\mu}$ is a linear function of y; it does not mean that $\hat{f}(x)$ need behave linearly as a function of x!) This class is quite large, and contains many popular estimators, as we'll see in the coming sections

- The k-nearest-neighbors estimator is *universally consistent*, which means $\mathbb{E}\|\hat{f} - f_0\|_2^2 \to 0$ as $n \to \infty$, with no assumptions other than $\mathbb{E}(Y^2) \leq \infty$, provided that we take $k = k_n$ such that $k_n \to \infty$ and $k_n/n \to 0$; e.g., $k = \sqrt{n}$ will do. See Chapter 6.2 of Gyorfi et al. (2002)

- Furthermore, assuming the underlying regression function f_0 is Lipschitz continuous, the k-nearest-neighbors estimate with $k \asymp n^{2/(2+d)}$ satisfies

$$\mathbb{E}\|\hat{f} - f_0\|_2^2 \lesssim n^{-2/(2+d)}. \tag{3}$$

See Chapter 6.3 of Gyorfi et al. (2002)

- Proof sketch: assume that $\mathrm{Var}(Y|X = x) = \sigma^2$, a constant, for simplicity, and fix (condition on) the training points. Using the bias-variance tradeoff,

$$\begin{aligned} \mathbb{E}\big[\big(\hat{f}(x) - f_0(x)\big)^2\big] &= \underbrace{\big(\mathbb{E}[\hat{f}(x)] - f_0(x)\big)^2}_{\mathrm{Bias}^2(\hat{f}(x))} + \underbrace{\mathbb{E}\big[\big(\hat{f}(x) - \mathbb{E}[\hat{f}(x)]\big)^2\big]}_{\mathrm{Var}(\hat{f}(x))} \\ &= \bigg(\frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} \big(f_0(x_i) - f_0(x)\big)\bigg)^2 + \frac{\sigma^2}{k} \\ &\leq \bigg(\frac{L}{k} \sum_{i \in \mathcal{N}_k(x)} \|x_i - x\|_2\bigg)^2 + \frac{\sigma^2}{k}. \end{aligned}$$

In the last line we used the Lipschitz property $|f_0(x) - f_0(z)| \leq L\|x - z\|_2$, for some constant $L > 0$. Now for "most" of the points we'll have $\|x_i - x\|_2 \leq C(k/n)^{1/d}$, for a constant $C > 0$. (Think of a having input points x_i, $i = 1, \ldots, n$ spaced equally over (say) $[0, 1]^d$.) Then our bias-variance upper bound becomes

$$(CL)^2 \bigg(\frac{k}{n}\bigg)^{2/d} + \frac{\sigma^2}{k},$$

We can minimize this by balancing the two terms so that they are equal, giving $k^{1+2/d} \asymp n^{2/d}$, i.e., $k \asymp n^{2/(2+d)}$ as claimed. Plugging this in gives the error bound of $n^{-2/(2+d)}$, as claimed

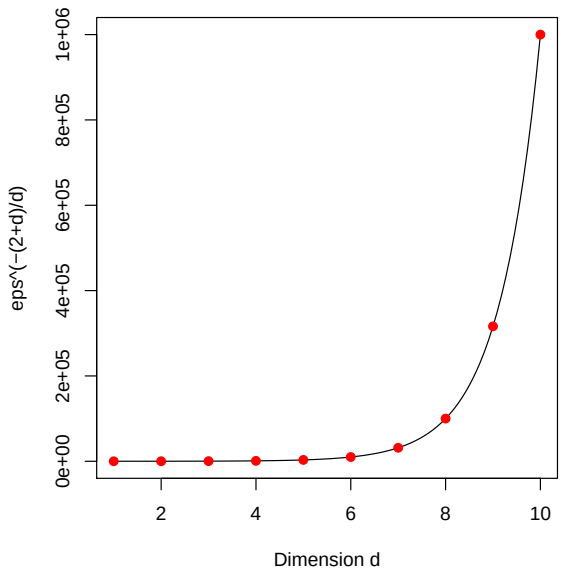

Figure 1: *The curse of dimensionality, with* $\epsilon = 0.1$

1.7 Curse of dimensionality

- Note that the above error rate $n^{-2/(2+d)}$ exhibits a very poor dependence on the dimension d. To see it differently: given a small $\epsilon > 0$, think about how large we need to make n to ensure that $n^{-2/(2+d)} \leq \epsilon$. Rearranged, this says $n \geq \epsilon^{-(2+d)/2}$. That is, as we increase d, we require *exponentially more samples* n to achieve an error bound of ϵ. See Figure 1 for an illustration with $\epsilon = 0.1$

- In fact, this phenomenon is not specific to k-nearest-neighbors, but a reflection of the *curse of dimensionality*, the principle that estimation becomes exponentially harder as the number of dimensions increases. This is made precise by minimax theory: we cannot hope to do better than the rate in(3) over $H_d(1, L)$, which we write for the space of L-Lipschitz functions in d dimensions, for a constant $L > 0$. It can be shown that

$$\inf_{\hat{f}} \sup_{f_0 \in H_d(1,L)} \mathbb{E}\|\hat{f} - f_0\|_2^2 \gtrsim n^{-2/(2+d)}, \tag{4}$$

  where the infimum above is over all estimators $\hat{f}$. See Chapter 3.2 of Gyorfi et al. (2002)

- So to circumvent this curse, we need to make more assumptions about what we're looking for in high dimensions. One such example is the additive model, covered near the end

2 Kernel smoothing, local polynomials

2.1 Kernel smoothing

- *Kernel regression* or *kernel smoothing* begins with a kernel function $K : \mathbb{R} \to \mathbb{R}$, satisfying

$$\int K(t)\,dt = 1, \quad \int tK(t)\,dt = 0, \quad 0 < \int t^2 K(t)\,dt < \infty.$$

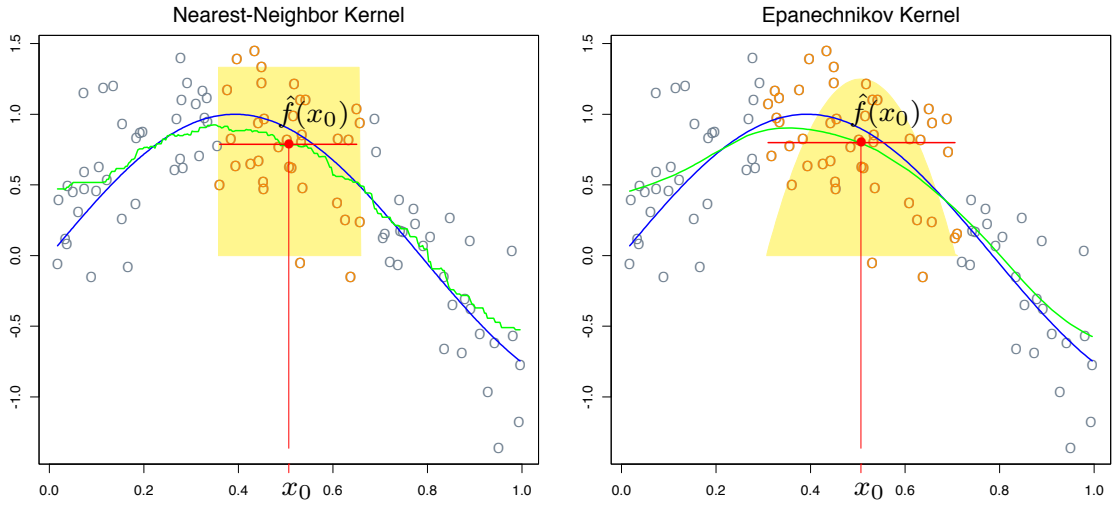

Figure 2: *Comparing k-nearest-neighbor and Epanechnikov kernels, when* $d = 1$. *From Chapter 6 of Hastie et al. (2009)*

Three common examples are the box-car kernel:

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

the Gaussian kernel:

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

and the Epanechnikov kernel:

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

- Given a bandwidth $h > 0$, the (Nadaraya-Watson) kernel regression estimate is defined as

$$\hat{f}(x) = \frac{\sum_{i=1}^{n} K\left(\frac{\|x - x_i\|_2}{h}\right) y_i}{\sum_{i=1}^{n} K\left(\frac{\|x - x_i\|_2}{h}\right)}. \tag{5}$$

  Hence kernel smoothing is also a linear smoother (2), with choice of weights $w_i(x) = K(\|x - x_i\|_2/h) / \sum_{j=1}^{n} K(\|x - x_j\|_2/h)$

- In comparison to the k-nearest-neighbors estimator in (1), which can be thought of as a raw (discontinuous) moving average of nearby responses, the kernel estimator in (5) is a smooth moving average of responses. See Figure 2 for an example with $d = 1$

- A shortcoming: the kernel regression suffers from poor bias at the boundaries of the domain of the inputs $x_1, \ldots x_n$. This happens because of the asymmetry of the kernel weights in such regions. See Figure 3

2.2 Local linear regression

- We can alleviate this boundary bias issue by moving from a local constant fit to a local linear fit, or a local higher-order fit
- To build intuition, another way to view the kernel estimator in (5) is the following: at each input x, it employs the estimate $\hat{f}(x) = \hat{\theta}_x$, where $\hat{\theta}_x$ is the minimizer of

$$\sum_{i=1}^n K\left(\frac{\|x - x_i\|_2}{h}\right)(y_i - \theta)^2,$$

over all $\theta \in \mathbb{R}$. Instead we could consider forming the local estimate $\hat{f}(x) = \hat{\alpha}_x + \hat{\beta}_x^T x$, where $\hat{\alpha}_x, \hat{\beta}_x$ minimize

$$\sum_{i=1}^n K\left(\frac{\|x - x_i\|_2}{h}\right)(y_i - \alpha - \beta^T x_i)^2.$$

over all $\alpha \in \mathbb{R}$, $\beta \in \mathbb{R}^d$. This is called *local linear regression*

- We can rewrite the local linear regression estimate $\hat{f}(x)$. This is just given by a weighted least squares fit, so

$$\hat{f}(x) = b(x)^T (B^T \Omega B)^{-1} B^T \Omega y,$$

where $b(x) = (1, x) \in \mathbb{R}^{d+1}$, $B \in \mathbb{R}^{n \times (d+1)}$ with ith row $b(x_i)$, and $\Omega \in \mathbb{R}^{n \times n}$ is diagonal with ith diagonal element $K(\|x - x_i\|_2/h)$. We can write more concisely as $\hat{f}(x) = w(x)^T y$, where $w(x) = \Omega B (B^T \Omega B)^{-1} b(x)$, which shows local linear regression is a linear smoother too

- The vector of fitted values $\hat{\mu} = (\hat{f}(x_1), \ldots, \hat{f}(x_n))$ can be expressed as

$$\hat{\mu} = \begin{pmatrix} w_1(x)^T y \\ \vdots \\ w_n(x)^T y \end{pmatrix} = B(B^T \Omega B)^{-1} B^T \Omega y,$$

which should look familiar to you from weighted least squares

- Now we'll sketch how the local linear fit reduces the bias, fixing (conditioning on) the training points. Compute at a fixed point x,

$$\mathbb{E}[\hat{f}(x)] = \sum_{i=1}^n w_i(x) f_0(x_i).$$

Using a Taylor expansion of f_0 about x,

$$\mathbb{E}[\hat{f}(x)] = f_0(x) \sum_{i=1}^n w_i(x) + \nabla f_0(x)^T \sum_{i=1}^n (x_i - x) w_i(x) + R,$$

where the remainder term R contains quadratic and higher-order terms, and under regularity conditions, is small. One can check that in fact for the local linear regression estimator $\hat{f}$,

$$\sum_{i=1}^n w_i(x) = 1 \quad \text{and} \quad \sum_{i=1}^n (x_i - x) w_i(x) = 0,$$

and so $\mathbb{E}[\hat{f}(x)] = f_0(x) + R$, which means that $\hat{f}$ is unbiased to first-order

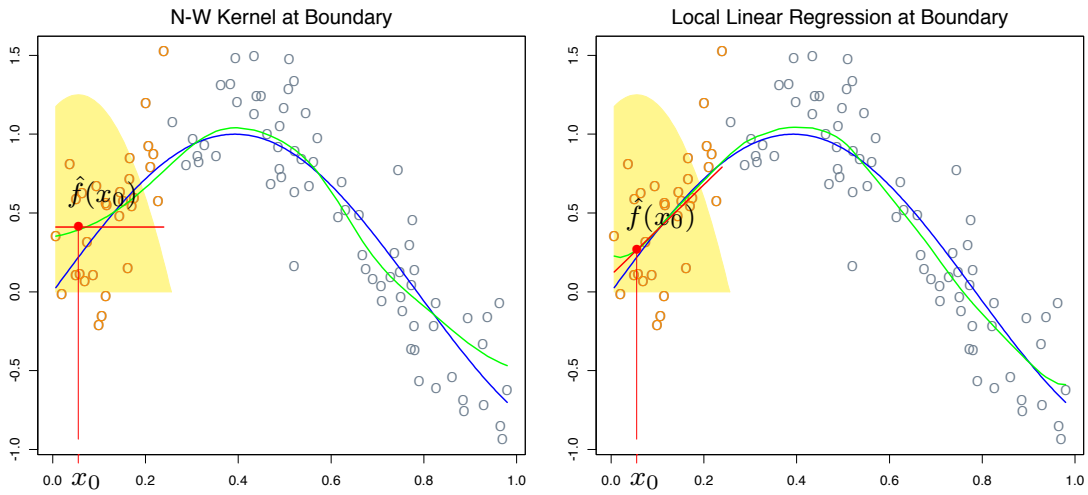

Figure 3: *Comparing (Nadaraya-Watson) kernel smoothing to local linear regression; the former is biased at the boundary, the latter is unbiased (to first-order). From Chapter 6 of Hastie et al. (2009)*

2.3 Consistency and error rates

- The kernel smoothing estimator is universally consistent (recall: $\mathbb{E}\|\hat{f} - f_0\|_2^2 \to 0$ as $n \to \infty$, with no assumptions other than $\mathbb{E}(Y^2) \leq \infty$), provided we take a compactly supported kernel K, and bandwidth $h = h_n$ satisfying $h_n \to 0$ and $nh_n^d \to \infty$ as $n \to \infty$. See Chapter 5.2 of Gyorfi et al. (2002)

- Now let us define the *Holder class* of functions $H_d(k+\gamma, L)$, for an integer $k \geq 0$, $0 < \gamma \leq 1$, and $L > 0$, to contain all k times differentiable functions $f : \mathbb{R}^d \to \mathbb{R}$ such that

$$\left| \frac{\partial^k f(x)}{\partial x_1^{\alpha_1} x_2^{\alpha_2} \ldots \partial x_d^{\alpha_d}} - \frac{\partial^k f(z)}{\partial x_1^{\alpha_1} x_2^{\alpha_2} \ldots \partial x_d^{\alpha_d}} \right| \leq L\|x - z\|_2^{\gamma}, \quad \text{for all } x, z, \text{ and } \alpha_1 + \ldots + \alpha_d = k.$$

  Note that $H_d(1, L)$ is the space of all L-Lipschitz functions, and $H_d(k+1, L)$ is the space of all functions whose kth-order partial derivatives are L-Lipschitz

- Assuming that the underlying regression function f_0 is Lipschitz, $f_0 \in H_d(1, L)$ for a constant $L > 0$, the kernel smoothing estimator with a compactly supported kernel K and bandwidth $h \asymp n^{-1/(2+d)}$ satisfies

$$\mathbb{E}\|\hat{f} - f_0\|_2^2 \lesssim n^{-2/(2+d)}. \tag{6}$$

  See Chapter 5.3 of Gyorfi et al. (2002)

- Recall from (4) we saw that this was the minimax optimal rate over $H_d(1, L)$. More generally, the minimax rate over $H_d(\alpha, L)$, for a constant $L > 0$, is

$$\inf_{\hat{f}} \sup_{f_0 \in H_d(\alpha, L)} \mathbb{E}\|\hat{f} - f_0\|_2^2 \gtrsim n^{-2\alpha/(2\alpha+d)}, \tag{7}$$

  see again Chapter 3.2 of Gyorfi et al. (2002)

- We also saw in (3) that the k-nearest-neighbors estimator achieved the same (optimal) error rate as kernel smoothing in (6), for Lipschitz f_0. Shouldn't the kernel estimator be better, because it is smoother? The answer is kind of both "yes" and "no"

- The "yes" part: kernel smoothing still achieves the optimal convergence rate over $H_d(1.5, L)$ whereas it is believed (conjectured) that this not true for k-nearest-neighbors; see Chapters 5.3 and 6.3 of Gyorfi et al. (2002)

- The "no" part: neither kernel smoothing nor k-nearest-neighbors are optimal over $H_d(2, L)$. (An important remark: here we see a big discrepancy between the L_2 and pointwise analyses. Indeed, it can be shown that both kernel smoothing and k-nearest-neighbors satisfy

$$\mathbb{E}\big(\hat{f}(x) - f_0(x)\big)^2 \lesssim n^{-4/(4+d)}$$

  for each fixed x, in the interior of the support of input density, when $f_0 \in H_d(2, L)$. But the same is not true when we integrate over x; it is boundary bias that kills the error rate, for both methods.)

- As an aside, why did we study the Holder class $H_d(k + \gamma, L)$? Because the analysis for kernel smoothing can be done via Taylor expansions, and it becomes pretty apparent that things will work out if we can bound the (partial) derivatives. So, in essense, it makes our proofs easier!

2.4 Higher-order smoothness

- How can we hope to get optimal error rates over $H_d(\alpha, d)$, when $\alpha \geq 2$? With kernels there are basically two options: use local polynomials, or use higher-order kernels

- Local polynomials build on our previous idea of local linear regression (itself an extension of kernel smoothing.) Consider $d = 1$, for concreteness. Define $\hat{f}(x) = \hat{\beta}_{x,0} + \sum_{j=1}^k \hat{\beta}_{x,j} x^j$, where $\hat{\beta}_{x,0}, \ldots, \hat{\beta}_{x,k}$ minimize

$$\sum_{i=1}^n K\left(\frac{|x - x_i|}{h}\right)\left(y_i - \beta_0 - \sum_{j=1}^k \beta_j x_i^j\right)^2.$$

  over all $\beta_0, \beta_1, \ldots, \beta_k \in \mathbb{R}$. This is called ($k$th-order) *local polynomial regression*

- Again we can express

$$\hat{f}(x) = b(x)(B^T \Omega B)^{-1} B^T \Omega y = w(x)^T y,$$

  where $b(x) = (1, x, \ldots, x^k)$, B is an $n \times (k+1)$ matrix with ith row $b(x_i) = (1, x_i, \ldots, x_i^k)$, and Ω is as before. Hence again, local polynomial regression is a linear smoother

- Assuming that $f_0 \in H_1(\alpha, L)$ for a constant $L > 0$, a Taylor expansion shows that the local polynomial estimator $\hat{f}$ of order k, where k is the largest integer strictly less than α and where the bandwidth scales as $h \asymp n^{-1/(2\alpha+1)}$, satisfies

$$\mathbb{E}\|\hat{f} - f_0\|_2^2 \lesssim n^{-2\alpha/(2\alpha+1)}.$$

  See Chapter 1.6.1 of Tsybakov (2009). This matches the lower bound in (7) (when $d = 1$)

- In multiple dimensions, $d > 1$, local polynomials become kind of tricky to fit, because of the explosion in terms of the number of parameters we need to represent a kth order polynomial in d variables. Hence, an interesting alternative is to return back kernel smoothing but use a *higher-order kernel*. A kernel function K is said to be of order k provided that

$$\int K(t)\,dt = 1, \quad \int t^j K(t)\,dt = 0, \quad j = 1, \ldots, k-1, \quad \text{and} \quad 0 < \int t^k K(t)\,dt < \infty.$$

  This means that the kernels we were looking at so far were of order 2

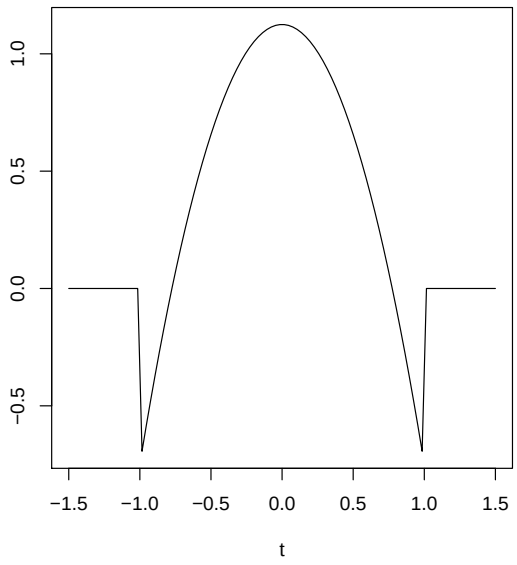

Figure 4: *A higher-order kernel function: specifically, a kernel of order 4*

- An example of a 4th-order kernel is $K(t) = \frac{3}{8}(3 - 5t^2)1\{|t| \leq 1\}$, plotted in Figure 4. Notice that it takes negative values! Higher-order kernels, in fact, have an interesting connection to smoothing splines, which we'll learn shortly

- Lastly, while local polynomial regression and higher-order kernel smoothing can help "track" the derivatives of smooth functions $f_0 \in H_d(\alpha, L)$, $\alpha \geq 2$, it should be noted that they don't share the same universal consistency property of kernel smoothing (or k-nearest-neighbors). See Chapters 5.3 and 5.4 of Gyorfi et al. (2002)

3 Regression splines, smoothing splines

3.1 Splines

- Regression splines and smoothing splines are motivated from a different perspective than kernels and local polynomials; in the latter case, we started off with a special kind of local averaging, and moved our way up to a higher-order local models. With regression splines and smoothing splines, we build up our estimate globally, from a set of select basis functions

- These basis functions, as you might guess, are *splines*. Let's assume that $d = 1$ for simplicity. (We'll stay in the univariate case, for the most part, in this section.) A kth-order spline f 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. Specifically, there are $t_1 < \ldots < t_p$ such that f is a polynomial of degree k on each of the intervals

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

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

- 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. See Figure 5

- 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

$$\begin{aligned} g_1(x) = 1,\ g_2(x) = x,\ \ldots\ g_{k+1}(x) = x^k, \\ g_{k+1+j}(x) = (x - t_j)_+^k, \quad j = 1, \ldots, p. \end{aligned} \tag{8}$$

  (Here x_+ denotes the positive part of x, i.e., $x_+ = \max\{x, 0\}$.) From this we can see that the space of kth-order splines with knots at $t_1, \ldots, t_p$ has dimension $p + k + 1$

- While these basis functions are natural, a much better computational choice, both for speed and numerical accuracy, is the *B-spline* basis. This was a major development in spline theory and is now pretty much the standard in software. The key idea: B-splines have local support, so a basis matrix that we form with them (to be defined below) is banded. See de Boor (1978) or the Appendix of Chapter 5 in Hastie et al. (2009) for details

3.2 Regression splines

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

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

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

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

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

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

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^{p+k+1}}{\operatorname{argmin}} \ \|y - G\beta\|_2^2,$$

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

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

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

  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

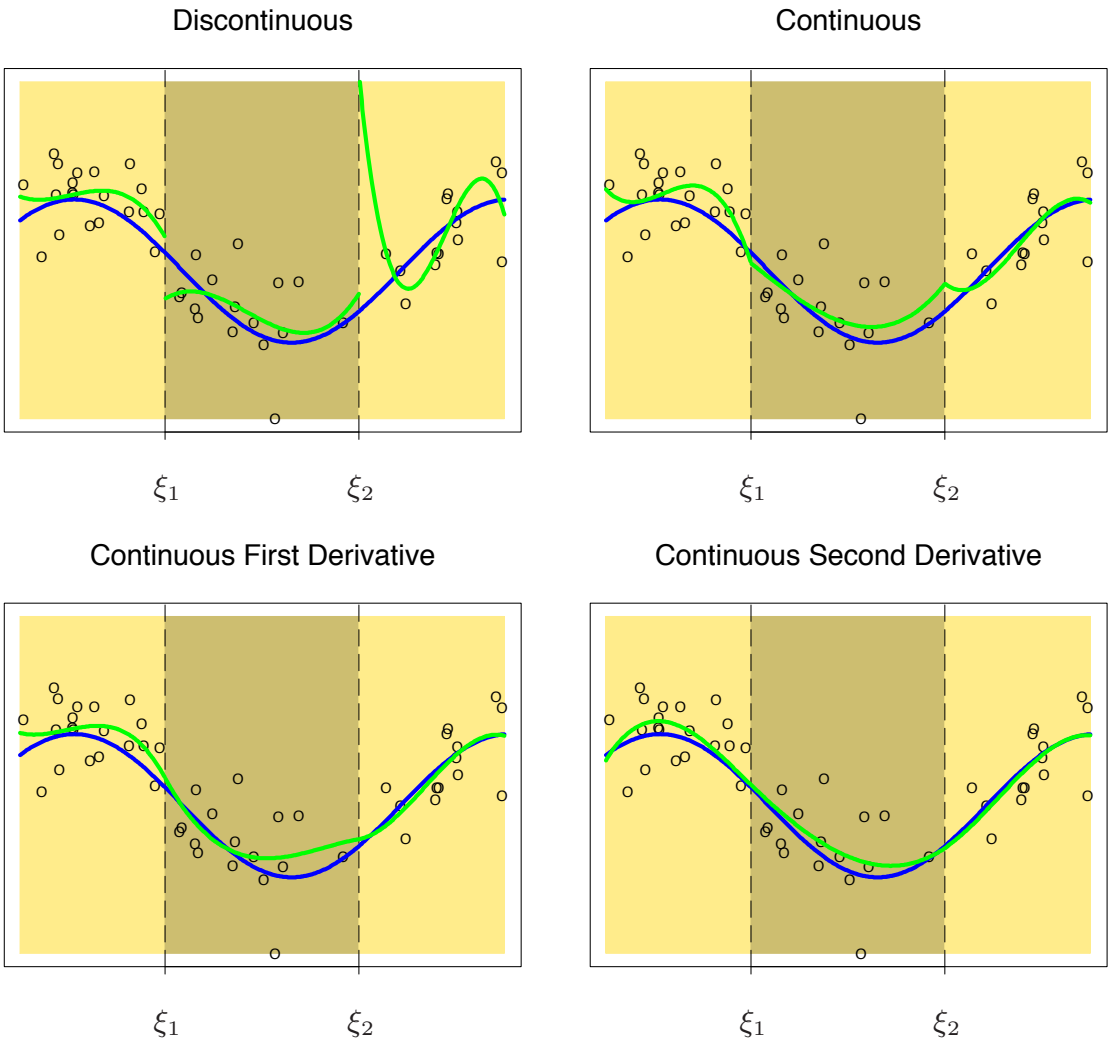

Figure 5: *Illustration of the effects of enforcing continuity at the knots, across various orders of the derivative, for a cubic piecewise polynomial. From Chapter 5 of Hastie et al. (2009)*

3.3 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 $t_1 < \ldots < t_p$, is a piecewise polynomial function f such that
  - f is a polynomial of degree k on each of $[t_1, t_2], \ldots, [t_{p-1}, t_p]$,
  - f is a polynomial of degree $(k-1)/2$ on $(-\infty, t_1]$ and $[t_p, \infty)$,
  - f is 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:

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

  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

3.4 Smoothing splines

- Smoothing splines, at the end of the day, are given by a regularized regression over the natural spline basis, placing knots at all inputs $x_1, \ldots, x_n$. They circumvent the problem of knot selection (as they just use the inputs as knots), and they control for overfitting by shrinking the coefficients of the estimated function (in its basis expansion)
- Interestingly, we can motivate and define a smoothing spline directly from a functional minimization perspective. With inputs $x_1, \ldots, x_n$ lying in an interval $[0, 1]$, the *smoothing spline* estimate $\hat{f}$, of a given odd integer order $k \geq 0$, is defined as

$$\hat{f} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \int_0^1 \big(f^{(m)}(x)\big)^2 \, dx, \quad \text{where } m = (k+1)/2. \tag{9}$$

  This is an infinite-dimensional optimization problem over all functions f for the which the criterion is finite. This criterion trades off the least squares error of f over the observed pairs (x_i, y_i), $i = 1, \ldots, n$, with a penalty term that is large when the mth derivative of f is wiggly. The tuning parameter $\lambda \geq 0$ governs the strength of each term in the minimization

- By far the most commonly considered case is $k = 3$, i.e., cubic smoothing splines, which are defined as

$$\hat{f} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \int_0^1 f''(x)^2 \, dx \tag{10}$$

- Remarkably, it so happens that the minimizer in the general smoothing spline problem (9) 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, $k = 3$, from Green & Silverman (1994) (see also Exercise 5.7 in Hastie et al. (2009))

  The key result can be stated as follows: if $\tilde{f}$ is any twice differentiable function on $[0, 1]$, and $x_1, \ldots, x_n \in [0, 1]$, then there exists a natural cubic spline f with knots at $x_1, \ldots, x_n$ such that $f(x_i) = \tilde{f}(x_i)$, $i = 1, \ldots, n$ and

$$\int_0^1 f''(x)^2 \, dx \leq \int_0^1 \tilde{f}''(x)^2 \, dx.$$

  Note that this would in fact prove that we can restrict our attention in (10) to natural splines with knots at $x_1, \ldots, x_n$

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

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

  Consider

$$\begin{aligned}
\int_0^1 f''(x)h''(x) \, dx &= f''(x)h'(x)\Big|_0^1 - \int_0^1 f'''(x)h'(x) \; dx \\
&= -\int_{x_1}^{x_n} f'''(x)h'(x) \; dx \\
&= -\sum_{j=1}^{n-1} f'''(x)h(x)\Big|_{x_j}^{x_{j+1}} + \int_{x_1}^{x_n} f^{(4)}(x)h'(x) \; dx \\
&= -\sum_{j=1}^{n-1} f'''(x_j^+)\big(h(x_{j+1}) - h(x_j)\big),
\end{aligned}$$

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

$$\int_0^1 f''(x)h''(x) \, dx = 0.$$

  From this, it follows that

$$\begin{aligned}
\int_0^1 \tilde{f}''(x)^2 \, dx &= \int_0^1 \big(f''(x) + h''(x)\big)^2 \, dx \\
&= \int_0^1 f''(x)^2 \, dx + \int_0^1 h''(x)^2 \, dx + 2\int_0^1 f''(x)h''(x) \, dx \\
&= \int_0^1 f''(x)^2 \, dx + \int_0^1 h''(x)^2 \, dx,
\end{aligned}$$

and therefore

$$\int_0^1 f''(x)^2\,dx \leq \int_0^1 \tilde{f}''(x)^2\,dx, \tag{11}$$

with equality if and only if $h''(x) = 0$ for all $x \in [0,1]$. Note that $h'' = 0$ 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 inequality (11) holds strictly except when $\tilde{f} = f$, so the solution in (10) is uniquely a natural spline with knots at the inputs

3.5 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 (9) as

$$\hat{\beta} = \underset{\beta\in\mathbb{R}^n}{\operatorname{argmin}} \sum_{i=1}^n \Big(y_i - \sum_{j=1}^n \beta_j \eta_j(x_i)\Big)^2 + \lambda \int_0^1 \Big(\sum_{j=1}^n \beta_j \eta_j^{(m)}(x)\Big)^2 dx. \tag{12}$$

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

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

$$N_{ij} = \eta_j(x_i) \quad \text{and} \quad \Omega_{ij} = \int_0^1 \eta_i^{(m)}(x)\eta_j^{(m)}(x)\,dx \quad \text{for } i,j = 1,\ldots,n, \tag{13}$$

  the problem in (12) can be written more succintly as

$$\hat{\beta} = \underset{\beta\in\mathbb{R}^n}{\operatorname{argmin}} \; \|y - N\beta\|_2^2 + \lambda\beta\Omega\beta, \tag{14}$$

  showing the smoothing spline problem to be a type of generalized ridge regression problem. In fact, the solution in (14) has the explicit form

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

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

$$\hat{\mu} = N(N^T N + \lambda\Omega)^{-1} N^T y. \tag{15}$$

  Therefore, once again, smoothing splines are a type of linear smoother

- A special property of smoothing splines: the fitted values in (15) 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 $N^T N + \Omega I$ ends up being banded (with a bandwidth that only depends on the polynomial order k). In practice, smoothing spline computations are extremely fast

3.6 Reinsch form

- It is informative to rewrite the fitted values in (15) is what is called Reinsch form,

$$\begin{aligned}
\hat{\mu} &= N(N^T N + \lambda\Omega)^{-1} N^T y \\
&= N\Big(N^T\big(I + \lambda(N^T)^{-1}\Omega N^{-1}\big)N\Big)^{-1} N^T y \\
&= (I + \lambda Q)^{-1} y,
\end{aligned} \tag{16}$$

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

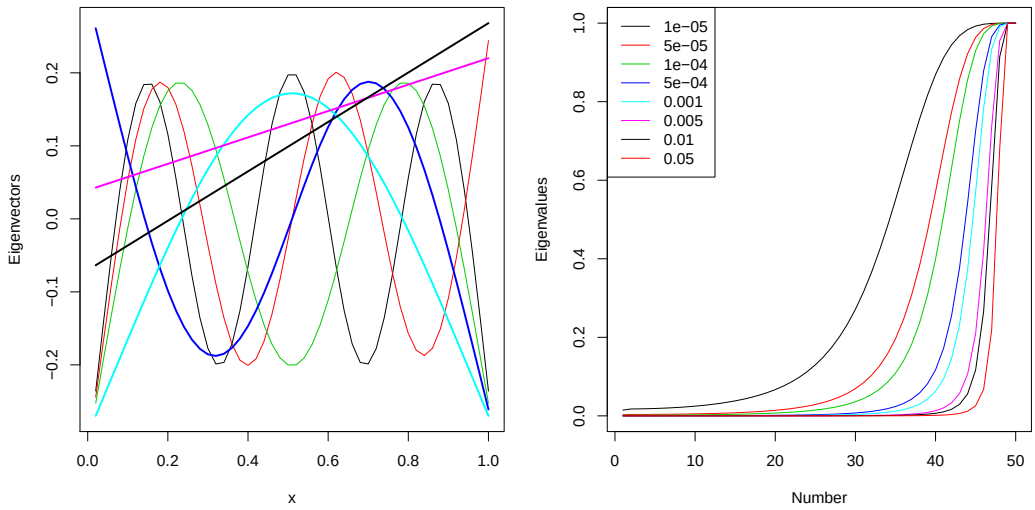

Figure 6: *Eigenvectors and eigenvalues for the Reinsch form of the cubic smoothing spline operator, defined over $n = 50$ evenly spaced inputs on $[0, 1]$. The left plot shows the bottom 7 eigenvectors of the Reinsch matrix Q. We can see that the smaller the eigenvalue, the "smoother" the eigenvector. The right plot shows the weights $w_j = 1/(1 + \lambda d_j)$, $j = 1, \ldots, n$ implicitly used by the smoothing spline estimator (17), 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*

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

$$S = \sum_{j=1}^{n} \frac{1}{1 + \lambda d_j} u_j u_j^T,$$

  where $D = \mathrm{diag}(d_1, \ldots, d_n)$

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

$$\hat{\mu} = \sum_{j=1}^{n} \frac{u_j^T y}{1 + \lambda d_j} u_j. \tag{17}$$

  Interpretation: smoothing splines perform a regression on the orthonormal basis $u_1, \ldots, u_n \in \mathbb{R}^n$, 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. See Figure 6

3.7 Kernel smoothing equivalence

- Something interesting happens when we plot the rows of the smoothing spline matrix S. For evenly spaced inputs, they look like the translations of a kernel! See Figure 7, left plot. For

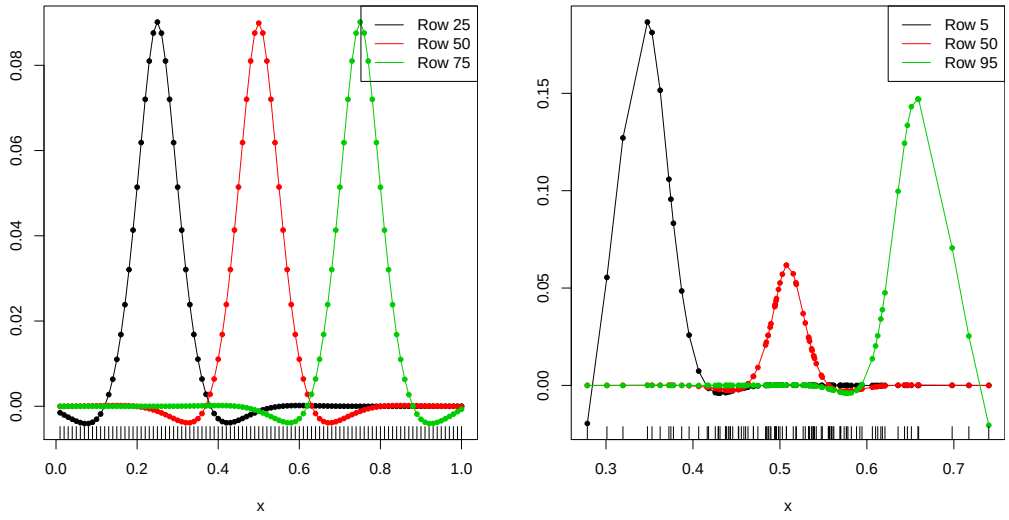

Figure 7: *Rows of the cubic smoothing spline operator S defined over $n = 100$ evenly spaced input points on $[0, 1]$. The left plot shows 3 rows of S (in particular, rows 25, 50, and 75) for $\lambda = 0.0002$. These look precisely like translations of a kernel. The right plot considers a setup where the input points are concentrated around 0.5, and shows 3 rows of S (rows 5, 50, and 95) for the same value of λ. These still look like kernels, but the bandwidth is larger in low-density regions of the inputs*

unevenly spaced inputs, the rows still have a kernel shape; now, the bandwidth appears to adapt to the density of the input points: lower density, larger bandwidth. See Figure 7, right plot

- What we are seeing is an empirical validation of a beautiful asymptotic result by Silverman (1984). It turns out that the cubic smoothing spline estimator is asymptotically equivalent to a kernel regression estimator, with an unusual choice of kernel. Recall that both are linear smoothers; this equivalence is achieved by showing that under some conditions the smoothing spline weights converge to kernel weights, under the "Silverman kernel":

$$K(x) = \frac{1}{2} \exp(-|x|/\sqrt{2}) \sin(|x|/\sqrt{2} + \pi/4), \tag{18}$$

  and a local choice of bandwidth $h(x) = \lambda^{1/4} q(x)^{-1/4}$, where $q(x)$ is the density of the input points. That is, the bandwidth adapts to the local distribution of inputs. See Figure 8 for a plot of the Silverman kernel

- The Silverman kernel is "kind of" a higher-order kernel. It satisfies

$$\int K(x)\,dx = 1, \quad \int x^j K(x)\,dx = 0, \quad j = 1, \ldots, 3, \quad \text{but} \quad \int x^4 K(x)\,dx = -24.$$

  So it lies outside the scope of usual kernel analysis

- There is more recent work that connects smoothing splines of all orders to kernel smoothing. See, e.g., Eggermont & LaRiccia (2006), Wang et al. (2013).

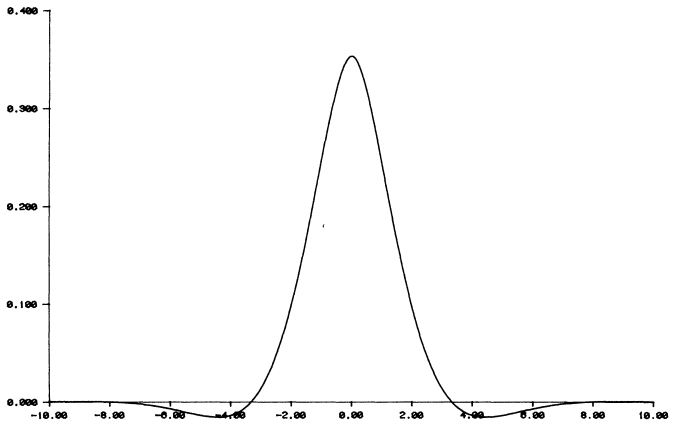

Figure 8: *The Silverman kernel in* (18)*, which is the (asymptotically) equivalent implicit kernel used by smoothing splines. Note that it can be negative. From Silverman (1984)*

3.8 Error rates

- Define the *Sobolev class* of functions $W_1(m, C)$, for an integer $m \geq 0$ and $C > 0$, to contain all m times differentiable functions $f : \mathbb{R} \to \mathbb{R}$ such that

$$\int \left(f^{(m)}(x)\right)^2 dx \leq C^2.$$

  (The Sobolev class $W_d(m, C)$ in d dimensions can be defined similarly, where we sum over all partial derivatives of order m.)

- Assuming $f_0 \in W_1(m, C)$ for the underlying regression function, where $C > 0$ is a constant, the smoothing spline estimator $\hat{f}$ in (9) of polynomial order $k = 2m-1$ with tuning parameter $\lambda \asymp n^{1/(2m+1)} \asymp n^{1/(k+2)}$ satisfies

$$\|\hat{f} - f_0\|_n^2 \lesssim n^{-2m/(2m+1)} \quad \text{in probability.}$$

  The proof of this result uses much more fancy techniques from empirical process theory (entropy numbers) than the proofs for kernel smoothing. See Chapter 10.1 of van de Geer (2000)

- This rate is seen to be minimax optimal over $W_1(m, C)$ (e.g., Nussbaum (1985)). Also, it is worth noting that the Sobolev $W_1(m, C)$ and Holder $H_1(m, L)$ classes are *equivalent* in the following sense: given $W_1(m, C)$ for a constant $C > 0$, there are $L_0, L_1 > 0$ such that

$$H_1(m, L_0) \subseteq W_1(m, C) \subseteq H_1(m, L_1).$$

  The first containment is easy to show; the second is far more subtle, and is a consequence of the Sobolev embedding theorem. (The same equivalences hold for the d-dimensional versions of the Sobolev and Holder spaces.)

3.9 Multivariate splines

- 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

$$\sum_{\alpha_1+\ldots+\alpha_d=m} \int \left| \frac{\partial^m f(x)}{\partial x_1^{\alpha_1} x_2^{\alpha_2} \ldots \partial x_d^{\alpha_d}} \right|^2 dx,$$

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))

- 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

4 Mercer kernels, RKHS

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

$$\hat{f} = \underset{f \in \mathcal{H}}{\operatorname{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda J(f), \tag{19}$$

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 $K : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$. 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:

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

and the Gaussian radial basis kernel:

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

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 eigenfunctions $\phi_i(x)$, $i = 1, 2, \ldots$ 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 $K(\cdot, z)$, $z \in \mathbb{R}^d$, i.e., elements in $\mathcal{H}_K$ are of the form

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

for a (possibly infinite) set M

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

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

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 $\|\cdot\|_{\mathcal{H}_K}$ on $\mathcal{H}_K$: we define

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

- The natural choice now is to take the penalty functional in (19) as this squared RKHS norm, $J(f) = \|f\|_{\mathcal{H}_K}^2$. This yields the RKHS problem

$$\hat{f} = \underset{f \in \mathcal{H}_K}{\operatorname{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \|f\|_{\mathcal{H}_K}^2. \tag{20}$$

A remarkable achievement of RKHS theory is that the infinite-dimensional problem (20) 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 (20) is uniquely attained by a function of the form

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

or in other words, a function f lying in the span of the functions $K(\cdot, x_i)$, $i = 1, \ldots, n$. Furthermore, we can rewrite the problem (20) in finite-dimensional form, as

$$\hat{\alpha} = \underset{\alpha \in \mathbb{R}^n}{\operatorname{argmin}} \ \|y - K\alpha\|_2^2 + \lambda \alpha^T K \alpha, \tag{21}$$

where $K \in \mathbb{R}^{n \times n}$ 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 $\hat{\alpha}$ in (21), the estimated function $\hat{f}$ in (20) is given by

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

- The solution in (21) is

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

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

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

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 (21): 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,

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

encodes a notion of distance between x_i, 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 (21) 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

5 Linear smoothers

5.1 Degrees of freedom and unbiased risk estimation

- Literally every estimator we have discussed so far, trained on $(x_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, n$, produces fitted values $\hat{\mu} = (\hat{f}(x_1), \ldots, \hat{f}(x_n))$ of the form

$$\hat{\mu} = Sy$$

for some matrix $S \in \mathbb{R}^{n \times n}$ depending on the inputs $x_1, \ldots, x_n$—and also possibly on a tuning parameter such as h in kernel smoothing, or λ in smoothing splines—but not on y. Recall that such estimators are called *linear smoothers*

- Consider the inputs as fixed (nonrandom), and assume y has i.i.d. components with mean 0 and variance σ^2. In this setting, we can define the degrees of freedom of an estimator $\hat{\mu}$

$$\mathrm{df}(\hat{\mu}) = \frac{1}{\sigma^2} \sum_{i=1}^{n} \mathrm{Cov}(\hat{\mu}_i, y_i).$$

In particular, recall that for linear smoothers $\hat{\mu} = Sy$, the degrees of freedom is

$$\mathrm{df}(\hat{\mu}) = \sum_{i=1}^{n} S_{ii} = \mathrm{tr}(S),$$

the trace of the smooth matrix S

- Example: for a regression spline estimator, of polynomial order k, with knots at the locations $t_1, \ldots, t_p$, recall that $\hat{\mu} = G(G^T G^{-1}) G^T y$ for $G \in \mathbb{R}^{n \times (p+k+1)}$ the order k spline basis matrix over the knots $t_1, \ldots, t_p$. Therefore

$$\mathrm{df}(\hat{\mu}) = \mathrm{tr}\big(G(G^T G)^{-1} G^T\big) = \mathrm{tr}\big(G^T G(G^T G)^{-1}\big) = p + k + 1,$$

the degrees of freedom of a regression spline is the number of knots + polynomial order + 1. The same calculation shows that the degrees of freedom of a regression natural spline is simply the number of knots (independent of the polynomial order)

- Example: for a smoothing spline estimator, recall that we were able to write the fitted values as $\hat{\mu} = (I + \lambda K)^{-1}$, i.e., as

$$\hat{\mu} = U(1 + \lambda D)^{-1} U^T y,$$

where UDU^T is the eigendecomposition of the Reinsch matrix $K = (N^T)^{-1} \Omega N^{-1}$ (and here K depends only on the input points $x_1, \ldots, x_n$ and the polynomial order k). The smoothing spline hence has degrees of freedom

$$\mathrm{df}(\hat{\mu}) = \mathrm{tr}\big(U(1 + \lambda D)^{-1} U^T\big) = \sum_{i=1}^{n} \frac{1}{1 + \lambda d_j},$$

where $D = \mathrm{diag}(d_1, \ldots, d_n)$. This is monotone decreasing in λ, with $\mathrm{df}(\hat{\mu}) = n$ when $\lambda = 0$, and $\mathrm{df}(\hat{\mu}) \to (k+1)/2$ when $\lambda \to \infty$, the number of zero eigenvalues among $d_1, \ldots, d_n$

- Degrees of freedom is generally a useful concept since it allows us to put two different estimators on equal footing. E.g., suppose we wanted to compare kernel smoothing versus smoothing splines; we could tune them to match their degrees of freedom, and then compare their performances

- A second more concrete motivation for considering degrees of freedom: it allows us to form an unbiased estimate of the error, or risk. Let $\mu = (f_0(x_1), \ldots, f_0(x_n)) \in \mathbb{R}^n$ be the vector given by evaluating the underlying regression function at the inputs, i.e., $\mu = \mathbb{E}(y)$. Then

$$\widehat{\mathrm{Err}} = \frac{1}{n} \|y - \hat{\mu}\|_2^2 - \sigma^2 + \frac{2\sigma^2}{n} \mathrm{df}(\hat{\mu})$$

serves as an unbiased estimate of the error $\mathrm{Err} = \mathbb{E}\|\mu - \hat{\mu}\|_2^2 / n$. This is simply

$$\widehat{\mathrm{Err}} = \frac{1}{n} \|y - Sy\|_2^2 - \sigma^2 + \frac{2\sigma^2}{n} \mathrm{tr}(S) \tag{22}$$

- Suppose our linear smoother of interest depends on a tuning parameter α (e.g., h for kernel smoothing, λ for smoothing splines, or λ for Mercer kernels), and express this as $\hat{\mu}_\alpha = S_\alpha y$. Then we could choose the tuning parameter α to minimize the estimated test error, as in

$$\hat{\alpha} = \underset{\alpha}{\mathrm{argmin}} \ \frac{1}{n} \|y - S_\alpha y\|_2^2 + \frac{2\sigma^2}{n} \mathrm{tr}(S_\alpha).$$

This is just like the C_p criterion, or AIC, in ordinary linear regression (we could also replace the factor of 2 above with $\log n$ to obtain something like BIC)

5.2 Leave-one-out and generalized cross-validation

- Of course, cross-validation gives us another way to perform error estimation and model selection. For linear smoothers $\hat{\mu} = (\hat{f}(x_1), \ldots \hat{f}(x_n)) = Sy$, leave-one-out cross-validation can be particularly appealing because in many cases we have the seemingly magical reduction

$$\mathrm{CV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \big(y_i - \hat{f}^{-i}(x_i)\big)^2 = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{f}(x_i)}{1 - S_{ii}} \right)^2, \tag{23}$$

where $\hat{f}^{-i}$ denotes the estimated regression function that was trained on all but the ith pair (x_i, y_i). This leads to a big computational savings since it shows us that, to compute leave-one-out cross-validation error, we don't have to actually ever compute $\hat{f}^{-i}$, $i = 1, \ldots, n$

- Why does (23) hold, and for which linear smoothers $\hat{\mu} = Sy$? Just rearranging (23) perhaps demystifies this seemingly magical relationship and helps to answer these questions. Suppose we knew that $\hat{f}$ had the property

$$\hat{f}^{-i}(x_i) = \frac{1}{1 - S_{ii}} \big(\hat{f}(x_i) - S_{ii} y_i \big). \tag{24}$$

  That is, to obtain the estimate at x_i under the function $\hat{f}^{-i}$ fit on all but (x_i, y_i), we take the sum of the linear weights (from our original fitted function $\hat{f}$) across all but the ith point, $\hat{f}(x_i) - S_{ii} y_i = \sum_{i \neq j} S_{ij} y_j$, and then renormalize so that these weights sum to 1

- This is not an unreasonable property; e.g., we can immediately convince ourselves that it holds for kernel smoothing. A little calculation shows that it also holds for smoothing splines (using the Sherman-Morrison update formula). How about for k-nearest-neighbors?

- From the special property (24), it is easy to show the leave-one-out formula (23). We have

$$y_i - \hat{f}^{-i}(x_i) = y_i - \frac{1}{1 - S_{ii}} \big(\hat{f}(x_i) - S_{ii} y_i \big) = \frac{y_i - \hat{f}(x_i)}{1 - S_{ii}},$$

  and then squaring both sides and summing over n gives (23)

- Finally, *generalized cross-validation* is a small twist on the right-hand side in (23) that gives an approximation to leave-one-out cross-validation error. It is defined as by replacing the appearences of diagonal terms S_{ii} with the average diagonal term $\mathrm{tr}(S)/n$,

$$\mathrm{GCV}(\hat{f}) = \frac{1}{n} \sum_{i=1}^{n} \left(\frac{y_i - \hat{f}(x_i)}{1 - \mathrm{tr}(S)/n} \right)^2.$$

  This can be of computational advantage in some cases where $\mathrm{tr}(S)$ is easier to compute that individual elements S_{ii}, and is also closely tied to the unbiased test error estimate in (22), seen by making the approximation $1/(1-x)^2 \approx 1 + 2x$

6 Locally adaptive estimators

6.1 Wavelet smoothing

- Not every nonparametric regression estimate needs to be a linear smoother (though this does seem to be very common), and *wavelet smoothing* is one of the leading nonlinear tools for nonparametric estimation. The theory of wavelets is elegant and we only give a brief introduction here; see Mallat (2008) for an excellent reference

- You can think of wavelets as defining an orthonormal function basis, with the basis functions exhibiting a highly varied level of smoothness. Importantly, these basis functions also display spatially localized smoothness at different locations in the input domain. There are actually many different choices for wavelets bases (Haar wavelets, symmlets, etc.), but these are details that we will not go into

- We assume $d = 1$. Local adaptivity in higher dimensions is not nearly as settled as it is with smooting splines or (especially) kernels (multivariate extensions of wavelets are possible, i.e., *ridgelets* and *curvelets*, but are complex)

- Consider basis functions, $\phi_1, \ldots, \phi_n$, evaluated over n equally spaced inputs over $[0, 1]$:

$$x_i = i/n, \quad i = 1, \ldots, n.$$

  The assumption of evenly spaced inputs is crucial for fast computations; we also typically assume with wavelets that n is a power of 2. We now form a wavelet basis matrix $W \in \mathbb{R}^{n \times n}$, defined by

$$W_{ij} = \phi_j(x_i), \quad i, j = 1, \ldots, n$$

- The goal, given outputs $y = (y_1, \ldots, y_n)$ over the evenly spaced input points, is to represent y as a sparse combination of the wavelet basis functions. To do so, we first perform a wavelet transform (multiply by W^T):

$$\tilde{\theta} = W^T y,$$

  we threshold the coefficients θ (the threshold function T_λ to be defined shortly):

$$\hat{\theta} = T_\lambda(\tilde{\theta}),$$

  and then perform an inverse wavelet transform (multiply by W):

$$\hat{\mu} = W\hat{\theta}$$

- The wavelet and inverse wavelet transforms (multiplication by W^T and W) each require $O(n)$ operations, and are practically extremely fast due do clever pyramidal multiplication schemes that exploit the special structure of wavelets

- The threshold function T_λ is usually taken to be hard-thresholding, i.e.,

$$[T_\lambda^{\text{hard}}(z)]_i = z_i \cdot 1\{|z_i| \geq \lambda\}, \quad i = 1, \ldots, n,$$

  or soft-thresholding, i.e.,

$$[T_\lambda^{\text{soft}}(z)]_i = \big(z_i - \text{sign}(z_i)\lambda\big) \cdot 1\{|z_i| \geq \lambda\}, \quad i = 1, \ldots, n.$$

  These thresholding functions are both also $O(n)$, and computationally trivial, making wavelet smoothing very fast overall

- We should emphasize that wavelet smoothing is not a linear smoother, i.e., there is no single matrix S such that $\hat{\mu} = Sy$ for all y

- We can write the wavelet smoothing estimate in a more familiar form, following our previous discussions on basis functions and regularization. For hard-thresholding, we solve

$$\hat{\theta} = \underset{\theta \in \mathbb{R}^n}{\text{argmin}} \ \|y - W\theta\|_2^2 + \lambda^2 \|\theta\|_0,$$

  and then the wavelet smoothing fitted values are $\hat{\mu} = W\hat{\theta}$. Here $\|\theta\|_0 = \sum_{i=1}^n 1\{\theta_i \neq 0\}$, the number of nonzero components of θ, called the "ℓ_0 norm". For soft-thresholding, we solve

$$\hat{\theta} = \underset{\theta \in \mathbb{R}^n}{\text{argmin}} \ \|y - W\theta\|_2^2 + 2\lambda \|\theta\|_1,$$

  and then the wavelet smoothing fitted values are $\hat{\mu} = W\hat{\theta}$. Here $\|\theta\|_1 = \sum_{i=1}^n |\theta_i|$, the ℓ_1 norm

6.2 The strengths of wavelets, the limitations of linear smoothers

- Apart from its computational efficiency, an important strength of wavelet smoothing is that it can represent a signal that has a *spatially heterogeneous* degree of smoothness, i.e., it can be both smooth and wiggly at different regions of the input domain. The reason that wavelet smoothing can achieve such local adaptivity is because it selects a sparse number of wavelet basis functions, by thresholding the coefficients from a basis regression

- We can make this more precise by considering convergence rates over an appropriate function class. In particular, we define the *total variation class* $F(k, C)$, for an integer $k \geq 0$ and $C > 0$, to contain all k times (weakly) differentiable functions whose kth derivative satisfies

$$\mathrm{TV}(f^{(k)}) = \sup_{0=z_1<z_2<\ldots<z_N<z_{N+1}=1} \sum_{j=1}^{N} |f^{(k)}(z_{i+1}) - f^{(k)}(z_i)| \;\leq\; C.$$

  (Note that if f has $k+1$ continuous derivatives, then $\mathrm{TV}(f^{(k)}) = \int_0^1 |f^{(k+1)}(x)|\, dx$.)

- For the wavelet smoothing estimator, denoted by $\hat{f}^{\mathrm{wav}}$, Donoho & Johnstone (1998) provide a seminal analysis. Assuming that $f_0 \in F(k, C)$ for a constant $C > 0$ (and further conditions on the setup), they show that (for an appropriate scaling of the smoothing parameter λ),

$$\mathbb{E}\|\hat{f}^{\mathrm{wav}} - f_0\|_2^2 \lesssim n^{-(2k+2)/(2k+3)} \quad \text{and} \quad \inf_{\hat{f}} \sup_{f_0 \in F(k,C)} \mathbb{E}\|\hat{f} - f_0\|_2^2 \gtrsim n^{-(2k+2)/(2k+3)}. \tag{25}$$

  Thus wavelet smoothing attains the minimax optimal rate over the function class $F(k, C)$. (For a translation of this result to the notation of the current setting, see Tibshirani (2014).)

- Some important questions: (i) just how big is the function class $F(k, C)$? And (ii) can a linear smoother also be minimax optimal over $F(k, C)$?

  It is not hard to check $F(k, C) \supseteq W_1(k+1, C')$, the (univariate) Sobolev space of order $k+1$, for some other constant $C' > 0$. We know from the previously mentioned theory on Sobolev spaces that the minimax rate over $W_1(k+1, C')$ is again $n^{-(2k+2)/(2k+3)}$. This suggests that these two function spaces might actually be somewhat close in size

  But in fact, the overall minimax rates here are sort of misleading, and we will see from the behavior of linear smoothers that the function classes are actually quite different. Donoho & Johnstone (1998) showed that the minimax error over $F(k, C)$, *restricted to linear smoothers*, satisfies

$$\inf_{\hat{f} \text{ linear}} \sup_{f_0 \in F(k,C)} \mathbb{E}\|\hat{f} - f_0\|_2^2 \gtrsim n^{-(2k+1)/(2k+2)}. \tag{26}$$

  (See again Tibshirani (2014) for a translation to the notation of the current setting.) Hence the answers to our questions are: (ii) linear smoothers cannot cope with the heterogeneity of functions in $F(k, C)$, and are are bounded away from optimality, which means (i) we can interpret $F(k, C)$ as being much larger than $W_1(k+1, C')$, because linear smoothers can be optimal over the latter class but not over the former. See Figure 9 for a diagram

- Let's back up to emphasize just how remarkable the results (25), (26) really are. Though it may seem like a subtle difference in exponents, there is actually a significant difference in the minimax rate and minimax linear rate: e.g., when $k = 0$, this is a difference of $n^{-1/2}$ (optimal) and $n^{-1/2}$ (optimal among linear smoothers) for estimating a function of bounded variation. Recall also just how broad the linear smoother class is: kernel smoothing, regression splines, smoothing splines, RKHS estimators ... none of these methods can achieve a better rate than $n^{-1/2}$ over functions of bounded variation

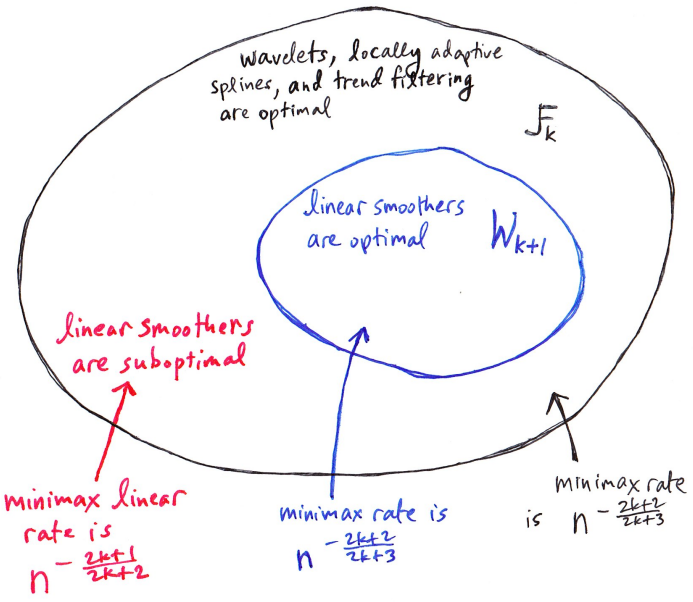

Figure 9: *A diagram of the minimax rates over $F(k, C)$ (denoted $\mathcal{F}_k$ in the picture) and $W_1(k+1, C)$ (denoted $\mathcal{W}_{k+1}$ in the picture)*

- Practically, the differences between wavelets and linear smoothers in problems with spatially heterogeneous smoothness can be striking as well. However, you should keep in mind that wavelets are not perfect: a shortcoming is that they require a highly restrictive setup: recall that they require evenly spaced inputs, and n to be power of 2, and there are often further assumptions made about the behavior of the fitted function at the boundaries of the input domain

- Also, though you might say they marked the beginning of the story, wavelets are not the end of the story when it comes to local adaptivity. The natural thing to do, it might seem, is to make (say) kernel smoothing or smoothing splines more locally adaptive by allowing for a local bandwidth parameter or a local penalty parameter. People have tried this, but it is both difficult theoretically and practically to get right. A cleaner approach is to redesign the kind of penalization used in constructing smoothing splines directly, which we discuss next

6.3 Locally adaptive regression splines

- *Locally adaptive regression splines* (Mammen & van de Geer 1997), as their name suggests, can be viewed as variant of smoothing splines that exhibit better local adaptivity. For a given integer order $k \geq 0$, the estimate is defined as

$$\hat{f} = \underset{f}{\operatorname{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \operatorname{TV}(f^{(k)}). \tag{27}$$

  The minimization domain is infinite-dimensional, the space of all functions for which the criterion is finite

- Another remarkable variational result, similar to that for smoothing splines, shows that (27) has a kth order spline as a solution (Mammen & van de Geer 1997). This *almost* turns the

minimization into a finite-dimensional one, but there is one catch: the knots of this kth-order spline are generally not known, i.e., they need not coincide with the inputs $x_1, \dots, x_n$. (When $k = 0, 1$, they do, but in general, they do not)

- To deal with this issue, we can redefine the locally adaptive regression spline estimator to be

$$\hat{f} = \underset{f \in \mathcal{G}_k}{\operatorname{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \operatorname{TV}(f^{(k)}), \tag{28}$$

  i.e., we restrict the domain of minimization to be $\mathcal{G}_k$, the space of kth-order spline functions with knots in T_k, where T_k is a subset of $\{x_1, \dots, x_n\}$ of size $n-k-1$. The precise definition of T_k is not important; it is just given by trimming away $k+1$ boundary points from the inputs

- As we already know, the space $\mathcal{G}_k$ of kth-order splines with knots in T_k has dimension $|T_k| + k + 1 = n$. Therefore we can choose a basis $g_1, \dots, g_n$ for the functions in $\mathcal{G}_k$, and the problem in (28) becomes one of finding the coefficients in this basis expansion,

$$\hat{\beta} = \underset{f \in \mathcal{G}_k}{\operatorname{argmin}} \sum_{i=1}^{n} \Big(y_i - \sum_{j=1}^{n} \beta_j g_j(x_i)\Big)^2 + \lambda \operatorname{TV}\Big\{\Big(\sum_{j=1}^{n} \beta_j g_j(x_i)\Big)^{(k)}\Big\}, \tag{29}$$

  and then we have $\hat{f}(x) = \sum_{j=1}^{n} \hat{\beta}_j g_j(x)$

- Now define the basis matrix $G \in \mathbb{R}^{n \times n}$ by

$$G_{ij} = g_j(x_i), \quad i = 1, \dots, n.$$

  Suppose we choose $g_1, \dots, g_n$ to be the truncated power basis. Denoting $T_k = \{t_1, \dots, t_{n-k-1}\}$, we compute

$$\Big(\sum_{j=1}^{n} \beta_j g_j(x_i)\Big)^{(k)} = k! + k! \sum_{j=k+2}^{n} \beta_j 1\{x \geq t_{j-k-1}\},$$

  and so

$$\operatorname{TV}\Big\{\Big(\sum_{j=1}^{n} \beta_j g_j(x_i)\Big)^{(k)}\Big\} = k! \sum_{j=k+2}^{n} |\beta_j|.$$

  Hence the locally adaptive regression spline problem (29) can be expressed as

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \ \|y - G\beta\|_2^2 + \lambda k! \sum_{i=k+2}^{n} |\beta_i|. \tag{30}$$

  This is a lasso regression problem on the truncated power basis matrix G, with the first $k+1$ coefficients (those corresponding to the pure polynomial functions, in the basis expansion) left unpenalized

- This reveals a key difference between the locally adaptive regression splines (30) (originally, problem (28)) and the smoothing splines (14) (originally, problem (9)). In the first problem, the total variation penalty is translated into an ℓ_1 penalty on the coefficients of the truncated power basis, and hence this acts a knot selector for the estimated function. That is, at the solution in (30), the estimated spline has knots at a subset of T_k (at a subset of the input points $x_1, \dots, x_n$), with fewer knots when λ is larger. In contrast, recall, at the smoothing spline solution in (14), the estimated function has knots at each of the inputs $x_1, \dots, x_n$. This is a major difference between the ℓ_1 and ℓ_2 penalties

- From a computational perspective, the locally adaptive regression spline problem in (30) is actually a lot harder than the smoothing spline problem in (14). Recall that the latter reduces to solving a single banded linear system, which takes $O(n)$ operations. On the other hand, fitting locally adaptive regression splines in (30) requires solving a lasso problem with a dense $n \times n$ regression matrix G; this takes something like $O(n^3)$ operations. So when $n = 10,000$, there is a big difference between the two!

- There is a tradeoff here, as with extra computation comes much improved local adaptivity of the fits. See Figure 10 for an example. Theoretically, when $f_0 \in F(k, C)$ for a constant $C > 0$, Mammen & van de Geer (1997) show the locally adaptive regression spline estimator, denoted $\hat{f}^{\text{lrs}}$, with $\lambda \asymp n^{1/(2k+3)}$, satisfies

$$\|\hat{f}^{\text{lrs}} - f_0\|_n^2 \lesssim n^{-(2k+2)/(2k+3)} \quad \text{in probability},$$

  so (like wavelets) it achieves the minimax optimal rate over $n^{-(2k+2)/(2k+3)}$. In this regard, as we discussed previously, they actually have a big advantage over any linear smoother (not just smoothing splines)

6.4 Trend filtering

- At a high level, you can think of trend filtering as computationally efficient version of locally adaptive regression splines, though their original construction (Steidl et al. 2006, Kim et al. 2009) comes from a fairly different perspective. We will begin by describing their connection to locally adaptive regression splines, following Tibshirani (2014)

- Revisit the formulation of locally adaptive regression splines in (28), where the minimization domain is $\mathcal{G}_k = \text{span}\{g_1, \ldots, g_n\}$, and $g_1, \ldots, g_n$ are the kth-order truncated power basis in (8) having knots in a set $T_k \subseteq \{x_1, \ldots x_n\}$ with size $|T_k| = n - k - 1$. The *trend filtering* problem is given by replacing $\mathcal{G}_k$ with a different function space,

$$\hat{f} = \underset{f \in \mathcal{H}_k}{\text{argmin}} \sum_{i=1}^{n} \big(y_i - f(x_i)\big)^2 + \lambda \, \text{TV}(f^{(k)}), \tag{31}$$

  where the new domain is $\mathcal{H}_k = \text{span}\{h_1, \ldots, h_n\}$. Assuming that the input points are ordered, $x_1 < \ldots < x_n$, the functions $h_1, \ldots, h_n$ are defined by

$$\begin{aligned} h_j(x) &= \prod_{\ell=1}^{j-1} (x - x_\ell), \quad j = 1, \ldots, k+1, \\ h_{k+1+j}(x) &= \prod_{\ell=1}^{k} (x - x_{j+\ell}) \cdot 1\{x \geq x_{j+k}\}, \quad j = 1, \ldots, n-k-1. \end{aligned} \tag{32}$$

  (Our convention is to take the empty product to be 1, so that $h_1(x) = 1$.) These are dubbed the *falling factorial basis*, and are piecewise polynomial functions, taking an analogous form to the truncated power basis functions in (8). Loosely speaking, they are given by replacing an rth-order power function in the truncated power basis with an appropriate r-term product, e.g., replacing x^2 with $(x-x_2)(x-x_1)$, and $(x-t_j)^k$ with $(x-x_{j+k})(x-x_{j+k-1}) \cdot \ldots, (x-x_{j+1})$

- Defining the falling factorial basis matrix

$$H_{ij} = h_j(x_i), \quad i, j = 1, \ldots, n,$$

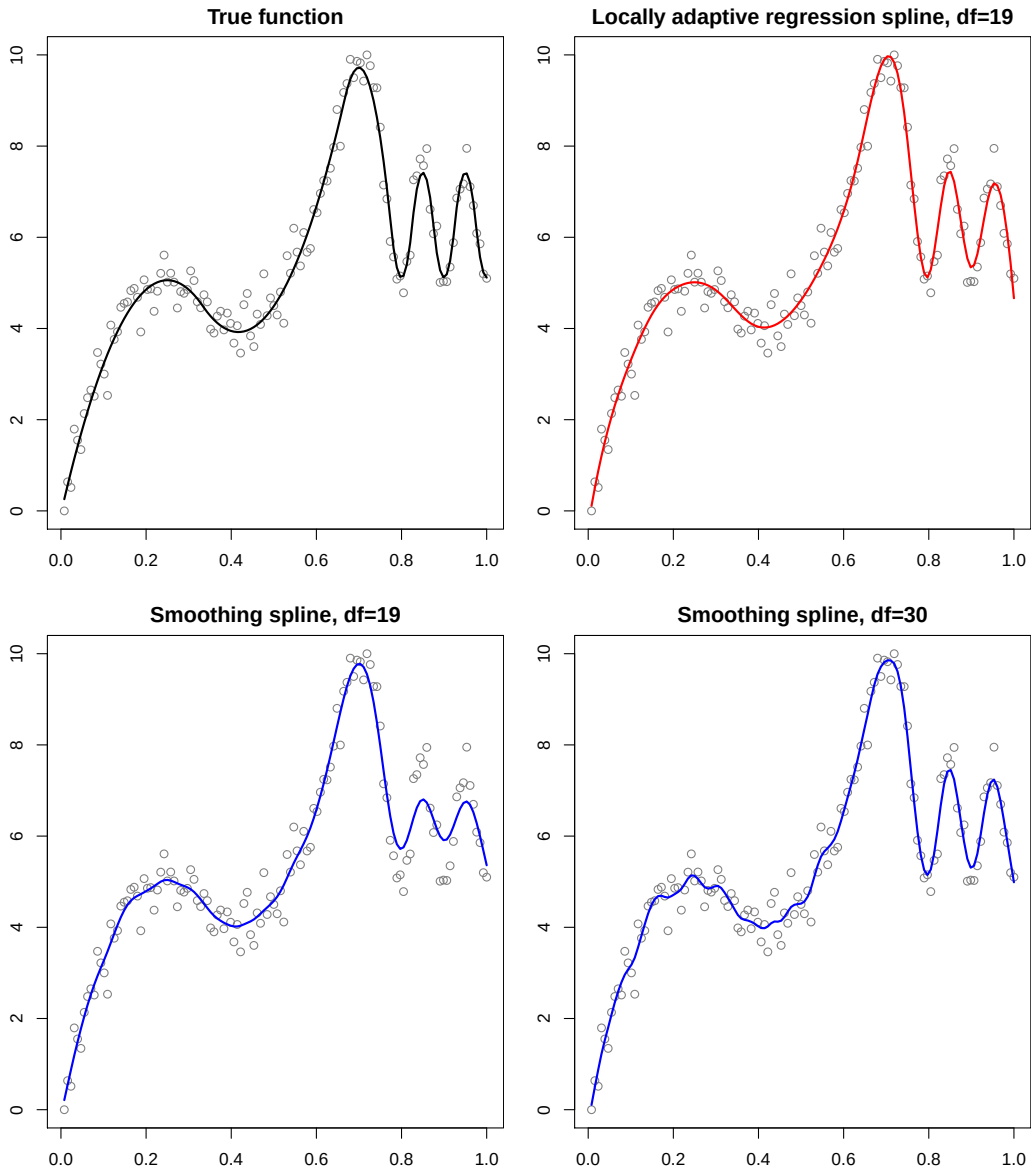

Figure 10: *The top left plot shows a simulated true regression function, which has inhomogeneous smoothness: smoother towards the left part of the domain, wigglier towards the right. The top right plot shows the locally adaptive regression spline estimate with 19 degrees of freedom; notice that it picks up the right level of smoothness throughout. The bottom left plot shows the smoothing spline estimate with the same degrees of freedom; it picks up the right level of smoothness on the left, but is undersmoothed on the right. The bottom right panel shows the smoothing spline estimate with 33 degrees of freedom; now it is appropriately wiggly on the right, but oversmoothed on the left. Smoothing splines cannot simultaneously represent different levels of smoothness at different regions in the domain; the same is true of any linear smoother*

it is now straightforward to check that the proposed problem of study, trend filtering in (31), is equivalent to

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \ \|y - H\beta\|_2^2 + \lambda k! \sum_{i=k+2}^{n} |\beta_i|. \tag{33}$$

This is still a lasso problem, but now in the falling factorial basis matrix H. Compared to the locally adaptive regression spline problem (30), there may not seem to be much of a difference here—like G, the matrix H is dense, and solving (33) would be slow. So why did we go to all the trouble of defining trend filtering, i.e., introducing the somewhat odd basis $h_1, \dots, h_n$ in (32)?

- The usefulness of trend filtering (33) is seen after reparametrizing the problem, by inverting H. Let $\theta = H\beta$, and rewrite the trend filtering problem as

$$\hat{\theta} = \underset{\theta \in \mathbb{R}^n}{\operatorname{argmin}} \ \|y - \theta\|_2^2 + \lambda \|D\theta\|_1, \tag{34}$$

where $D \in \mathbb{R}^{(n-k-1)\times n}$ denotes the last $n-k-1$ rows of $k! \cdot H^{-1}$. Explicit calculation shows that D is a banded matrix (Tibshirani 2014, Wang et al. 2014). For simplicity of exposition, consider the case when $x_i = i$, $i = 1, \dots, n$. Then, e.g., the first 3 orders of difference operators are:

$$D = \begin{bmatrix} -1 & 1 & 0 & \dots \\ 0 & -1 & 1 & \dots \\ \vdots & & & \end{bmatrix}, \quad \text{when } k = 0$$

$$D = \begin{bmatrix} 1 & -2 & 1 & 0 & \dots \\ 0 & 1 & -2 & 1 & \dots \\ 0 & 0 & 1 & -2 & \dots \\ \vdots & & & & \end{bmatrix}, \quad \text{when } k = 1$$

$$D = \begin{bmatrix} -1 & 3 & -3 & 1 & \dots \\ 0 & -1 & 3 & -3 & \dots \\ 0 & 0 & -1 & 3 & \dots \\ \vdots & & & & \end{bmatrix}, \quad \text{when } k = 2.$$

One can hence interpret D as a type of discrete derivative operator, of order $k+1$. This also suggests an intuitive interpretation of trend filtering (34) as a discrete approximation to the original locally adaptive regression spline problem in (27)

- The bandedness of D means that the trend filtering problem (34) can be solved efficiently, in close to linear time (complexity $O(n^{1.5})$ in the worst case). Thus trend filtering estimates are much easier to fit than locally adaptive regression splines

- But what of their statistical relevancy? Did switching over to the falling factorial basis (32) wreck the local adaptivity properties that we cared about in the first place? Fortunately, the answer is no, and in fact, trend filtering and locally adaptive regression spline estimates are extremely hard to distinguish in practice. See Figure 11

- Moreover, Tibshirani (2014), Wang et al. (2014) prove that the estimates from trend filtering and locally adaptive regression spline estimates, denoted $\hat{f}^{\text{tf}}$ and $\hat{f}^{\text{lrs}}$, respectively, when the tuning parameter λ for each scales as $n^{1/(2k+3)}$, satisfy

$$\|\hat{f}^{\text{tv}} - \hat{f}^{\text{lrs}}\|_n^2 \lesssim n^{-(2k+2)/(2k+3)} \quad \text{in probability.}$$

This coupling shows that trend filtering converges to the underlying function f_0 at the rate $n^{-(2k+2)/(2k+3)}$ whenever locally adaptive regression splines do, making them also minimax optimal over $F(k, C)$. In short, trend filtering offers provably significant improvements over linear smoothers, with a computational cost that is not too much steeper than a single banded linear system solve

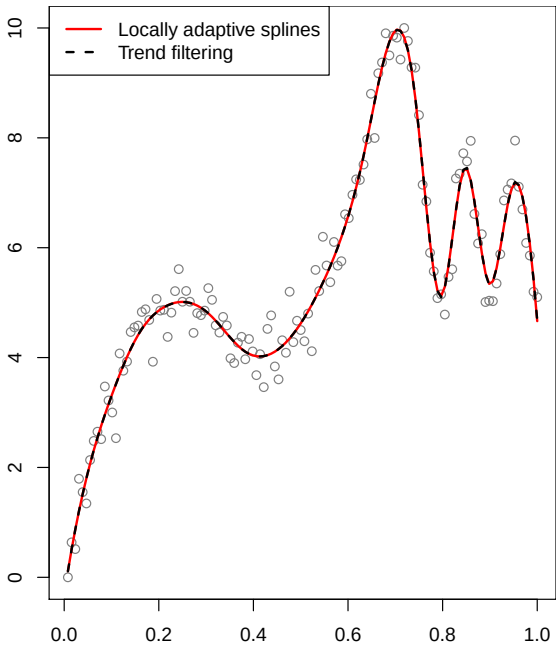

Figure 11: *Trend filtering and locally adaptive regression spline estimates, fit on the same data set as in Figure 10. The two are tuned at the same level, and the estimates are visually indistinguishable*

7 Additive models

7.1 Motivation and definition

- Computational efficiency and statisical efficiency are both very real concerns as the dimension d grows large, in nonparametric regression. If you're trying to fit a kernel, thin-plate spline, or RKHS estimate in > 20 dimensions, without any other kind of structural constraints, then you'll probably be in trouble (unless you have a very fast computer and tons of data)

- Recall from (7) that the minimax rate over the Holder class $H_d(\alpha, L)$ is $n^{-2\alpha/(2\alpha+d)}$, which has an exponentially bad dependence on the dimension d. This is usually called the curse of dimensionality (though the term apparently originated with Bellman (1962), who encountered an analogous issue but in a separate context—dynamic programming)

- What can we do? One answer is to change what we're looking for, and fit estimates with less flexibility in high dimensions. Think of a linear model in d variables: there is a big difference between this and a fully nonparametric model in d variables. Is there some middle man that we can consider, that would make sense?

- *Additive models* play the role of this middle man. Instead of considering a full d-dimensional function of the form

$$f(x) = f(x_{\cdot 1}, \ldots, x_{\cdot d}), \tag{35}$$

we restrict our attention to functions of the form

$$f(x) = f_1(x_{\cdot 1}) + \ldots + f_d(x_{\cdot d}). \tag{36}$$

(Here the notation $x_{\cdot j}$ denotes the jth component of $x \in \mathbb{R}^d$, slightly cumbersome notation used so as not to confuse with the labeling of the d-dimensional inputs $x_1, \ldots, x_n$). As each

function f_j, $j = 1, \ldots, d$ is univariate, fitting an estimate of the form (36) is certainly less ambitious than fitting one of the form (35). On the other hand, the scope of (36) is still big enough that we can capture interesting (marginal) behavior in high dimensions

- The choice of modeler (36) need not be regarded as an assumption we make about the true function f_0, just like we don't always assume that the true model is linear when using linear regression. In many cases, we fit an additive model because we think it may provide a useful approximation to the truth, and is able to scale well with the number of dimensions d

- A classic result by Stone (1985) encapsulates this idea precisely. He shows that, while it may be difficult to estimate an arbitrary regression function f_0 in multiple dimensions, we can still estimate its *best additive approximation* $\bar{f}^{\text{add}}$ well. Assuming each component function $\bar{f}^{\text{add}}_{0,j}$, $j = 1, \ldots, d$ lies in the Holder class $H_1(\alpha, L)$, for constant $L > 0$, and we can use an additive model, with each component $\hat{f}_j$, $j = 1, \ldots, d$ estimated using an appropriate kth degree spline, to give

$$\mathbb{E}\|\hat{f}_j - \bar{f}^{\text{add}}_j\|_2^2 \lesssim n^{-2\alpha/(2\alpha+1)}, \quad j = 1, \ldots, d.$$

  Hence each component of the best additive approximation $\bar{f}^{\text{add}}$ to f_0 can be estimated at the optimal univariate rate. Loosely speaking, though we cannot hope to recover f_0 arbitrarily, we can recover its major structure along the coordinate axes

7.2 Backfitting

- Estimation with additive models is actually very simple; we can just choose our favorite univariate smoother (i.e., nonparametric estimator), and cycle through estimating each function f_j, $j = 1, \ldots, d$ individually (like a block coordinate descent algorithm). Denote the result of running our chosen univariate smoother to regress $y = (y_1, \ldots, y_n) \in \mathbb{R}^n$ over the input points $z = (z_1, \ldots, z_n) \in \mathbb{R}^n$ as

$$\hat{f} = \text{Smooth}(z, y).$$

  E.g., we might choose $\text{Smooth}(\cdot, \cdot)$ to be a cubic smoothing spline with some fixed value of the tuning parameter λ, or even with the tuning parameter selected by generalized cross-validation

- Once our univariate smoother has been chosen, we initialize $\hat{f}_1, \ldots, \hat{f}_d$ (say, to all to zero) and cycle over the following steps for $j = 1, \ldots, d, 1, \ldots, d, \ldots$:

  1. define $r_i = y_i - \sum_{\ell \neq j} \hat{f}_\ell(x_{i\ell})$, $i = 1, \ldots, n$;
  2. smooth $\hat{f}_j = \text{Smooth}(x_{\cdot j}, r)$;
  3. center $\hat{f}_j = \hat{f}_j - \frac{1}{n}\sum_{i=1}^n \hat{f}_j(x_{ij})$.

  This algorithm is known as *backfitting*. In last step above, we are removing the mean from each fitted function $\hat{f}_j$, $j = 1, \ldots, d$, otherwise the model would not be identifiable. Our final estimate therefore takes the form

$$\hat{f}(x) = \bar{y} + \hat{f}_1(x_{\cdot 1}) + \ldots + \hat{f}_d(x_{\cdot d}),$$

  where $\bar{y} = \frac{1}{n}\sum_{i=1}^n y_i$. Hastie & Tibshirani (1990) provide a very nice exposition on the some of the more practical aspects of backfitting and additive models

- In many cases, backfitting is equivalent to blockwise coordinate descent performed on a joint optimization criterion that determines the total additive estimate. E.g., for the additive cubic smoothing spline optimization problem,

$$\hat{f}_1, \ldots, \hat{f}_d = \underset{f_1, \ldots, f_d}{\text{argmin}} \sum_{i=1}^n \Big(y_i - \sum_{j=1}^d f_j(x_{ij}) \Big)^2 + \sum_{j=1}^d \lambda_j \int_0^1 f_j''(t)^2 \, dt,$$

backfitting is exactly blockwise coordinate descent (after we reparametrize the above to be in finite-dimensional form, using a natural cubic spline basis)

- The beauty of backfitting is that it allows us to think *algorithmically*, and plug in whatever we want for the univariate smoothers. This allows for several extensions. One extension: we don't need to use the same univariate smoother for each dimension, rather, we could mix and match, choosing $\text{Smooth}_j(\cdot,\cdot)$, $j = 1,\ldots,d$ to come from entirely different methods or giving estimates with entirely different structures

- Another extension: to capture correlations, we can perform smoothing over (small) groups of variables instead of individual variables; e.g., if we thought that variables $1,2$ might have reasonable correlation, then we could lump the backfitting steps over variables $1,2$ together and perform (say) a 2-dimensional kernel smooth, giving an estimate of the form

$$\hat{f}(x) = \bar{y} + \hat{f}_{12}(x_{\cdot 1}, x_{\cdot 2}) + \hat{f}_3(x_{\cdot,3}) + \ldots + \hat{f}_d(x_{\cdot d}),$$

7.3 Error rates

- Error rates for additive models are both kind of what you'd expect and suprising. What you'd expect: if the underlying function f_0 is additive, and we place standard assumptions on its component functions, such as $f_{0,j} \in W_1(m, C)$, $j = 1,\ldots,d$, for a constant $C > 0$, a somewhat straightforward argument building on univariate minimax theory gives us the lower bound

$$\inf_{\hat{f}} \sup_{f_0 \in \oplus_{j=1}^d W_1(m,C)} \mathbb{E}\|\hat{f} - f_0\|_2^2 \gtrsim d n^{-2m/(2m+1)}.$$

  This is simply d times the univariate minimax rate. (Note that we have been careful to track the role of d here, i.e., it is not being treated like a constant.) Also, standard methods like backfitting with univariate smoothing splines of polynomial order $k = 2m-1$, will also match this upper bound in error rate (though the proof to get the sharp linear dependence on d is a bit trickier)

- Surprising: an additive model with different levels of smoothness among its component functions behaves in an interesting manner. Just in $d = 2$ dimensions, let us consider $f_0(x) = f_{0,1}(x_{\cdot 1}) + f_{0,2}(x_{\cdot 2})$, where $f_{0,1}$ is a lot smoother than $f_{0,2}$, e.g., $f_{0,1} \in W_1(2, C_1)$ and $f_{0,2} \in F(0, C_2)$, so

$$\int_0^1 f_{0,1}''(t)^2\, dt \leq C_1 \quad \text{and} \quad \text{TV}(f_{0,2}) \leq C_2,$$

  for constants $C_1, C_2 > 0$. Suppose also that we used an additive model to estimate f_0, with (say) a 3rd-order smoothing spline for the first component smoother, and a 0th-order locally adaptive regression spline for the second component smoother. Now, assuming each smoother was appropriately tuned, should we expect that

$$\|\hat{f}_1 - f_{0,1}\|_n^2 \lesssim n^{-4/5} \quad \text{and} \quad \|\hat{f}_2 - f_{0,2}\|_n^2 \lesssim n^{-2/3}, \tag{37}$$

  each having the error rate associated with their corresponding univariate problem, or

$$\|\hat{f}_1 - f_{0,1}\|_n^2 \lesssim n^{-2/3} \quad \text{and} \quad \|\hat{f}_2 - f_{0,2}\|_n^2 \lesssim n^{-2/3}, \tag{38}$$

  where the rougher of the two components dictates both rates? Recent work by van de Geer & Muro (2015) shows that (provided $x_{\cdot 1}, x_{\cdot 2}$ are not too correlated) it is the first case (37) that occurs

- This is somewhat surprising, because if you think about it from the perspective of backfitting, at convergence, we have

$$\hat{f}_1 = \text{Smooth}_1\Big(x_{\cdot 2}, \big(y_i - \hat{f}_2(x_{i2})\big)_{i=1}^n\Big),$$

a cubic smoothing spline fit to the effective responses $r_i = y_i - \hat{f}_2(x_{i2})$, $i = 1, \ldots, n$. If we were actually fitting a smoothing spline to $f_{0,1}(x_{i1}) + \epsilon_i$, $i = 1, \ldots, n$, then we'd see a $n^{-4/5}$ error rate. But we're not; instead we're fitting a cubic smoothing spline to

$$y_i - \hat{f}_2(x_{i2}) = f_{0,1}(x_{i1}) + \epsilon_i + \underbrace{\big(f_{0,2}(x_{i2}) - \hat{f}_2(x_{i2})\big)}_{e_{2i}}, \quad i = 1, \ldots, n.$$

The terms denoted e_{2i}, $i = 1, \ldots, n$ above are the errors in estimating the second component function at the input points. In the best case, we should hope for $\|e_2\|_2^2/n \asymp n^{-2/3}$. Doesn't this $n^{-2/3}$ pertubation mess up our estimation of $f_{0,1}$? Surprisingly, it does not.

It is worth noting that the proof given by van de Geer & Muro (2015) is very intricate, and does not obviously extend beyond $d = 2$ components. (Also, it is worth noting that this result relates to older, classic results from semiparametric estimation.)

7.4 Sparse additive models

- Recently, *sparse additive models* have received a good deal of attention. In truly high dimensions, we might believe that only a small subset of the variables play a useful role in modeling the regression function, so might posit a modification of (36) of the form

$$f(x) = \sum_{j \in S} f_j(x_{\cdot j}),$$

where $S \subseteq \{1, \ldots, d\}$ is an unknown subset of the full set of dimensions

- This is a natural idea, and to estimate a sparse additive model, we can use methods that are like nonparametric analogies of the lasso (more accurately, the group lasso). This is a research topic still very much in development; some recent works are Lin & Zhang (2006), Ravikumar et al. (2009), Raskutti et al. (2012). We'll cover this in more detail when we talk about the sparsity, the lasso, and high-dimensional estimation

8 Nonparametric classification

8.1 From regression to classification

- What about nonparametric classification? In all fairness, this topic deserves a more thorough treatment on its own, but here we show that many of our ideas from nonparametric regression carry over nicely to classification. An excellent general, theoretical reference is Devroye et al. (1996) (similar to to the book by Gyorfi et al. (2002) for nonparametric regression)

- For $(X, Y) \in \mathbb{R}^d \times \{0, 1\}$, consider the regression function, defined as usual as

$$f_0(x) = \mathbb{E}(Y|X = x) = \mathbb{P}(Y = 1|X = x),$$

which now becomes the conditional probability of observing class 1, given $X = x$. Given i.i.d. samples $(x_i, y_i) \in \mathbb{R}^d \times \{0, 1\}$, $i = 1, \ldots, n$, that have the same joint distribution as (X, Y), we

can use any nonparametric regression method to form an estimate $\hat{f}$ of f_0, and then define a *plug-in classifier* via

$$\hat{C}(x) = \begin{cases} 0 & \text{if } \hat{f}(x) > 1/2 \\ 1 & \text{if } \hat{f}(x) \leq 1/2 \end{cases}. \tag{39}$$

This of course estimates the optimal classification rule, called the *Bayes classifier*,

$$C_0(x) = \begin{cases} 0 & \text{if } f_0(x) > 1/2 \\ 1 & \text{if } f_0(x) \leq 1/2 \end{cases} \tag{40}$$

- Consider now the risk, with respect to the 0-1 loss, of a generic classifier C, defined as

$$r(C) = \mathbb{P}\big(Y \neq C(X)\big).$$

It is not hard to show that the Bayes classifier defined in (40) it optimal precisely in the sense that it achieves the smallest risk, i.e., $r(C_0)$ is minimal among all classifiers C. We consider, for any classifier C and any fixed x,

$$\begin{aligned} \mathbb{P}\big(Y \neq C(X) | X = x\big) &= 1 - \big[\mathbb{P}\big(Y = 1, C(X) = 1 | X = x\big) + \mathbb{P}\big(Y = 0, C(X) = 0 | X = x\big) \\ &= 1 - \big[C(x) f_0(x) + \big(1 - C(x)\big)\big(1 - f_0(x)\big)\big] \\ &= f_0(x) + \big(1 - 2 f_0(x)\big) C(x). \end{aligned}$$

Thus, we can compute the conditional risk difference between C and C_0 as

$$\mathbb{P}\big(Y \neq C(X) | X = x\big) - \mathbb{P}\big(Y \neq C_0(X) | X = x\big) = \big(2 f_0(x) - 1\big)\big(C_0(x) - C(x)\big).$$

When $f_0(x) > 1/2$, we have $C_0(x) = 1$ by construction, and so the right-hand side above is nonnegative. When $f_0(x) \leq 1/2$, we have $C_0(x) = 0$ by construction, and again the above is nonnegative. Therefore we have shown $\mathbb{P}(Y \neq C(X) | X = x) - \mathbb{P}((Y \neq C_0(X) | X = x) \geq 0$ for every x; integrating out with respect to the distribution of X gives the result

- Moreover, from the above calculation, we can learn a very important fact about any plug-in classifier, as defined in (39). We have, as before

$$\begin{aligned} \mathbb{P}\big(Y \neq \hat{C}(X) | X = x\big) - \mathbb{P}\big(Y \neq C_0(X) | X = x\big) &= \big(2 f_0(x) - 1\big)\big(C_0(x) - \hat{C}(x)\big) \\ &= 2 |f_0(x) - 1/2| 1\{C_0(x) \neq \hat{C}(x)\}. \end{aligned}$$

When $C_0(x)$ and $\hat{C}(x)$ do not match, note that $|f_0(x) - 1/2| \leq |f_0(x) - \hat{f}(x)|$ (because $f_0(x)$ and $\hat{f}(x)$ must be on opposite sides of $1/2$). Thus

$$\begin{aligned} r(\hat{C}) - r(C_0) &= \int \Big[\mathbb{P}\big(Y \neq \hat{C}(X) | X = x\big) - \mathbb{P}\big(Y \neq C_0(X) | X = x\big)\Big] \, dP_X(x) \\ &= 2 \int |f_0(x) - 1/2| 1\{C_0(x) \neq \hat{C}(x)\} \, dP_X(x) \\ &\leq 2 \int |f_0(x) - \hat{f}(x)| 1\{C_0(x) \neq \hat{C}(x)\} \, dP_X(x) \\ &\leq 2 \int |f_0(x) - \hat{f}(x)| \, dP_X(x) \\ &\leq 2 \sqrt{\int \big(f_0(x) - \hat{f}(x)\big)^2 \, dP_X(x)}, \end{aligned}$$

where in the last line we used the Cauchy-Schwartz inequality

- Abbreviate the L_2 regression risk by $R(\hat{f}) = \mathbb{E}\|\hat{f} - f_0\|_2^2 = \int (\hat{f}(x) - f_0(x))^2 \, dP_X(x)$. Then to rewrite the above, have established

$$r(\hat{C}) - r(C_0) \leq 2\sqrt{R(\hat{f})}, \tag{41}$$

and hence, any bound on the L_2 risk of the estimator $\hat{f}$ for regression function f_0 translates into a bound on the excess classification risk of the plug-in classifier $\hat{C}$. This sounds pretty great, because we have seen a lot of risk guarantees so far for nonparametric regression, and so we should be able to generate a lot of classification guarantees for plug-in classifiers! E.g., if $f_0 \in H_d(\alpha, L)$ for a constant $L > 0$, then for a suitable regression estimate $\hat{f}$,

$$R(\hat{f}) \lesssim n^{-2\alpha/(2\alpha+d)} \;\Rightarrow\; r(\hat{C}) - r(C_0) \lesssim n^{-\alpha/(2\alpha+d)}$$

8.2 Classification: easier or harder?

- Of course, the result in (41) is just an upper bound, and we don't necessarily know that it is tight. An interesting result in Chapter 6.7 of Devroye et al. (1996) suggests that actually (41) could be quite pessimistic, and hence classification could be a lot easier than regression (in a sense): they show that if $R(\hat{f}) \to 0$ as $n \to \infty$, and $\hat{C}$ is the plug-in classifier defined based on $\hat{f}$, then

$$\frac{r(\hat{f}) - r(C_0)}{R(\hat{f})^{1/2}} \to 0 \quad \text{as } n \to \infty. \tag{42}$$

The proof is straightforward. From the above calculation (leading up to (41)) we have, for any fixed $\epsilon > 0$,

$$\begin{aligned}
r(\hat{C}) - r(C_0) &\leq 2 \int |f_0(x) - \hat{f}(x)| 1\{C_0(x) \neq \hat{C}(x)\} 1\{f_0(X) \neq 1/2\} \, dP_X(x) \\
&\leq 2\mathbb{E}|f_0(X) - \hat{f}(X)| 1\{C_0(X) \neq \hat{C}(X)\} 1\{|f_0(X) - 1/2| \leq \epsilon, f_0(X) \neq 1/2\} + \\
&\quad 2\mathbb{E}|f_0(X) - \hat{f}(X)| 1\{C_0(X) \neq \hat{C}(X)\} 1\{|f_0(X) - 1/2| > \epsilon\} \\
&\leq 2\sqrt{R(\hat{f})} \bigg(\Big[\underbrace{\mathbb{P}\big(C_0(X) \neq \hat{C}(X), |f_0(X) - 1/2| \leq \epsilon, f_0(X) \neq 1/2\big)}_{a} \Big]^{1/2} + \\
&\qquad \Big[\underbrace{\mathbb{P}\big(C_0(X) \neq \hat{C}(X), |f_0(X) - 1/2| > \epsilon\big)}_{b} \Big]^{1/2} \bigg),
\end{aligned}$$

where in the last line we again applied the Cauchy-Schwartz inequality. For the second term in brackets above, note $b \leq \mathbb{P}(|f_0(X) - \hat{f}(X)| > \epsilon) \leq R(\hat{f})/\epsilon^2 \to 0$ as $n \to \infty$, where we used Markov's inequality. On the other hand, the term a can be made arbitrarily small as $\epsilon \to 0$. This verifies (42)

- While it is striking at first, the result in (42) says nothing about how slow this convergence is. So instead of just accepting that the excess classification risk is smaller than the L_2 regression risk, we should ask about rate. E.g., for Holder functions of smoothness α in d dimensions, we should ask: is an excess classification risk of $n^{-\alpha/(2\alpha+d)}$ good or bad?

- As usual, minimax theory provides us with a good grounding here. Yang (1999) showed that, as long as the class of functions (say) $\mathcal{F}$ that we consider for f_0 is sufficiently rich, the upper bound in (41) will be tight up to rate, in that the minimax rate for $r(\hat{C}) - r(C_0)$ over $\mathcal{F}$ will be the square root of that for $R(\hat{f})$ over $\mathcal{F}$, i.e.,

$$\inf_{\hat{f}} \sup_{f_0 \in \mathcal{F}} R(\hat{f}) \asymp r_n^2 \quad \text{and} \quad \inf_{\hat{C}} \sup_{f_0 \in \mathcal{F}} r(\hat{C}) - r(C_0) \asymp r_n \tag{43}$$

- The richness condition given in Yang (1999) is described in terms the metric entropy (or log covering number) of the function class $\mathcal{F}$ in question. (This concept is often used to derive not only minimax lower bounds, but also upper bounds, e.g., for smoothing splines over Sobolev smoothness classes. We will cover this concept precisely later in the course.) It can be shown this richness condition is satisfied by the (d-dimensional) Holder class with $H_d(\alpha, C)$ for any $\alpha > 0$, and the (univariate) total variation class $F(k, C)$ for any $k \geq 0$

- In summary, provided the function class is rich enough—like the generic Holder and bounded variation classes—we can simply do plug-in classification to get a classification risk bound like (41), and we can be certain from the minimax theory in (43) that we are not missing anything in terms of rates

8.3 k-nearest-neighbors classification

- The plug-in estimator (39) when we estimate $\hat{f}$, using k-nearest-neighbors regression, is called *k-nearest-neighbors classification*. This idea is itself quite natural and worth discussing. Recall that here the regression estimate is $\hat{f}(x) = \frac{1}{k} \sum_{i \in \mathcal{N}_k(x)} y_i$, and so $\hat{C}(x) = 1$ if and only if more than half of the neighbors $\mathcal{N}_k(x)$ of x are of class 1. In words, this classifier "votes" based on class memberships of neighboring points to decide the predicted class

- From (41) and the result in (3) for k-nearest-neighbors regression, from near the beginning of these lectures notes, we know that the excess classification risk of k-nearest-neighbors classification, when f_0 is Lipschitz, and $k \asymp n^{2/(2+d)}$, satisfies

$$r(\hat{C}) - r(C_0) \lesssim n^{-1/(2+d)}. \tag{44}$$

  We also know from the remarks in the previous subsection that this rate cannot be improved in general for the excess classification risk over Lipschitz functions (recalling that L-Lipschitz functions are precisely $H_d(1, L)$)

- There are some well-known and somewhat remarkable distribution-free results for k-nearest-neighbors classification. The oldest, and most famous result, is due to Cover & Hart (1967). They show that, for basically any joint distribution of the data (X, Y), the 1-nearest-neighbor classifier $\hat{C} = \hat{C}_n$, where the notation here emphasizes that it has been trained on n samples (x_i, y_i), $i = 1, \ldots, n$, satisfies

$$\lim_{n \to \infty} r(\hat{C}_n) = \lim_{n \to \infty} \mathbb{P}\big(Y \neq \hat{C}_n(X)\big) = 2\mathbb{E}\big(f_0(X)(1 - f_0(X)\big).$$

  (Here $n \to \infty$ with d fixed.) Furthermore,

$$2\mathbb{E}\big(f_0(X)(1 - f_0(X)\big) \leq 2\mathbb{E} f_0(X)\big(1 - \mathbb{E} f_0(X)\big) = 2r(C_0)(1 - r(C_0)) \leq 2r(C_0).$$

  The first inequality follows from direct integration, and the subsequent equality follows as the Bayes risk is $r(C_0) = \mathbb{E} \min\{f_0(X), 1 - f_0(X)\}$. Putting the above two displays together, we get what is sometimes called the Cover-Hart inequality,

$$\lim_{n \to \infty} r(\hat{C}_n) \leq 2r(C_0),$$

  or in words, the asymptotic probability of error of the 1-nearest-neighbor classifier is no more than twice the Bayes error. For more, see Chapter 5 of Devroye et al. (1996)

8.4 Nonparametric logisitic regression

- One opposition to "naive" plug-in methods: it might seem a bit funny to estimate f_0 using a standard nonparametric regression tool, because these tools are usually defined by a squared error loss criterion, and f_0 always lies between 0 and 1. An alternative approach would be to change the loss function so that it is appropriate for the classification setting. A favorite of statisticians is to instead directly model the conditional log odds,

$$g_0(x) = \log\Big(\frac{\mathbb{P}(Y=1|X=x)}{\mathbb{P}(Y=1|X=x)}\Big),$$

  and this leads to a nonparametric logistic regression model

- As an example, the *logistic smoothing spline* estimate of polynomial order $k \geq 0$ is defined by

$$\hat{g} = \underset{g}{\operatorname{argmin}} \sum_{i=1}^n \big(-y_i g(x_i) + \log(1 + e^{-g(x_i)})\big) + \lambda \big(g^{(m)}(x)\big)^2 \, dx, \quad \text{where } m = (k+1)/2.$$

  Let $\eta_1, \ldots, \eta_n$ denote the natural kth-order spline basis with knots over the inputs $x_1, \ldots, x_n$, and define the basis matrix $N \in \mathbb{R}^{n \times n}$ and penalty matrix $\Omega \in \mathbb{R}^{n \times n}$ just as we did before, for smoothing splines in regression, in (13). Then we can reformulate the above problem as

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^n}{\operatorname{argmin}} \; -y^T N\beta + \sum_{i=1}^n \log(1 + e^{-(N\beta)_i}) + \lambda \beta^T \Omega \beta,$$

  a logistic regression problem with a generalized ridge penalty. Our classifier $\hat{C}(x)$ now outputs 1 when $\hat{g}(x) = \sum_{j=1}^n \hat{\beta}_j \eta_j(x) > 0$, and outputs 0 otherwise

- Deriving an error bound for $\hat{g}$ as an estimate of the conditional log odds g_0 is more difficult than the analogous calculation for smoothing splines in regression; but altogether, the final bounds are fairly similar. See Chapter 10.2 of van de Geer (2000)

- Finally, it is worth noting that an approach like that above extends seamlessly to the additive model setting, and also to any loss derived from an exponential family distribution. This lies at the heart of *generalized additive models* for producing flexible nonparametric estimates in multiple dimensions, whether predicting a continuous response (regression), a binary response (logistic regression), or counts (Poisson regression), etc. See Hastie & Tibshirani (1990)

References

Bellman, R. (1962), *Adaptive Control Processes*, Princeton University Press.

Cover, T. & Hart, P. (1967), 'Nearest neighbor pattern classification', *IEEE Transactions on Information Theory* **13**(1), 21–27.

de Boor, C. (1978), *A Practical Guide to Splines*, Springer.

Demmler, A. & Reinsch, C. (1975), 'Oscillation matrices with spline smoothing', *Numerische Mathematik* **24**(5), 375–382.

Devroye, L., Gyorfi, L., & Lugosi, G. (1996), *A Probabilistic Theory of Pattern Recognition*, Springer.

Donoho, D. L. & Johnstone, I. (1998), 'Minimax estimation via wavelet shrinkage', *Annals of Statistics* **26**(8), 879–921.

Eggermont, P. P. B. & LaRiccia, V. N. (2006), 'Uniform error bounds for smoothing splines', *IMS Lecture Notes–Monograph Series* **51**, 220–237.

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.

Hastie, T. & Tibshirani, R. (1990), *Generalized Additive Models*, Chapman and Hall.

Hastie, T., Tibshirani, R. & Friedman, J. (2009), *The Elements of Statistical Learning; Data Mining, Inference and Prediction*, Springer. Second edition.

Johnstone, I. (2011), *Gaussian estimation: Sequence and wavelet models*, Under contract to Cambridge University Press. Online version at http://www-stat.stanford.edu/~imj.

Kim, S.-J., Koh, K., Boyd, S. & Gorinevsky, D. (2009), 'ℓ_1 trend filtering', *SIAM Review* **51**(2), 339–360.

Kimeldorf, G. & Wahba, G. (1970), 'A correspondence between bayesian estimation on stochastic processes and smoothing by splines', *Annals of Mathematical Statistics* **41**(2), 495–502.

Lin, Y. & Zhang, H. H. (2006), 'Component selection and smoothing in multivariate nonparametric regression', *Annals of Statistics* **34**(5), 2272–2297.

Mallat, S. (2008), *A wavelet tour of signal processing*, Academic Press. Third edition.

Mammen, E. & van de Geer, S. (1997), 'Locally apadtive regression splines', *Annals of Statistics* **25**(1), 387–413.

Nussbaum, M. (1985), 'Spline smoothing in regression models and asymptotic efficiency in l_2', *Annals of Statistics* **13**(3), 984–997.

Raskutti, G., Wainwright, M. & Yu, B. (2012), 'Minimax-optimal rates for sparse additive models over kernel classes via convex programming', *Journal of Machine Learning Research* **13**, 389–427.

Ravikumar, P., Liu, H., Lafferty, J. & Wasserman, L. (2009), 'Sparse additive models', *Journal of the Royal Statistical Society: Series B* **75**(1), 1009–1030.

Scholkopf, B. & Smola, A. (2002), 'Learning with kernels'.

Silverman, B. (1984), 'Spline smoothing: the equivalent variable kernel method', **12**(3), 898–916.

Simonoff, J. (1996), *Smoothing Methods in Statistics*, Springer.

Steidl, G., Didas, S. & Neumann, J. (2006), 'Splines in higher order TV regularization', *International Journal of Computer Vision* **70**(3), 214–255.

Stone, C. (1985), 'Additive regression models and other nonparametric models', *Annals of Statistics* **13**(2), 689–705.

Tibshirani, R. J. (2014), 'Adaptive piecewise polynomial estimation via trend filtering', *Annals of Statistics* **42**(1), 285–323.

Tsybakov, A. (2009), *Introduction to Nonparametric Estimation*, Springer.

van de Geer, S. (2000), *Empirical Processes in M-Estimation*, Cambdrige University Press.

van de Geer, S. & Muro, A. (2015), 'Penalized least squares estimation in the additive model with different smoothness for the components', *Journal of Statistical Planning and Inference* **162**, 43–61.

Wahba, G. (1990), *Spline Models for Observational Data*, Society for Industrial and Applied Mathematics.

Wang, X., Du, P. & Shen, J. (2013), 'Smoothing splines with varying smoothing parameter', *Biometrika* **100**(4), 955–970.

Wang, Y., Smola, A. & Tibshirani, R. J. (2014), 'The falling factorial basis and its statistical properties', *International Conference on Machine Learning* **31**.

Wasserman, L. (2006), *All of Nonparametric Statistics*, Springer.

Yang, Y. (1999), 'Nonparametric classification–Part I: Rates of convergence', *IEEE Transactions on Information Theory* **45**(7), 2271–2284.