Chapter 2
Splines, Smoothers, and Kernels

2.1 Introduction

This chapter launches a more detailed examination of statistical learning within a regression framework. Once again, the focus is on conditional distributions. But for now, the mean function for a response variable is central. How does the mean vary with different predictor values? The intent is to begin with procedures that have much the same look and feel as conventional linear regression and gradually move toward procedures that do not.

2.2 Regression Splines

A "spline" is a thin strip of wood that can be easily bent to follow a curved line (Green and Silverman 1994: 4). Historically, it was used in drafting for drawing smooth curves. Regression splines, a statistical translation of this idea, are a way to represent nonlinear, but unknown, mean functions.

Regression splines are not used a great deal in empirical work. As we show later, there are usually better ways to proceed. Nevertheless, it is important to consider them, at least briefly. They provide an instructive transition between conventional parametric regression and the kinds of smoothers commonly seen in statistical learning settings. They also introduce concepts and concerns that will be relevant throughout this chapter and in subsequent chapters.

Recall the general expression for function estimation with a quantitative response variable Y.

$$Y = f(\mathbf{X}) + \varepsilon, \tag{2.1}$$

The original version of this chapter was revised: See the "Chapter Note" section at the end of this chapter for details. The erratum to this chapter is available at https://doi.org/10.1007/978-3-319-44048-4_10.

© Springer International Publishing Switzerland 2016

R.A. Berk, *Statistical Learning from a Regression Perspective*,
Springer Texts in Statistics, DOI 10.1007/978-3-319-44048-4_2

where $\mathbf{X}$ is a set of predictors, $f(\mathbf{X})$ is not specified, and ε can be nothing more than a residual or nothing less than a traditional disturbance term depending, respectively, on whether a level I or level II regression is being undertaken.

Regression splines are an algorithmic way to empirically arrive at a $f(\mathbf{X})$. For a level I analysis, one tries to show how $\mathbf{X}$ is related to Y. For a conventional level II analysis, one tries to estimate nature's true response surface. For the kind of level II analysis we are using, one estimates an approximation of that true response surface. That is, the estimation target is acknowledged to be an approximation the $f(\mathbf{X})$, not the true $f(\mathbf{X})$, but one attempts to get close to the true response surface. The most common estimator is ordinary least squares. That estimator is linear, and the loss function is quadratic.

For a piecewise linear basis, the goal is to fit the data with a broken line (or hyperplane) such that at each break point the left-hand edge meets the right-hand edge. When there is a single predictor, for instance, the fit is a set of straight line segments, connected end to end, sometimes called "piecewise linear."

2.2.1 Applying a Piecewise Linear Basis

Consider as an example a wonderfully entertaining paper in which Zelterman (2014) documents associations by state in the U.S. between the number of internet references to zombies and census features for those states. The number of zombie references was determined through a "Google search" state by state. Like Zelterman, we let y be the number of zombie references by state per 100,000 people. We let x be the average in each state of minutes per day spent watching cable TV. The values used here for both variables are total fabrications constructed for didactic purposes.

Figure 2.1 shows an example of a piecewise linear function constructed in three steps.[1] The first step is to decide where the break points on x will be. Based on prior market research, suppose there are known tipping points at 90 and 180 min of TV watching a day. At 90 min a day, viewers often become fans of zombies on TV and shows with related content. At 180 min a day, zombie saturation is reached. Break points are defined at $x = a$ and $x = b$ (with $b > a$) so that in Fig. 2.1, $a = 90$ and $b = 180$. Such break points are often called "knots."

The next step is to define two indicator variables to represent the break points. The first, I_a, is equal to 1 if x is greater than 90 and equal to 0 otherwise. The second, I_b, is equal to 1 if x is greater than 180 and equal to 0 otherwise. Both are step functions. We let x_a be the value of x at the first break point, and x_b be the value of x at the second break point (i.e. 90 and 180 respectively).

The final step is to define the mean function that allows for changes in the slope and the intercept:

$$f(x_i) = \beta_0 + \beta_1 x_i + \beta_2 (x_i - x_a) I_a + \beta_3 (x_i - x_b) I_b. \tag{2.2}$$

[1] Using conventional multiple regression, Zelterman (2014: 40–41) finds more internet references to zombies in states with fewer Miss America winners and fewer shopping centers per capita. Who knew?

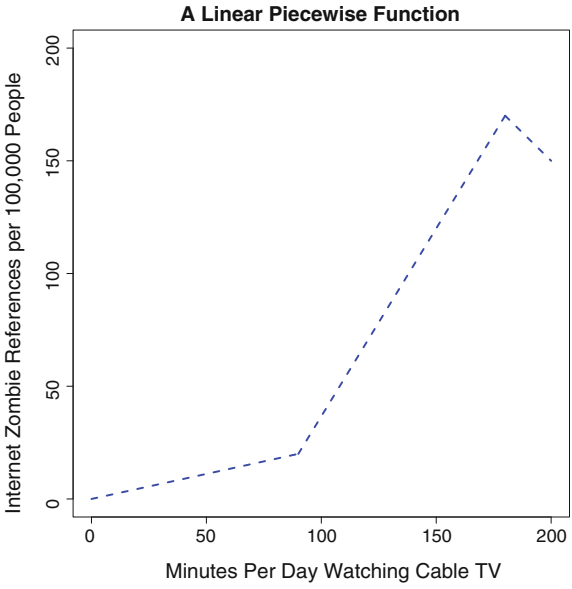

Fig. 2.1 A piecewise linear function with two knots for the number of zombie references on the internet as a function of minutes per day watching cable TV

Looking back how linear basis expansions are defined in Eq. 1.17, it is apparent that there are four transformations of X ($m = 1, 2, \ldots, 4$), each denoted by $h_m(X)$, in which the first function of x is a constant. One has a set of predictors constructed as a linear basis expansion of X. Equation 2.2 is, therefore, the mean function for a conventional multiple regression with coefficient values that can be obtained from ordinary least squares.

What exactly is the point of Eq. 2.2? Equation 2.2 is *not* a model representing how the data were generated. But, it can be used as a mean function in a least squares regression to help find fitted values that summarize associations in the data. Going no farther, this can result in a level I analysis through which relationships in the data are described.

For a level II analysis, Eq. 2.2 can play two related roles. First, it can represent the mean function to be estimated. The approach taken here is that Eq. 2.2 is the expression for the piecewise linear function that is a feature of a joint probability distribution, but with no claims made about its correspondence to the truth. It is the estimation target. When embedded in the population least squares framework discussed in the last chapter, it can be viewed as the best (i.e., by least squares), piecewise linear approximation of the true response surface and an appropriate estimation target. Second, used as the mean function in a least squares procedure applied to training data, it is the mean function for the least squares estimator.

Whether for a level I or level II analysis, the piecewise linear function can be decomposed into its constituent line segments. The mean function for x less than a is

$$f(x_i) = \beta_0 + \beta_1 x_i. \tag{2.3}$$

In Fig. 2.1, β_0 is zero, and β_1 is positive.

For values of x greater than a but smaller than b, the mean function becomes

$$f(x_i) = (\beta_0 - \beta_2 x_a) + (\beta_1 + \beta_2)x_i. \tag{2.4}$$

For a positive β_1 and β_2, the line beyond $x = a$ is steeper because the slope is $(\beta_1 + \beta_2)$. The intercept is lower because of the second term in $(\beta_0 - \beta_2 x_a)$. This too is consistent with Fig. 2.1. If β_2 were negative, the reverse would apply.

For values of x greater than b, the mean function becomes,

$$f(x_i) = (\beta_0 - \beta_2 x_a - \beta_3 x_b) + (\beta_1 + \beta_2 + \beta_3)x_i. \tag{2.5}$$

For these values of x, the slope is altered by adding β_3 to the slope of the previous line segment. The intercept is altered by subtracting $\beta_3 x_b$. The sign and magnitude of β_3 determine by whether the slope of the new line segment is positive or negative and how steep it is. The intercept will shift accordingly. In Fig. 2.1, β_3 is negative and large enough to make the slope negative. The intercept is increased substantially.

Expressions like Eq. 2.2 are all one needs for a level I regression analysis. For a level II regression analysis, a credible data generating process also must be articulated. As before, we will generally employ a joint probability distribution as the "population" from which each observation is realized as a random, independent draw. Very often this approach is reasonable. But each level II regression analysis must be considered on a case by case basis. For example, if time is a predictor (e.g., month or year), there can be important conceptual complications as we will now see.

Figure 2.2 shows a three-piece linear regression spline applied to water use data from Tokyo over a period of 27 years.[2] The data were collected as part of a larger research project motivated by concerns about the provision of potable water to large metropolitan areas as human-induced climate change proceeds. Residential water use in 1000s of cubic feet is on the vertical axis. Year is on the horizontal axis. The locations of the break points were chosen using subject matter expertise about residential water use in Japan. The R code is shown in Fig. 2.3.

For a level I analysis, it is clear that water use was flat until about 1980, increased linearly until about 1996, and then flattened out again. The first break point may correspond to a transition toward much faster economic and population growth. The second break point may correspond to the introduction of more water-efficient technology. But why the transitions are so sharp is mysterious. One possibility is

[2]The data were provided by the Tokyo Municipal Water Works as part of a project funded by The Asian-Pacific Network for Global Change Research.

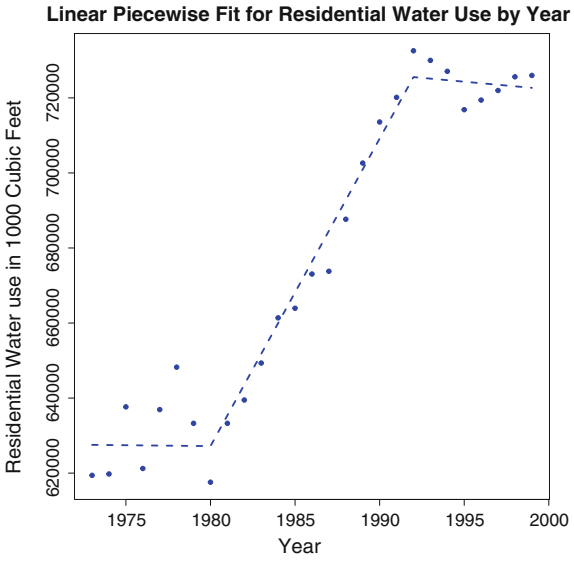

Fig. 2.2 A piecewise linear basis applied to water use in Tokyo by year

```
xa<-ifelse(year>1980,1,0)
xb<-ifelse(year>1992,1,0)
x1<-(year-1980)*xa
x2<-(year-1992)*xb
working<-data.frame(hhwater,year,xa,xb,x1,x2)
out1<-lm(hhwater~year+x1+x2,data=working)
plot(year,hhwater,xlab="Year",ylab="Residential
       Water use in 1000 Cubic Feet", main="Linear
       Piecewise Fit for Residential Water Use by Year",
       col="blue",pch=19)
lines(year,out1$fitted.values, lty="dashed",col="blue",lwd=3)
```

Fig. 2.3 R code for piecewise linear fit

that the break points correspond in part to changes in how the water use data were collected or reported.

In Fig. 2.2, the end-to-end connections between line segments work well with processes that unfold over time. But there is nothing about linear regression splines requiring that time be a predictor. For example, the response could be crop production per acre and the sole predictor could be the amount of phosphorus fertilizer applied to the soil. Crop production might increase in approximately a linear fashion until there is an excess of phosphorus causing of metabolic difficulties for the crops. At that point, crop yields might decline in roughly a linear manner.

More generally, fitting line segments to data provides an example of "smoothing" a scatterplot, or applying a "smoother." The line segments are used to summarize how x and y are related. The intent is to highlight key features of any association while removing unimportant details. This can often be accomplished by constructing fitted values in a manner that makes them more homogeneous than the set of conditional means of y computed for each unique value of x or binned values of x.[3]

Imagine a scatterplot in which the number of observations is large enough so that for each value of x there are at least several values of y. One could compute the mean of y for each x-value. If one then drew straight lines between adjacent conditional means, the resulting smoother would be an interpolation of the conditional means and as "rough" as possible. At the other extreme, imposing a single linear fit on all of the means at once would produce the "smoothest" fit possible. Figure 2.2 falls somewhere in between. How to think about the degree of smoothness more formally is addressed later. Effective degrees of freedom and the bias-variance tradeoff discussed in the last chapter are a start.

For a piecewise linear basis, one can simply compute mean functions such as Eq. 2.2 with ordinary least squares. With the regression coefficients in hand, fitted values are easily constructed. Indeed, many software packages compute and store fitted values on a routine basis. Also widely available are procedures to construct the matrix of regressors, although it is not hard to do so one term at a time using common transformation capabilities (See Fig. 2.3.). For example, the library *spline* has a procedure *bs()* that constructs a B-spline basis (discussed later) that can be easily used to represent the predictor matrix for piecewise linear regression.

In contrast to most applications of conventional linear regression, there would typically be little interest in the regression coefficients themselves; they are but a means to an end. The point of the exercise is to superimpose the fitted values on a scatterplot so that the relationship between y and x can be more effectively visualized. The story is in the visualization not the regression coefficients.

As already noted, for a level II analysis the estimation target is the same mean function in the population or joint probability distribution. But, with the longitudinal data, the inferential issues can be tricky. In particular, treating the data as independent, random realizations from a joint probability distribution does not work well. An obvious difficulty is that allowing year to be a random variable means that for any given dataset, some years might not be represented at all and some years might be represented more than once. In fact, the data were assembled with these particular years in mind; the data collection process took the specified years as given.

If in practice, year is treated as fixed, it probably makes sense to focus on a data generating process that also treats year as fixed. Then, one can imagine nature conditioning on year so that for each year, there is a probability distribution of water consumption values that in principle could be realized. The data on hand are realized from these conditional distributions. But after conditioning on year, are the water use observations realized independently? For example, if in one year average water

[3]For these data, there is only one value of y for each unique values of x. They may be treated as if they were means.

consumption increases, does that by itself have implications for the realized values of water consumption the next year? Without extensive subject-matter knowledge it is hard to know. Alternatively, it might be possible to specify a mean function that addressed possible dependence, but that would require a reconceptualization of how the data were realized.

There are significant complications if the break points were chosen through data snooping. For example, there is data snooping if the scatter plot for water consumption by year was examined to choose the best break point locations. Those locations become features of the regression specification itself so that the mean function specification could well be different with new realizations of the data (Leeb and Pötscher 2005, 2006, 2008). But, with the year treated as fixed, valid asymptotic inferences can be made to the same best, linear piecewise approximation, although conventional confidence intervals and tests are no longer valid, even asymptotically. If one assumes independence after conditioning on year, there can be valid asymptotic confidence intervals and tests using other procedures (Berk et al. 2010, 2014; Lockhart et al. 2014). However, the sample size here is very small. There are only 27 observations, and 4 degrees of freedom are used by the piecewise linear regression.[4]

The difficulties with a level II analysis may be even more fundamental. Nature's data generation process assumes, in effect, that history can repeat itself over and over, but we only get to see one random set of realized water consumption values from the joint probability distribution for the years in question. For Fig. 2.2, one must be comfortable assuming that Tokyo's water consumption values from 1970 to 1997 could have been different. Superficially, this may seem reasonable enough. But to be convincing, one would need to describe the ways in which nature makes this happen. In the absence of such an account, one has started down the slippery slope of assume-and-proceed statistics. It may be wise here to remain at level I.

2.2.2 Polynomial Regression Splines

Smoothing a scatterplot using a piecewise linear basis has the great advantage of simplicity in concept and execution. And by increasing the number of break points, very complicated relationships can be approximated. However, in most applications there are good reasons to believe that the underlying relationship is not well represented with a set of straight line segments. Another kind of approximation is needed.

Greater continuity between line segments can be achieved by using polynomials in x for each segment. Cubic functions of x are a popular choice because they strike a nice balance between flexibility and complexity. When used to construct regression splines, the fit is sometimes called "piecewise cubic." The cubic polynomial serves as a "truncated power series basis" in x.

[4] If there is dependence, one is in uncharted waters for post-model selection inference.

Unfortunately, simply joining polynomial segments end to end is unlikely to result in a visually appealing fit where the polynomial segments meet. The slopes of the two lines will often appear to change abruptly even when that is inconsistent with the data. Far better visual continuity usually can be achieved by constraining the first and second derivatives on either side of each break point to be the same.

One can generalize the piecewise linear approach and impose those continuity requirements. Suppose there are K interior break points, usually called "interior knots." These are located at $\xi_1 < \cdots < \xi_K$ with two boundary knots added at ξ_0 and ξ_{K+1}. Then, one can use piecewise cubic polynomials in the following mean function exploiting, as before, linear basis expansions of X:

$$f(x_i) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \sum_{j=1}^{K} \theta_j (x_i - x_j)_+^3, \tag{2.6}$$

where the "+" indicates the positive values from the expression inside the parentheses, and there are $K + 4$ parameters whose values need to be computed. This leads to a conventional regression formulation with a matrix of predictor terms having $K + 4$ columns and N rows. Each row would have the corresponding values of the piecewise cubic polynomial function evaluated at the single value of x for that case. There is still only a single predictor, but now there are $K + 4$ basis functions.

The output for the far-right term in Eq. 2.6 may not be apparent at first. Suppose the values of the predictor are arranged in order from low to high. For example, $x = [1, 2, 4, 5, 7, 8]$. Suppose also that x_j is located at an x-value of 4. Then, $(x - x_j)_+^3 = [0, 0, 0, 1, 27, 64]$. The knot-value of 4 is subtracted from each value of x, the negative numbers set to 0, and the others cubed. All that changes from knot to knot is the value of x_j that is subtracted. There are K such knots and K such terms in the regression model.

Figure 2.4 shows the water use data again, but with a piecewise cubic polynomial overlaid that imposes the two continuity constraints. The code is shown in Fig. 2.5. Figure 2.4 reveals a good eyeball fit, which captures about 95 % of the variance in water use. But, in all fairness, the scatterplot did not present a great challenge. The point is to compare Fig. 2.2 to Fig. 2.4 and note the visual difference. The linear piecewise fit also accounted for about 95 % of the variance. Which plot would be more instructive in practice would depend on the use to be made of the fitted values and on prior information about what a sensible $f(X)$ might be. This is a general point to which we will return many times. It can be very risky to rely on statistical summaries alone, such as fit statistics, to select the most instructive response surface approximation. Subject-matter knowledge, potential applications and good judgments need to play an important role.[5]

[5]This also is a good example of the problems one faces trying to adjust for the degrees of freedom used by the regression. There seems to be no way explicitly to introduce the constraints on the first and second derivatives, although they are built into fitting process. More is involved than the K estimates of θ_j.

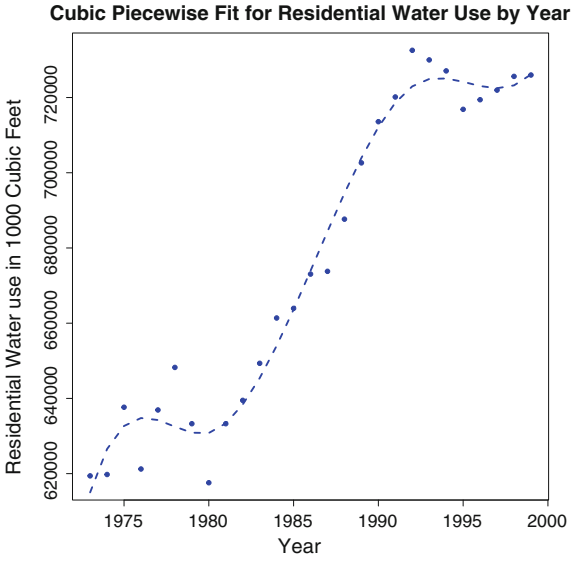

Fig. 2.4 A piecewise cubic basis applied to water use in Tokyo by year

```
library(splines)
cubic<-bs(year,knots=c(1980,1992))
out2<-lm(hhwater~cubic)
plot(year,hhwater,xlab="Year",ylab="Residential Water use
       in 1000 Cubic Feet", main="Cubic Piecewise Fit for
       Residential Water Use by Year",col="blue",pch=19)
lines(year,out2$fitted.values,lty="dashed",col="blue",lwd=3)
```

Fig. 2.5 R code for piecewise cubic fit

The regression coefficients ranged widely and, as to be expected, did not by themselves add any useful information. Any story was primarily in the fitted values. The issues for a level I and level II analysis are essentially the same as for a piecewise linear approach.

2.2.3 Natural Cubic Splines

Fitted values for piecewise cubic polynomials near the boundaries of x can be unstable because they fall at the ends of polynomial line segments where there are no continuity constraints, and where the data may be sparse. By "unstable" one means that a very few observations, which might vary substantially over random realizations of the

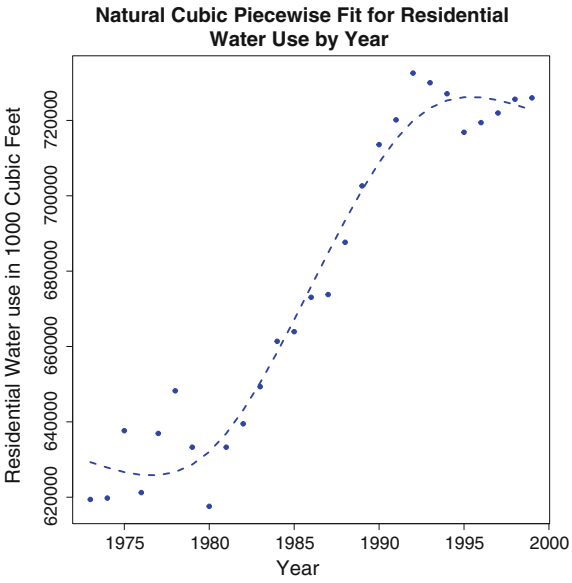

Fig. 2.6 A natural cubic piecewise basis applied to water use in Tokyo by year

data, could produce rather different fitted values near the boundaries of x. As a result, the plot of the fitted values near the boundaries might look somewhat different from sample to sample.

Sometimes, constraints for behavior at the boundaries are added to increase stability. One common constraint imposes linearity on the fitted values beyond the boundaries of x. This introduces a bit of bias because it is very unlikely that if data beyond the current boundaries were available, the relationship would be linear. However, the added stability is often worth it. When these constraints are added, the result is a "natural cubic spline."

Figure 2.6 shows again a plot of the water use data on year, but now with a smoother constructed from natural cubic splines. The code can be found in Fig. 2.7. One can see that the fitted values near the boundaries of x are somewhat different from the fitted values near the boundaries of x in Fig. 2.4. The fitted values in Fig. 2.6 are smoother, which is the desired result. There is one less bend near both boundaries, but the issues for a level I or level II analysis have not changed.[6]

The option of including extra constraints to help stabilize the fit provides an example of the bias–variance tradeoff discussed in the previous chapter, but for piecewise cubic polynomials and natural cubic splines, the degree of smoothness is primarily a function of the number of interior knots. In practice, the smaller the number of knots, the smoother are the fitted values. A smaller number of knots means that there are more constraints on the pattern of fitted values because there

[6]More generally, how one can formulate the boundary constraints is discussed in Hastie et al. (2009: Sect. 5.2.1).

```
library(splines)
Ncubic<-ns(year,knots=c(1980,1992))
out3<-lm(hhwater~Ncubic)
plot(year,hhwater,xlab="Year",ylab="Residential Water use
       in 1000 Cubic Feet", main="Natural Cubic Piecewise
      Fit for Residential Water Use by Year",col="blue",pch=19)
lines(year,out3$fitted.values,lty="dashed",col="blue",lwd=3)
```

Fig. 2.7 R code for natural piecewise cubic fit

are fewer end-to-end, cubic line segments used in the fitting process. Consequently, less provision is made for a complex response surface.

Knot placement matters too. Ideally, knots should be placed where one believes, before looking at the data, the $f(X)$ is changing most rapidly. But it will often be very tempting to data snoop. In some cases, inspection of the data, coupled with subject matter knowledge, can be used to determine the number and placement of knots. Alternatively, the number and placement of knots can be approached as a conventional model selection problem. But that means determining a candidate set of models with different numbers of knots and different knot locations. That set could be very large. Absent subject matter information, knot placement has been long known to be a difficult technical problem, especially when there is more than one predictor (de Boors 2001). The fitted values are related to where the knots are placed in a very complicated manner. Fortunately, methods discussed later sidestep the knot number and location problem.

Even if a good case for a small number of candidate models can be made, one must be careful about taking any of their fit measures too literally. There will often be several models with rather similar values, whatever the kind of fit statistic used. Then, selecting a single model as "best" using the fit measure alone may amplify a small numerical superiority into a large difference in the results, especially if the goal is to interpret how the predictors are related to the response. Some call this "specious specificity." Also, one must be a very careful not to let small differences in the fit statistics automatically trump subject matter knowledge. The risk is arriving at a model that may be difficult to interpret, or effectively worthless. Finally, one has introduced the demanding complications that come with model selection. All is well for a level I analysis. But moving to a level II analysis can introduce difficult problems already mentioned. Toward the end of the chapter, we will consider an empirical example in which a split sample approach is very helpful for valid statistical inference.

In summary, for level II regression splines of the sort just discussed, there is no straightforward way to arrive at the best tradeoff between the bias and the variance because there is no straightforward way to determine knot location. A key implication is that it is very difficult to arrive at a model that is demonstrably the "best." Fortunately, there are other approaches to smoothing that are more promising. A broader

point is that we have begun the transition from models to black box algorithms. As the substantive role for fitted values has become more prominent, the substantive role for regression coefficients has becomes less prominent.

2.2.4 B-Splines

In practice, data analyses using piecewise cubic polynomials and natural cubic splines are rarely constructed directly from polynomials of x. They are commonly constructed using a B-spline basis, largely because of computational convenience.[7] A serious discussion of B-splines would take us far afield and accessible summaries can be found in Gifi (1990: 366–370) and Hastie et al. (2009: 186–189). Nevertheless several observations are worth making even if they are a bit of a detour.

The goal is to construct a piecewise fit from linear basis expansions of x with nice numerical properties. B-splines meet this test. They are computed in a recursive manner from very simple functions to more complex ones.

For a set of knots, usually including some beyond the upper and lower boundaries of x, the recursion begins with indicator variables for each neighborhood defined by the knots. If a value of x falls within a given neighborhood, the indicator variable for that neighborhood is coded 1, and coded 0 otherwise. For example, if there is a knot at an x-value of 2 and the next knot is at an x-value of 3, the x-values between them constitute a neighborhood with its own indicator variable coded 1 if the value of x falls in that neighborhood (e.g., $x = 2.3$). Otherwise the coded value is 0. In the end, there is a set of indicator variables, with values of 1 or 0, depending on the neighborhood. These indicator variables define a set of degree zero B-splines.

Figure 2.8 is an illustration with interior knots at -2, -1, 0, 1, and 2. With five interior knots, there are four neighborhoods and four indicator variables. Using indicator variables as regressors will produce a step function when y is regressed on x; they are the linear basis expansion for a step function fit. The steps will be located at the knots and for this example, the mean function specification will allow for four levels, one for each indicator variable. With a different set of knots, the indicator variables will change.

As usual, one of the indicator variables is dropped from any regression analysis that includes an intercept. The deleted indicator becomes the baseline. In R, the procedure *lm()* automatically drops one of the indicator variables in a set if an intercept is included.

Next, a transformation can be applied to the degree zero B-splines (See Hastie et al. 2009: 186–189). The result is a set of degree one B-splines. Figure 2.9 shows the set of degree one B-splines derived from the indicator variables shown in Fig. 2.8.

[7] But there can be lots of options, depending on the application. For example, there are special issues when the intent is to smooth a 2-dimensional surface. An excellent discussion can be found in Wood (2006: Sect. 4.1).

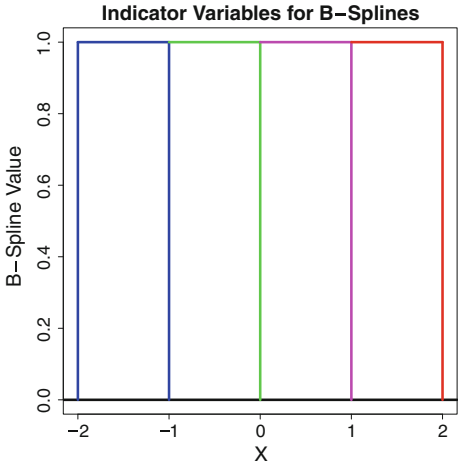

Fig. 2.8 Degree zero B-splines

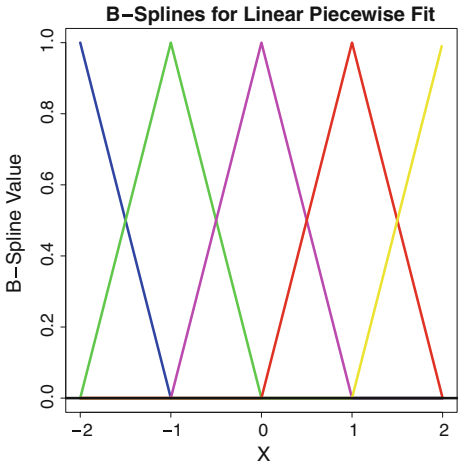

Fig. 2.9 Degree one B-splines

The triangular shape is characteristic of degree one B-splines, and indicates that the values for each spline are no longer just 0 or 1, but proportions in-between as well.

In Fig. 2.9, each new basis function is color coded. Starting from the left, the blue line maps x onto a set of B-spline values. From x-values of -2 to -1, the B-Spline values decline from 1 to 0 but are 0 for the rest of the x-values. These B-spline values would be the first column in a new predictor matrix. For x-values between -2 and 0, the green upside down V indicates that the B-spline values are between 0 and 1, but equal to 0 otherwise. These B-spline values would be the second column in a new predictor matrix. The same reasoning applies to the purple and red upside down Vs and to the yellow line. In the end, there would be six columns of B-spline values with the sixth column coded to have no impact because it is redundant,

given other five columns. That column is not shown inasmuch as it has B-spline values that are 0 for all x-values.

Degree one B-splines are the basis for linear piecewise fits. In this example, regressing a response on the B-spline matrix would produce a linear piecewise fit with four slopes, one for each neighborhood defined by the indicator variables. For different numbers and locations of knots, the piecewise fit would vary as well.

A transformation of the same form can now be applied to the degree one B-splines. This leads to a set of degree two B-splines. A set of such B-splines is shown in Fig. 2.10. As before, each new basis function is color coded, and the shapes are characteristic. For this illustration, there is now a matrix having seven columns with one redundant column coded as all 0s. Should the B-spline matrix be used in a regression analysis, a piecewise quadratic fit would be produced. There would be one quadratic function for each neighborhood defined by the indicator variables.

The same kind of transformation can then be applied to the degree two B-splines. The result is a set of degree three B-splines. Figure 2.11 shows the set of degree three color-coded splines, whose shapes are, once again, characteristic. The regressor matrix now contains eight columns with one redundant column coded as all 0s. When these are used as regressors, there will one cubic function for each neighborhood defined by the original indicator variables.

All splines are linear combinations of B-splines; B-splines are a basis for the space of all splines. They are also a well-conditioned basis because they are not highly correlated, and they can be computed in a stable and efficient manner. For our purposes, the main point is that B-splines are a computational device used to construct cubic piecewise fitted values. No substantive use is made of the associated regression coefficients because they too are just part of the computational machinery. Our trek toward black box algorithms continues.

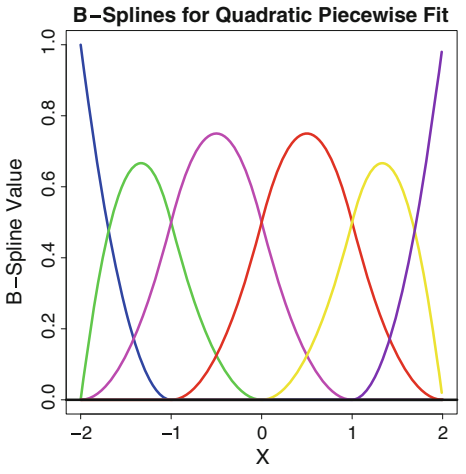

Fig. 2.10 Degree two B-splines

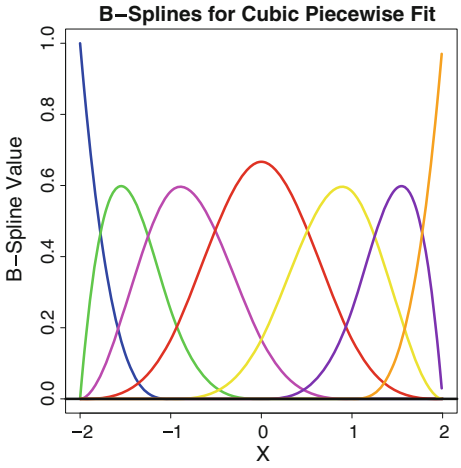

Fig. 2.11 Degree three *B*-splines

2.3 Penalized Smoothing

The placement of knots, the number of knots, and the degree of the polynomial are subject to manipulation by a data analyst. For a level I regression analysis, the goal is to arrive at an instructive fit of the data. Is one learning what one can about associations in the data? For a level II regression analysis, the goal is to estimate a useful approximation of the true response surface. Beyond documenting associations of various kinds, do the fitted values provide helpful information that might help guide future decisions?

Whether for a level I or level II analysis, the data analyst is engaged in "tuning." Therefore, the placement of knots, the number of knots, and the degree of the polynomial can seen as "tuning parameters." Unlike the usual parameters of a regression model, they typically are of little substantive interest. More like dials on a piece of machinery, they are set to promote good performance.

There are at least two problems with the tuning parameters for regression splines. First, there are at least three of them so that the tuning process can be quite complicated. For example, should one increase the number of knots or the degree of the polynomial? Usually, the only guidance is sketchy craft lore. Second, there is little or no formal theory to justify the tuning. To many, the tuning process feels like a "hack." The entire process is at least inelegant.

A useful alternative is to alter the fitting process itself so that the tuning is accomplished automatically, guided by clear statistical reasoning. One popular approach is to combine a mathematical penalty with the loss function to be optimized. The penalty imposes greater losses as a mean function becomes more complicated. For greater complexity to be accepted, the fit must be improved by an amount that is

larger than the penalty. The greater complexity has to be "worth it." This leads to a very popular approach called "penalized regression."[8]

2.3.1 Shrinkage and Regularization

To get a feel for penalized regression, consider a conventional regression analysis with an indicator variable as the sole regressor. If its regression coefficient equals zero, the fitted values will be a straight line, parallel to the x-axis, located at the unconditional mean of the response. As the regression coefficient increases in absolute value, the resulting step function will have a step of increasing size. The difference between the conditional means of Y when the indicator is 0 compared to the conditional means of Y when the indicator is 1 is larger. In language we have begun to use, the fit becomes more rough. Or in still other language, the fit is more complex. In short, the larger the regression coefficient the rougher the fitted values.

For a level I regression analysis, less complexity can mean that important features of the fitted values are overlooked. More complexity can complicate unnecessarily interpretations of the fitted values. For a level II regression analysis, less complexity means that a smoother approximation of the true response surface is being estimated, which can increase bias with respect to nature's true response surface. More complexity can increase the variance of estimates of that response surface. We have the bias-variance tradeoff once again, and we are once again seeking a Goldilocks solution. The fitted values should not be too rough. The fitted values should not be too smooth. They should be just right.

Popular Goldilocks strategies are sometimes called "shrinkage" (Hastie et al. 2009: Sect. 3.4) and sometimes called "regularization" (Hastie et al. 2009: Chap. 5). In the context of statistical learning, both are tools for trying to address the bias-variance tradeoff. But it can be helpful to think of shrinkage as a special case of regularization in which the loss function for a conventional linear regression is altered to include a penalty for complexity. We will see in later chapters that regularization can apply to a much wider range of procedures and take many different forms. For now, we focus on shrinkage.

A number of proposals have been offered for how to control the complexity of the fitted values by constraining the magnitude of regression coefficients (See Ruppert et al. 2003: Sect. 3.5 for a very accessible discussion.). Two popular suggestions are:

1. constrain the sum of the absolute values of the regression coefficients to be less than some constant C (sometimes called an L_1-penalty); and

[8] Suppose the error sum of squares for a given amount of fitted value complexity is 1000. Ordinarily, an increase in the complexity of the fitted values that reduces the error sum of squares to less than 1000 would accepted. But suppose there is a penalty of 100. Now the penalized error sum of squares is 1100. Still, the error sum of squares threshold remains at 1000. Improvement in the fit has to overcome the penalty of 100.

2. constrain the sum of the squared regression coefficients to be less than some constant C (sometimes called an L_2-penalty).

The smaller the value of C is, the smaller the sum. The smaller the sum, the smaller is the typical magnitude of the regression coefficients. The smaller the typical magnitude of the regression coefficients, the smoother the fitted values. In part because the units in which the regressors are measured will affect how much each regression coefficient contributes to the sum, it can make sense to work with standardized regressors.[9] Often, very little interpretive weight is carried by the regression coefficients in any case if interest centers on the fitted values. The intercept does not figure in either constraint and is usually addressed separately.

For a level I analysis, both constraints can impose different amounts of smoothness in the fitted values. Description of the relationships between the response and the predictors can be affected. For a level II analysis, both constraints lead to shrinkage methods. The regression coefficients can be "shrunk" toward zero, making the fitted values more homogeneous. The population approximation is altered in the same fashion. When the intent is to represent the true response surface, one is prepared to introduce a small amount of bias into the estimated regression coefficients in trade for a substantial reduction in their variance so that the same is true of the fitted values.

One also can recast some measures of fit discussed in the last chapter within a shrinkage framework. The total number of regression coefficients to be estimated can serve as a constraint and is sometimes called an L_0-penalty. Maximizing the adjusted R^2, for example, can be seen as minimizing the usual error sum of squares subject to a penalty for the number of regression coefficients in the model (Fan and Li 2006).

Shrinkage methods can be applied with the usual regressor matrix or with smoother matrices of the sort we introduced earlier. For didactic purposes, we start within a conventional multiple regression framework and p predictors.

2.3.1.1 Ridge Regression

Suppose that for a conventional fixed X regression, one adopts the constraint that the sum of the p squared regression coefficients is less than C. This constraint leads directly to ridge regression. The task is to obtain values for the regression coefficients so that

$$\hat{\boldsymbol{\beta}} = \min_{\beta} \left[\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} \beta_j^2 \right]. \tag{2.7}$$

In Eq. 2.7, the usual expression for the error sum of squares has a new component. That component is the sum of the squared regression coefficients multiplied by a

[9] For example, 3.2 additional years of age may count the same as smoking 11.4 additional cigarettes a day. They may both be equal to one standard deviation of their respective predictors.

constant λ. When Eq. 2.7 is minimized in order to obtain $\hat{\boldsymbol{\beta}}$, the sizes of the squared regression coefficients are taken into account. This is an L_2 penalty.

For a given value of λ, the larger the $\sum_{j=1}^{p} \beta_j^2$ is, the larger the increment to the error sum of squares. The $\sum_{j=1}^{p} \beta_j^2$ can be thought of as the penalty function. For a given value of $\sum_{j=1}^{p} \beta_j^2$, the larger the value of λ is, the larger the increment to the error sum of squares; λ determines how much weight is given to the penalty. In short, $\sum_{j=1}^{p} \beta_j^2$ is what is being constrained, and λ imposes the constraint. C is inversely related to λ. The smaller the value of C, the larger is the value of λ.

It follows that the ridge regression estimator is

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}, \tag{2.8}$$

where $\mathbf{I}$ is a $p \times p$ identity matrix. The column of 1s for the intercept is dropped from $\mathbf{X}$. β_0 is estimated separately.

In Eq. 2.8, λ plays same role as in Eq. 2.7, but can now be seen as a tuning parameter. It is not a feature of a population or a stochastic process. Its role is to help provide an appropriate fit to the data and can be altered directly by the data analyst. As such, it has a different status from the regression coefficients, whose values are determined through the minimization process itself, conditional upon the value of λ.

The value of λ is added to the main diagonal of the cross-product matrix $\mathbf{X}^T\mathbf{X}$, which determines how much the estimated regression coefficients are "shrunk" toward zero (and hence, each other). A λ of zero produces the usual least squares result. As λ increases in size, the regression coefficients approach zero, and the fitted values are smoother. In effect, the variances of the predictors are being increased with no change in the covariances between predictors and the response variable. This is easy to appreciate in the case of a single predictor. For a single predictor, the regression coefficient is the covariance of the predictor with the response divided by the variance of the predictor. So, if the covariance is unchanged and the variance is increased, the absolute value of the regression coefficient is smaller.

In ridge regression, the regression coefficients and fitted values obtained will differ in a complicated manner depending on the units in which the predictors are measured. It is common, therefore, to standardize the predictors before the estimation begins. However, standardization is just a convention and does not solve the problem of the scale dependent regression coefficients. There also can be some issues with exactly how the standardization is done (Bing 1994). In practice, the standardized regression coefficients are transformed back into their original units when time comes to interpret the results, but that obscures the impact of the standardization on Eq. 2.7. The penalty function has the standardized coefficients as its argument.

In whatever manner the value of λ is determined, a valid level I analysis may be undertaken. What varies is the smoothness of the fitted values from which descriptive summaries are constructed. For a level II analysis, if the value of λ is determined before the data analysis begins, one can resurrect a conventional "true model" approach. For reasons already addressed, that seems like a bad idea. Moreover, the shrinkage is a new source of bias. Using the "wrong model" perspective, one is esti-

mating in an asymptotically unbiased manner an approximation of the true response surface determined by the value of λ.

If the value of λ is determined as part of the data analysis, there are significant complications for level II analysis. A key issue is how the value of λ is chosen. Ideally, there are training data, evaluation data, and test data as described in the last chapter. Using the training data and the evaluation data, one option is trial and error. Different values of λ are tried with the training data until there is a satisfactory fit in the evaluation data by some measure such as mean squared error. With modern computing power, a very large number of potential values can be searched very quickly. Once a value for λ is determined, a level II analysis properly can be undertaken with the test data, although if the degrees of freedom are used in the calculations, there are complications (Dijkstra 2011). If all one cares about is a level I analysis, one can simply search over all the data at once for a value of λ that leads to good results in statistical and subject-matter terms. However, it will be useful to keep in mind how much searching has been done. It is easy to get carried away by complex results that are essentially a byproduct of noisy data. After a lot of searching, complex results that come as a surprise need to be very cautiously interpreted.

If there are no evaluation and test data and the dataset is too small to partition, there are the fallback options noted in the last chapter such as cross-validation. The value of λ is chosen to maximize some cross-validation measure of fit. But no matter what the method, search for the best values of λ can lead to overfitting, especially if the searching is very extensive. In cross validation, for example, the training data are reused many times.[10]

One must also be careful with how ridge regression results are interpreted. Even within the wrong model perspective, the regression coefficient values are regularized to achieve a desirable set of fitted values. Features of the fitted values are driving the results, and regression coefficients are but a means to that end. For both level I and level II analyses, it is not entirely clear why a better fit of the data implies more instructive regression coefficients. For example, in a properly implemented randomized experiment, average treatment effect estimates are unbiased and have causal interpretations even though the overall fit may be poor. Also, there is nothing especially compelling about the L_2 ridge penalty. Other kinds of defensible penalties exist that can produce very different results.

[10]Suppose in training data with 100 observations there are 15 observations that are by chance anomalous and affect the regression results. About 15 % of the observations in random splits of the data will be composed of these observations. So, in split after split, the regression results will be affected in a similar way. Yet, in real test data, that 15 % might not be represented at all or at least no nearly so commonly. In other words, one is held captive to whatever features characterize the training data, even if those features are essentially noise. This is why cross-validation needs to be justified asymptotically. In all fairness, split samples can have less extreme versions of the same problems.

2.3.1.2 A Ridge Regression Illustration

Perhaps an example will help fix these ideas. The data come from a survey of 95 respondents in which a key question is how various kinds of social support may be related to depression.[11] There are 19 predictors and for this illustration, we will work with three: (1) "emotional" — a "summary of 5 questions on emotional support availability," (2) "affect" — a "summary of 3 questions on availability of affectionate support sources," and (3) "psi" — a "summary of 3 questions on availability of positive social interaction." The response "BDI," which stands for Beck depression inventory, is a 21-item inventory based on self-reports of attitudes and symptoms characteristic of depression (Beck et al. 1961). We are treating the data as random realizations so that the predictors and the response are random variables. No doubt the model is misspecified by conventional criteria. We adopt, therefore the best linear approximation approach, and the usual output from *lm()* takes the following form.

```
Call:
lm(formula = BDI ~ emotional + affect + psi, data =
socsupport)

Residuals:
    Min      1Q  Median      3Q     Max
-14.141  -5.518  -0.764   3.342  32.667

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  31.5209     5.0111   6.290 1.09e-08 ***
emotional     0.2445     0.3458   0.707  0.48133
affect       -0.4736     0.4151  -1.141  0.25693
psi          -1.6801     0.5137  -3.270  0.00152 **
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1  1

Residual standard error: 8.6 on 91 degrees of freedom
Multiple R-squared:  0.216, Adjusted R-squared:  0.1901
F-statistic: 8.355 on 3 and 91 DF,  p-value: 5.761e-05
```

Consider first a level I analysis. The predictors "affect" and "psi" are negatively related to depression, and "emotional" is positively related to depression. As is often the case with constructed scales, it is difficult to know how substantively important the regression coefficients are. How big is big, and how small is small? For example, psi ranges from 5 to 15, and BDI ranges from 0 to 48. For each additional psi point, the average of the depression scale is about 1.7 points smaller. Even over the range of psi, the full range of BDI is not covered. Moreover, how does one translate variation

[11] The data, named *socsupport*, can be obtained as part of the *DAAG* library in R.

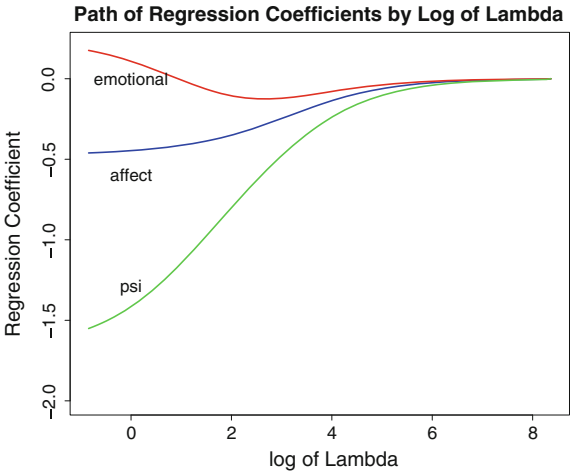

Fig. 2.12 Ridge regression results as a function of the log of the ridge penalty λ for each of the three predictors

in any of the variables into *clinical* importance? How many points make a clinical difference?

For level II analysis, one has in principle an estimate of a best linear approximation of the true response surface as a property of the joint probability distribution responsible for the data. But, we know far too little about how the data were collected to make such a case one way or another. What joint probability distribution are we talking about? And were the data actually realized independently? If these problems could be resolved, proper estimation, confidence tests, and statistical tests can follow with sandwich estimates of the standard errors and an asymptotic justification. In this case, one would again reject the null hypothesis of 0.0 for the psi regression coefficient but not for the other regression coefficients. Still, with only 91 residual degrees of freedom it is not clear if one can count on the asymptotics.[12]

The conventional least squares estimates provide a benchmark for ridge regression results. Using the same mean function, Fig. 2.12 shows how the regression coefficients change as the ridge penalty is given more weight. When the ridge penalty is ignored, one has the ordinary least squares estimates. But as the ridge penalty gets larger, all three coefficients are shrunk toward zero in a proportional manner. The larger the coefficient, the greater the shrinkage. For λ values greater than about 1100 (i.e., approximately e^7), all three coefficients are effectively zero, and all three arrive at zero together. This is a characteristic of ridge regression. The code is provided in Fig. 2.13.

But what value of λ should be used? Figure 2.13 shows with the red dotted line how the average mean-squared error from a tenfold cross-validation changes with

[12]Several different forms of sandwich standard errors can be estimated with hccm() in the *car* library. There is little formal guidance on which to use. One might as well work with the default.

```
### Get Data and Ridge Software
library(DAAG)
library(glmnet)
data(socsupport)
attach(socsupport)

### Least Squares
X<-as.matrix(data.frame(emotional,affect,psi)) #  Needs a matrix
out1<-lm(BDI~emotional+affect+psi,data=socsupport) # OLS results

### Ridge Regression
out2<-glmnet(X,BDI,family="gaussian",alpha=0) # Ridge results
plot(log(out2$lambda),out2$beta[2,],ylim=c(-2,.2),type="l",
     col="blue",lwd=3,xlab="log of Lambda", ylab="Regression
     Coefficient", main="Path of Regression Coefficients by
     Log of Lambda")
lines(log(out2$lambda),out2$beta[1,],type="l",col="red",lwd=3)
lines(log(out2$lambda),out2$beta[3,],type="l",col="green",lwd=3)
text(0,0,"emotional",cex=1.5)
text(0,-.6,"affect",cex=1.5)
text(0,-1.3,"psi",cex=1.5)
```

Fig. 2.13 R code for a least squares analysis and a ridge regression plot of coefficients as a function or λ

the log of λ.[13] The band around the average mean squared error is constructed as plus or minus two standard deviations computed from the 10 cross-validation folds. The blue vertical line shows the value of the log of λ for which the average mean squared error is the smallest. A logged value of λ equal to 1 does about as well as one can do. Looking back at Fig. 2.12, the regression coefficient for psi is shrunk from about -1.6 to about -1.0, affect is shrunk from about $-.5$ to about $-.4$, and the regression coefficient for emotional support is shrunk from about .25 to near 0.0. The regression coefficients and, consequently, the fitted values, have been regularized. But none of the regression coefficients are shrunk to exactly 0.0. The sequence of 3s across the top of the graph means that for each value of the log of λ, no predictors are dropped; all three predictors are retained in the regression.[14]

[13] The "canned" code from *glmnet* was used much as in the R documentation. Note that the regression coefficients are transformed back into their original units.

[14] In the output object from *glmnet()*, the 3 s are labeled "df", presumably for degrees of freedom. But the documentation makes clear that df is actually the number of predictors. In conventional linear regression, the degrees of freedom is usually taken to be the number of predictors plus 1 for the intercept. As noted in Chap. 1, when the value of λ is determined empirically, more degrees of freedom are used than the number of parameters to be estimated in the regression mean function. Moreover, one can get degrees of freedom that is not in integers (Dijkstra 2011). In short, the number of predictors should not be confused with the degrees of freedom used.

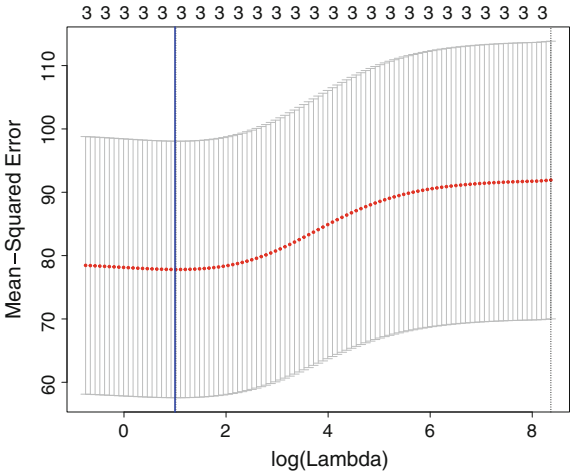

Fig. 2.14 Using cross-validation to choose the value of λ in ridge regression

Now what? Shrinkage is motivated by level II concerns, so the level I least squares analysis stands. For a level II analysis, the estimation target is the exact same ridge formulation with the selected value of λ as a feature of the joint probability distribution responsible for the data. It is not apparent why the ridge estimation target would be of interest. And even it it were, any level II interpretations must confront the model selection complications produced by the large number of cross-validation exercises undertaken (i.e., the default is 100). Estimates of the ridge regression approximation will be biased, even asymptotically. Confidence intervals and statistical tests will be invalid. It is with good reason that no statistical tests are provided by the software (Fig. 2.14).

In summary, the greatest substantive payoff from this mental health application probably comes from the level I analysis using ordinary least squares. The level II analysis from the least squares results would have been more persuasive had there been good reason to treat the data as random realizations from a substantively meaningful joint probability distribution or finite population. A larger sample would have helped too. Finally, there seems to be little that was gained applying the ridge regression formulation. One has to buy the focus on regression coefficients, the use of the L_2 penalty, and a lambda selected by cross-validation. And why was shrinkage a good idea to begin with in a very shaky level II setting? Perhaps the major take-home message is that ridge regression showcases some importance concepts and tools, but will not likely to be a useful data analysis procedure. We need to do better.

2.3.1.3 The Least Absolute Shrinkage and Selection Operator (LASSO)

Suppose that one proceeds as in ridge regression but now adopts the constraint that the sum of the absolute values of the regression coefficients is less than some constant. Just like for ridge regression, all of the predictors usually are standardized

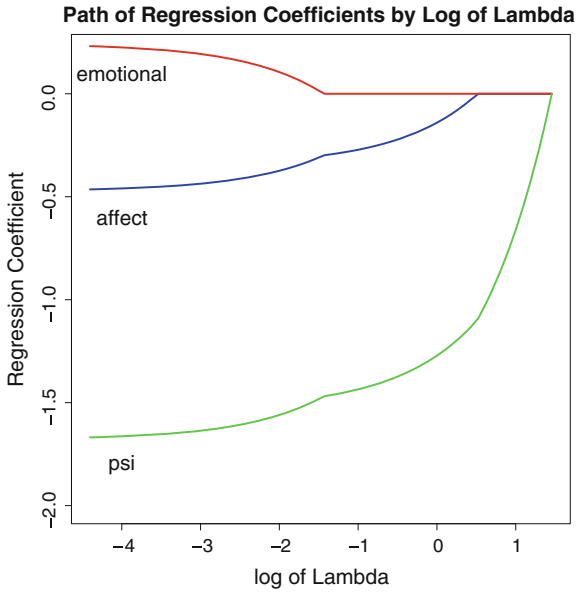

Fig. 2.15 Lasso regression results as a function of the log of the ridge penalty λ for each of the three predictors

for the calculations, but the regression coefficients are transformed back into their original units when time comes to interpret the results. The L_1 constraint leads to a regression procedure known as the lasso[15] (Tibshirani 1996) whose estimated regression coefficients are defined by

$$\hat{\beta} = \min_{\beta} \left[\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda \sum_{j=1}^{p} |\beta_j| \right]. \tag{2.9}$$

Unlike the ridge penalty, the lasso penalty leads to a nonlinear estimator, and a quadratic programming solution is needed. As before, the value of λ is a tuning parameter, typically determined empirically, usually through some measure of fit or prediction error. Just as with ridge regression, a λ of zero yields the usual least squares results. As the value of λ increases, the regression coefficients are shrunk toward zero.

2.3.1.4 A Lasso Regression Illustration

Using the same data as for the ridge regression analysis, Fig. 2.15 shows that in contrast to ridge regression, the regression coefficients are not shrunk proportionately. (The code is provided in Fig. 2.16.) The regression coefficients are shrunk by a

[15]LASSO is sometimes written as "lasso," Lasso," or "LASSO."

```
### lasso Results
out3<-glmnet(X,BDI,family="gaussian",alpha=1)  # lasso
plot(log(out3$lambda),out3$beta[2,],ylim=c(-2,.2),type="l",
     col="blue",lwd=3,xlab="log of Lambda", ylab="Regression
     Coefficient", main="Path of Regression Coefficients by
     Log of Lambda")
lines(log(out3$lambda),out3$beta[1,],type="l",col="red",lwd=3)
lines(log(out3$lambda),out3$beta[3,],type="l",col="green",lwd=3)
text(-4,.1,"emotional",cex=1.5)
text(-4,-.6,"affect",cex=1.5)
text(-4,-1.8,"psi",cex=1.5)
```

Fig. 2.16 R code for lasso regression plot of coefficients as a function or λ

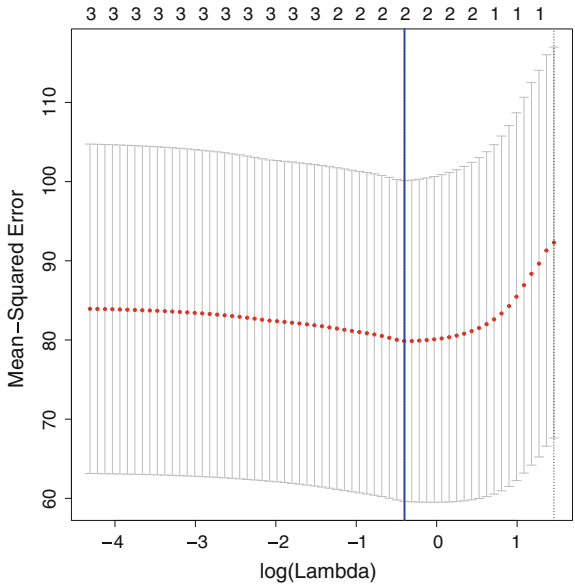

Fig. 2.17 Using cross-validiation to choose the value of λ in lasso regression

constant factor λ (Hastie it al. 2009: 69) so that some are shrunk relatively more than others as λ increases. But for a sufficiently large λ, all are shrunk to 0.0. This is a standard result that allows Hastie and his colleagues (2009: Sect. 3.4.5) to place ridge regression and the lasso in a larger model selection context. The lasso performs in a manner that has some important commonalities with model selection procedures used to choose a subset of regressors. When a coefficient value of 0.0 is reached, that predictor is no longer relevant and can be dropped.

Figure 2.17 shows how. We learn that a logged value for λ of about $-.4$ leads to the smallest average mean-squared error in the tenfold cross-validation. Looking back at Fig. 2.15, the predictor emotional has been shrunk to 0.0 and plays no role

in the fitted values. On the top margin of the plot, one can see that the number of predictors has been reduced from three to two. A form of model selection has been implemented. The other two predictors remain active with psi still dominant. As the value of λ reaches about 2.5, the affect predictor is dropped as well. In practice, one would likely settle on the two predictors that are not shrunk to 0.0 and then use them in an ordinary least squares analysis. That is, the lasso chooses the predictors, but the analysis meant to inform subject-matter concerns is done with conventional least squares regression. Once a preferred set of regressors is chosen, the motivation for a fitting penalty is far less compelling.

Unfortunately, the lasso does not solve any of the level II difficulties that undermined the level II analysis with ridge regression. The main difference is in the use of an L_1 penalty rather than an L_2 that can make the lasso a useful, variable selection tool. But as before, this creates very difficult problems for estimation, confidence intervals, and statistical tests.

Were one just going to make use of the fitted values, logged $\lambda = -.4$ would produce the best performing results according to the cross-validation mean squared error and ideally, by the true generalization error that it is meant to approximate. There would be no need to revert to ordinary least squares. But all of the level II problems remain.

Although the lasso is certainly a very slick technique, it is unlikely in practice to find the "correct" mean function. At the very least, all of the regressors responsible for nature's true response function would have to be in the data set (how would you know?) and in the real world, combined as the regression mean function specifies (i.e., as a linear combination). In addition, there can be empirical obstacles such as high correlations among the predictors and whether some predictors that should be included contribute sufficiently to the fit after covariance adjustments to be retained. In effect, the predictors that survive are just those having a sufficiently large partial correlation with the response, given the set of predictors being empirically considered.

The lasso has generated an enormous amount of interest among statisticians. Rosset and Zhu (2007) consider the path that the regression coefficients take as the value of λ changes, place the lasso in a class of regularization processes in which the solution path is piecewise linear, and then develop a robust version of the lasso. Wang and colleagues (2007) combine quantile regression with the lasso to derive another robust model selection approach. Zou (2006) has proposed an adaptive version of the lasso when correlations between predictors are high so that unimportant coefficients are shrunk more aggressively. Zou and Hastie (2005) combine the ridge and lasso penalties and call the results "elastic net." Thus,

$$\hat{\beta} = \min_{\beta} \left[\sum_{i=1}^{n} (y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j)^2 + \lambda_1 \sum_{j=1}^{p} |\beta_j| + \lambda_2 \sum_{j=1}^{p} \beta_j^2 \right]. \tag{2.10}$$

Elastic net can earn its keep in settings where the lasso stumbles: when the number of predictors is larger than the number of observations (which is common with

microarray data) and when there are high correlations between predictors. Elastic net is a feature of *glmnet()* in R, and there are some promising extensions available in the R procedure *c060()* (Sill et al. 2014).

But, much of the work on the lasso has been directed toward model selection as an end in itself (Fan and Li 2006; Fan and Lv 2008; Meinshausen and Bühlmann 2006; Bühlmann and van de Geer 2011; Lockhart et al. 2014). There are a host of complications associated with model selection briefly noted in the last chapter. There are also many unsolved problems that can make real applications problematic. But perhaps most important here, a discussion model selection as a freestanding enterprise would take us far afield. Our emphasis will continue to be how to get good fitted values.[16]

In summary, ridge regression and lasso regression introduce the very important idea of penalized fitting and show some ways in which regularization can work. As the value of λ is increases, the regression coefficients are shrunk toward 0.0, and the fitted values become less rough. In the process, bias can be traded against variance with the hope of reducing generalization error in the fitted values. But with our focus on a set of inputs and their fitted values, the lasso is essentially a linear regression and is not even as flexible as regression splines; the linear mean function is very restrictive. We can do a lot better, and we will. The lasso can also be used as a model selection tool, but that is peripheral to our discussion.[17]

However, one point about model selection may be worth making. Before proceeding with model/variable selection, there needs to be ample justification. The dataset should have too many predictors for the number of observations available. Collinearity between predictors should be no worse than modest. And, the coefficients associated with the predictors should be sparse. In practice, whether these conditions are sufficiently met can be very difficult to ascertain.

2.4 Smoothing Splines

For the spline-based procedures considered earlier, the number and location of knots had to be determined a priori or by some measure of fit. We now consider an alternative that does not require a priori knots. A key feature of this approach is to effectively saturate the predictor space with knots and then protect against overfitting by constraining the impact the knots can have on the fitted values. The influence that knots

[16] Should one be interested in model selection per se, there are at least two other visible players within the penalized regression perspective: The Danzig Selector (Candes and Tao 2007; Gareth and Radchenko 2007; Liu et al. 2012) and the Regularization and Derivative Expectation Operator — RODEO — (Lafferty and Wasserman 2008). As mentioned earlier, model selection is sometimes called variable selection or feature selection.

[17] Lasso regularization is an example of "soft thresholding" because the regression coefficients gradually arrive at 0. Backward stepwise regression selects predictors by "hard thresholding" because regression coefficients, or functions of regression coefficients, smaller than some value are abruptly dropped from the analysis.

have can be diluted; the initial number of knots does not have to change but the impact of some can be shrunk to zero. We are proceeding in the same spirit as ridge regression and the lasso, but we are allowing for nonlinear associations between X and Y and introducing a different kind of penalty function.

We begin by returning to the wrong model perspective in which the predictors and the response are random variables. For a single predictor and a quantitative response variable, there is a function $f(X)$ with two derivatives over its entire surface. This is a common assumption in the statistical learning literature and in practice does not seem to be particularly restrictive. The goal is to minimize a penalized error sum of squares of the form

$$\text{RSS}(f, \lambda) = \sum_{i=1}^{N} [y_i - f(x_i)]^2 + \lambda \int [f''(t)]^2 dt, \tag{2.11}$$

where λ is, as before, a tuning parameter. The first term on the right-hand side captures how close the fitted values are to the actual values of y. It is just the usual error sum of squares. The second imposes a cost for the complexity of the fit, much in the tradition of penalized regression, where t is a placeholder for the unknown function. The integral quantifies the roughness penalty, and λ once again determines the weight given to that penalty in the fitting process.[18] At one extreme, as λ increases without limit, the fitted values approach the least squares line. Because no second derivatives are allowed, the fitted values are as smooth as they can be. At the other extreme, as λ decreases toward zero, the fitted values approach an interpolation of the values of the response variable. For a level I analysis, the larger the value of λ, the smoother the representation of the association between X and Y. For a level II analysis, the estimation target is the smoothing splines function with the empirically determined values of λ as a feature of the joint probability distribution. When for a level II analysis one is trying to construct fitted values that usefully approximate the true response surface, if λ is larger, the smoother fitted values will likely lead to more bias and less variance. If λ is smaller, the rougher fitted values will likely lead to less bias and more variance. The value of λ can be used in place of the number of knots to tune the bias-variance tradeoff.

Equation 2.11 can be minimized with respect to the $f(x)$, given a value for λ. Hastie et al. (2009: Sect. 5.4) explain that a unique solution results, based on a set of natural cubic splines with N knots.[19] In particular,

$$f(x) = \sum_{j=1}^{N} N_j(x)\theta_j, \tag{2.12}$$

[18] The second derivative at a given point will be larger the more rapidly the function is changing at that location. The integral is, in effect, the sum of such second derivatives. When the integral is larger, the function is rougher.

[19] There will be fewer knots if there are less than N distinct values of x.

where θ_j is a set of weights, $N_j(x)$ is an N-dimensional set of basis functions for the natural cubic splines being used, and j stands for the number of knots, of which there can be a maximum of N.

Consider the following toy example, in which x takes on values 0 to 1 in steps of .20. In this case, suppose $j = 6$, and Eq. 2.12, written as $f(x) = \mathbf{N}\theta$, then takes the form of

$$f(x) = \begin{pmatrix} -.267 & 0 & 0 & -.214 & .652 & -.429 \\ .591 & .167 & 0 & -.061 & .182 & -.121 \\ .158 & .667 & .167 & -.006 & .019 & -.012 \\ 0 & .167 & 0.667 & .155 & .036 & -.024 \\ 0 & 0 & .167 & .596 & .214 & .024 \\ 0 & 0 & 0 & -.143 & .429 & .714 \end{pmatrix} \begin{pmatrix} \theta_1 \\ \theta_2 \\ \theta_2 \\ \theta_4 \\ \theta_5 \\ \theta_6 \end{pmatrix}. \tag{2.13}$$

Equation 2.11 can be rewritten using a natural cubic spline basis and then the solution becomes

$$\hat{\theta} = (\mathbf{N}^T\mathbf{N} + \lambda\boldsymbol{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y}, \tag{2.14}$$

with $[\boldsymbol{\Omega}_N]_{ij} = \int N_j''(t)N_k''(t)dt$, where the second derivatives are for the function that transforms x into its natural cubic spline basis. $\boldsymbol{\Omega}_N$ has larger values where the predictor is rougher, and given the linear estimator, this is where the fitted values can be rougher as well. The penalty is the same as in Eq. 2.11.

To arrive at fitted values,

$$\hat{\mathbf{y}} = \mathbf{N}(\mathbf{N}^T\mathbf{N} + \lambda\boldsymbol{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y} = \mathbf{S}_\lambda\mathbf{y}, \tag{2.15}$$

where $\mathbf{S}_\lambda$ is a smoother matrix. For a given value of λ, we have a linear estimator of the fitted values.

Equation 2.14 can be seen as a generalized form of ridge regression. With ridge regression, for instance, $\boldsymbol{\Omega}_N$ is an identity matrix. In practice, N is replaced by a basis of B-splines that is used to compute the natural cubic splines.

The requirement of N knots may seem odd because it appears to imply that for a linear estimator all N degrees of freedom are used up. However, for values of λ greater than zero, the fitted values are shunk toward a linear fit, and the fitted values are made more smooth. Less than N degrees of freedom are being used.

As with the number of knots, the value of λ can be determined a priori or through model selection procedures. One common approach is based on N-fold (drop-one) cross-validation, briefly discussed in the last chapter. The value of λ is chosen so that

$$\text{CV}(\hat{f}_\lambda) = \sum_{i=1}^{N}[y_i - \hat{f}_i^{(-i)}(x_i)]^2 \tag{2.16}$$

is as small as possible. In standard notation, $\hat{f}_i^{(-i)}(x_i)$ is the fitted value with case i removed. Using CV to select λ can be automated to find a promising balance between

the bias and the variance in the fitted values. The same reasoning can be used with an in-sample estimate of CV called the generalized cross-validation statistic (GCV), which is computed as,

$$\text{GCV} = \frac{1}{N}\sum_{i=1}^{N}\left[\frac{y_i - \hat{f}(x_i)}{1 - \text{trace}(\mathbf{S})/N}\right]^2, \tag{2.17}$$

where **S** is the smoother (or hat) matrix as before. Whether, the CV or GCV is used, all of the earlier caveats apply. For a level II analysis, we are back in the model selection business with all of its complications. For example, the trace in Eq. 2.17 is not a proper expression for the degrees of freedom expended by the smoother.

2.4.1 A Smoothing Splines Illustration

To help fix all these ideas, we turn to an application of smoothing splines. Figure 2.18 shows four smoothed scatterplots based on Eqs. 2.11 and 2.12. The R code can be found in Fig. 2.19.

Each plot in Fig. 2.18 has the number of users of a particular server (centered) over time.[20] Time is measured in minutes. The penalty weight is *spar*, which can be thought of as a monotonic and standardized function of λ that is ordinarily set at a value from 0 to 1. The data, available in R — see Fig. 2.19 — constitute a time series of 100 observations. Because the documentation in R does not explain how the data were collected, we proceed with a level I regression analysis.[21]

The number of users connected to the server varies dramatically over time in a highly nonlinear manner. Because details of the nonlinearity probably would have been unanticipated, an inductive approach available with smoothing splines is appropriate. For Fig. 2.18, there are four different plots with four different values of *spar*. The quality of the fitted values changes dramatically. But there is no right answer. For a level I analysis, the goal is description. The value of *spar* depends heavily on subject-matter expertise and how the results might be used.

It could be very misleading to automatically choose the value of *spar* by some overall measure of in-sample fit. The fitted values are responding to both signal and noise. For example, is the dip around minute 40 to be taken seriously? It would seem to be "real" when *spar* is .4 or .6, but not when *spar* is .8 or 1. Someone very familiar with the setting in which the data were collected and more generally knowledgeable about patterns of server use would need to make that call.

For a level II analysis, one might want to consider the bias-variance tradeoff and in the absence of test data, set the value of *spar* using some approximation of out-of-sample performance. For these data, the default leave-one-out cross-validation

[20] The plotted fitted values vary around the mean of the response. More will be said about this later.

[21] The procedure *gam()* in the library *gam* was used.

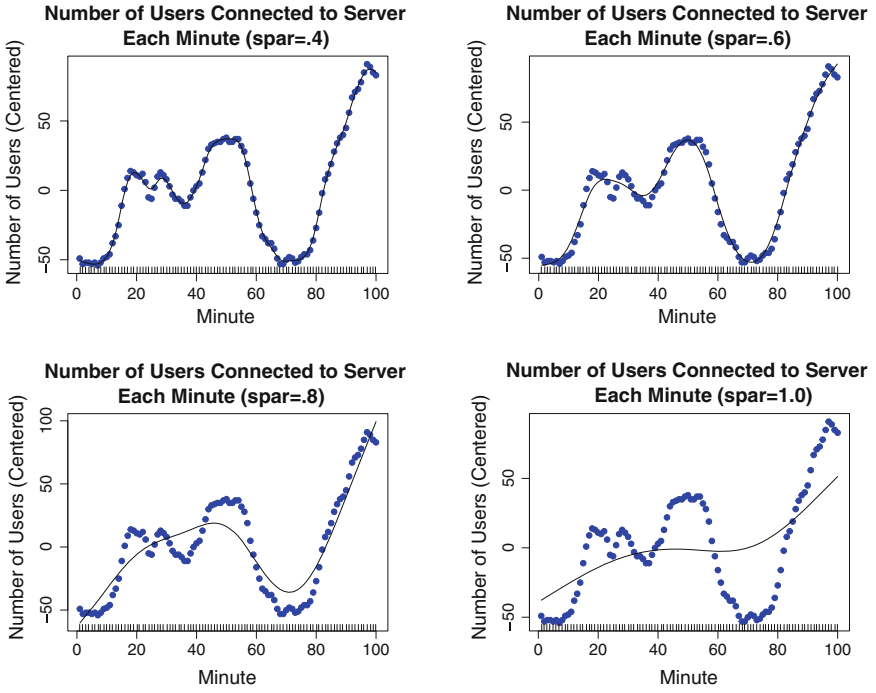

Fig. 2.18 Penalized smoothing splines for use by minute with different penalty weights indexed by the tuning parameter *spar*

selects a value of 8.5. But it is difficult to know what "out-of-sample" means. The data on hand are a sample of exactly what? To what does "out-of-sample" refer? Moreover, cross-validation is probably not a sensible procedure for time series data because cross-validation breaks up the temporal sequence of the data.[22]

More generally, the issues surrounding a level II analysis are largely the same as those addressed for ridge regression and the lasso. If the value of λ (or the equivalent) is chosen before before the data analysis begins, one can proceed as described in Chap. 1 with the response surface approximation perspective. If the value of λ is chosen as part of the data analysis, one is engaging in model selection. With true test data or split samples, there may be a framework in which to proceed, but the longitudinal nature of the data would have to be maintained.[23]

[22] When there is an empirical determination of spar, the *gam* procedure reports the degrees of freedom used, not spar. With a little trial and error, however, one can figure out what the value of spar is.

[23] For example, one might have another dataset for the same 100 min at the same time the next day. If day to day variation is essentially noise, one might well have valid test data.

```
# Data
data(WWWusage)
Minute<-1:100
NumUsers<-WWWusage
internet<-data.frame(NumUsers,Minute)

# Smoothing Splines
library(gam)
par(mfrow=c(2,2))
out<-gam(NumUsers~s(Minute, spar=.4),data=internet)
plot(out,xlab="Minute",ylab="Number of Users (Centered)",
     main="Number of Users Connected to Server
     Each Minute (spar=.4)", col="blue", pch=19,
     residuals=T)

out<-gam(NumUsers~s(Minute, spar=.6),data=internet)
plot(out,xlab="Minute",ylab="Number of Users (Centered)",
     main="Number of Users Connected to Server
     Each Minute (spar=.6)",col="blue",pch=19,
     residuals=T)

out<-gam(NumUsers~s(Minute, spar=.8),data=internet)
plot(out,xlab="Minute",ylab="Number of Users (Centered)",
     main="Number of Users Connected to Server
     Each Minute (spar=.8)",col="blue",pch=19,
     residuals=T)

out<-gam(NumUsers~s(Minute, spar=1.0),data=internet)
plot(out,xlab="Minute",ylab="Number of Users (Centered)",
     main="Number of Users Connected to Server
     Each Minute (spar=1.0)",col="blue",pch=19,
     residuals=T)
```

Fig. 2.19 R code for penalized smoothing splines

2.5 Locally Weighted Regression as a Smoother

Thus far, the discussion of smoothing has been built upon a foundation of conventional linear regression. Another approach to smoothing also capitalizes on conventional regression, but through nearest neighbor methods. We start with those.

2.5.1 Nearest Neighbor Methods

Consider Fig. 2.20 in which the ellipse represents a scatter plot of points for values for X and Y. There is a target value of X, labeled x_0, for which a conditional mean $\bar{y}_0$ is to be computed. There may be only one such value of X or a relatively small number of such values. As a result, a conditional mean computed from those values alone risks being very unstable. One possible solution is to compute $\bar{y}_0$ from observations with values of X close to x_0. The rectangle overlaid on the scatterplot illustrates a region of "nearest neighbors" that might be used. Insofar as the conditional means for Y are not changing systematically within that region, a useful value for $\bar{y}_0$ can be obtained. For a level I description, the conditional mean is a good summary for Y derived from the x-values in that neighborhood. If that conditional mean is to be used as an estimate in a level II analysis of the true response surface, it will be unbiased and likely be more stable than the conditional mean estimated only for the observations with $X = x_0$. In practice, however, some bias is often introduced because Y actually does vary systematically in the neighborhood. As before, one hopes that the increase in the bias is small compared to the decrease in the variance.

A key issue is how the nearest neighbors are defined. One option is to take the k closest observations using the metric of X. For example, if X is age, x_0 is 24 years old, and k is 10, the ten closest x-values might range from 23 to 27 years of age. Another option is take some fixed fraction f of the observations that are closest to x_0. For example, if the closest 25 % of the observations were taken, k might turn out to be 30, and the age-values might range between 21 and 29. Yet another option is to vary either k or f depending on the variability in Y within a neighborhood. For example, if there is more heterogeneity that is likely to be noise, larger values of k or f can be desirable to improve stability. For any of these approaches, the neighborhoods will likely overlap for different target values for X. For another target value near x_0, some near neighbors will likely be in both neighborhoods. There also is no requirement that the neighborhood be symmetric around x_0.

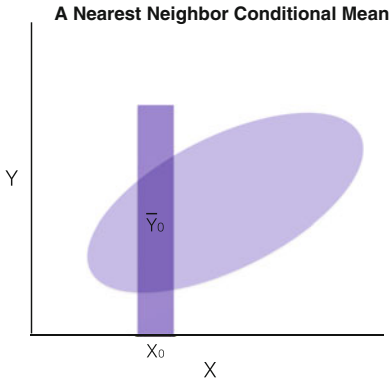

Fig. 2.20 A conditional mean $\bar{Y}_0$ for X_0, a target value of X

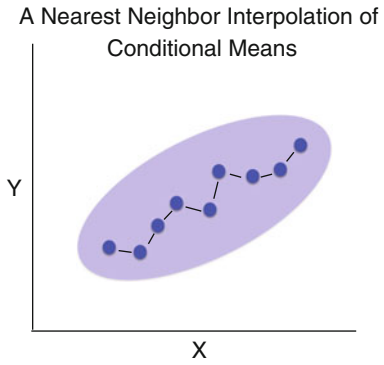

Fig. 2.21 Interpolated conditional means

Suppose now that for each unique value of x, a nearest neighbor conditional mean for y is computed using one of the approaches just summarized. Figure 2.21 shows a set of such means connected by straight lines. The pattern provides a visualization of how the means of y vary with x. As such, the nearest neighbor methods can been seen as a smoother.

The values k or f are often referred to as the "bandwidth," "window," or "span" of a neighborhood. The larger the values of k or f, the larger the size of the neighborhood, and Fig. 2.21 will change in response. Larger neighborhoods will tend to make the smoothed values less variable. If the smoothed values are to be treated as level II estimates of the true response surface, they will likely be more biased and more stable. Smaller neighborhoods will tend to make the smoothed values more variable. If the smoothed values are to be treated as level II estimates of the true response surface, they will likely be less biased and less stable.

2.5.2 *Locally Weighted Regression*

Nearest neighbor methods can be effective in practice and have been elaborated in many ways (Ripley 1996; Shakhnarovich 2006). In particular, what if within each neighborhood the conditional means of Y vary systematically? At the very least, there is information being ignored that could improve the estimate of $\bar{y}_0$.

Just as in conventional linear regression, if Y is related to X in a systematic fashion, there can be less variation in the regression residuals than around the neighborhood mean of Y. More stable estimates can follow. The idea of applying linear regression within each neighborhood leads to a form of smoothing based on locally weighted regressions. The smoother commonly is known as "lowess".[24]

[24]Lowess is sometimes said to stand for locally weighted scatter plot smoothing. But Cleveland, who invented the procedure (Cleveland 1979), seems to prefer the term "local regression" known as "loess" (Cleveland 1993: 94).

We stick with the one predictor case a bit longer. For any given value of the predictor x_0, a polynomial regression is constructed only from observations with x-values that are nearest neighbors of x_0. Among these, observations with x-values closer to x_0 are weighted more heavily. Then, $\hat{y}_0$ is computed from the fitted regression and used as the smoothed value of the response y at x_0. The process is repeated for all other values of x.

Although the lowess polynomial is often of degree one (linear), quadratic and cubic polynomials are also used. It is not clear that much is gained in practice using the quadratic or cubic form. In some implementations, one can also employ a degree zero polynomial, in which case no regression is computed, and the conditional mean of y in the neighborhood is used as $\hat{y}_0$. This is just the nearest neighbor approach except for the use of distance weighting. Perhaps surprisingly, the lowess estimator is linear for a given value of k or f (Hastie et al. 2009; Sect. 6.1.1).

The precise weight given to each observation depends on the weighting function employed. The normal distribution is one option. That is, the weights form a bell-shaped curve centered on x_0 that declines with distance from x_0. The tricube is another option. Differences between x_0 and each value of x in the window are divided by the length of the window along x. This standardizes the differences. Then the differences are transformed as $(1 - |z|^3)^3$, where z is the standardized difference. Values of x outside the window are given weights of 0.0. As an empirical matter, most of the common weighting functions give about the same results, and there seems to be no formal justification for any particular weighting function.

As discussed for nearest neighbor methods, the amount of smoothing depends on the value of k or f. For f, proportions between .25 and .75 are common. The larger the proportion of observations included, the smoother are the fitted values. The span plays the same role as the number of knots in regression splines or λ in smoothing splines. Some software also permits the span to be chosen in the units of the regressor. For example, if the predictor is population size, the span might be defined as 10,000 people wide.

More formally, each local regression at each x_0 is constructed by minimizing the weighted sum of squares with respect to the intercept and slope for the $M \leq N$ observations included in the window. Thus,

$$\text{RSS}^*(\boldsymbol{\beta}) = (\mathbf{y}^* - \mathbf{X}^*\boldsymbol{\beta})^T \mathbf{W}^* (\mathbf{y}^* - \mathbf{X}^*\boldsymbol{\beta}). \tag{2.18}$$

The asterisk indicates that only the observations in the window are included. The regressor matrix $\mathbf{X}^*$ can contain polynomial terms for the predictor, should that be desired. $\mathbf{W}^*$ is a diagonal matrix conforming to $\mathbf{X}^*$, with diagonal elements w_i^*, which are a function of distance from x_0. This is where the weighting-by-distance gets done.

The overall algorithm then operates as follows.

1. Choose the smoothing parameter such as bandwidth, f, which is a proportion between 0 and 1.
2. Choose a point x_0 and from that the ($f \times N = M$) nearest points on x.

3. For these M nearest neighbor points, compute a weighted least squares regression line for y on x.
4. Construct the fitted value $\hat{y}_0$ for that single x_0.
5. Repeat Steps 2 through 4 for each value of x. Near the boundary values of x, constraints are sometimes imposed much like those imposed on cubic splines and for the same reasons.
6. To enhance visualization, connect adjacent $\hat{y}$s with straight lines.

There is also a robust version of lowess. After the entire fitting process is completed, residuals are computed in the usual way. Weights are constructed from these residuals. Larger residuals are given smaller weights and smaller residuals larger weights. Using these weights, the fitting process is repeated. This, in turn, can be iterated until the fitted values do not change much (Cleveland 1979) or until some predetermined number of iterations is reached (e.g., three). The basic idea is to make observations with very large residuals less important in the fitting.

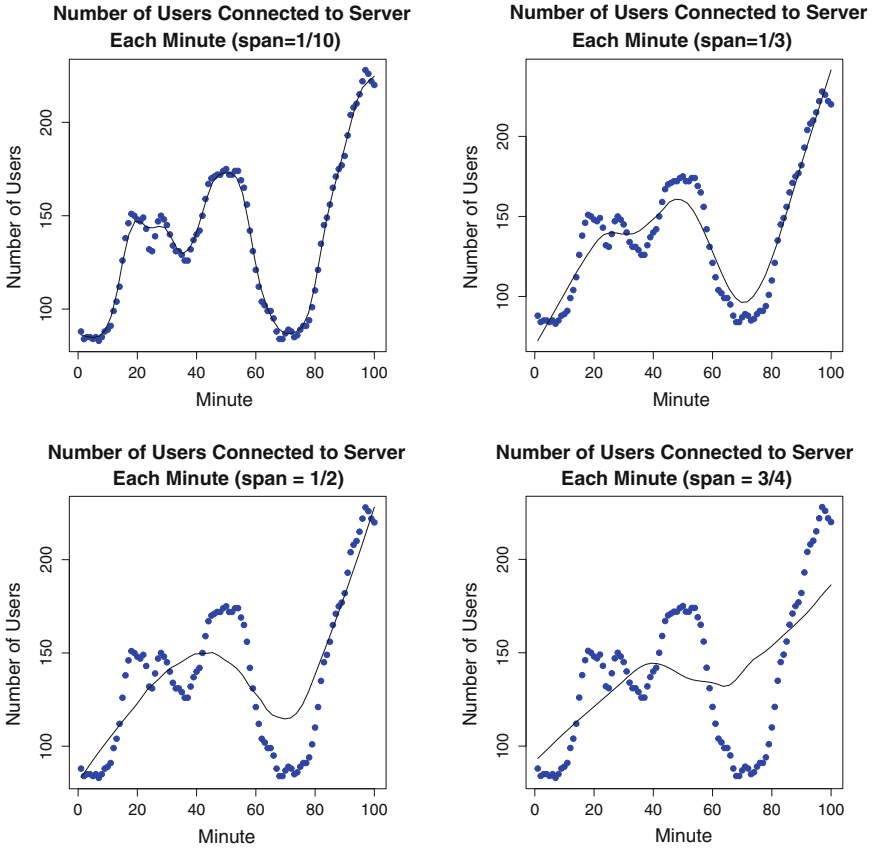

Fig. 2.22 Lowess smoothing for use by minute for different spans

Whether the "robustification" of lowess is useful will be application-specific and depend heavily on the window size chosen. Larger windows will tend to smooth the impact of outlier residuals. But because the scatterplot being smoothed is easily plotted and examined, it is usually easy to spot the possible impact of outlier residuals and, if necessary, remove them or take them into account when the results are reported. In short, there is no automatic need for the robust version of lowess when there seem to be a few values of the response that perhaps distort the fit.

Just as with penalized smoothing splines, a level I analysis is descriptive. A level II analysis can entail estimation of a response surface approximation of the same form and with the same values of the tuning parameters as used for the training data. That approximation can also be used to provide estimates of the true response surface that are subject to the same sort of bias-variance tradeoffs discussed earlier.

2.5.2.1 A Lowess Illustration

Figure 2.22 repeats the earlier analysis of server use, but applies lowess rather than smoothing splines. The results are much the same over a set of different spans. The fraction for each reported span is the proportion of observations that define a given neighborhood. (See Fig. 2.23 for the R code.)

```
# Data
data(WWWusage)
Minute<-1:100
NumUsers<-WWWusage

# Lowess
par(mfrow=c(2,2))
scatter.smooth(Minute,NumUsers,xlab="Minute",ylab="Number of
     Users", main="Number of Users Connected to Server
     Each Minute (span=1/10)", span=1/10,pch=19,col="blue")
scatter.smooth(Minute,NumUsers,xlab="Minute",ylab="Number of
     Users", main="Number of Users Connected to Server
     Each Minute (span=1/3)", span=1/3, pch=19, col="blue")
scatter.smooth(Minute,NumUsers,xlab="Minute",ylab="Number of
     Users", main="Number of Users Connected to Server
     Each Minute (span = 1/2)", span=1/2, pch=19, col="blue")
scatter.smooth(Minute,NumUsers,xlab="Minute",ylab="Number of
     Users", main="Number of Users Connected to Server
     Each Minute (span = 3/4)", span=3/4, pch=19,col="blue")
```

Fig. 2.23 R code lowess smooth

Figure 2.22 was produced in R by *scatter.smooth()*.[25] One can also proceed with *loess()*, which has more options and separates the plotting from the fitting. Both procedures require that the span (or an equivalent turning parameter) be hard coded although there have been proposals to automate the tuning, much as done for smoothing splines (Loader 2004: Sect. 4).

In summary, lowess provides a good, practical alternative to smoothing splines except that the span is not determined automatically (at least in R). Otherwise, it has pretty much the same strengths and weaknesses, and performance will be similar. For example, the same issues arise about whether the dip in use at about minute 40 is "real." Or even if it is, whether the dip is substantively or practically important. The lower lefthand plot in Fig. 2.22 may be the most instructive rendering for these data.

2.6 Smoothers for Multiple Predictors

The last set of figures is only the most recent example in which the limitations of a single predictor are apparent. Many more things could be related to server use than time alone. We need to consider smoothers when there is more than one predictor.

In principle, it is a simple matter to include many predictors and then smooth a multidimensional space. However, there are three significant complications in practice. The first problem is the curse of dimensionality addressed in the last chapter. As the number of predictors increases, the space the data need to populate increases as a power function. Consequently, the demand for data increases very rapidly, and one risks data that are far too sparse to produce a meaningful fit. There are too few observations, or those observations are not spread around sufficiently to provide the support needed. One must, in effect, extrapolate into regions where there is little or no information. To be sensible, such extrapolations would depend on knowing the $f(X)$ quite well. But it is precisely because the $f(X)$ is unknown that smoothing is undertaken to begin with.

The second problem is that there are often conceptual complications associated with multiple predictors. In the case of lowess, for example, how is the neighborhood near x_0 to be defined (Fan and Gijbels 1996: 299–300)? One option is to use Euclidian distance. But then the neighborhood will depend on the units in which predictors happen to be measured. The common practice of transforming the variables into standard deviation units solves the units problem, but introduces new problems. When does it make substantive sense to claim that two observations that are close in standard deviations are close in subject-matter units?

[25] The plots produced by *scatter.smooth()* are not centered around the mean of the response, but whether the plot is centered or not should have no effect on an interpretation of the relationship between Y and X unless variation relative to the mean of Y matters. It also might if the actual values of the approximate response surface were important (e.g., in forecasting).

Consider a simple case of two predictors. Suppose the standard deviation for one predictor is five years of age, and the standard deviation for the other predictor is two years of education. Now suppose one observation falls at x_0's value of education, but is five years of age higher than x_0. Suppose another observation falls at x_0's value for age, but is two years higher in eduction than x_0. Both are one standard deviation unit away from x_0 in Euclidian distance. But do we really want to say they are equally close to x_0?

Another approach to neighborhood definition is to use the same span (e.g., .20) for both predictors, but apply it separately in each direction. Why this is a better definition of a neighborhood is not clear. And one must still define a distance metric by which the observation in the neighborhood will be weighted.

The third problem is that gaining meaningful access to the results is no longer straightforward. When there are more than two predictors, one can no longer graph the fitted surface in the usual way. How does one make sense of a surface in more than three dimensions?

2.6.1 Smoothing in Two Dimensions

Given the problems just summarized, the step from a single predictor to two predictors can be challenging. But there are extensions of the single-predictor setting that can work well and that introduce some tools that will be important as we move to applications with more than two predictors. We will proceed drawing heavily on an empirical example.

A key issue in the study of climate is cyclical variation in rainfall for areas on the Pacific rim. An important driver is measured by the Southern Oscillation Index (SOI), which is the difference in sea level barometric pressure between Tahiti and Darwin, Australia. Negative values are associated with a phenomenon called "El Niño." Positive values are associated with a phenomenon called "La Niña." Both are connected to patterns of rainfall in ways that are not well understood.

The illustrative data we will use has 101 observations. Predictors are year from 1900 to 2001 and the average yearly SOI. Average yearly rainfall in Australia is the response. The data are shown in Fig. 2.24, and the code is shown in Fig. 2.25.[26]

Much as in the case of conventional multiple linear regression, the goal is to fit a surface to the data. Rainfall is the response. Year and SOI are the predictors. Unlike conventional linear regression, no model is imposed. In particular, the correct fit is not assumed to be a plane produced by a linear combination of year and SOI. A smoothing splines formulation for a single predictor is applied, but the single predictor is the product of year and SOI, just as one might represent an interaction effect between the two. The product variable is smoothed and then plotted in the 2-dimensional predictor space.

[26]The data can be obtained from the *DAAG* library under the name *bomsoi*.

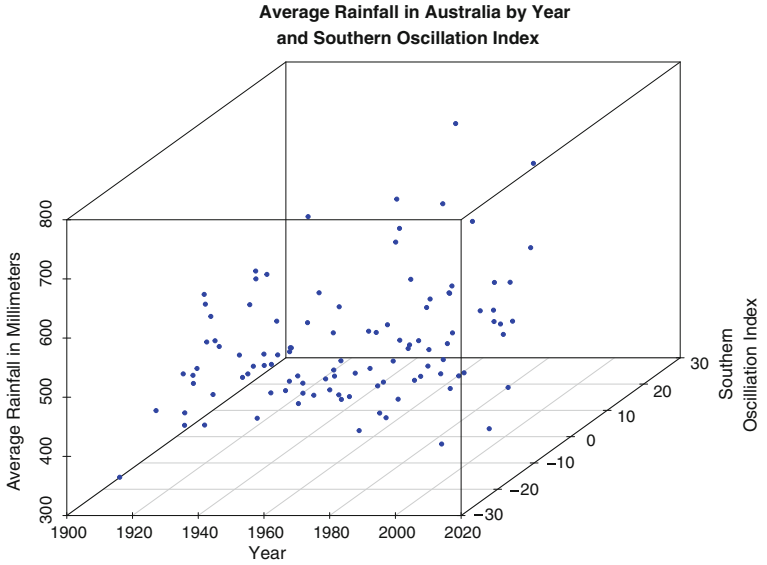

Fig. 2.24 A 3-dimensional scatter plot of rainfall in Australia as a function year and the southern oscillation index

```
library(DAAG)
data(bomsoi)
attach(bomsoi)
library(scatterplot3d)
scatterplot3d(Year,SOI,avrain,xlab="Year", ylab="Southern
              Oscilliation Index",zlab="Average Rainfall in
              Millimeters", main="Average Rainfall in Australia
              by Year and Southern Oscillation Index",pch=19,
              color="blue")
```

Fig. 2.25 R code for the 3-D scatter plot

Figure 2.26 shows the result for the Australian rainfall data from 1900 to 2001.[27] Values of the tuning parameter range from .2 to 1.0. As before, larger values of spar produce smoother surfaces. Any one of the values (or some other) could be preferred,

[27]The procedure used was *gam* in the *gam* library. There is an alternative implementation of *gam* in the *mgcv* library, written by Simon Wood, that has a very large number of somewhat daunting options. For example, there are several rather different ways to smooth a nonadditive function of two predictors. Documentation is extensive, but challenging in spots. Fortunately, the defaults seem to work well and yielded much the same results as shown in Fig. 2.26.

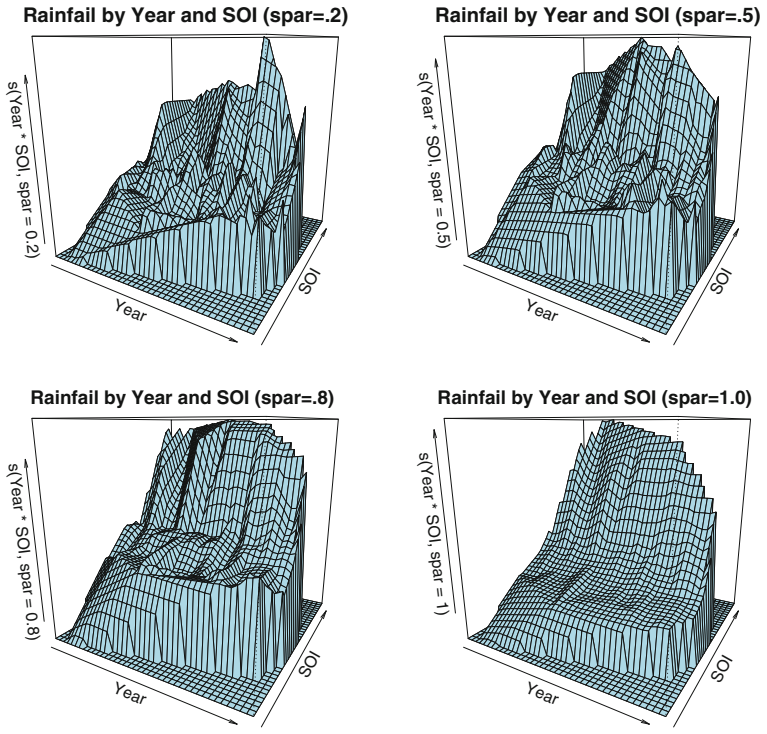

Fig. 2.26 3-Dimensional scatter plot of rainfall in Australia as a function of year and the southern oscillation index

but a spar-value of 1.0 was chosen using leave-one-out cross-validation. The label on the vertical axis shows the expression by which the fitted values were computed.[28]

Within a level I analysis, there is a modest increase in rainfall over the time period with most of that increase occurring early. Rainfall increases when the SOI is larger, but the relationship is highly nonlinear. The association is especially pronounced for larger values of the SOI. For some smaller values of spar, the surface is "torqued." The relationship between SOI and rainfall is substantially stronger in some years than others. For example, with spar = .2, the strongest association is in the 1980s. However, with only 101 observations spread across the 2-dimensional predictor space, there are relatively few observations behind such results. That may be reason enough to

[28]The use of the $*$ operator means multiplication. (See Fig. 2.27.) Year is multiplied by SOI. Currently, the plotting procedure "knows" when it sees $*$ that the plotting surface should be in 2-D predictor space. If one does the multiplication in advance and uses the product as a predictor, *gam* produces the same fit and summary statistics. But the plot treats the product as the vector it is. The procedure does not "know" that the vector is the product of two predictors. Figure 2.26 shows the results as perspective plots. Contour plots would convey much the same information with more numerical detail but in a less easily understood visual format. Computations in *gam* are undertaken using cubic smoothing splines with B-splines the behind-the-scenes workhorse as usual.

```
library(DAAG)
data(bomsoi)
attach(bomsoi)
llbrary(gam)
library(akima)
par(mfrow=c(2,2))
out1<-gam(avrain~s(Year*SOI,spar=.2),data=bomsoi,
                  family=gaussian)
plot(out1,theta=30, main="Rainfail by Year and SOI (spar=.2)",
       col="light blue")
out2<-gam(avrain~s(Year*SOI,spar=.5),data=bomsoi,
                  family=gaussian)
plot(out2, theta=30, main="Rainfail by Year and SOI (spar=.5)",
       col="light blue")
out3<-gam(avrain~s(Year*SOI,spar=.8),data=bomsoi,
                  family=gaussian)
plot(out3, theta=30, main="Rainfail by Year and SOI (spar=.8)",
      col="light blue")
out4<-gam(avrain~s(Year*SOI,spar=1),data=bomsoi,
                  family=gaussian)
plot(out4, theta=30, main="Rainfall by Year and SOI (spar=1.0)",
       col="light blue")
```

Fig. 2.27 R code for 3-Dimensional smooth of Australian rainfall data as a function of year and the southern oscillation index

prefer the fitted values with spar = 1.0 and provides a small-scale object lesson about the curse of dimensionality.

Much as in several earlier analyses, a level II analysis would be very challenging. The data are longitudinal with year as one of the predictors. In the same fashion as the Tokyo water use data, the data collection conditions on year. As a result, the data may be best considered as random realizations from a set of conditional distributions. But then, we still have to rely on a theory of how nature is able to repeat the rainfall patterns for a given set of years. Perhaps this could be worked out by researchers steeped in climate science, but for now at least, no level II analysis will be attempted.

2.6.2 The Generalized Additive Model

Moving beyond two predictors usually requires a different strategy. A more practical and accessible means needs to be found to approximate a response surface when the predictor space is greater than two. One approach is to resurrect an additive formulation that in practice can perform well.

The Generalized Additive Model (GAM) is superficially an easy extension of the Generalized Linear Model (GLM). GAM tries to circumvent the curse of dimensionality by assuming that the conditional mean of the response is a linear combination of functions of the predictors. Thus, the generalized additive model with p predictors can be written as

$$Y = \alpha + \sum_{j=1}^{p} f_j(\mathbf{X}_j) + \varepsilon, \tag{2.19}$$

where α is fixed at the mean of Y. We once again minimize the penalized regression sum of squares (PRSS) but with respect to all the p f_j's (Hastie et al. 2009: 297):

$$\text{PRSS}(\alpha, f_1, f_2, \ldots, f_p) = \sum_{i=2}^{N} \left(y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij}) \right)^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t_j)^2 dt_j. \tag{2.20}$$

Equation 2.20 is a generalization of single-predictor smoothing splines that allows for a different value of λ_j for each function in the linear combination of functions; there are p values for λ that need to be specified in advance or more typically, determined as part of the data analysis. The pth second derivatives correspond to the pth function only.[29]

In the same manner as the generalized linear model, the generalized additive model permits several different link functions and disturbance distributions. For example, with a binary response, the link function can be the log of the odds (the "logit") of the response, and the disturbance distribution can be logistic. This is analogous to logistic regression within the generalized linear model. But, there are no regression coefficients associated with the predictors. Regression coefficients would just scale up or scale down the functions of predictors. Whatever impact they would have is absorbed in the function itself. In other words, the role of the regression coefficients cannot be distinguished from the role of the transformation and therefore, the regression coefficients are not identified.

Each predictor can have its own functional relationship to the response. Because these functions are usually estimated using single-predictor smoothers of the sort addressed earlier, the term nonparametric is commonly applied despite the a priori commitment to an additive formulation. Alternatively, all of the functions may be specified in advance with the usual linear model as a special case.

All of the common regression options are available, including the wide range of transformations one sees in practice: logs, polynomials, roots, product variables (for interaction effects), and indicator variables. As a result, GAM can be parametric as well and in this form, is really no different from the generalized linear model. The parametric and nonparametric specifications can be mixed so that some of the

[29] The notation t_j is a placeholder for the unknown jth function.

functions are derived empirically from the data, and some are specified in advance. Then the model is often called semiparametric.

One can use for GAM the same conception of "holding constant" that applies to conventional linear regression. Suppose that for a conventional regression analysis each of the predictors is transformed in a known manner. With least squares, each transformed predictor is covariance adjusted; the relationship between a given transformed predictor and the response is determined with the linear dependence between that transformed predictor and all other transformed predictors removed. One would like to do the same thing when each transformation is not known. But there can be no covariance adjustments until the transformations are determined, and there can be no transformations until each predictor is covariance adjusted. The backfitting algorithm provides a solution.

2.6.2.1 A GAM Fitting Algorithm

The backfitting algorithm is a common way to estimate the functions and coefficient α in Eq. 2.20 (Hastie et al. 2009: Sect. 9.1.1) using the following steps.

1. Initialize with $\hat{\alpha} = \frac{1}{N}\sum_1^N y_i$, $\hat{f}_j \equiv 0, \forall i, j$. Each function is given initial values of 0.0, with α fixed at the mean of y.
2. Cycle: $j = 1, \ldots, p, 1, \ldots, p, \ldots,$

$$\hat{f}_j \leftarrow S_j \left[\{y - \hat{\alpha} - \sum_{k \neq j} \hat{f}_k(x_{ik})\}_1^N \right].$$

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

   Fitted values from all predictors but predictor j are linearly combined and subtracted from the response. A smoother S_j is applied to the resulting "residuals." The result is a new set of fitted values for predictor j. These fitted values are then centered. All of the other predictors are cycled through one at a time in the same manner until each of the p predictors has a revised set of fitted values.
3. Repeat Step 2 until $\hat{f}_j$ changes less than some small, pre-determined amount. In the process, adjustments are made for nonlinear dependence between predictors.

The backfitting algorithm is quite general and quite fast. A wide variety of smoothers can be applied and in the past have been. For example, both lowess and penalized smoother splines are available in R. A range of smoothing basis functions can be employed as well (Wood 2006: Sect. 4.1). Some procedures also permit the use of functions of two predictors at a time, so that the smoothed values represent a surface rather than a line, just as in Fig. 2.26; one can work with a linear combination

of bivariate smoothed values. Excellent implementations in R include the procedure *gam()* in the *gam* library and the procedure *gam()* in the *mgcv* library.[30]

So what's not to like? The linear combination of smooths is not as general as a smooth of an entire surface, and sometimes that matters. Looking back at the El Niño example, the response surface was "torqued." The nonlinear function along one predictor dimension varied by the values of the other predictor dimension. This is a generalization of conventional interaction effects for linear regression in which the slope for one predictor varies over values of another predictor. The generalized additive model does not allow for interaction effects unless they are built into the mean function as product variables; interaction effects are not arrived at inductively. This is no different from conventional linear regression.

2.6.2.2 An Illustration Using the Generalized Additive Model

Although population counts from the U.S. census are highly accurate, they are certainly not perfect. In the absence of perfection, small differences between the actual number of residents in an area and the counted number of residents in that area can have very important consequences for definitions of voting districts, the number of elected representatives a county or state can have, and the allocations of federal funds. Beginning with the 1980 U.S. census, there were particular concerns about population undercounts in less affluent, minority-dominated voting districts.

The data we will now use come for a study by Ericksen, Kadane and Tukey (1989) that sought correlates of census undercounts. Sixty-six geographical areas were included, 16 being large cities. The other geographical units were either the remainder of the state in which the city was located or other states entirely. Sampling was purposive.

We use the following variables:

1. Undercount — the undercount as a percentage of the total count;
2. Minority — the percentage of residents who are Black or Hispanic;
3. Language — the percentage of residents who have difficulty with English;
4. Housing — the percentage of residential buildings that is small and multi-unit; and
5. Crime — reported crimes per 1000 residents.

The first variable is the response. The others are predictors thought to be related to census undercounts. No doubt there are any number of other predictors that subject-matter experts would claim should have been included. For example, we know nothing about the census enumeration procedures or the race and gender of enumerators.

[30]There are some important differences in the computational details, output, plotting facility, and ease of use, many of which boil down to personal taste. From Hastie et al. (2009), it is natural to work with the *gam* library. From Wood (2006), it is natural work with the *mgcv* library. Both references provide important background because the documentation in R is not meant to teach the material and makes reference to terms and concepts that may be unfamiliar.

We also know nothing about any efforts by local authorities to encourage residents to cooperate. By conventional criteria, the mean function is misspecified.

Using the procedure *gam()* in the library *mgcv*, we applied the generalized additive model to the data. An examination of the tabular output indicated that nearly 80 % of the deviance was accounted for. There was also lots of other information, much like that provided for conventional linear regression, and if there had been predictors that were not smoothed (e.g., factors), their regression coefficients would have been displayed. For each smoothed predictor, the effective degrees of freedom was reported, although as noted earlier, an inductively determined value of λ makes those values suspect. Still, large values for the effective degrees of freedom indicate that a function is more complex.

Figure 2.28 shows how the predictors are related to the response. (See Fig. 2.29 for the R code.) For each graph, the response is centered around $\hat{\alpha}$, and there are rug plots just above the horizontal axes. Also shown are the response values after

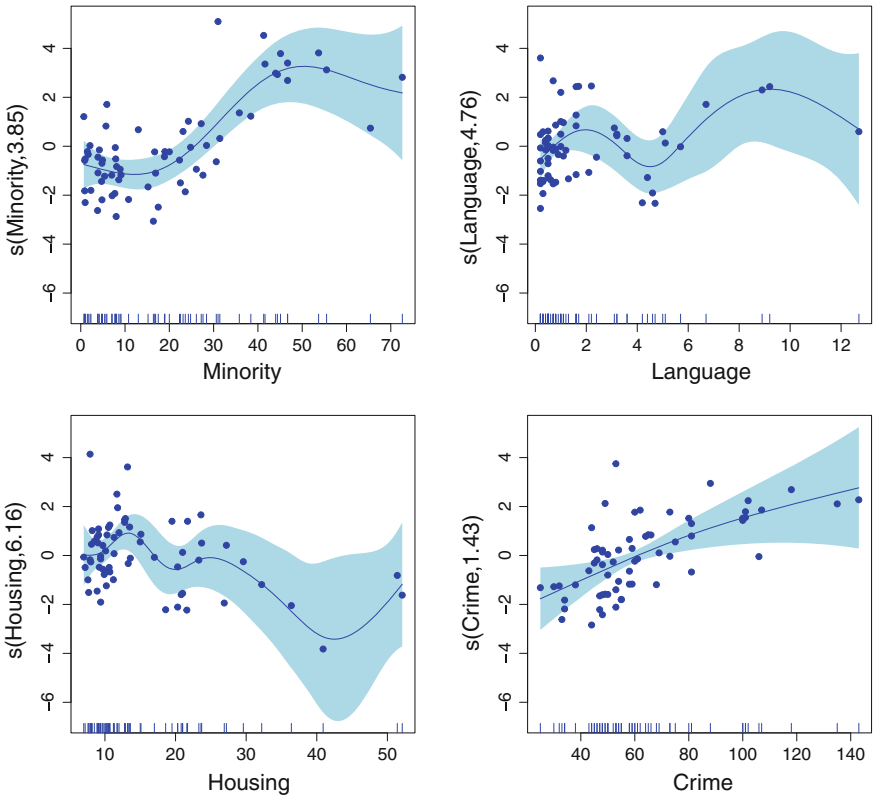

Fig. 2.28 Correlates of the estimated undercount percentage for the U.S. 1980 census (The predictor is on each *horizontal* axis, the centered fitted values are on each *vertical* axis, the *shaded* areas are error bands, rug plots are shown, and $N = 66$.)

```
library(car)
data(Ericksen)
library(mgcv)

# make better variable names plotting
attach(Ericksen)
City<-as.numeric(ifelse(city=="city",1,0))
Minority<-minority
Language<-language
Housing<-housing
Crime<-crime
Undercount<-undercount
temp<-data.frame(Undercount,Crime,Housing,Language,Minority,City)

# GAM from mgcv
out<-gam(Undercount~s(Minority)+s(Language)+s(Housing)+
         s(Crime)+City,data=temp,family=gaussian)
par(mfrow=c(2,2))
plot(out,residual=T,cex=1,pch=19,shade=T,
     shade.col="light blue",col="blue")
```

Fig. 2.29 R code for undercount analysis

adjustments consistent with the earlier discussion of holding constant. The vertical axis label includes the effective number of degrees of freedom used by the particular smoothed function, determined by tuning the equivalent of λ_j with the GCV statistic. For example, there are a little more than six degrees of freedom used by the housing variable and a little more than one degree of freedom used by the crime variable. The former is highly nonlinear. The latter is very nearly linear. The four values for effective degrees of freedom (or alternatively, spar) were determined by an automated search over values of the generalized cross-validation statistic.

There are also shaded areas representing plus and minus two standard errors for the fitted values. Were one doing a level II analysis, they are supposed to convey uncertainty in the estimates. But it is difficult to know what to make of this rendering of uncertainty. Constant disturbance variance is assumed, there is almost certainly bias in the estimated fitted values (Wood 2006, Sect. 4.4.1), and model selection by the GCV statistic makes any conventional level II analysis problematic. As discussed in the first chapter, the consequences of model selection are far more weighty than just some additional uncertainty to contend with. Perhaps the major take-away is that the "error bands" widen dramatically where the data are most sparse. Fitted values in those regions need very careful scrutiny and perhaps have no interpretive value.

In general, all four predictors show positive relationships with the size of the undercount in regions where the data are not sparse. For example, the relationship with the percentage of residents having problems with English is positive until the value

exceeds about 2 %. The relationship then becomes negative until a value of about 5 % when the relationship turns positive again. But there is no apparent substantive explanation for the changes in the slope, which are based on very few observations. The twists and turns could be the result of noise or neglected predictors, but with so little data, very little can be concluded. The changing width of the shaded area makes the same point.

Do any of the estimated relationships matter much? The observed values for the undercount can serve as an effective benchmark. They range from a low of −2.3 % to a high of 8.2 %. Their mean is 1.9 %, and their standard deviation is 2.5 %. Where the data are not sparse, each predictor has fitted values that vary by at least 1 percentage point. Whether changes variable by variable of a percentage point or two matter would need to be determined by individuals with subject-matter expertise. But perhaps more telling is what happens when the four fitted relationships are combined in a linear fashion to arrive at fitted values. Fitted values range from little below −1 % to about 7.5 %, and as Fig. 2.30 shows, there is a substantial number of areas with fitted undercounts greater than 4 %. Moreover, the distribution tends to decline over positive values up to about 4 %, after which there is an uncharacteristic increase. One might have expected continuing declines in the right tail. At least from a policy point of view, it might be instructive to know which geographical areas fall on the far right of the histogram. In short, insofar as the observed variability in the undercount matters, so does the variability in the fitted values.

Ignoring the data snooping for a moment, does one have the makings of a level II analysis? One would need clarification on what the estimation target is. In addition,

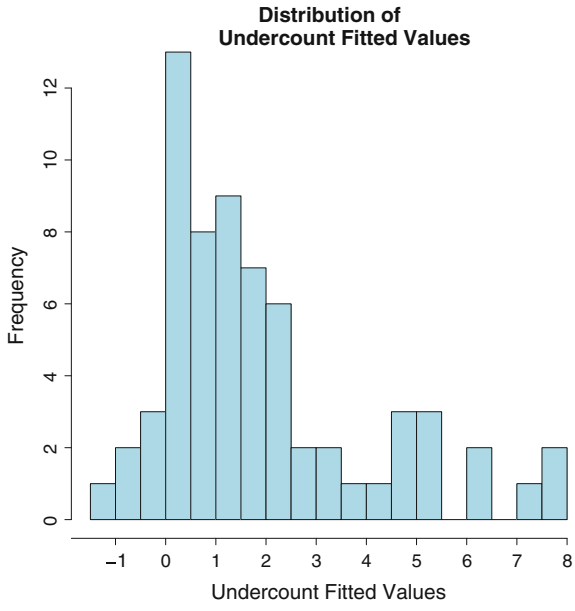

Fig. 2.30 Histogram of the undercount percentage fitted values from the GAM procedure (N = 66)

the data are not a probability sample of anything, and one would be hard pressed to provide some reasonable joint probability distribution responsible for the data. These present significant challenges before one gets to the data analysis.

2.7 Smoothers with Categorical Variables

Smoothers can be used with categorical variables. When a predictor is categorical, however, there is really nothing to smooth. A binary predictor can have only two values. The "smoother" is then just a straight line connecting the two conditional means of the response. For a predictor with more than two categories, there is no way to order the categories along the predictor axis. Any imposed order would imply assigning numbers to the categories. How the numbers were assigned could make an enormous difference in the resulting fitting values, and the assigned numbers necessarily would be arbitrary. Consequently, the categories are reconfigured as indicator variables.

When the response is categorical and binary, smoothing can be a very useful procedure. All of the earlier benefits apply. In addition, because it is very difficult to see much in a scatterplot with a categorical response, a smoother may be the only way to gain some visual leverage on what may be going on. However, the earlier caveats apply too.

Within the generalized additive model (GAM), the analysis of binary response variables can be seen as an extension of binomial regression from the generalized linear model (GLM). The right hand side is a linear combination of predictor functions. The left hand side is the response transformed by a link function to logit units (i.e., the log of the odds). What is different is that the functions of each predictor are unknown.

2.7.1 An Illustration Using the Generalized Additive Model with a Binary Outcome

We consider again the low birthweight data, but now the response is binary: low birthweight or not. A birthweight is low when it is less than 2.5 kg. The following predictors are used:

1. Age — in years;
2. Mother's weight — in pounds;
3. Uterine — presence of uterine irritability; and
4. Smoke — whether the mother smokes.

Using the defaults in *gam* (in the library *mgcv*), about 9 % of the deviance can be attributed to the four predictors. Consider first the role of the two binary predictors. When the mother smokes, the odds of a low birthweight baby are multiplied by

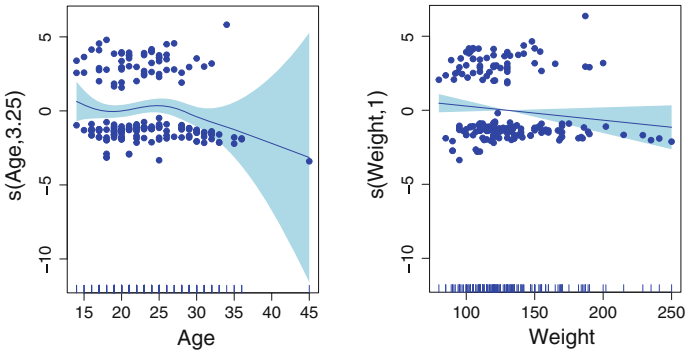

Fig. 2.31 A GAM analysis of a new born low birthweight as a function of background characteristics of mothers (The mother's age and weight are on the *horizontal* axes, the centered fitted logits are on the *vertical* axes, and the *shaded* area represents error bars. $N = 189$)

```
library(mgcv)
library(MASS)
data(birthwt)
attach(birthwt)

# Rename and Clean up variables
Low<-as.factor(low)
Age<-age
Weight<-(lwt)
Uterine<-as.factor(ifelse(ui==1,"Yes","No"))
Smokes<-as.factor(ifelse(smoke==1,"Yes","No"))
temp<-data.frame(Low,Age,Weight,Uterine,Smokes)

# Apply GAM
out<-gam(Low~s(Age)+s(Weight)+Uterine+Smokes,family=binomial,
                data=birthwt)
par(mfrow=c(1,2))
plot(out,se=T,residuals=T,pch=19,col="blue",shade=T,
      shade.col="light blue")
```

Fig. 2.32 R code for low birthweight analysis

2.32. When the mother has uterine irritability, the odds of a low birthweight baby are multipled by 2.05. Both are exponentiated regression coefficients like those that can be obtained from a logistic multiple regression.[31] In practical terms, both associations are likely to be seen as substantial.

[31] The regression coefficients are usually reported with the response in logit units. They are part of the routine tabular output from *gam()*.

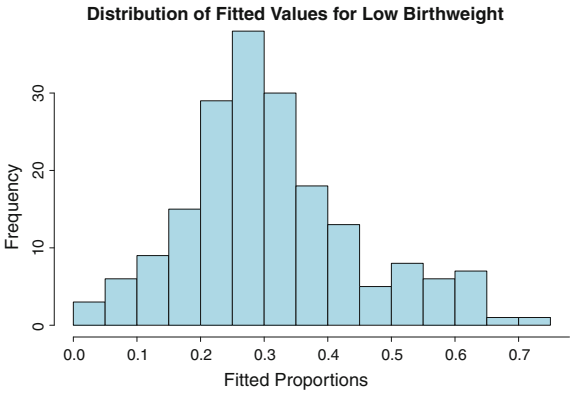

Fig. 2.33 Histogram of the fitted values in proportion units for low birthweight GAM analysis ($N = 189$)

Figure 2.31 (code in Fig. 2.32) shows the smoothed plots for the mother's age and weight. The units on the vertical axis are logits centered on the mean of the response in logit units. Weight has a linear relationship with the logit of low birth weight. The relationship between age and the logit of birth weight is roughly negative overall, but positive between about the age of 20 and 25. Yet, to the eye, both relationships do not appear to be strong, and the impression of a negative relationship with age is driven in part by one observation for a woman who is 45 years old. However, looks can be deceiving. When the logit units are transformed into probability units,[32] the difference in the proportion of low birthweight babies can be about .30 greater when a mother of 35 is compared to a mother of 25. A similar difference in the proportion of low birthweight babies is found when women weighing around 100 pounds are compared to women weighing around 200 pounds.[33]

GAM output can include two kinds of fitted values for binary response variables: the linear combination of predictors in logit units and fitted proportions. The former can be useful for diagnostic purposes because many of the conventional regression diagnostics apply. One has an additive model in logit units. The latter are useful for interpretation as approximate values of the response surface.

Figure 2.33 is a histogram of the fitted values in proportion units. One can see that substantial variation is found between different cases. Fitted proportion ranges from a little above 0.0 to nearly .80. Some prefer to see the proportions as probabilities, but such interpretations require a clear and credible explanation of the stochastic process by which the fitted values are produced. In effect, one is moving from a level I to a level II analysis.

The formulation we have favored that depends on a joint probability distribution may work as a start, but we know so little about how the data were collected that

[32] How to transform logits into probabilities and interpret them properly will be discussed in later chapters. There can be subtle issues.

[33] The relationship reported in Chap. 1 was rather different. In Fig. 1.2, the relationship between birthweights and mothers' weights was generally positive. But for that analysis, the mother's weight was the only predictor. Conditioning on other predictors makes a difference.

making a credible case would be very difficult. For example, perhaps these births occurred in a medical facility for women with difficult pregnancies. If women there are all served by the same medical staff, birthweights may not be independently realized. How a given pregnancy proceeds may affect how subsequent pregnancies are handled. The estimation process introduces further problems. Empirical determination of λ means that model selection is in play. There might be work-arounds were there real test data or if the dataset is large enough to be split into three subsets.

The default output included the usual information in the standard format about the statistical tests undertaken. The plots came with error bands. The issues raised are the same as just addressed. If the value of λ (or its equivalent) were determined before the GAM analysis began, one could at least in principle proceed with the formulation derived for response surface approximations, although a lot of hard thinking would be required to pin down the reality to which the generating joint probability distribution applies. However, the value of λ was determined empirically as part of the fitting process. We are again put in harm's way by model selection. It is difficult to know what properties the regression coefficients and fitted values have as estimates. One is probably best off sticking with a level I analysis unless there are real test data or the dataset is large enough to subdivide.

2.8 An Illustration of Statistical Inference After Model Selection

Significant parts of the discussion in this chapter have emphasized the risks in a level II analysis when penalty weights are determined empirically as part of the data analysis. For readers who rely on "statistical significance" to extract subject-matter conclusions, the tone may be disappointing and even annoying. As a possible corrective, consider now a level II analysis that perhaps can be properly be defended.

Figure 2.34 in a rendering of the overall inferential strategy introduced in Chap. 1. Tuning is done working back and forth between training data and evaluation data. Once the procedure is tuned, test data are used to get an honest performance assessment. It can then be possible in an additional step to get a sense of the uncertainty in any honest performance assessment. That process is based on bootstrap resampling and is discussed in a bit more detail below, with more to follow in later chapters.

There are two kinds of honest assessments. If the goal is to estimate *generalization error*, results determined with the help of evaluation data are fixed. Using GAM as an example, the values of λ and the associated functions of the predictors are set. Fitted values determined from the x-values for all cases in the test data are then compared to the actual values of Y in the test data using a statistic such as MSE. If the goal is to estimate *expected prediction error*, matters are more complicated because tuning implies model selection. Perhaps the safest approach is to treat the values of any tuning parameters as fixed once they are determined with the help of evaluation data. Then the procedure is applied to the test data. Again using GAM as

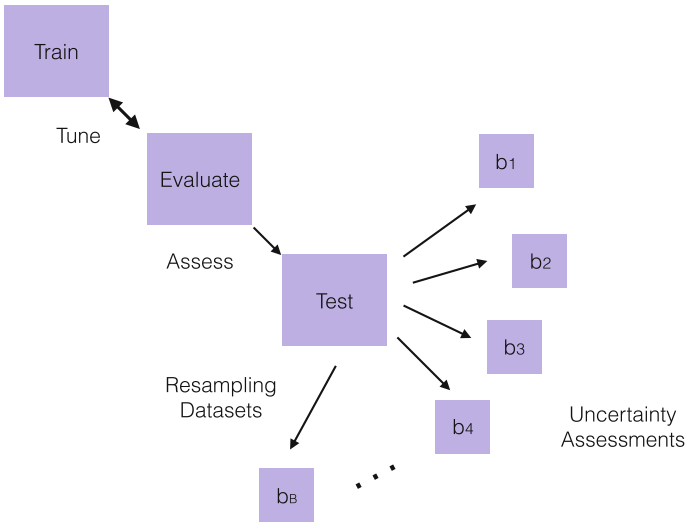

Fig. 2.34 A split sample approach for a level II analysis that includes tuning with an evaluation sample, assessment with a test sample, and uncertainty addressed with bootstrap resampling

an illustration, with the values of λ fixed, the response function for each predictor is estimated again, and one or more measures of fit are computed. All results are then honest with respect to data snooping for the given set tuning parameter values.

Uncertainty in test data performance assessment can be addressed with a nonparametric bootstrap. A total of B samples with replacement are drawn from the test data. Whether for generalization error or expected prediction error, a performance assessment is undertaken in each bootstrap sample. The distribution of performance assessments over samples from the test data can be used to characterize some forms of uncertainty — details to follow.

Let's see how this can play out. The data, available in R, are taken from the U.S. Panel Study of Income Dynamics (PSID). Funded by the National Science Foundation, the study began in 1968 and has been going for nearly 45 years. Households are followed over time and as the children of those households leave and form their own households, they are followed as well. New households are added when needed and feasible. Topics addressed in the survey have expanded over the years to include more than economic well-being: health, child development, time use and others (McGonagle et al. 2012). The data available in R are on married women in 1975. (See the code in Fig. 2.35.)

Given the sampling design, it may be reasonable to treat each observation in the dataset as a random realization from a joint probability distribution. In this case, the population is real and finite, but very large. The response is whether a married woman is in the labor force (even if unemployed). Predictors include (1) the wife's age, (2) family income excluding the wife's income in thousands of dollars, (3) whether the wife attended college, (4) whether the husband attended college, and (5) the number

of children in the household under 6 years of age. Clearly, there are other potentially important predictors such as the wife's health and the local availability of jobs. We are working within the wrong model perspective once again.

The R code used is shown in Fig. 2.35. With a sample of 753 cases, it is practical to construct a training sample, an evaluation sample, and a test sample. The code starts by providing more accessible names for the variables and recoding some stray observations — one cannot have negative family income. Then, three randomly chosen, disjoint, data subsets of equal size are constructed. The training data are analyzed using the generalized additive model as implemented in *gam()* from the *mgcv* library. Several different values for the penalty parameters (i.e., *sp*) are tried beginning the smallest at .01. For each, performance is then assessed using the evaluation data.[34]

There is a wealth of information in the usual output, but it can be difficult to arrive at an overall assessment of what the *sp* value for each regressor function should be. For binary outcomes, a "confusion table" can be a very effective overall assessment tool. A confusion table is nothing more than a cross-tabulation of the actual binary outcome and the fitted binary outcome. The smaller the proportion of cases misclassified, the better the fitted values perform.

Confusion tables will play a key role in later chapters, and there are a number of complications and subtleties. For now, we simply will classify a case as in the labor force if the fitted value in the response metric is greater than .50 and not in the labor force if the fitted value in the response metric is equal to or smaller than .50. The marginal distribution of labor force participation provides a baseline for the GAM fitted values. Nearly 57 % of the sample are in the labor force. Applying the Bayes classifier to the marginal distribution, classification error is minimized if all cases are classified as in the labor force. The proportion misclassified is then .43. How much better can be done using the predictors and good values for *sp*?

Regardless of the *sp* values tried, performance was modest. In the training data, about 10 % of the deviance could be attributed to the five predictors with the misclassification proportion a little less than .33. Improvement over the baseline was noticeable, but not dramatic.

For the *sp* values tried, out-of-sample performance did not vary much. Confusion tables from the evaluation data were all rather similar. For each, the misclassification proportion was about .36. Because the functions for both smoothed predictors were essentially straight lines, smaller *sp* values did not improve the fit enough to overcome the loss of degrees of freedom. Therefore, the values of *sp* for both quantitative predictors were set to the relatively large value 1.0.[35]

---

[34]The evaluation data provide out-of-sample performance measures, but with each new *sp* value, the out-of-sample benefits decline a bit. The model is being tuned to the evaluation data so that the fitting responds to both its random and systematic variation. Nevertheless, the evaluation data are very instructive. It was easy to see that with small values of *sp*, the overfitting in the training is substantial.

[35]Effectively the same results were obtained when the default of fitting by the GCV statistic was used.

```
## Set Up Data
library(car)
data(Mroz)
# Clean up labels and stray observations
Participates<-Mroz$lfp # in labor force
Age<-Mroz$age # age
FamIncome<-ifelse(Mroz$inc < 0,0,Mroz$inc) # family income
WifeColl<-Mroz$wc # Wife college degree
HusColl<-Mroz$hc # husband colleage degree
Kids5<-Mroz$k5 # Number of kinds under 6
temp1<-data.frame(Participates,Age,FamIncome,
                  WifeColl,HusColl,Kids5)

# Construct 3 random disjoint splits
index<-sample(1:753,753,replace=F) # shuffle row numbers
temp2<-temp1[index,] # put in random order
Train<-temp2[1:251,] # training data
Eval<-temp2[252:502,] # evaluation data
Test<-temp2[503:753,] # test data

## Determine Value of spar
library(mgcv)
# Applications to Training Data
out1<-gam(Participates~s(Age,sp=.01)+s(FamIncome,sp=.01)+
          WifeColl+HusColl+Kids5,data=Train,family=binomial)
Tab<-table(out1$fitted.values>.5,
           Train$Participates) # Confusion table
Tab
(Tab[1,2]+Tab[2,1])/sum(Tab) # Proportion Misclassified

# Apply to Evaluation Data
out2<-predict(out1,newdata=Eval,type="response") # Fitted values
Tab<-table(out2>.5,Eval$Participates) # Confusion table
Tab
(Tab[1,2]+Tab[2,1])/sum(Tab) # Proportion Misclassified

## Get Honest Performance Estimate
# With best values for sp, apply to test data
out3<-gam(Participates~s(Age,sp=1)+s(FamIncome,sp=1)+
            WifeColl+HusColl+Kids5,data=Test,family=binomial)
Tab<-table(out3$fitted.values>.5,
           Test$Participates) # Confusion table
Tab
(Tab[1,2]+Tab[2,1])/sum(Tab) # Proportion Misclassified
```

Fig. 2.35 R code for a level II analysis of the PSID data

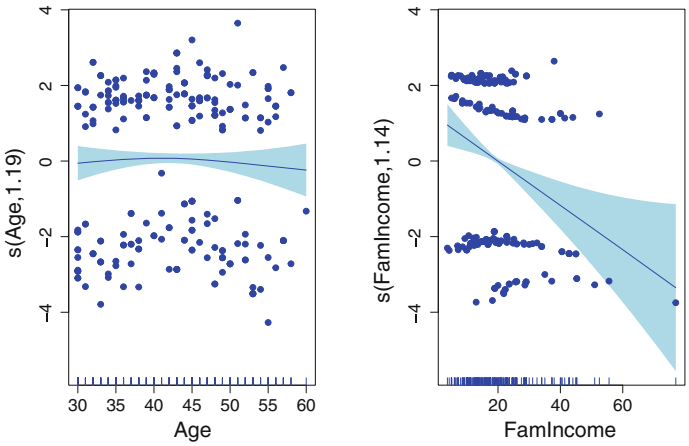

Fig. 2.36 Smoother estimates for age and family income in the test data (Fitted values in the units of centered log odds are on the *vertical* axes, predictors age and family income are on the *horizontal* axes, the *shaded* areas show error bands, and $N = 251$.)

The test data provide an antidote to model selection. Because the goal was to represent how the predictors were related to labor force participation, and no forecasting was anticipated, an estimate of expected prediction error was used as the performance yardstick. With the value of *sp* fixed at the "best" value, the generalized additive model was applied using the test data. Again, about 33 % of the cases were misclassified, and a little more than 10 % of the deviance could be attributed to the predictors. There was no evidence of meaningful overfitting. Among the three linear predictors, if a wife had attended college, the odds of labor force participation are multiplied by a factor of 3.3. A husband's college attendance hardly matter. For each additional child under six, the odds of labor force participation are multiplied by a factor of .20.[36]

Figure 2.36 shows in the test data the relationships between labor force participation and the two smoothed predictors. The near-linearity is apparent. Age has virtually no relationship with labor force participation, although there is a hint of decline for wives over 50. Family income (excluding the wife's income) has a very strong effect. The odds increase by a factor of about 3.9 when the lowest income households are compared to the highest. Lower income can dramatically increase a wife's labor force participation. So much for the level I analysis.

Thanks to the test data, it is relatively easy to move to level II. All of the results just reported from the test data can be used as asymptotically unbiased estimates for the population's same generalized additive model with the same sp values. But, there is considerable uncertainty to address. Although original sample had 753 observations,

[36]The number of children under 6 ranged from 0 to 2. With only 3 predictor values, smoothing was not an option.

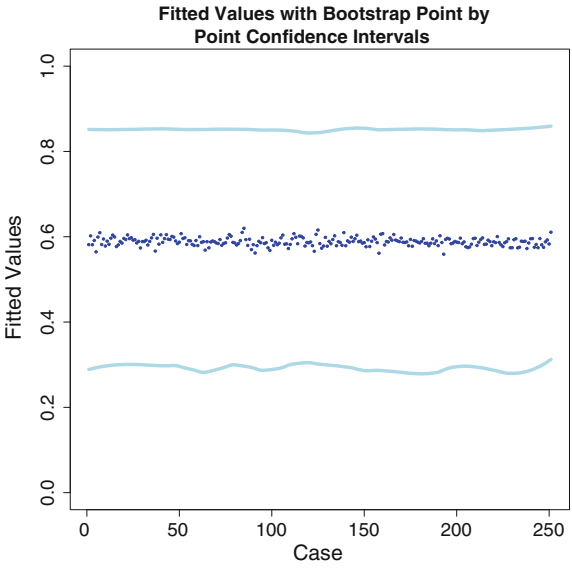

Fig. 2.37 Average bootstrap fitted values and Point-By-Point Error bands for the labor force analysis using 200 nonparametric bootstrap samples ($N = 251$)

the test sample had only 251, and that is the relevant sample size. Still, some use can be made of the nonparametric bootstrap with the test sample.

The bootstrap will be addressed in some depth in later chapters. For now, take on faith that Fig. 2.37 shows the bootstrapped point-by-point error bands around the average of the fitted values over 200 bootstrap samples. As such, they can be viewed as legitimate asymptotic estimates of point-by-point 95 % confidence intervals.

But confidence intervals around what? If the estimation target is the true response surface, the error bands are not confidence intervals. Because the estimates include bias, the error bands only capture the variance around the biased estimates. If the estimation target is an approximation of the true response surface, one has asymptotically valid confidence intervals around estimates of that approximation surface.

The major message from Fig. 2.37 that the wiggle around the fitted values is very large: approximately plus or minus .25. This is consistent with the small improvement in fit compared to the marginal Bayes classifier. It is also consistent with the small fraction of the deviance that could be attributed to the predictors and with the modest sample size.

A key implication is that the estimates of classification error can vary widely as well. Figure 2.38 shows a histogram of the proportion of cases misclassified over the 200 bootstrap samples. Although of the estimates cluster between .35 and .40, estimates range from about .28 to .45. Yet, the estimated probability of a misclassification proportion of less than the marginal Bayes classifier of .43 is about .98. Even the very weak model is likely to be an improvement. And there is more good news. The stronger relationships reported — for the wife attending college, the number of

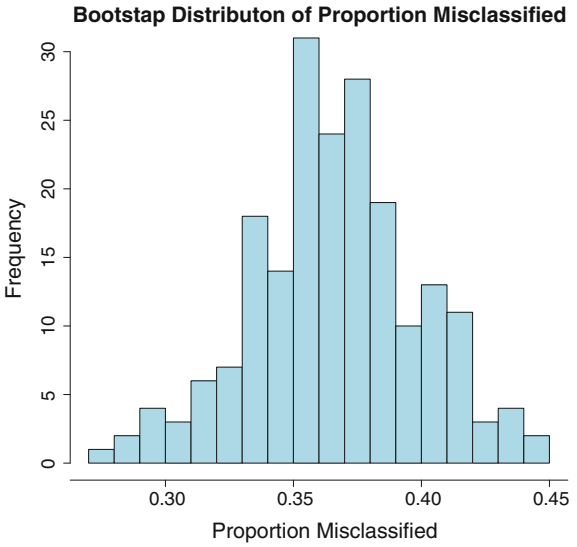

Fig. 2.38 Bootstrap distribution of proportion misclassified over 200 bootstrap samples ($N = 251$)

children under 6 and family income — all have reported p-values far smaller than .05 for a proper two-tailed test and a null hypothesis of no relationship.[37]

Still, there are at least three important caveats. First, the split sample approach comes at a price. Working with fewer observations split by split means that all level II results are less precise than had a full sample procedure been used (Faraway 2014). A more subtle point is that for smoothers and statistical learning more generally, more complex associations can be found with larger samples. In the case of smoothing splines, a smaller value for λ may be justified in a substantially larger sample. For a level I regression analysis, richer descriptions can follow. For a level II analysis, there can be a reduction in bias if there is focus on the true response surface and true conditional relationships.

Second, all of the statistical inference is asymptotic. A sample of 251 is probably large enough so that the inference is not misleading, but there is no way to know for sure. It is at least encouraging that the bootstrap distribution of the estimated misclassification error when examined with normal QQ plot (*qqnorm()*), showed no troubling departures from normality. The same can be said about the distributions of fitted values from which each of the error bands were constructed.[38] The sample of 251 may well be large enough for our purposes.

[37] These come from the standard *gam()* output. This is playing somewhat fast and loose. Ideally, the test should use a standard error estimate taking into account that the predictors are random variables. In principle, some version of the sandwich should apply. Alternatively, a nonparametric bootstrap could be used. One would resample the test data and obtain empirical distributions for each regression coefficient. Statistical tests or confidence intervals could follow. But with p-values so small, it is very likely that the test results would be confirmed.

[38] A random subset of 25 were examined.

Third and more fundamentally, the thought experiment behind the nonparametric bootstrap envisions random samples from a population, or realizations from a joint probability distribution, with the model and value of sp already known. This is consistent with conventional statistical inference. But, one might also wonder about uncertainty caused by the random data splitting. A very different thought experiment is implied, and its relevance can be argued. In principle, one could address that uncertainty by wrapping the entire analysis, including the sample splitting, in a nonparametric bootstrap.

There is also a need to briefly address again the role of all level II analyses. This example works because the way the data were assembled comports well with the joint probability distribution formulation. Other examples can work without real random sampling when a credible case can be made from subject matter knowledge, past research and an examination of the data on hand. For example, suppose as a criminal justice researcher, one wanted to study probation failures, and suppose one had access to data over several years on all individuals released on probation in a large city. Such data could have well over 100,000 observations. One can certainly imagine a joint probability distribution for the outcomes and predictors of interest as a product of the social processes determining how individuals do on probation. But, over time that joint probability distribution could well change as new laws are passed, new administrative regulations are promulgated, and the mix of convicted offenders varies. To what time interval does the data on hand apply?

There is no definitive way to address this question, but knowledge of the setting and surrounding circumstances can be very helpful. For example, have there been any important changes in the governing statutes? One can also make some headway using the data. It would be a simple matter to look at subsets of the data for several months at a time. Is there evidence that the joint probability distribution responsible for the data is changing in ways that matter? For example, is the fraction of the offenders being convicted for drug crimes increasing substantially? If key features of the data appear to be stable over the period the data were collected, the data may be seen as realized from a single joint probability distribution. One might also infer that the joint probability distribution changes slowly so that forecasts made for new cases in the not-too-distant future may be valid. As with all real data analyses, it is difficult to know for sure, but one properly can capitalize on the balance of evidence.

More difficult would be making the case that the observations were realized independently. Sometimes there are co-offenders for a given crime, and some crimes can be serial in nature. One might want to use the data to see how common both were for different kinds of crimes. For many crimes, both may be relatively rare. In addition, it may be possible to condition on certain predictors so that most dependence is removed, although that may lead to a revision of the plain vanilla, joint probability distribution approach. We will have a lot more to say about these issues in later chapters. Level II analyses with statistical learning can be a challenge.

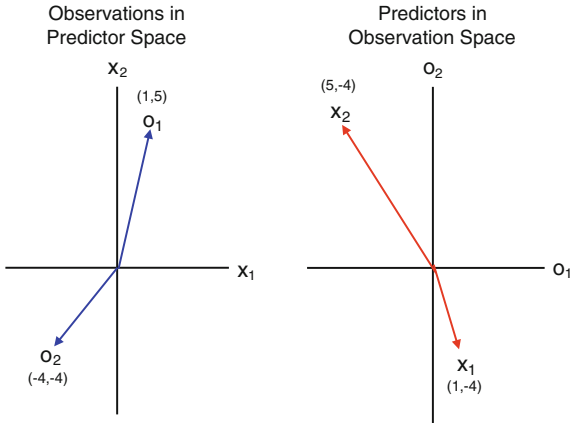

Fig. 2.39 2-Dimensional vector spaces that change the roles of variables and observations (The *left* figure has observations in variable space, and the *right* figure has variables in observation space.)

2.9 Kernelized Regression

In this chapter, much has been made of linear basis expansions as a way to make better use of the information a set of predictors contains. But each kind of expansion retained a matrix structure in which rows were observations and columns were variables. There is a very different and powerful way to transform the data from observations located in variable space to variables located in observation space. The result can be a kernel matrix that with some regularization can form a new kind of predictor matrix for a wide range of regression applications.[39]

Figure 2.39 illustrates how one can represent observations in variable space or variables in observation space. O_1 and O_2 are observations, and X_1 and X_2 are variables. The arrows represent vectors. Recall that vectors are lines with direction and length. We use them here primarily as a visualization tool.

The left plot shows observations in variable space and is the way one normally thinks about a scatter plot. There are two observations in a space defined by two predictors. The right plot shows variables in observation space, and is one way to think about kernels. There are two predictors in a space defined by two observations. For example, in the plot on the left, O_1 has a value of 1 for X_1 and a value 5 for X_2. In the plot on the right, X_1 has a value of 1 for O_1 and a value of -4 for O_2. In practice, there would be many more predictors and many more observations.

But why bother with predictors in observation space? Thinking back to the discussion of linear basis expansions in Chap. 1, kernels can alter the number of dimensions in which the values of a response variable are located. By increasing the number of dimensions, one may find good separation more easily. This is an important rationale for working with kernels. Kernels also have other desirable properties that can

[39] In statistics, there are other meanings of "kernel" such as when the term is used for localized estimators (Hastie et al. 2009: Sect. 6.6.1). In computer science, "variables" are sometimes called "feature vectors" that are located in "input space."

Fig. 2.40 R code inner products of **X**

```
X # The predictor matrix

       V1 V2 V3
Ilene   1  2  3
Jim     4  2  0
Ken     1  0  0
Linda   5  3  5
Mary    3  2  4

# Cross Product Matrix t(X)%*%X

    V1 V2 V3
V1 52 31 40
V2 31 21 29
V3 40 29 50

# Kernal Matrix X%*%t(X)

      Ilene Jim Ken Linda Mary
Ilene    14   8   1    26   19
Jim       8  20   4    26   16
Ken       1   4   1     5    3
Linda    26  26   5    59   41
Mary     19  16   3    41   29
```

make them a very handy tool. These will be considered as the discussion of kernels proceeds.[40]

Consider Eq. 2.21, a very simple predictor matrix **X** with five rows and three columns. Rows and columns are labeled. The rows are people and the columns are variables that are features of those people. Linear basis expansions of the sort we have considered so far could be applied to all three predictors or a subset.

$$\mathbf{X} = \begin{pmatrix} & V1 & V2 & V3 \\ Ilene & 1 & 2 & 3 \\ Jim & 4 & 2 & 0 \\ Ken & 1 & 0 & 0 \\ Linda & 5 & 3 & 5 \\ Mary & 3 & 2 & 4 \end{pmatrix}. \tag{2.21}$$

Figure 2.40 shows the **X** matrix, and the results from two different forms of matrix multiplication. The first is $\mathbf{X}^T\mathbf{X}$, which produces the usual sum of cross products, a symmetric matrix that plays such an important role in the ordinary least squares

[40] Adam Kapelner and Justin Bleich helped extensively with the exposition of kernels. They deserve much of the credit for what clarity there is. Lack of clarity is my doing.

estimator. Its main diagonal contains for each variable the sum of its squared values. For example, value of 21 in the second row and second column is the sum of the squared values of V2. The off-diagonal elements contain for each pair of variables their sum of element by element products, called inner products. The result is a scalar. For example, the value of 40 in the first row and third column and also in the third row and the first column results from $(1 \times 3) + (4 \times 0) + \cdots + (3 \times 4)$.

The second matrix is derived from $\mathbf{X}$ by computing $\mathbf{X}\mathbf{X}^T$. It too is symmetric. There are again sums of squared values or sums of cross-products, but the roles of variables and observations are switched. The main diagonal now contains for each *person* the sum of that person's squared values over the three variables. For example, Linda's diagonal value is 59: $5^2 + 3^2 + 5^2$. The off-diagonal elements are the sums of cross products for *person* pairs over the three variables. For example, the sum of cross products for Jim and Ilene is $(4 \times 1) + (2 \times 2) + (3 \times 0) = 8$. As before, these are sums of cross-products that result in a scalar.

Notice that this matrix has 5 columns rather than 3. Were one to use the matrix columns as a set of predictors, there would be 5 regressors. The response variable values would now reside in a 5-D predictor space, not a 3-D predictor space. The number of dimensions has been increased by 2.

$\mathbf{X}\mathbf{X}^T$ is often called a "linear kernel" and can be viewed as a similarity matrix (Murphy 2012: 479). The off-diagonal elements can be measures of the association between the different rows of $\mathbf{X}$. One can learn which observations are more alike over the full set of variables. In this example, a close look at $\mathbf{X}$ indicates that Mary and Linda have the most similar values for V1, V2, and V3, and from the kernel matrix, the value of 41 is the largest off-diagonal element. A kernel matrix is conventionally denoted by $\mathbf{K}$.

There are many kinds of kernels constructed with different kernel functions denoted in general by $\kappa(x, x')$. The notation x and x' means one row and another row, although it can also mean a row with itself in which case, each sum of cross products is non-negative (i.e., $\kappa(x, x') \geq 0$). Any kernel matrix, $\mathbf{K}$, is symmetric (i.e., $\kappa(x, x') = \kappa(x', x)$). For regression applications, it is common to work with Mercer kernels for which $\mathbf{K}$ is positive semi-definite.

The preference for Mercer kernel begins with $\mathbf{X}$. Imagine linear basis expansions for the full set of predictors each represented by $h(x)$. For Mercer kernels, $\kappa(x, x') = \langle h(x), h(x') \rangle$, which means the inner products of the expansions are contained in Mercer kernels (Hastie et al. 2009: Sect. 12.3.1).[41] There is no need the know the actual expansions because for regression applications one can proceed directly with the kernel. This is a very convenient computational shortcut, which means that in practice, model specification is usually a choice between different kinds of kernels

[41] Imagine a predictor $\mathbf{X}$ expanded so that each original column can now be many columns defined by some linear basis expansion (e.g., polynomials or indicator variables). In this context, the inner product means multiplying two rows (i.e. vectors) of the expanded predictor so that a scalar is produced. As before, it is just the sum of cross products. More generally, if there are two (column) vectors v_1 and v_2, the inner product is $v_1^T v_2$. The outer product is $v_1 v_2^T$, which results in a matrix. The same reasoning applies when a vector is multiplied by itself.

without much regard for the implied basis expansions. More will be said about kernels in the chapter on support vector machines.[42]

However, there are several complications. Because the kernel function requires that all elements in **X** be numeric, categorical predictors are a problem. At best, they can be transformed into 1/0 indicator variables, but the gap between a 1 and a 0 is arbitrary. And actually, the problem is more general. **K** depends on the units in which each column of **X** is measured. With different units, there are different kernels even when the kernel function is the same. Standardization of each column of **X** is, therefore, a common practice. But the common units chosen are effectively arbitrary and make it difficult to understand what the similarities mean. Two rows that are the much alike in standard deviation units, may be very different in their original units, which is how one normally thinks about those rows. One should always ask, therefore, "similar with respect to what?"

A second complication is that a kernel matrix is necessarily $N \times N$. Therefore, some form of dimension reduction is required in a regression setting. Regularization is required. Options include using a subset of **K**'s principal components as regressors or a form penalized regression. For example, the latter can lead to a ridge regression approach. In the notation of Hastie et al. (2009: Sect. 12.3.7),

$$\hat{f}(x) = h(x)^T \hat{\beta} = \sum_{i=1}^{N} \hat{\alpha}_i K(x, x_i), \tag{2.22}$$

and

$$\hat{\alpha} = (K(x, x_i) + \lambda \mathbf{I})^{-1} \mathbf{y}. \tag{2.23}$$

Equation 2.22 shows the fundamental equivalence between regressors as basis functions and regressors as columns of **K**. Equation 2.23 shows how the new regression coefficients $\hat{\alpha}$ for **K** are computed. Equation 2.23 is a close cousin of conventional ridge regression.

With $\hat{\alpha}$ in hand, the fitted values can follow as usual as long as one remembers to use **K** not **X**. For fitted values from *new* observations, the same reasoning carries over, but the new observations **Z** need to be "kernelized" (Exterkate et al. 2011: Sect. 2.2). A prediction kernel is constructed as $\kappa(x, z') = \langle h(x), h(z') \rangle$ not as $\kappa(x, x') = \langle h(x), h(x') \rangle$. That is, the inner products are undertaken with respect to **X** and **Z**, not with respect to **X** itself. For the linear kernel one computes $\mathbf{X}\mathbf{Z}^T$ rather than $\mathbf{X}\mathbf{X}^T$. That is,

$$\hat{f}(x, z) = \sum_{i=1}^{N} \hat{\alpha}_i K(x, z_i). \tag{2.24}$$

[42] There is a lot of very nice math involved that is actually quite accessible if it is approached step by step. For readers who want to pursue this, there are many lectures taught by excellent instructors that can be viewed for free on the web. These are generally more helpful than formal textbook treatments because the intuitions behind the math are often well explained. A good example is the lecture by MIT professor Patrick Winston "Lecture 16 – Learning: Support Vector Machines."

Also as before, λ in Eq. 2.23 is a tuning parameter whose value needs to be specified in advance or determined empirically. This leads to a third complication. Often it is desirable for λ to be large because, in effect, one starts with N predictors (much like for smoothing splines). But, empirically determining a sensible value for λ can be challenging, as we will soon see. There are usually additional tuning parameters.

A fourth complication is that kernel matrices produce a new kind of black box. In Eq. 2.22, for example, the regressors are columns of **K** not columns of **X**, and the estimated regression coefficients in Eq. 2.23 are $\hat{\alpha}$ not $\hat{\beta}$. It is a bit like trying to make sense of the regression coefficients associated with B-splines. Moreover, only in very special cases is it practical to work backwards from **K** to $h(x)$. The linear expansions of **X** typically are not accessible. As before, therefore, the story will be in the fitted values.

Finally, there many different kinds of kernels, and several different kinds of Mercer kernels that can be used in practice (Murphy 2012: Sect. 14.2; Duvenaud et al. 2013). Because of the black box, it is very difficult to know which kernel to use. The decision is usually based on experience with particular subject-matter applications and craft lore. We turn to two kernels that are popular for regression analysis.

2.9.1 Radial Basis Kernel

The bottom matrix in Fig. 2.40 is an example of a linear basis kernel. Formally, it is relatively easy to work with, but is not used much because there are other kernels that usually perform better. A good example is the radial basis kernel that is sometimes characterized as an "all purpose" kernel. Perhaps its most important departure from the linear kernel is that row comparisons are made initially by subtraction not multiplication. With each variable standardized and $\|.\|$ denoting the Euclidian distance (i.e. the "norm"), the radial basis kernel is defined by the function

$$k(x, x') = exp(-\sigma \|x - x'\|^2), \tag{2.25}$$

where $\|x - x'\|^2$ is the squared Euclidian distance between two rows.

The first step is to compute the sum of squared *differences*. For the second and third row of our toy **X**, one has for the sum of squared differences: $(4 - 1)^2 + (2 - 0)^2 + (0 - 0)^2 = 13$. The sum of squared differences is multiplied by $-\sigma$ and then exponentiated. For the radial basis kernel, otherwise known as the Gaussian radial basis kernel (Murphy 2012: Sect. 14.2.1), σ is a scale parameter specifying the spread in the kernel values. The kernel matrix **K** is always symmetric and $N \times N$. If the scale parameter σ happens to be 0.5, the value in **K** for the second and third row of

```
library(kernlab)

# Radial Basis Kernel

tune<-rbfdot(sigma=.5)
kernelMatrix(tune,X)
An object of class "kernelMatrix"
         Ilene       Jim       Ken        Linda       Mary
Ilene 1.0000e+00 1.2341e-04 1.5034e-03 2.7536e-05 8.2085e-02
Jim   1.2341e-04 1.0000e+00 1.5034e-03 1.3710e-06 2.0347e-04
Ken   1.5034e-03 1.5034e-03 1.0000e+00 1.3888e-11 6.1442e-06
Linda 2.7536e-05 1.3710e-06 1.3888e-11 1.0000e+00 4.9787e-02
Mary  8.2085e-02 2.0347e-04 6.1442e-06 4.9787e-02 1.0000e+00

# ANOVA Basis Kernel

tune<-anovadot(sigma=2.0,degree=2)
kernelMatrix(tune,X)
An object of class "kernelMatrix"
       Ilene     Jim      Ken        Linda       Mary
Ilene 9.000000 1.000000 1.0007e+00 1.8407e-02 1.2898e+00
Jim   1.000000 9.000000 1.0007e+00 7.3263e-02 1.2810e+00
Ken   1.000671 1.000671 9.0000e+00 2.3195e-16 4.5014e-07
Linda 0.018407 0.073263 2.3195e-16 9.0000e+00 7.3444e-02
Mary  1.289748 1.288986 4.5014e-07 7.3444e-02 9.0000e+00
```

Fig. 2.41 R code for radial basis and ANOVA basis kernels

$\mathbf{X}$ is $e^{(13\times-.5)} = .0015034$. The top matrix in Fig. 2.41 follows in the same manner.[43]

The diagonal entries of the radial basis kernel are always 1 ($e^0 = 1$), and the off-diagonal entries are between 0 and 1. Because radial kernels build on Euclidian distances, they can be viewed as similarity matrices. With a smaller distance between a pair observations, there is greater similarity. Thanks to the negative sign the associated with σ, a larger kernel value then conveys greater similarity.

When the value of σ is larger, the off-diagonal kernel values become smaller, so their measured similarities are reduced. The rows become more heterogeneous, which is consistent with the idea a larger scale value. In language we used earlier, the bandwidth, span, or window has gotten smaller. A more complex set of fitted values can be accommodated. Consequently, σ typically is treated as a tuning parameter.

[43]For consistency, the kernel expressions and notation are the same as in the documentation for *kernlab* (Karatzoglou et al. 2004), the excellent library containing the kernel procedures used. In some expositions, λ is used instead of σ and then $\lambda = \frac{1}{2\sigma^2}$, where σ is another constant. In that form, the radial basis kernel commonly is called the Gaussian radial basis kernel. Note that this λ is not the same λ as in Eq. 2.23.

Radial basis kernels have proved to be useful in a wide variety of applications but for regression, there can be a better choice (Karazolou et al. 2004: Sect. 2.4).

2.9.2 ANOVA Radial Basis Kernel

The ANOVA radial basis kernel builds on the radial basis kernel. Using common notation for the ANOVA kernel,

$$k(x, x') = \left(\sum_{k=1}^{p} exp(-\sigma(x^k - x'^k)^2) \right)^d, \tag{2.26}$$

where x^k and x'^k are the two values for predictor k, p is the number of predictors in $\mathbf{X}$, and σ is again a scale parameter typically used for tuning. As before, larger values of σ allow for a more complex fit.[44] The values for d are usually 1, 2, or 3. Because the computations begin with differences that after being transformed are added together, the calculations are linear when $d = 1$, and one has a linear, additive effects expression. When $d = 2$, one has an expression with products that can be seen as two-way interaction variables. By the same reasoning, when $d = 3$, one has three-way interaction variables. In practice, d is treated as a tuning parameter along with σ.[45] Larger values for d allow for a more complex set of fitted values.

The lower matrix in Fig. 2.41 shows the results for the same predictor matrix $\mathbf{X}$ when σ is set to 2.0 and d is set to 2. Because there are 3 predictors in $\mathbf{X}$, the main diagonal elements are all equal to 9 (i.e., $(1 + 1 + 1)^2$). Off-diagonal elements no longer have an upper bound of 1.0.

2.9.3 A Kernel Regression Application

In a regression context, the radial kernel and the anova kernel can be used as predictor matrices that replace $\mathbf{X}$ in a regression analysis. Both kernels provide a very rich

[44] The computational translation is a little tricky. These are the steps for any given entry i, j in $\mathbf{K}$. (1) As before, one does an element by element subtraction of observations i and j over each of the predictors. These are rows in $\mathbf{X}$. (2) Square each of the differences. (3) Multiply each of these squared differences by minus σ. (4) Exponentiate each of these products. (5) Sum the exponentiated products. (6) Raise the sum to the power of d.

[45] To illustrate, consider $\mathbf{X}$ with three predictors. For the pair of observations from, say, the first and second row of $\mathbf{X}$ and $d = 1$, the sum of differences is $(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + (x_{13} - x_{23})^2$. This is linear and additive in the squared differences. For $d = 2$, the result is $[(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + (x_{13} - x_{23})^2]^2$. All of the terms are now products of two squared differences, which are two-way interaction effects. For $d = 3$, the result is $[(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + (x_{13} - x_{23})^2]^3$. All of the terms are now products of three squared differences, which are three-way interaction effects. Hence the name ANOVA kernel.

menu of complicated transformations that are directly given to the fitting machinery. One hopes that **K** can find relationships with the response that **X** cannot. Complicated nonlinear relationships are in play through what is effectively a nonparametric formulation.

As usual, "nonparametric" can be used in different ways. What one means for kernelized regression is that the regression structure is not meant to be a model of anything. The regression structure is just part of an algorithm linking input to outputs. There are regression coefficients for the columns in **K**, but they have no subject-matter interpretation. Just as for smoothing splines, the goal is to interpret and use the relationship between inputs and outputs.

The heavy emphasis on fitted values has meant that for kernel regression, ways to visualize how inputs are related to outputs are not as well developed. Some of the visualization procedures discussed in later chapters could be applied, but at least in R, they have not been applied yet. Where kernel regression method can shine is in forecasting.

A natural question, therefore, is what kernel methods estimate. Kernel regression methods can be used to estimate an approximation of the true response surface as a feature of nature's joint probability distribution. That approximation has the same structure, and same values for the tuning parameters used with the training data. The fitted values from the data can be taken as biased (even asymptotically) estimates of the true response surface. In short, a level II analysis has the same features as a level II analysis for smoothing splines. And as before, one has to make a credible case that a level II analysis is justified.

Even if the choice of kernel is made before looking at the data, in practice the kernel's parameters will be tuned, and the regularization parameter will be tuned as well. So, here too the data snooping issues have been addressed. One is usually in the model selection business once again.

Consider Eqs. 2.22 and 2.23. What may look to be a simple application of ridge regression is not so simple. There are three tuning parameters, two for the ANOVA kernel and one for the regularization, that typically need to be determined empirically. A first impulse might be to use some in-sample fit measure such as the GCV statistic. A search is undertaken over values of the tuning parameters. Their values are determined by the best fit value. However, any credible in-sample fit statistics should take the effective degrees of freedom (EDF) into account because the effective degrees of freedom is changing as the values of the tuning parameters are varied. If the effective degrees of freedom is ignored, a better fit may result simply from more degrees of freedom being used in the fitting process. This matters even for a level I analysis because the data analyst could be faced with unnecessarily complicated fitted values that will be challenging to interpret.

Yet, as discussed in Chap. 1, even the idea of an effective degrees of freedom (or effective number of parameters) in such a setting is being questioned. What does the effective degrees of freedom mean when there is tuning? In practice, therefore, a sensible level I analysis requires a careful examination of the fitted values and plots of the actual response values against the fitted values. Subject-matter knowledge can be critical for determining which sets of fitted values are most instructive. In short,

the level I concerns for conventional ridge regression carry over but now with more tuning parameters to specify.

For a level II analysis, we are again faced with model section implications that can introduce challenging problems for statistical inference. Rather than using some in-sample fit statistics, why not employ cross-validation? The issues are tricky. A kernel matrix is $N \times N$, and the tuning is done with respect to the full kernel matrix. Yet, cross-validation necessarily fits subsets of the data, each with fewer than N observations. Tuning could be quite different, and with respect to the full kernel matrix, misleading. Probably the best cross-validation approach is N-fold because for each pass through the data only one observation is lost.

Alternatively, one might employ a split sample strategy. As before, the training and evaluation samples are exploited to determine the values for the tuning parameters. The kernel regression coefficients $\hat{\boldsymbol{\alpha}}$ from the training sample are used to obtain fitted values in the evaluation data, from which one or more performance measures are computed. With the values of the tuning parameters determined, the test data can be employed to obtain honest performance assessments. But one is still faced with a smaller N than had the data not be split. If one is prepared to settle for working with and tuning for a kernel fit based on a substantially smaller sample, a split sample approach can work well. The new estimation target is test sample version.

There are apparently no fully satisfactory procedures currently in R that implement the kind of penalized kernel regression shown Eqs. 2.22 and 2.23.[46] But with split samples and the *kernelMatrix()* procedure from the library *kernlab* to construct the requisite kernels, it is relatively easy to write a "one-off" R-script that implements a version of Eqs. 2.23 and 2.24 cycling through the training data many times using different sets of values for σ, d, and λ.[47] Each set of $\hat{\boldsymbol{\alpha}}$ is then used to produce fitted values in the evaluation data. Once acceptable tuning parameter values are determined, they are used to compute a penalized kernel ridge regression in the test data.

To see how this plays out, suppose one wanted to analyze variation in the gross domestic earnings for movies made in the United States immediately after the movie opens. There are data on 471 movies for which one has the following regressors: (1) the budget for making each movie, (2) the number of theaters in which it opened, and (3) opening day's earnings. The response is gross domestic earnings over the next 24 months. These data were randomly split into a training sample, an evaluation sample, and a test sample of equal sizes. The performance criterion was mean squared error.

From the search over the movie training data and evaluation data, σ was a chosen to be 10, d was chosen to be 2.0, and λ was chosen to be 3. There were several sets

[46]There is in the *CVST* library a procedure with which to do penalized kernel ridge regression. However, the documentation is very spare, and the references cited often do not seem germane. It is, therefore, very difficult to know what the underlying code is really doing, and some of the output is problematic.

[47]The library *kernlab()* was written by Alexandros Karatzoglou, Alex Smola, and Kurt Hornik. It is has an excellent collection of functions for working with kernels in a wide variety of ways.

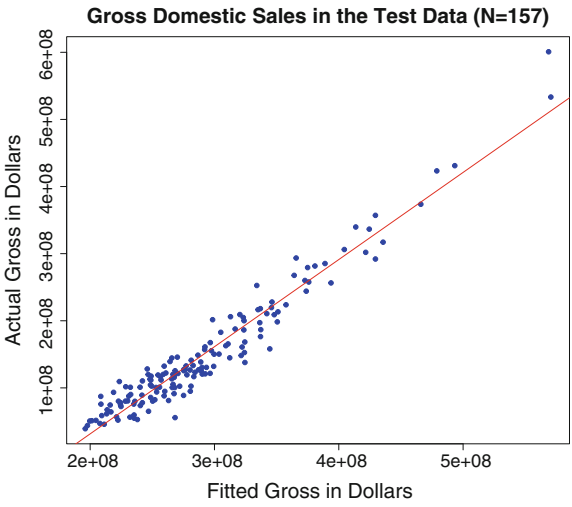

Fig. 2.42 Domestic gross sales in million of dollars by the fitted values from a penalized kernel regression using the test data (N = 157)

of tuning parameters that performed approximately as well, and the among those, the set with the smallest tuning parameter values for the kernel and the largest value for penalty parameter was selected. There seemed to be no reason to unnecessarily use up degrees of freedom.

A kernel regression with the same number of observations and tuning parameters values was used with the test data. In the same spirit as expected prediction error, a plot of the observed response values against the fitted response values for the test data is shown in Fig. 2.42. Overlaid is the least squares line from which the R^2 of .92 was computed. Overall, the scatterplot has a sensible pattern although the fitted values do not track some of the highest or lowest gross sales as well. This means that if the goal is to represent very soon after a movie is released its gross domestic sales over the next two years, the results look promising except for the few "flops" and "blockbusters." It is easy to capture those response values too with a more complex set of fitted values (e.g. with a σ of 20, a d of 3, and a λ of 1), but that leads to unrealistic measures of fit (e.g., an R^2 of .99).

For a level II analysis, the estimation target is the kernel regression approximation of the true response surface with the same values for the tuning parameters. If one can define a substantively relevant joint probability distribution or finite population from which each observation was independently realized, the test data results can provide an asymptotically unbiased estimates. And much as in the previous example, one can then apply a nonparametric bootstrap to the test data and obtain information on the uncertainty built into the R^2 of .92.

2.10 Summary and Conclusions

Regression splines and regression smoothers can be very useful level I tools for describing relationships between a response variable and one or more predictors. As long as one is content to "merely" describe, these methods are consistent with the goals of an exploratory data analysis. Moving to a level II analysis can be challenging because there needs to be a credible data generation backstory consistent with the formal requirements of statistical inference. In addition, when model selection is part of the data analysis, there are significant obstacles that in practice can be difficult to overcome. In the absence of real experiments, a level III analysis will depend, as usual, on a compelling interpretive overlay addressing why one should believe that the manipulation of given predictors will alter the distribution of the response. And all three regression analysis levels are undertaken with estimates of an approximation of the true response surface. In conventional terms, one is working misspecified regression models.

Experience suggests that for most datasets, it does not make a great difference which brand of smoother one uses. The dominant factor is usually the values of λ or other tuning parameters that determine smoothness and the bias-variance tradeoff. Less clear is how their values are best determined. Most methods emphasize some measure of generalization error. This is certainly sensible given the empirical focus on fitted values. But fitted values with low generalization error do not necessarily make scientific or policy sense. Moreover, any overall measure of out-of-sample performance can neglect that performance will usually be better for some predictors than others. Often a subset of predictors are of special interest, and it is on their performance that a useful evaluation should rest. These complications and others suggest that it can be a mistake to automatically defer to default tuning values or default tuning procedures. There is no substitute for subject-matter expertise and a careful examination of a wide range of data analysis output. Judgment matters. It is very important to avoid what a number of philosophers and social scientists call "statisticism" (Finch 1976; Lamiell 2013).

Finally, there are other smoothers that have not been discussed either because they perform best in a relatively narrow set of applications or because they are not yet ready for widespread use. An example of the former is wavelet smoothing (Hastie et al. 2009: Sect. 5.9). "Wavelet bases are very popular in signal processing and compression, since they are able to represent both smooth and/or locally bumpy functions in an efficient way — a phenomenon dubbed time and frequency localization" (Hasite et al. 2009: 175). An example of the latter is very recent work that applies trend filtering to nonparametric regression (Tibshirani 2015). The key idea is to define the fitting penalty a novel way, not using second derivatives, by a discrete differencing operator on the regression coefficients. The result is a smoother that adapts locally. It will fit a rougher function where the data are more rough and a smoother function

where the data are more smooth.[48] In short, the book is not closed on smoothers, and readers interested in such procedure should at least skim the relevant journals from time to time.

For a wide range of problems, there are statistical learning techniques that arguably perform better than the procedures discussed in this chapter. They can fit the data better, are less subject to overfitting, and permit a wider range of information to be brought to bear. One price, however, is that the links to conventional regression analysis become even more tenuous. In the next chapter, we continue down this path.

Demonstrations and Exercises

Just as for the first chapter, these demonstrations and exercises emphasize the analysis of data. What substantive insights can be properly extracted? You may need to install some packages depending on what you have already installed. (Have you updated R and the procedures you will be using lately?)

Set 1: Smoothers with a Single Predictor

1. Load the dataset called airquality using the command *data(airquality)*. Attach the data with the command *attach(airquality)*. Use *gam()* from the *gam* library with Ozone as the response and Temp as the sole predictor. Estimate the following three specifications assigning the output of each to its own name (e.g., output1 for the first model).

```
gam(Ozone ~ Temp)
gam(Ozone ~ as.factor(Temp))
gam(Ozone ~ s(Temp))
```

   The first model is the smoothest model possible. Why is that? The second model is the roughest model possible. Why is that? The third model is a compromise between the two in which the degree of smoothing is determined by the GCV statistic. (See the *gam()* documentation followed by the smoothing spline documentation.)

   For each model, examine the numerical output and plot the fitted values against the predictor. For example, if the results of the first model are assigned to the name "output1," use *plot.gam (output1, residuals=TRUE)*. Also, take a look at the output object for the variety of gam features and output that can be accessed. Extractor functions are available.

   Which model has the best fit judging by the residual deviance? Which model has the best fit judging by the AIC? Why might the choice of the best model differ depending on which measure of fit is used? Which model seems to be most useful

[48] The idea of allowing for locally varying complexity in fitted values is an old one. Fan and Gijbels (1992, 1996), who have made important contributions to the topic, attribute the idea to Breiman, Meisel and Purcell (1977).

judging by the plots? Why is that?

2. Using *scatter.smooth()*, overlay a lowess smooth on a scatterplot with the variable Ozone on the vertical axis and the variable Temp on the horizontal axis. Vary three tuning parameters: span: .25, .50, .75; degree: 0, 1, 2; family as Gaussian or symmetric. How do the fitted values change as each tuning parameter is varied? Which tuning parameter seems to matter most? (You can get the same job done with *loess()*, but a few more steps are involved.)

3. The relationship between temperature and ozone concentrations should be positive and monotonic. From the question above, select a single set of tuning parameter values that produces a fit you like best. Why do you like that fit best? If there are several sets of fitted values you like about equally, what it is about these fitted values that you like also?

4. For the overlay of the fitted values you like best (or select a set from among those you like best) describe how temperature is related to ozone concentrations.

Set 2: Smoothers with Two Predictors

1. From the library *assist* load the dataset TXtemp. Load the library *gam*. With mmtemp as the response and longitude and latitude as the predictors, apply *gam()*. Construct the fitted values using the sum of a 1-D smoothing spline of longitude and a 1-D smoothing spline of latitude. Try several different values for the degrees of freedom of each. You can learn how to vary these tuning parameters with *help(gam)* and *help(s)*. Use the *summary()* command to examine the output, and *plot.gam()* to plot the two partial response functions. To get both plots on the same page use *par(mfrow=c(2,1))*. How are longitude and latitude related to temperature? (If you want to do this in *gam()* in the *mgcv* library, that works too. But the tuning parameters are a little different.)

2. Repeat the analysis in 1, but now construct the fitted values using a single 2-D smoother of longitude and latitude together. Again, try several different values for the degrees of freedom. Examine the tabular output with *summary()* and the plot using *plot.gam()*. You will need to load the library *akima* for the plotting. How do these results compare to those using two 1-D predictor smooths? (For 2-D smoothing, the plotting at the moment is a little better using *gam* in the *mgcv* library.)

Set 3: Smoothers with More Than Two Predictors

1. Still working in *gam()*, build an additive model for mmtemp with the predictors longitude, latitude, year, and month. Use a lowess smooth for each. Try different spans and polynomial degrees. Again use the *summary()* and *plot.gam()* command. To get all four graphs on the same page use *par(mfrow=c(2,2))*. How is

temperature related to each of the four predictors?

2. Repeat the analysis just done using smoothing splines in *gam()*. See if you can tune the model so that you get very close to same graphs. Does it matter which kind of smoother you use? Why or why not? (Keep in mind that you tune *s()* differently from *lo()*.)

Set 4: Smoothers with a Binary Response Variable

1. From the *car* library, load the dataset *Mroz*. Using *glm()*, regress labor force participation on age, income, and the log of wages. From the library *gam*, use *gam()* to repeat the analysis, smoothing each of the predictors with the smoother of your choice. Note that labor force participation is a binary variable. Compare and contrast your conclusions from the two sets of results. Which procedure seems more appropriate here? Why?

http://www.springer.com/978-3-319-44047-7

Statistical Learning from a Regression Perspective

Berk, R.A.
2016, XXV, 347 p. 120 illus., 91 illus. in color.,
Hardcover
ISBN: 978-3-319-44047-7