3

From functional data to smooth functions

3.1 Introduction

This chapter serves to introduce some ideas that are central to the next two chapters, where we will develop methods for turning raw discrete data into smooth functions.

Our goals in this chapter are:

- To understand what we mean when we refer to data as "functional".
- To explore the concept of "smoothness" of a function, and relate smoothness to the function's derivatives.
- To consider how noise or error of measurement combines with smooth functional variation to produce the observed data.

We will use linear combinations of basis functions as our main method for representing functions. The use of basis functions is a computational device well adapted to storing information about functions, and gives us the flexibility that we need combined with the computational power to fit even hundreds of thousands of data points. Moreover, it permits us to express the required calculations within the familiar context of matrix algebra.

Most of the functional analyses that we discuss can be expressed directly in terms of functional parameters using more advanced methods such as the calculus of variations and functional analysis, but we consider these approaches to be too technical to be useful to the readers that we have in

mind. Moreover, the basis function approach has not, in our experience, imposed any practical limitations on what we have needed to do.

We will consider in detail two basis function systems: The Fourier basis and the B-spline basis. The former tends to be used to describe periodic data, and the latter for functional information without any strongly cyclic variation. We will not neglect, however, several other types of basis systems, each having its own merits in certain contexts.

3.2 Some properties of functional data

The basic philosophy of functional data analysis is to think of observed data functions as single entities, rather than merely as a sequence of individual observations. The term *functional* in reference to observed data refers to the intrinsic structure of the data rather than to their explicit form. In practice, functional data are usually observed and recorded discretely as n pairs (t_j, y_j), and y_j is a snapshot of the function at time t_j, possibly blurred by measurement error. Time is so often the continuum over which functional data are recorded that we may slip into the habit of referring to t_j as such, but certainly other continua may be involved, such as spatial position, frequency, weight, and so forth.

3.2.1 What makes discrete data functional?

What would it mean for a functional observation to be known in functional form x? We do not mean that x is actually recorded for every value of t, because that would involve storing an uncountable number of values! Rather, we mean that we assume the existence of a function x giving rise to the observed data.

In addition, we usually want to declare that the underlying function x is *smooth*, so that a pair of adjacent data values, y_j and y_{j+1} are necessarily linked together to some extent and unlikely to be too different from each other. If this smoothness property did not apply, there would be nothing much to be gained by treating the data as functional rather than just multivariate.

By smooth, we usually mean that function x possesses one or more derivatives, which we indicate by Dx, D^2x, and so on, so that $D^m x$ refers to the derivative of order m, and $D^m x(t)$ is the value of that derivative at argument t. We will usually want to use the discrete data $y_j, j = 1, \ldots, n$ to estimate the function x and at the same time a certain number of its derivatives. For example, if we are tracking the position x of a moving object such as a rocket, we will want, also, to estimate its velocity Dx and its acceleration D^2x. The modelling of a system's rates of change is often

called the analysis of a system's *dynamics*. The many uses of derivatives will be a central theme of this book.

The actual observed data, however, may not be at all smooth due to the presence of what we like to call noise or measurement error. Some of this extraneous variation may indeed have all the characteristics of noise, that is, be formless and unpredictable, or it may be high-frequency variation that we could in principle model, but for practical reasons choose to ignore. Sometimes this noise level is a tiny fraction of the size of the function that it reflects, and then we say that the *signal-to-noise ratio* (S/N ratio) is high. However, higher levels of variation of the y_j's around the corresponding $x(t_j)$'s can make extracting a stable estimate of the the function and some of its derivatives a real challenge.

Most of this chapter and the next are given over to how to estimate x and some of its derivatives from noisy data.

3.2.2 Samples of functional data

In general, we are concerned with a collection or sample of functional data, rather than just a single function x. Specifically, the record or observation of the function x_i might consist of n_i pairs (t_{ij}, y_{ij}), $j = 1, \ldots, n_i$. It may be that the argument values t_{ij} are the same for each record, but they may also vary from record to record. It may be that the interval $\mathcal{T}$ over which data are collected also varies from record to record.

Normally, the construction of the functional observations x_i using the discrete data y_{ij} takes place separately or independently for each record i. Therefore, in this chapter, we will usually simplify notation by assuming that a single function x is being estimated. However, where the signal-to-noise ratio is low, or the data are sparsely sampled or few in number, it can be essential to use information in neighboring or similar curves to get more stable estimates of a specific curve.

Sometimes time t is considered cyclically, for instance when t is the time of year, and this means that the functions satisfy *periodic boundary conditions*, where the function x at the beginning of the interval $\mathcal{T}$ picks up smoothly from the values of x at the end. Data for functions which do not naturally wrap around in this way are called *non-periodic.*

Finally, a lot of functional data are distributed over multidimensional argument domains. We may have data observed over one or more dimensions of space as well as over time, for example. A photograph or a brain image is a functional observation where the intensity and possibly color composition is a function of spatial location.

3.2.3 The interplay between smooth and noisy variation

Smoothness, in the sense of possessing a certain number of derivatives, is a property of the latent function x, and may not be at all obvious in the raw

data vector $y = (y_1, \ldots, y_n)$ owing to the presence of observational error or noise that is superimposed on the underlying signal by aspects of the measurement process. We express this in notation as

$$y_j = x(t_j) + \epsilon_j, \tag{3.1}$$

where the noise, disturbance, error, perturbation or otherwise exogenous term ϵ_j contributes a roughness to the raw data. One of the tasks in representing the raw data as functions may be to attempt to filter out this noise as efficiently as possible. However, in other cases we may pursue the alternative strategy of leaving the noise in the estimated function; and instead require smoothness of the results of our analysis, rather than of the data that are analyzed.

Vector notation leads to much cleaner and simpler expressions, and so we express the "signal plus noise" model (3.1) as

$$\mathbf{y} = x(\mathbf{t}) + \mathbf{e} \tag{3.2}$$

where $\mathbf{y}, x(\mathbf{t}), \mathbf{t}$ and $\mathbf{e}$ are all column vectors of length n.

The variance-covariance matrix for the vector of observed values $\mathbf{y}$ is equal to the variance-covariance matrix for the corresponding vector $\boldsymbol{\epsilon}$ of residual values since the values $x(t_j)$ are here considered fixed effects with variance 0. Let $\boldsymbol{\Sigma}_e$ be our notation for residual variance-covariance matrix, which expresses how the residuals vary over repeated samples that are identical in every respect except for noise or error variation.

3.2.4 The standard model for error and its limitations

The standard or textbook statistical model for the distribution of the ϵ_j's is to assume that they are independently distributed with mean zero and constant variance σ^2. Consequently, according to the standard model,

$$\mathtt{Var}(\mathbf{y}) = \boldsymbol{\Sigma}_e = \sigma^2 \mathbf{I} \tag{3.3}$$

where the identity matrix $\mathbf{I}$ is of order n.

These assumptions in the standard model, in spite of being routinely made, are almost surely too simple for most functional data. Rather, for example, we must often recognize that the variance of the residuals will itself vary over argument t. We will see in Chapter 5, for example, that the standard error of measurement of the height of children is about eight millimeters in infancy, but declines to around five millimeters by age six.

We may also have to take into account a correlation among neighboring ϵ_j's. The *autocorrelation* that we often see in functional residuals reflects the fact that the functional variation that we choose to ignore is itself probably smooth at a finer scale of resolution.

In fact, the concept of independently distributed error in the standard model, which, as n increases, becomes what is called *white noise,* is not realistic or realizable in nature because white noise would require infinite

energy to achieve. For example, fluctuations in a large stock market are often treated as having white noise properties, but in reality only a limited number of stocks can be traded within a short time interval such as a millisecond, and consequently scale stock prices will exhibit some structure within a time scale that is small enough.

This does not necessarily mean that it will be always essential to model the variable variance or autocorrelation structure in the residuals or errors. Such models for $\mathbf{\Sigma}_e$ can burn up precious degrees of freedom, slow down computation significantly, and finally result in estimates of functions that are virtually indistinguishable from what is achieved by assuming independence in residuals. Nevertheless, a model specifically for variance heterogeneity and/or autocorrelation can pay off in terms of better estimation, and this type of structure may be in itself interesting. A thoughtful application of functional data analysis will always be open to these possibilities.

We should also keep in mind the possibility that errors or disturbances might multiply rather than add when the data are intrinsically positive, in which case it is more sensible to work with the logarithms of the data. We will do this, for example, with the precipitation data for Canadian weather stations in Chapter 14.

3.2.5 The resolving power of data

The *sampling rate* or *resolution* of the raw data is a key determinant of what is possible in the way of functional data analysis. This is essentially a local property of the data, and can be described as the density of the argument values t_j relative to the amount of curvature in the data, rather than simply the number n of argument values. The *curvature* of a function x at argument t is usually measured by the size of the second derivative, as reflected in either $|D^2x(t)|$ or $[D^2x(t)]^2$.

Where curvature is high, it is essential to have enough points to estimate the function effectively. What is enough? This depends on the amount of error ϵ_j; when the error level is small and the curvature is mild, we can get away with a low sampling rate. The gait data in Figure 1.8 exhibit little error and only mild curvature, and thus the sampling rate of 20 values per cycle is enough for our purposes. The human growth data in Figure 1.1 have moderately low error levels, amounting to about 0.3% of adult height, but the curvature in the second derivative functions is fairly severe, so that a sampling rate of measurements every six months for these data is barely sufficient for making inferences about growth acceleration.

3.2.6 Data resolution and derivative estimation

Figure 3.1 provides an interesting example of functional data. The letters "fda" were written on a flat surface by one of the authors. The pen positions

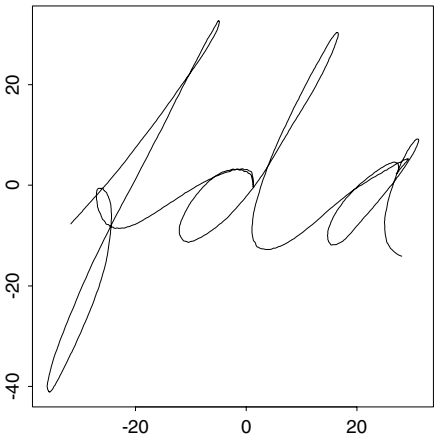

Figure 3.1. A sample of handwriting in which the X-Y coordinates are recorded 600 times per second.

were recorded by an Optotrak system that gives the position of an infrared emitting diode in three-dimensional space 600 times per second with an error level of about 0.5 millimeters. The X and Y position functions `ScriptX` and `ScriptY` are plotted separately in Figure 3.2, and we can see that the error level is too small to be visible. The total event took about 2.3 seconds, and the plotted functions each have 1401 discrete values. This is certainly a lot of resolution, but the curvature is rather high in places, and it turns out that even with the small error level involved, this level of resolution is not excessive.

Because the observed function looks reasonably smooth, the sampling rate is high, and the error level is low, one might be tempted to use the first *forward difference* $(y_{j+1} - y_j)/(t_{j+1} - t_j)$, or the *central difference* $(y_{j+1} - y_{j-1})/[(t_{j+1} - t_{j-1})]$, to estimate $Dx(t_j)$, but Figure 3.3 shows that the resulting derivative estimate for `ScriptX` is rather noisy. The second central difference estimate of D^2`ScriptX`

$$D^2x(t_j) \approx (y_{j+1} + y_{j-1} - 2y_j)/(\Delta t)^2$$

is shown in Figure 3.3 to be a disaster. The reason for this failure is precisely the high sampling rate for the data; taking differences between extremely close values magnifies the influence of error enormously. Press et al. (1999) comment on how simple differencing to estimate derivatives can go wrong even when functions are available analytically.

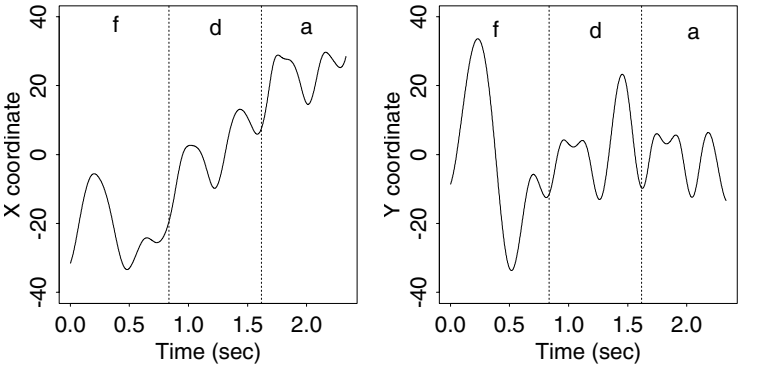

Figure 3.2. The X and Y coordinates for the handwriting sample plotted separately. Note the strongly periodic component with roughly two cycles per second.

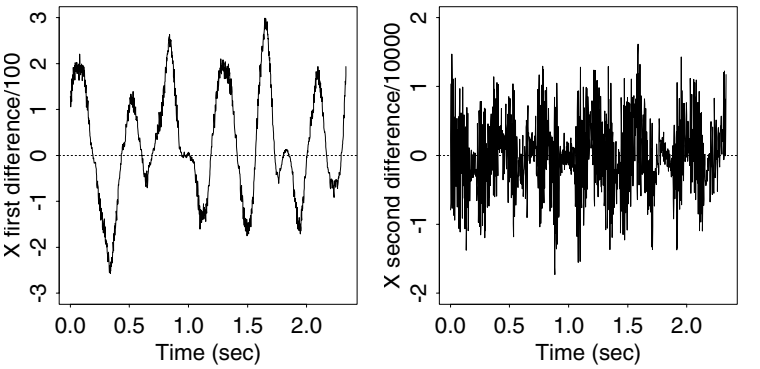

Figure 3.3. The first and second central differences for the X coordinate for the handwriting sample. The high sampling rate causes differencing to greatly magnify the influence of noise.

We will give a lot of attention to derivative estimation in this and the next chapter, including methods for estimating confidence intervals for derivative estimates. Many challenges remain, however, and there is plenty of room for improvement in existing techniques.

3.3 Representing functions by basis functions

A basis function system is a set of known functions ϕ_k that are mathematically independent of each other and that have the property that we can approximate arbitrarily well any function by taking a weighted sum or *linear combination* of a sufficiently large number K of these functions. The

most familiar basis function system is the collection of *monomials* that are used to construct power series,

$$1, t, t^2, t^3, \ldots, t^k, \ldots \ .$$

Right behind the power series in our list of golden oldies is the *Fourier series* system,

$$1,\ \sin(\omega t), \cos(\omega t),\ \sin(2\omega t), \cos(2\omega t),\ \sin(3\omega t), \cos(3\omega t), \ldots,$$

$$\sin(k\omega t), \cos(k\omega t), \ldots$$

Basis function procedures represent a function x by a linear expansion

$$x(t) = \sum_{k=1}^{K} c_k \phi_k(t) \tag{3.4}$$

in terms of K known basis functions ϕ_k.

By letting $\mathbf{c}$ indicate the vector of length K of the coefficients c_k and $\boldsymbol{\phi}$ as the functional vector whose elements are the basis functions ϕ_k, we can also express (3.4) in matrix notation as

$$x = \mathbf{c}'\boldsymbol{\phi} = \boldsymbol{\phi}'\mathbf{c} \ . \tag{3.5}$$

In effect, basis expansion methods represent the potentially infinite-dimensional world of functions within the finite-dimensional framework of vectors like c. The *dimension* of the expansion is therefore K. It would be a mistake, though, to conclude that functional data analysis in this case simply reduces to multivariate data analysis; a great deal also depends on how the basis system, $\boldsymbol{\phi}$, is chosen.

An exact representation or *interpolation* is achieved when $K = n$, in the sense that we can choose the coefficients c_k to yield $x(t_j) = y_j$ for each j. Therefore the degree to which the data y_j are *smoothed* as opposed to interpolated is determined by the number K of basis functions. Consequently, we do not view a basis system as defined by a fixed number K of parameters, but rather we see K as itself a parameter that we choose according to the characteristics of the data.

Ideally, basis functions should have features that match those known to belong to the functions being estimated. This makes it easier to achieve a satisfactory approximation using a comparatively small number K of basis functions. The smaller K is and the better the basis functions reflect certain characteristics of the data,

- the more degrees of freedom we have to test hypotheses and compute accurate confidence intervals,
- the less computation is required, and
- the more likely it is that the coefficients themselves can become interesting descriptors of the data from a substantive perspective.

Consequently, certain classic off-the-rack bases such as polynomials and Fourier series may be ill-advised in some applications; there is no such thing as a universally good basis.

The choice of basis is particularly important for a derivative estimate

$$D\hat{x}(t) = \sum_{k}^{K} \hat{c}_k D\phi_k(t) = \hat{\mathbf{c}}' D\boldsymbol{\phi}(t). \tag{3.6}$$

Bases that work well for function estimation may give rather poor derivative estimates. This is because an accurate representation of the observations may force $\hat{x}$ to have small but high-frequency oscillations that have dreadful consequences for its derivatives. Put more positively, one of the criteria for choosing a basis may be whether or not one or more of the derivatives of the approximation behave reasonably.

Chapter 21 touches on tailoring a basis to fit a particular problem. For now, we discuss some popular bases that are widely used in practice and when to use them. To summarize what follows, most functional data analyses these days involve either a Fourier basis for periodic data, or a B-spline basis for non-periodic data. Where derivatives are not required, wavelet bases are seeing more and more applications. Poor old polynomials are now the senior citizens of the basis world, relegated to only the simplest of functional problems.

3.4 The Fourier basis system for periodic data

Perhaps the best known basis expansion is provided by the Fourier series:

$$\hat{x}(t) = c_0 + c_1 \sin \omega t + c_2 \cos \omega t + c_3 \sin 2\omega t + c_4 \cos 2\omega t + \ldots \tag{3.7}$$

defined by the basis $\phi_0(t) = 1, \phi_{2r-1}(t) = \sin r\omega t$, and $\phi_{2r}(t) = \cos r\omega t$. This basis is periodic, and the parameter ω determines the period $2\pi/\omega$. If the values of t_j are equally spaced on $\mathcal{T}$ and the period is equal to the length of interval $\mathcal{T}$, then the basis is *orthogonal* in the sense that the cross product matrix $\boldsymbol{\Phi}'\boldsymbol{\Phi}$ is diagonal, and can be made equal to the identity by dividing the basis functions by suitable constants, $\sqrt{n}$ for $j = 0$ and $\sqrt{n/2}$ for all other j.

The Fast Fourier transform (FFT) makes it possible to find all the coefficients extremely efficiently when n is a power of 2 and the arguments are equally spaced, and in this case we can find both the coefficients c_k and all n smooth values at $x(t_j)$ in $O(n \log n)$ operations. This is one of the features that has made Fourier series the traditional basis of choice for long time series, but newer techniques such as B-splines and wavelets can match and even exceed this computational efficiency.

Derivative estimation in a Fourier basis is simple since

$$\begin{aligned} D\sin r\omega t &= r\omega\cos r\omega t \\ D\cos r\omega t &= -r\omega\sin r\omega t \end{aligned} \tag{3.8}$$

This implies that the Fourier expansion of Dx has coefficients

$$(0, c_1, -\omega c_2, 2\omega c_3, -2\omega c_4, \ldots)$$

and of D^2x has coefficients

$$(0, -\omega^2 c_1, -\omega^2 c_2, -4\omega^2 c_3, -4\omega^2 c_4, \ldots).$$

Similarly, we can find the Fourier expansions of higher derivatives by multiplying individual coefficients by suitable powers of $r\omega$, with sign changes and interchange of sine and cosine coefficients as appropriate.

The Fourier series is so familiar to statisticians, engineers and applied mathematicians that it is worth stressing its limitations. Invaluable though it may often be, neither it nor any other basis should be used uncritically. A Fourier series is especially useful for extremely stable functions, meaning functions where there are no strong local features and where the curvature tends to be of the same order everywhere. Ideally, the periodicity of the Fourier series should be reflected to some degree in the data, as is certainly the case for the temperature and gait data. Fourier series generally yield expansions which are uniformly smooth. But they are inappropriate to some degree for data known or suspected to reflect discontinuities in the function itself or in low order derivatives. A Fourier series is like margarine: It's cheap and you can spread it on practically anything, but don't expect that the result will be exciting eating. Nevertheless, we find many applications for Fourier series expansion in this book.

3.5 The spline basis system for open-ended data

Spline functions are the most common choice of approximation system for non-periodic functional data or parameters. They have more or less replaced polynomials, which in any case they contain within the system. Splines combine the fast computation of polynomials with substantially greater flexibility, often achieved with only a modest number of basis functions. Moreover, basis systems have been developed for spline functions that require an amount of computation that is proportional to n, a vital property since many applications involve thousands or millions of observations. In this section we first examine the structure of a spline function, and then describe the usual basis system used to construct it, the B-spline system.

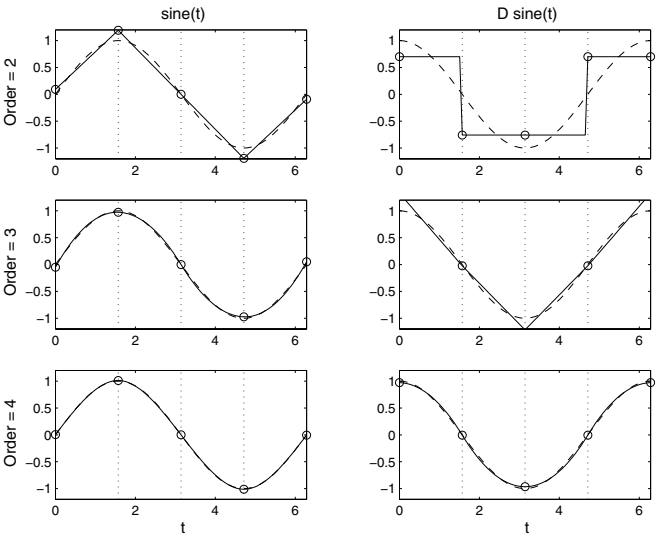

Figure 3.4. In the left panels the solid line indicates spline function of a particular order that fits the sine function shown as a dashed line. In the right panels the corresponding fits to its derivative, a cosine function, are shown. The vertical dotted lines are the interior breakpoints or knots defining the spline fits.

3.5.1 Spline functions and degrees of freedom

The anatomy of a spline is illustrated in Figure 3.4, where three spline functions are fit to $\sin(t)$ over the interval $[0, 2\pi]$ in the left panels, and where we also see the fit to its derivative, $\cos(t)$, in the right panels.

The first step in defining a spline is to divide the interval over which a function is to be approximated into L subintervals separated by values $\tau_\ell, \ell = 1, \ldots, L-1$ that are called *breakpoints* or *knots.* The former term is, strictly speaking, more correct for reasons that will be indicated shortly. We see in the figure that three breakpoints divide the interval into four subintervals. If we include the endpoints 0 and 2π as breakpoints, we may number them $\tau_0, \ldots, \tau_L$, where $L = 4$.

Over each interval, a spline is a polynomial of specified order m. The *order* of a polynomial is the number of constants required to define it, and is one more than its *degree*, its highest power. Thus, the spline in the top left of Figure 3.4 is piecewise linear, the center left spline is piecewise quadratic, and the bottom left piecewise cubic, corresponding to orders 2, 3 and 4, respectively. An order one spline can be seen in the top right panel, and this is a step function of degree zero.

Adjacent polynomials join up smoothly at the breakpoint which separates them for splines of order greater than one, so that the function values are constrained to be equal at their junction. Moreover, derivatives up to

order $m-2$ must also match up at these junctions. For example, for the commonly used order four cubic spline, the second derivative is a polygonal line and the third derivative is a step function. See a few paragraphs further on in this section, however, for an account of the possibility of reducing these smoothness constraints by using multiple knots at junction points.

We see in the top left panel of Figure 3.4, where an order two spline is fit to the sine curve, that only the function values join. Thus that there is one constraint on adjacent lines. Since there are two degrees of freedom in a line, and we have four lines, the total number of degrees of freedom in this line is calculated as follows. We count a total of 2×4 coefficients to define the four line segments, but we subtract one degree of freedom for each of the continuity constraints at each of the three junctions. This makes five in all.

Similarly, in the center left panel, the piecewise polynomials are quadratic, giving $3 \times 4 = 12$ coefficients, but this time both the function value and the first derivative join smoothly, so that we subtract six to get six remaining degrees of freedom. Finally, in the third row, where the polynomials are cubic, and where the function values, first derivatives and second derivatives must join, the accounting gives $4 \times 4 = 16$ less $3 \times 3 = 9$ constraints, leaving us with seven degrees of freedom. The rule is simple:

> The total number of degrees of freedom in the fit equals the order of the polynomials plus the the number of *interior* breakpoints.

If there are no interior knots, the spline reverts to being a simple polynomial.

We see that with increasing order comes a better and better approximation to both the sine and its derivative, and that by order four the fit is very good indeed. In fact, if we were to increase the order to five or beyond, we would also get a fine fit to the second derivative as well.

The main way to gain flexibility in a spline is to increase the number of breakpoints. Here we have made them equally spaced, but in general, we want more breakpoints over regions where the function exhibits the most complex variation, and fewer where the function is only mildly nonlinear. A subsidiary consideration is that we certainly do not want intervals that do not contain data, but then this seems reasonable since we cannot expect to capture a function's features without data.

We mentioned above that breakpoints are not quite the same thing as knots. This is because we can have two or more breakpoints that move together to coalesce or be coincident. When this happens, there is a loss of continuity condition for each additional coincident breakpoint. In this way, we can engineer abrupt changes in a derivative or even a function value at pre-specified breakpoints. The interested reader should consult de Boor (2001) for further details.

Thus, the term *breakpoint*, strictly speaking, refers to the number of unique knot values, while the term *knot* refers to the sequence of values at breakpoints, where some breakpoints can be associated with multiple knots. The knots are all distinct in most applications, and consequently breakpoints and knots are then the same thing. But we will encounter data input/output systems where the inputs are varied in a discrete step-wise way, and these will require coincident knots to model these sharp changes in level.

To review, a spline function is determined by two things: The order of the polynomial segments, and the knot sequence τ. The number of parameters required to define a spline function in the usual situation of one knot per breakpoint is the order plus the number of interior knots, $m + L - 1$.

3.5.2 The B-spline basis for spline functions

We have now defined a spline function, but have given no clue as to how to actually construct one. For this, we specify a system of basis functions $\phi_k(t)$, and these will have the following essential properties:

- Each basis function $\phi_k(t)$ is itself a spline function as defined by an order m and a knot sequence τ.
- Since a multiple of a spline function is still a spline function, and since sums and differences of splines are also splines, any linear combination of these basis functions is a spline function.
- Any spline function defined by m and τ can be expressed as a linear combination of these basis functions.

Although there are many ways that such systems can be constructed, the *B-spline* basis system developed by de Boor (2001) is the most popular, and code for working with B-splines is available in a wide range of programming languages, including R, S-PLUS and MATLAB®. Other spline basis systems are truncated power functions, M-splines and natural splines, and these and others are discussed by de Boor (2001) and Schumaker (1981).

Figure 3.5 shows the thirteen B-spline basis functions for an order four spline defined by the nine equally spaced interior breakpoints, which are also shown in this figure. Notice that each of the seven basis functions in the center only is positive over four adjacent sub-intervals. Because cubic splines have two continuous derivatives, each basis function makes a smooth transition to the regions over which it is zero. These central basis splines have the same shape because of the equal spacing of breakpoints; unequally spaced breakpoints would define splines varying in shape. The left-most three basis functions and their three right counterparts do differ in shape, but nevertheless are also positive over no more than four adjacent sub-intervals.

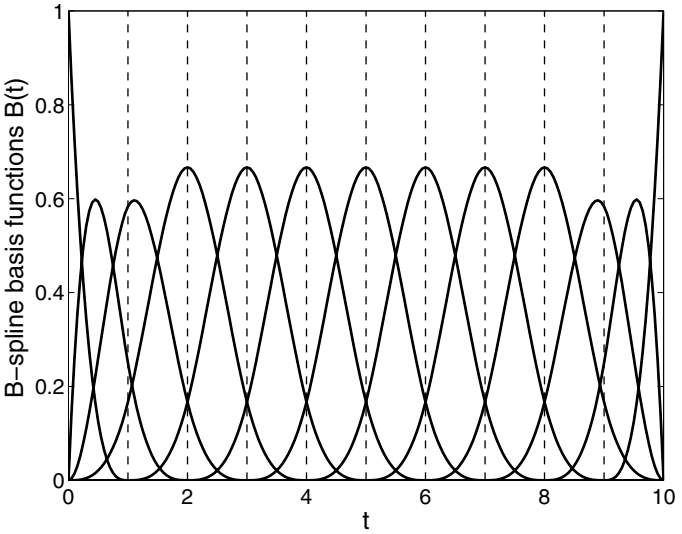

Figure 3.5. The thirteen basis functions defining an order four spline with nine interior knots, shown as vertical dashed lines.

The property that an order m B-spline basis function is positive over no more than m intervals, and that these are adjacent, is called the *compact support* property, and is of the greatest importance for efficient computation. If there are K B-spline basis functions, then the order K matrix of inner products of these functions will be band-structured, with only $m-1$ sub-diagonals above and below the main diagonal containing nonzero values. This means that no matter how large K is, and we will be dealing with values in the thousands, the computation of spline function can be organized so as to increase only linearly with K. Thus splines share the computational advantages of potentially *orthogonal* basis systems such as as Fourier and wavelet bases.

The three basis functions on the left and the three on the right are different. As we move from the left boundary towards the center, the intervals over which the basis functions are positive increase from one to four, but always make the same smooth twice-differentiable transition to the zero region. On the other hand, their transition to the left boundary varies in smoothness, with the left-most spline being discontinuous, the next being continuous only, and the third being once-differentiable. The same thing happens on the right side, but in reverse order. That we lose differentiability at the boundaries makes good sense, since we normally have no information about what the function we are estimating is doing beyond the interval on which we collect data. We therefore are allowing for the possibility that the function may be discontinuous beyond the boundaries.

This boundary behavior of B-spline basis functions is achieved by placing, in effect, m knots at the boundaries. That is, when B-splines are actually computed, the knot sequence τ is extended at each end to add an additional $m-1$ replicates of the boundary knot value. As we noted before, there are some applications where we do not want $m-2$ continuous derivatives at certain fixed points in the interior of the interval. This can be readily accommodated by B-splines. We place a knot at such fixed points, and then for each reduction in differentiability an additional knot is placed at that location as well. For example, if were were working with order four splines, and wanted the derivative to be able to change abruptly at a certain value of t but still wanted the fitted function to be continuous, we would place three knots at that value.

The notation $B_k(t,\tau)$ is often used to indicate the value at t of the B-spline basis function defined by the breakpoint sequence τ. Here k refers to the number of the largest knot at or to the immediate left of value t. The $m-1$ knots added to the initial breakpoint are also counted in this scheme, and this is consistent with the fact that the first m B-spline basis functions all have supports all beginning at the left boundary. This notation gives us $m+L-1$ basis functions, as required in the usual case where all interior knots are discrete. According to this notation, a spline function $S(t)$ with discrete interior knots is defined as

$$S(t) = \sum_{k=1}^{m+L-1} c_k B_k(t,\tau) \ . \tag{3.9}$$

It remains to give some guidance as to where the interior breakpoints or knots τ_ℓ should be positioned. Many applications default to equal spacing, which is fine as long as the data are relatively equally spaced. If they are not, it may be wiser to place a knot at every jth data point, were j is a number fixed in advance. This amounts to placing interior knots at the *quantiles* of the argument distribution. A special case is that of *smoothing splines* that we will take up in the next chapter, where a breakpoint is placed at each argument value. Finally, one can depart from either of these simple procedures to place more knots in regions known to contain high curvature, and fewer where there is less.

Figure 3.6 shows an example of using coincident knots to measurements of the level of a fluid in a tray in a oil refinery distillation column, previously shown in Figure 1.4. At time 67 a valve was turned and the flow of fluid into the tray changed abruptly, whereupon the fluid level increases rapidly at first, and then more and more slowly as it approaches its final value. It is clear that the first derivative should be discontinuous at time 67, but that the fluid level is essentially smooth elsewhere. These data were fit with B-splines of order four, with a single knot mid-way between times 0 and 67, three equally spaced knots between times 67 and 193, and three coincident knots at time 67. Now an order four spline has a third derivative that is

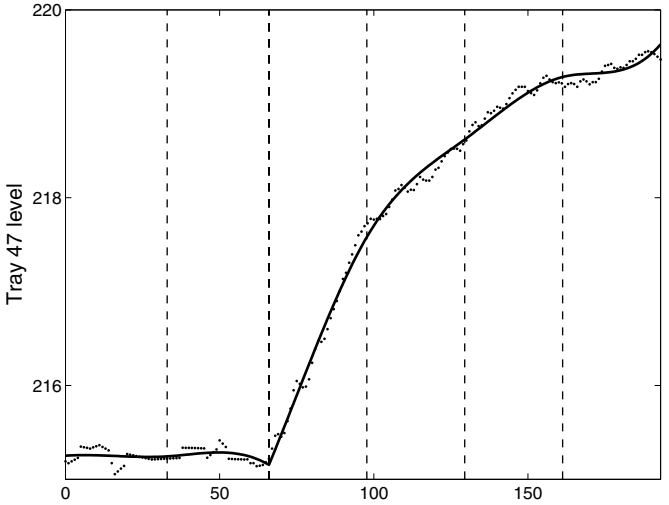

Figure 3.6. The oil refinery tray 47 level shown in Figure 1.4. The heavy smooth line is a fit to the data using B-spline basis functions with knots located as shown by the vertical dashed lines. There are three coincident knots at time 67 in order to achieve the discontinuity in the first derivative of the fit.

discontinuous at single knot locations, and, recalling that each additional coincident knot decreases the order of continuity by one, we achieve first derivative discontinuity at time 67. This can be seen in the smooth line fit to the data by the methods described in the next chapter. Go to Chapter 17 for further analyses of these data.

One possibly disconcerting feature of spline bases is that increasing K does not always improve certain aspects of the fit to the data. This is because, when the order of a spline is fixed, the function space defined by K B-splines is not necessarily contained within that defined by $K + 1$ B-splines. Complicated effects due to knot spacing relative to sampling points can result in a lower-dimensional B-spline system actually producing better results than a higher-dimensional system. However, if K is increased by either adding a new breakpoint to the current τ, or by increasing the order and leaving τ unchanged, then the K-space is contained within the $(K + 1)$-space.

There are data-driven methods for breakpoint positioning. Some approaches begin with a dense set of breakpoints, and then eliminate unneeded ones by an algorithmic procedure similar to variable selection techniques used in multiple regression. See, for example, Friedman and Silverman (1989). Alternatively, one can optimize the fitting criterion with respect to knot placement at the same time that one estimates the coefficients of the expansion. However, this can lead to computational problems,

since fitting criteria can vary in highly complex ways as a function of knot placement. Some useful techniques for improving knot placement are discussed by de Boor (2001).

It is not easy to find a readable introduction to splines, but the functional data analysis website, `www.functionaldata.org`, offers a beginner's treatment. The most comprehensive reference is de Boor (2001), which contains a wealth of information on computational as well as theoretical issues. But it is for advanced readers only, whereas Eubank (1999) and Green and Silverman (1994) are at a more intermediate level.

3.6 Other useful basis systems

We must not, however, forget about a number of other potentially important basis systems. In fact, two contrasting developments in recent years are having a large impact on data analysis. On the side of great mathematical sophistication we have wavelets that combine the frequency-specific approximating power of the Fourier series with the time- or spatially-localized features of splines. On the other hand, we have seen a fascinating resurgence of interest in exceedingly simple bases such as step functions (order one splines in effect) and polygons (order two splines) (Hastie, et al. 2001).

3.6.1 Wavelets

We can construct a basis for all functions on $(-\infty, \infty)$ that are square-integrable by choosing a suitable *mother wavelet* function ψ and then considering all dilations and translations of the form

$$\psi_{jk}(t) = 2^{j/2}\psi(2^j t - k)$$

for integers j and k. We construct the mother wavelet to ensure that the basis is orthogonal, in the sense that the integral of the product of any two distinct basis functions is zero. Typically, the mother wavelet has compact support, and hence so do all the basis functions. The wavelet basis idea is easily adapted to deal with functions defined on a bounded interval, most simply if periodic boundary conditions are imposed.

The wavelet expansion of a function f gives a *multiresolution analysis* in the sense that the coefficient of ψ_{jk} yields information about f near position $2^{-j}k$ on scale 2^{-j}, i.e., at frequencies near $c2^j$ for some constant c. Thus wavelets provide a systematic sequence of degrees of locality. In contrast to Fourier series, wavelet expansions cope well with discontinuities or rapid changes in behavior; only those basis functions whose support includes the region of discontinuity or other bad behavior are affected. This property, as well as a number of more technical mathematical results, means that it is often reasonable to assume that an observed function is well approximated

by an economical wavelet expansion with few non-zero coefficients, even if it displays sharp local features.

Suppose a function x is observed without error at $n = 2^M$ regularly spaced points on an interval $\mathcal{T}$. just as with the Fourier transformation, there is a discrete wavelet transform (DWT) which provides n coefficients closely related to the wavelet coefficients of the function x. We can calculate the DWT and its inverse in $O(n)$ operations, even faster than the $O(n \log n)$ of the FFT. As a consequence, most estimators based on wavelets can be computed extremely quickly, many of them in $O(n)$ operations.

Now suppose that the observations of x are subject to noise. The fact that many intuitively attractive classes of functions have economical wavelet expansions leads to a simple *nonlinear* smoothing approach: Construct the DWT of the noisy observations, and threshold it by throwing away the small coefficients in the expansion and possibly shrinking the large ones. The basic motivation of thresholding is the notion that any coefficient that is small is entirely noise and does not reflect any signal at all. This nonlinear thresholding has attractive and promising theoretical properties (see, for example, Donoho, Johnstone, Kerkyacharian and Picard, 1995), indicating that thresholded wavelet estimators should adapt well to different degrees of smoothness and regularity in the function being estimated.

3.6.2 Exponential and power bases

Exponential basis systems consist of a series of exponential functions,

$$e^{\lambda_1 t}, e^{\lambda_2 t}, \ldots, e^{\lambda_k t}, \ldots$$

where the *rate parameters* λ_k are all distinct, and often $\lambda_1 = 0$. Linear differential equations with constant coefficients have as solutions expansions in terms of exponential bases.

Power bases,

$$t^{\lambda_1}, t^{\lambda_2}, \ldots, t^{\lambda_k}, \ldots$$

likewise are important from time to time, often when t is strictly positive so that negative powers are possible.

3.6.3 Polynomial bases

The monomial basis $\phi_k(t) = (t - \omega)^k, k = 0, \ldots, K$ is also classic, where ω is a shift parameter that is usually chosen to be in the center of the interval of approximation. Care must be taken to avoid rounding error in the computations, since monomial values are more and more highly correlated as the degree increases. However, if the argument values t_j are equally spaced or can be chosen to exhibit a few standard patterns, orthogonal polynomial expansions can be obtained, implying $O((n+m)K)$ operations for all

smooth values. Otherwise we are condemned to contemplate $O((n+m)K^2)$ operations.

Like the Fourier series expansion, polynomials cannot exhibit very local features without using a large K. Moreover, polynomials tend to fit well in the center of the data but exhibit rather unattractive behavior in the tails. They are usually a poor basis for extrapolation or forecasting, for example.

Although derivatives of polynomial expansions are simple to compute, they are seldom satisfactory as estimators of the true derivative because of the rapid localized oscillation typical of high order polynomial fits.

3.6.4 The polygonal basis

Smoothing the observed rough data is not always necessary, and especially if our interest is not in the fit to the data itself, but rather in some functional parameter not directly connected to the data. In the chapters on the functional linear model, we will see that we can interpolate the data with a simple basis, and move the smoothing issue to where it belongs, namely the estimation of the functional parameter. In fact, polygonal or piecewise linear data fits have much to recommend them, and can even offer a crude estimate of the first derivative.

3.6.5 The step-function basis

Data mining problems often involve huge numbers of variables combined with phenomenal sample sizes. Because computational constraints can become critical, and we look for the simplest methods that will work. One of the great success stories in modern statistics has the usefulness of simple splits of variables into two categories, and the construction of tree-based representations of relationships or classification schemes. A split of variable values can be viewed as a functional transformation with a basis consisting of two step functions. This is, in effect, an order one B-spline system with a single interior knot. A recent reference on data mining in general that has considerable material on tree-based classification is Hastie, Tibshirani and Friedman (2001). It is somehow refreshing that we return to basics from time to time and rediscover that "effective" is not always the same thing as "sophisticated".

3.6.6 The constant basis

We shouldn't neglect the most humble of bases, the single basis function $1(t)$ whose value is one everywhere. It comes in handy surprisingly often. Firstly, it provides a useful point of reference or null hypothesis when we estimate regression coefficient functions for the functional linear model and elsewhere. Secondly, it is explicitly in systems like the Fourier, and

implicitly into B-spline bases. Finally, we can view a scalar observation as a functional datum whose value is the same everywhere, and consequently its value becomes the coefficient for the constant basis. Using this device we can, in effect, include most multivariate statistical techniques into functional data analysis in a seamless manner.

3.6.7 Empirical and designer bases

If choosing a basis that matches the characteristics of the data is important, can't we design our own basis systems? The answer is positive in two ways. First, we will discuss basis systems associated with differential equation models for functional data in Chapter 21, and the chapter preceding this will show how such models can be fit empirically to the data.

Designer bases can also be constructed empirically using *functional principal components analysis*, the subject of Chapters 8 to 10. Such bases have the property of optimizing the amount of variance in the data explained by basis systems of size K. If one wants the most compact basis possible with the sole objective of fitting the data, principal components analysis is usually the method that is first considered.

3.7 Choosing a scale for t

From the perspective of mathematics, the choice of unit of measurement for argument t may appear to be of no great consequence. But the implications for computation can be dramatic, and especially when we work with derivatives.

The two main bases that we intend to work with, the B-spline and Fourier systems, have *normalized* basis function, meaning that basis function values are bounded. In the case of B-splines, the bounds are zero and one, and at any point t the sum of B-spline basis functions that are nonzero at that point is exactly one. The only B-splines that attain the upper limit of one are those at the extreme ends of the interval. In the Fourier series case, function values are found within [-1,1].

As a result, if large number of basis functions are packed into a small interval, their derivatives are bound to be large. This is particularly easy to see in the Fourier series case, where the mth derivative of $\sin k\omega t$ will attain limits $\pm(k\omega)^m$. For example, if we opt to define the unit of time for the daily weather data to be one year, we decide to work with a saturated basis containing 365 basis functions, then we will be looking at values of the fourth derivative oscillating between $\pm 1.7 \times 10^{12}$, as opposed to about ± 1560 if we use the day as the unit of time.

The same applies to B-splines. For example, the handwriting data has 1401 sampling points equally spaced between zero and 2.3 seconds. On

this time scale, if we use 1405 basis functions of order six with knots at the time points, which is not an unreasonable proposal, we will see fourth derivative values of about 2^{13}. On the other hand, if we use a time scale of milliseconds, then we see the same derivatives reaching values of only about 20.

Why does this matter? We will at many points in our investigation want to combine derivatives of various orders. For example, in Chapter 5, we will use the fourth derivative to stabilize or smooth estimates of the second derivative, and will do this by combining within the same fitting criterion B-spline basis functions values with their fourth derivatives. When you try to add together quantities of the order of one with quantities of the order of 10^{12}, it is easy to run into prodigious rounding error problems if you are not careful. All this trouble can be avoided by using using a unit for t which is roughly equal to the period of oscillation of the most rapidly varying basis function that you will use. In any case, we tend to find that our clients do not take well to seeing plots or tables of quantities far beyond magnitudes that they can imagine.

3.8 Further reading and notes

We imagine that the Fourier series needs little introduction for most of our readers. Most introductory calculus texts cover the topic, and many branches of statistics apply it.

Spline functions are another thing entirely. We have not found many treatments that are for beginners, and have often been brought up short when asked for something to read. This is why we have supplied the rather lengthy account that this chapter contains, at the risk of boring spline experts. The introduction to splines in Hastie and Tibshirani (1990) has proven helpful, and Green and Silverman (1994) is useful those with more intermediate exposures to mathematics and statistics. Even after a revision, de Boor (2001) remains a challenging book, but is unequalled in its coverage of splines. Texts devoted to smoothing and nonparametric regression such as Eubank (1999) and Simonoff (1996) are also useful references. Schumaker (1981) is an important but more advanced treatment of splines. Wahba (1990) is often cited, but if you can understand that book, you shouldn't be reading these early chapters!

Wavelet bases are comparatively recent, and they have considerable promise in many functional data analysis contexts. For further reading, see Chui (1992), Daubechies (1992), Press et al. (1992), Nason and Silverman (1994), Donoho et al. (1995) and Johnstone and Silverman (1997), as well as the many references contained in these books and papers. An entire issue in 1999 of the *Philosophical Transactions of the Royal Society of London, Series A,* was devoted to wavelet applications and theory, and

the papers there by Silverman (1999) and Silverman and Vassilicos (1999), as well as Silverman (2000) are to be recommended to newcomers to this exciting field.

Polynomial and power bases appear often under other titles. Power series, treated in all calculus texts, and the Taylor and Maclaurin expansions found there are specialized methods for estimating polynomial expansions. Later in Chapter 21 we will consider ways of generalizing these important tools.