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

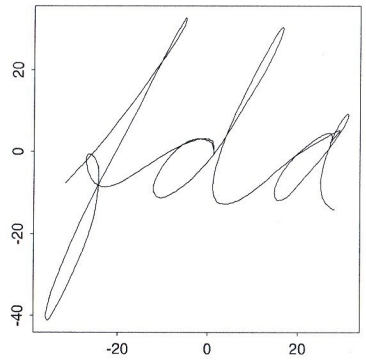

Figure 3.1. A sample of handwriting in which the X-Y coordinates are recorded 600 times per second.

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.

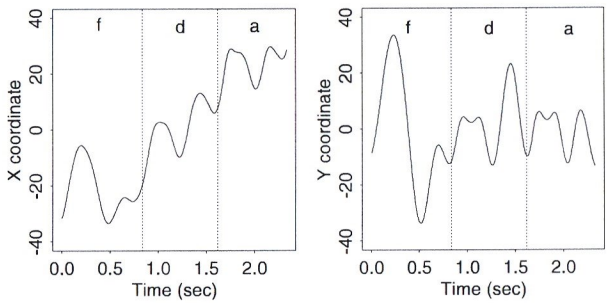

Figure 3.2. The X and Y coordinates for the handwriting sample plotted separately. Note the strongly periodic component with roughly two cycles per second.

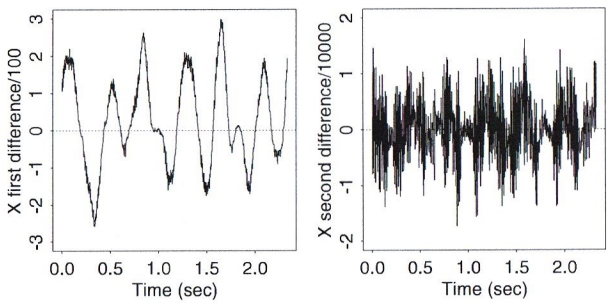

Figure 3.3. The first and second central differences for the X coordinate for the handwriting sample. The high sampling rate causes differencing to greatly magnify the influence of noise.

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.

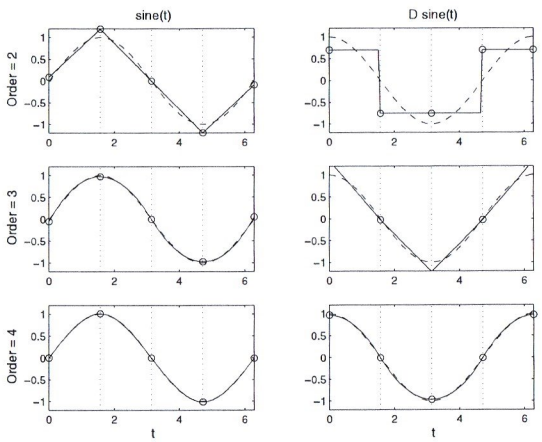

Figure 3.4. In the left panels the solid line indicates spline function of a particular order that fits the sine function shown as a dashed line. In the right panels the corresponding fits to its derivative, a cosine function, are shown. The vertical dotted lines are the interior breakpoints or knots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

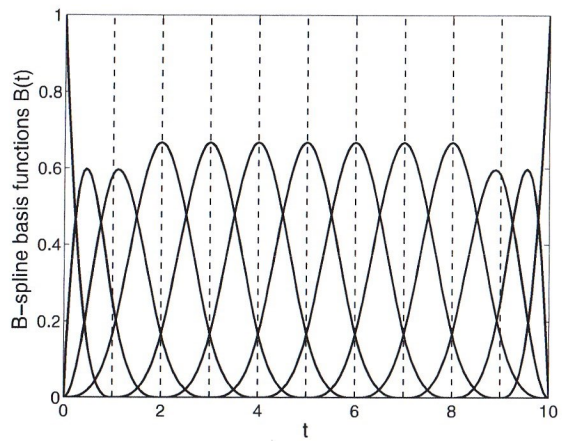

Figure 3.5. The thirteen basis functions defining an order four spline with nine interior knots, shown as vertical dashed lines.

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

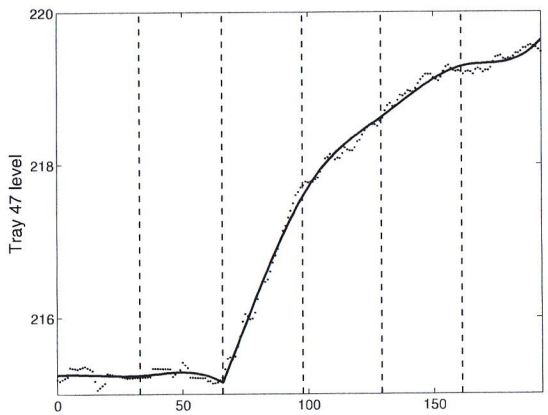

Figure 3.6. The oil refinery tray 47 level shown in Figure 1.4. The heavy smooth line is a fit to the data using B-spline basis functions with knots located as shown by the vertical dashed lines. There are three coincident knots at time 67 in order to achieve the discontinuity in the first derivative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

4

Smoothing functional data by least squares

4.1 Introduction

In this chapter and the next we turn to a discussion of specific smoothing methods. Our goal is to give enough information to those new to the topic of smoothing to launch a functional data analysis. Here we focus on the more familiar technique of fitting models to data by minimizing the sum of squared errors, or *least squares estimation*. This approach ties in functional data analysis with the machinery of multiple regression analysis. A number of tools taken from this classical field are reviewed here, and especially those that arise because least squares fitting defines a model whose estimate is a linear transformation of the data.

The treatment is far from comprehensive, however, and primarily because we will tend to favor the more powerful methods using roughness penalties to be taken up in the next chapter. Rather, notions such as degrees of freedom, sampling variance, and confidence intervals are introduced here as a first exposure to topics that will be developed in greater detail in Chapter 5.

4.2 Fitting data using a basis system by least squares

Recall that our goal is to fit the discrete observations $y_j, j = 1, \ldots, n$ using the model $y_j = x(t_j) + \epsilon_j$, and that we are using a basis function expansion

for $x(t)$ of the form

$$x(t) = \sum_{k}^{K} c_k \phi_k(t) = \mathbf{c}'\boldsymbol{\phi}.$$

The vector $\mathbf{c}$ of length K contains the coefficients c_k. Let us define the n by K matrix $\boldsymbol{\Phi}$ as containing the values $\phi_k(t_j)$.

4.2.1 Ordinary or unweighted least squares fits

A simple linear smoother is obtained if we determine the coefficients of the expansion c_k by minimizing the least squares criterion

$$\texttt{SMSSE}(\mathbf{y}|\mathbf{c}) = \sum_{j=1}^{n}[y_j - \sum_{k}^{K} c_k \phi_k(t_j)]^2. \tag{4.1}$$

The criterion is expressed more cleanly in matrix terms as

$$\texttt{SMSSE}(\mathbf{y}|\mathbf{c}) = (\mathbf{y} - \boldsymbol{\Phi}\mathbf{c})'(\mathbf{y} - \boldsymbol{\Phi}\mathbf{c}) \; . \tag{4.2}$$

The right side is also often written in functional notation as $\|\mathbf{y} - \boldsymbol{\Phi}\mathbf{c}\|^2$.

Taking the derivative of criterion $\texttt{SMSSE}(\mathbf{y}|\mathbf{c})$ with respect to $\mathbf{c}$ yields the equation

$$2\boldsymbol{\Phi}\boldsymbol{\Phi}'\mathbf{c} - 2\boldsymbol{\Phi}'\mathbf{y} = 0$$

and solving this for $\mathbf{c}$ provides the estimate $\hat{\mathbf{c}}$ that minimizes the least squares solution,

$$\hat{\mathbf{c}} = (\boldsymbol{\Phi}'\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}'\mathbf{y} \; . \tag{4.3}$$

The vector $\hat{\mathbf{y}}$ of fitted values is

$$\hat{\mathbf{y}} = \boldsymbol{\Phi}\hat{\mathbf{c}} = \boldsymbol{\Phi}(\boldsymbol{\Phi}'\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}'\mathbf{y} \; . \tag{4.4}$$

Simple least squares approximation is appropriate in situations where we assume that the residuals ϵ_j about the true curve are independently and identically distributed with mean zero and constant variance σ^2. That is, we prefer this approach when we assume the *standard model for error* discussed in Section 3.2.4.

As an example, Figure 4.1 shows the daily temperatures in Montreal averaged over 34 years, 1960–1994, for 101 days in the summer and 101 days in the winter. There is some higher frequency variation that seems to require fitting in addition to the smooth quasi-sinusoidal long-term trend. For example, there is a notable warming period from about January 16 to January 31 that is present in the majority of Canadian weather stations. The smooth fit shown in the figure was obtained with 109 Fourier basis functions, which would permit $108/2 = 54$ cycles per year, or roughly one per week. The curve seems to track nicely these shorter-term variations in temperature.

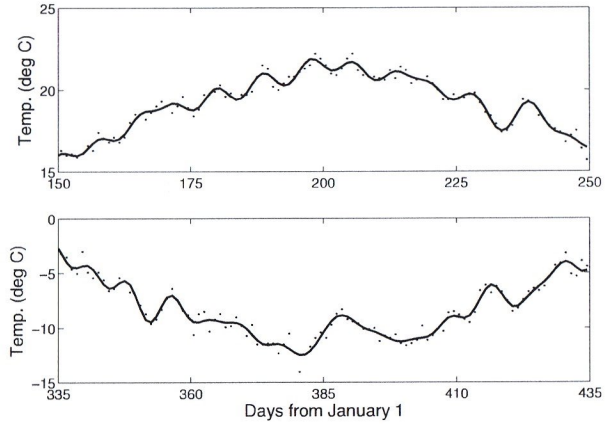

Figure 4.1. The upper panel shows the average daily temperatures for 101 days over the summer in Montreal, and the lower panel covers 101 winter days, with the day values extended into the following year. The solid curves are unweighted least squares smooths of the data using 109 Fourier basis functions.

4.2.2 Weighted least squares fits

As we noted in Section 3.2.4, the standard model for error will often not be realistic. To deal with nonstationary and/or autocorrelated errors, we may need to bring in a differential weighting of residuals by extending the least squares criterion to the form

$$\texttt{SMSSE}(\mathbf{y}|\mathbf{c}) = (\mathbf{y} - \mathbf{\Phi c})'\mathbf{W}(\mathbf{y} - \mathbf{\Phi c}) \tag{4.5}$$

where $\mathbf{W}$ is a symmetric positive definite matrix that allows for unequal weighting of squares and products of residuals.

Where do we get $\mathbf{W}$? If the variance-covariance matrix $\mathbf{\Sigma}_e$ for the residuals ϵ_j is known, then

$$\mathbf{W} = \mathbf{\Sigma}_e^{-1} .$$

In applications where an estimate of the complete $\mathbf{\Sigma}_e$ is not feasible, the covariances among errors are often assumed to be zero, and then $\mathbf{W}$ is diagonal with, preferably, reciprocals of the error variance associated with the y_j's in the diagonal. We will consider various ways of estimating $\mathbf{\Sigma}_e$ in Section 4.6.2. But in the meantime, we will not lose anything if we always include the weight matrix $\mathbf{W}$ in results derived from least squares estimation: we can always set it to $\mathbf{I}$ if the standard model is assumed.

The weighted least squares estimate $\hat{\mathbf{c}}$ of the coefficient vector $\mathbf{c}$ is

$$\hat{\mathbf{c}} = (\mathbf{\Phi}'\mathbf{W}\mathbf{\Phi})^{-1}\mathbf{\Phi}'\mathbf{W}\mathbf{y} . \tag{4.6}$$

Whether the approximation is by simple least squares or by weighted least squares, we can express what is to be minimized in the more universal functional notation $\texttt{SMSSE}(\mathbf{y}|\mathbf{c}) = \|\mathbf{y} - \mathbf{\Phi c}\|^2$.

4.3 A performance assessment of least squares smoothing

It may be helpful to see what happens when we apply least squares smoothing to a situation where we know what the right answer is, and can therefore check the quality of various aspects of the fit to the data, as well as the accuracy of data-driven bandwidth selection methods.

We turn now to the growth data, where a central issue was obtaining a good estimate of the acceleration or second derivative of the height function. For example, can we trust the acceleration curves displayed in Figure 1.1?

The parametric growth curve proposed by Jolicoeur (1992) has the following form:

$$h(t) = a\frac{\sum_{\ell=1}^{3}[b_\ell(t+e)]^{c_\ell}}{1+\sum_{\ell=1}^{3}[b_\ell(t+e)]^{c_\ell}}. \tag{4.7}$$

Jolicoeur's model is now known to be a bit too smooth, and especially in the period before the pubertal growth spurt, but it does offer a reasonable account of most growth records for the comparatively modest investment of estimating eight parameters, namely a, e and (b_ℓ, c_ℓ), $\ell = 1, 2, 3$. The model has been fit to the Fels growth data (Roche, 1992) by R. D. Bock (2000), and from these fits it has been possible to summarize the variation of parameter values for both genders reasonably well using a multivariate normal distribution. The average parameter values are $a = 164.7, e = 1.474, \mathbf{b} = (0.3071, 0.1106, 0.0816)', \mathbf{c} = (3.683, 16.665, 1.474)'$. By sampling from this distribution, we can simulate the smooth part of as many records as we choose.

The standard error of measurement has also been estimated for the Fels data as a function of age by one of the authors, and Figure 4.2 summarizes this relation. We see height measurements are noisier during infancy, where the standard error is about eight millimeters, but by age six or so, the error settles down to about five millimeters. Simulated noisy data were generated from the smooth curves by adding independent random errors having a mean of zero and standard deviation defined by this curve to the smooth values at the sampling points. The reciprocal of the square of this function was used to define the entries of the weight matrix $\mathbf{W}$, which in this case was diagonal. The sampling ages were those of the Berkeley data, namely

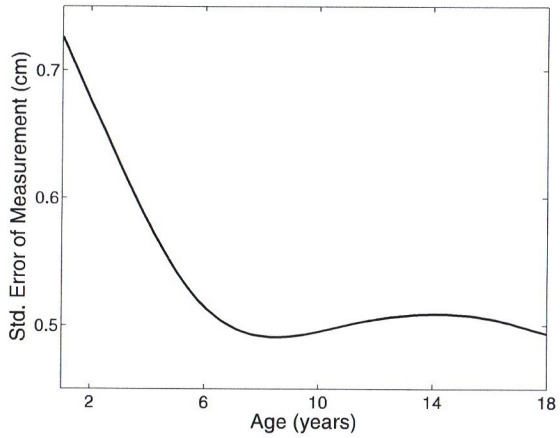

Figure 4.2. The estimated relation between the standard error of height measurements and age for females based on the Fels growth data.

quarterly between one and two years, annually between two to eight years, and twice a year after that to eighteen years of age.

We estimated the growth acceleration function by fitting a single set of data for a female. For the analysis, a set of 12 B-spline basis functions were used of order six and with equally spaced knots. We chose order six splines so that the acceleration estimate would be a cubic spline and hence smooth. A weighted least squares analysis was used with $\mathbf{W}$ being diagonal and with diagonal entries being the reciprocals of the squares of the standard errors shown in Figure 4.2.

Figure 4.3 shows how well we did. The maximum and minimum for the pubertal growth spurt are a little underestimated, and there are some peaks and valleys during childhood that aren't in the true curve. However, the estimate is much less successful at the lower and upper boundaries, and this example is a warning that we will have to look for ways to get better performance in these regions. On the whole, though, the important features in the true acceleration curve are reasonably reflected in the estimate.

4.4 Least squares fits as linear transformations of the data

The smoothing methods described in this chapter all have the property of being *linear*. Linearity simplifies computational issues considerably, and

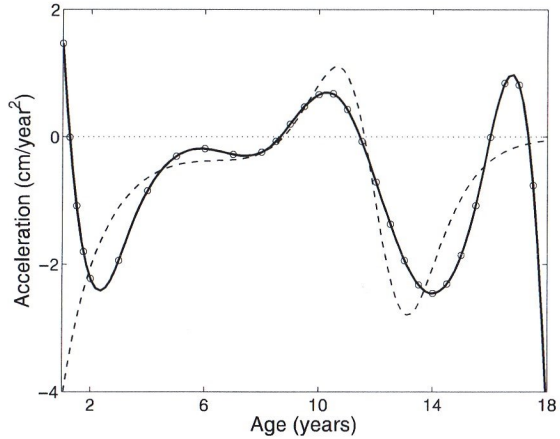

Figure 4.3. The solid curve is the estimated growth acceleration for a single set of simulated data, and the dashed curve is the errorless curve. The circles indicate the ages at which simulated observations were generated.

is convenient in a number of other ways. Most smoothing in practice gets done by linear procedures. Consequently, before we turn to other smoothing methods, we need to consider what linearity in a smoothing procedure means.

4.4.1 How linear smoothers work

A *linear smoother* estimates the function value $\hat{y}_j = \hat{x}(t_j)$ by a linear combination of the discrete observations

$$\hat{x}(t_j) = \sum_{\ell=1}^{n} S_j(t_\ell) y_\ell \ , \tag{4.8}$$

where $S_j(t_\ell)$ weights the ℓth discrete data value in order to generate the fit to y_j.

In matrix terms,

$$\hat{x}(\mathbf{t}) = \mathbf{S}\mathbf{y} \ , \tag{4.9}$$

where $\hat{x}(\mathbf{t})$ is a column vector containing the values of the estimate of function x at each sampling point t_j.

In the unweighted least squares case, for example, we see in (4.4) that

$$\mathbf{S} = \mathbf{\Phi}(\mathbf{\Phi}'\mathbf{\Phi})^{-1}\mathbf{\Phi}'. \tag{4.10}$$

In regression analysis, this matrix is often called the "hat matrix" because it converts the dependent variable vector $\mathbf{y}$ into its fit $\hat{\mathbf{y}}$.

In the context of least squares estimation, the smoothing matrix has the property of being a *projection matrix*. This means that it creates an image of data vector y on the space spanned by the columns of matrix $\mathbf{\Phi}$ such that the residual vector $\mathbf{e} = \mathbf{y} - \hat{\mathbf{y}}$ is orthogonal to the fit vector $\hat{y}$,

$$(\mathbf{y} - \hat{\mathbf{y}})'\hat{\mathbf{y}} = 0 \ .$$

This in turn implies that the smoothing matrix has the property $\mathbf{SS} = \mathbf{S}$, a relation called *idempotency*. In the next chapter on roughness-penalized least squares smoothing, we shall see that property does not hold.

The corresponding smoothing matrix for weighted least squares smoothing is

$$\mathbf{S} = \mathbf{\Phi}(\mathbf{\Phi}'\mathbf{W}\mathbf{\Phi})^{-1}\mathbf{\Phi}'\mathbf{W} \ . \tag{4.11}$$

Matrix $\mathbf{S}$ is still an orthogonal projection matrix, except that now the residual and fit vectors are orthogonal in the sense that

$$(\mathbf{y} - \hat{\mathbf{y}})'\mathbf{W}\hat{\mathbf{y}} = 0 \ .$$

In this case $\hat{\mathbf{y}} = \mathbf{S}\mathbf{y}$ is often said to be a *projection in the metric* $\mathbf{W}$.

Figure 4.4 shows the weights associated with estimating the growth acceleration curve in Figure 4.3 for ages six, twelve, and eighteen. For ages away from the boundaries, the weights have a positive peak centered on the age being estimated, and two negative side-lobes. For age twelve in the middle of the pubertal growth spurt for females, the observations receiving substantial weight, of either sign, range from ages seven to seventeen. This is in marked contrast to second difference estimates

$$D^2x(t_j) \approx \left(\frac{y_{j+1} - y_j}{t_{j+1} - t_j} - \frac{y_j - y_{j-1}}{t_j - t_{j+1}}\right)/(t_{j+1} - t_{j-1}),$$

which would only use three adjacent ages.

At the upper boundary, we see why there is likely to be considerable instability in the estimate. The final observation receives much more weight than any other value, and only observations back to age fifteen are used at all. The boundary estimate pools much less information than do interior estimates, and is especially sensitive to the boundary observations.

Many widely used smoothers are linear. The linearity of a smoother is a desirable feature for various reasons: The linearity property

$$\mathbf{S}(a\mathbf{y} + b\mathbf{z}) = a\mathbf{S}\mathbf{y} + b\mathbf{S}\mathbf{z}$$

is important for working out various properties of the smooth representation, and the simplicity of the smoother implies relatively fast computation. On the other hand, some nonlinear smoothers may be more adaptive to different behavior in different parts of the range of observation, and may be robust to outlying observations. Smoothing by the thresholded

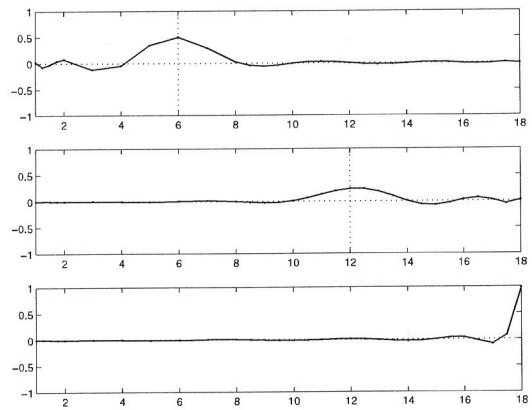

Figure 4.4. The top panel indicates how observations are weighted in order to estimate growth acceleration at age six in figure 4.3. The central panel shows the weights for age twelve, and the bottom for age eighteen. The dots indicate the ages at which simulated observations were generated.

wavelet transform, discussed in Section 3.6.1, is an important example of a nonlinear smoothing method.

Speed of computation can be critical: a smoother that is useful for a few hundred data points can be completely impractical for thousands. Smoothers that require a number of operations that is proportional to n to compute n smoothed values $\hat{x}(s_j)$, abbreviated $O(n)$ operations, are virtually essential for large n. If $\mathbf{S}$ is band-structured, meaning that only a small number K of values on either side of its diagonal in any row are nonzero, then $O(n)$ computation is assured.

4.4.2 The degrees of freedom of a linear smooth

We are familiar with the idea that the model for observed data offers an image of the data that has fewer *degrees of freedom* than are present in the original data. In most textbook situations, the concept of the degrees of freedom of a fit means simply the number of parameters estimated from the data that are required to define the model.

The notion of degrees of freedom applies without modification to data smoothing using least squares, where the number of parameters is the length K of the coefficient vector $\mathbf{c}$. The number of *degrees of freedom for error* is therefore $n - K$.

When we begin to use roughness penalty methods in Chapter 5, however, things will not be so simple, and we will need a more general way of computing the effective degrees of freedom of a smooth fit to the data, and consequently the corresponding degrees of freedom for error. We do this by using the "hat" matrix $\mathbf{S}$ by defining the degrees of freedom of the smooth fit to be

$$df = \text{trace}\,\mathbf{S} \tag{4.12}$$

where the trace of a square matrix means the sum of its diagonal elements. This more general definition yields exactly K for least squares fits, and therefore does not represent anything new. But this definition will prove invaluable in our later chapters.

There are also situations in which it may be more appropriate to use the alternative definition

$$df = \text{trace}\,(\mathbf{S}\mathbf{S}') \tag{4.13}$$

but most of the time (4.12) is employed. In any case, the two definitions give the same answer for least squares estimation.

4.5 Choosing the number K of basis functions

How do we choose the order of the expansion K? The larger K, the better the fit to the data, but of course we then risk also fitting noise or variation that we wish to ignore. On the other hand, if we make K too small, we may miss some important aspects of the smooth function x that we are trying to estimate.

4.5.1 The bias/variance trade-off

This trade-off can be expressed in another way. For large values of K, n the *bias* in estimating $x(t)$, that is

$$\texttt{Bias}[\hat{x}(t)] = x(t) - \text{E}[\hat{x}(t)], \tag{4.14}$$

is small. In fact, if the notion of additive errors having expectation zero expressed in (3.1) holds, then we know that the bias will be zero for $K = n$.

But of course, that is only half of the story. One of the main reasons that we do smoothing is to reduce the influence of noise or ignorable variation on the estimate $\hat{x}$. Consequently we are also interested in the *variance of estimate*

$$\texttt{Var}[\hat{x}(t)] = \text{E}[\{\hat{x}(t) - \text{E}[\hat{x}(t)]\}^2] \; . \tag{4.15}$$

If $K = n$, this is almost certainly going to be unacceptably high. Reducing variance leads us to look for smaller values of K, but of course not so small

as to make the bias unacceptable. The worse the signal-to-noise ratio in the data, the more reducing sampling variance will outweigh controlling bias.

One way of expressing what we really want to achieve is *mean-squared error*

$$\texttt{MSE}[\hat{x}(t)] = \text{E}[\{\hat{x}(t) - x(t)\}^2] \; , \tag{4.16}$$

also called the $\mathcal{L}^2$ *loss function*. In most applications we can't actually minimize this since we have no way of knowing what $x(t)$ is without using the data. However, one of the most important equations in statistics links mean squared error to bias and sampling variance by the simple additive decomposition

$$\texttt{MSE}[\hat{x}(t)] = \texttt{Bias}^2[\hat{x}(t)] + \texttt{Var}[\hat{x}(t)] \; . \tag{4.17}$$

What this relation tells us is that it would be worthwhile to tolerate a little bias if the result is a big reduction in sampling variance. In fact, this is almost always the case, and is the fundamental reason for smoothing data in order to estimate functions. We will return to this matter in Chapter 5.

Figure 4.5 shows some total squared error measures as a function of various numbers of basis functions. The measures were computed by summing mean squared error, sampling variance and squared bias across the ages ranging from three to sixteen. This range was used to avoid ages near the boundaries, where the curve estimates tend to have much greater error levels. The results are based on smoothing 10,000 random samples constructed in the same manner as that in Figure 4.3.

Notice that the measures for sampling variance and squared bias sum to those for mean squared error, as in (4.17). Sampling variance increases rapidly when we use too many basis functions, but squared bias tends to decay more gently to zero at the same time. We see there that the best results for totaled mean squared error are obtained with ten and twelve basis functions, and we broke the tie by opting for the result with the least bias.

It may see surprising that increasing K does not always decrease bias. If so, recall that, when the order of a spline is fixed and knots are equally spaced, K B-splines do not span a space that lies 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.

Although the decomposition mean squared error (4.17) is helpful for expressing the bias/variance tradeoff in a neat way, the principle applies more widely. In fact, there are many situations where it is preferable to use other loss functions. For example, minimizing $\text{E}[|\hat{x}(t) - x(t)|]$, called the $\mathcal{L}^1$ norm, is more effective if the data contain outliers. For this and nearly any fitting criterion or loss function for smoothing, we can assume that when bias goes down, sampling variance goes up, and some bias must be tolerated to achieve a stable estimate of the smooth trend in the data.

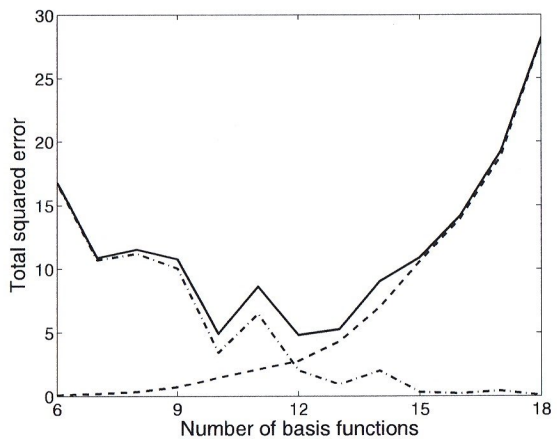

Figure 4.5. The heavy solid line indicates mean squared error totaled across the ages of observation between three and sixteen. The dashed line shows the totaled sampling variance, and the dotted-dashed line shows the totaled squared bias.

4.5.2 Algorithms for choosing K

The vast literature on multiple regression contains many ideas for deciding how many basis functions to use. For example, *stepwise variable selection* would proceed in a step-up fashion by adding basis functions one after another, testing at each step whether the added function significantly improves fit, and also checking that the functions already added continue to play a significant role. Conversely, *variable-pruning* methods are often used for high-dimensional models, and work by starting with a generous choice of K and dropping a basis function on each step that seems to not account for a substantial amount of variation.

These methods all have their limitations, and are often abused by users who do not appreciate these problems. The fact that there is no one gold standard method for the variable selection problem should warn us at this point that we face a difficult task in attempting to fix model dimensionality. The discrete character of the K-choice problem is partly to blame, and the methods described in Chapter 5 providing a continuum of smoothing levels will prove helpful.

4.6 Computing sampling variances and confidence limits

4.6.1 Sampling variance estimates

The estimation of the coefficient vector $\mathbf{c}$ of the basis function expansion $x = \mathbf{c}'\boldsymbol{\phi}$ by minimizing least squares defines a linear mapping (4.6) from the raw data vector $\mathbf{y}$ to the estimate. With this mapping in hand, it is a relatively simple matter to compute the sampling variance of the coefficient vector, and of anything that is linearly related to it.

We begin with the fact that if a random variable y is normally distributed with a variance-covariance matrix $\boldsymbol{\Sigma}_y$, then the random variable $\mathbf{Ay}$ defined by any matrix $\mathbf{A}$ has the variance-covariance matrix

$$\mathtt{Var}[\mathbf{Ay}] = \mathbf{A}\boldsymbol{\Sigma}_y\mathbf{A}' \; . \tag{4.18}$$

Now in this and other linear modelling situations that we will encounter, the model for the data vector $\mathbf{y}$, in this case $x(\mathbf{t})$, is regarded as a fixed effect having zero variance. Consequently, the variance-covariance matrix of y using the model $\mathbf{y} = x(\mathbf{t}) + \boldsymbol{\epsilon}$ is the variance-covariance matrix $\boldsymbol{\Sigma}_e$ of the residual vector $\boldsymbol{\epsilon}$. We must in some way use the information in the actual residuals to replace the population quantity $\boldsymbol{\Sigma}_e$ by a reasonable sample estimate $\hat{\boldsymbol{\Sigma}}_e$.

For example, to compute the sampling variances and covariances of the coefficients themselves in $\mathbf{c}$, we use that fact that in this instance

$$\mathbf{A} = (\boldsymbol{\Phi}'\mathbf{W}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}'\mathbf{W} \; .$$

to obtain

$$\mathtt{Var}[\mathbf{c}] = (\boldsymbol{\Phi}'\mathbf{W}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}'\mathbf{W}\boldsymbol{\Sigma}_e\mathbf{W}\boldsymbol{\Phi}(\boldsymbol{\Phi}'\mathbf{W}\boldsymbol{\Phi})^{-1} \; . \tag{4.19}$$

When the standard model is assumed, $\boldsymbol{\Sigma}_e = \sigma^2\mathbf{I}$, and if unweighted least squares is used, then we obtain the simpler result that appears in textbooks on regression analysis

$$\mathtt{Var}[\mathbf{c}] = \sigma^2(\boldsymbol{\Phi}'\boldsymbol{\Phi})^{-1} \; . \tag{4.20}$$

However, in our functional data analysis context there will seldom be much interest in interpreting the coefficient vector $\mathbf{c}$ itself. Rather, we will want to know the sampling variance of some quantity computed from these coefficients. For example, we might want to know the sampling variance of the the fit to the data defined by $x(t) = \boldsymbol{\phi}(t)'\mathbf{c}$. Since we now have in hand the sampling variance of $\mathbf{c}$ through (4.19) or (4.20), we can simply apply result (4.18) again to get

$$\mathtt{Var}[\hat{x}(t)] = \boldsymbol{\phi}(t)'\mathtt{Var}[\mathbf{c}]\boldsymbol{\phi}(t) \tag{4.21}$$

and the variances of all the fitted values corresponding to the sampling values t_j are in the diagonal of the matrix

$$\mathtt{Var}[\hat{\mathbf{y}}] = \mathbf{\Phi}\mathtt{Var}[\mathbf{c}]\mathbf{\Phi}'$$

which, in the standard model/unweighted least squares case, and using (4.10), reduces to

$$\mathtt{Var}[\hat{\mathbf{y}}] = \sigma^2\mathbf{\Phi}(\mathbf{\Phi}'\mathbf{\Phi})^{-1}\mathbf{\Phi}' = \sigma^2\mathbf{S}\ .$$

4.6.2 Estimating $\mathbf{\Sigma}_e$

Clearly our estimates of sampling variances are only as good as our estimates of the variances and covariances among the residuals ϵ_j.

When we are smoothing a single curve, the total amount of information involved is insufficient for much more than estimating either a single constant variance σ^2 assuming the standard model for error, or at most a variance function with values $\sigma^2(t)$, that has fairly mild variation over t. It is important to use methods which produce relatively unbiased estimate of variance in order to avoid underestimating sampling variance. For example, if the standard model for error is accepted,

$$s^2 = \frac{1}{n-K}\sum_j^n (y_j - \hat{y}_j)^2 \tag{4.22}$$

is much preferred as an estimate of σ^2 than the maximum likelihood estimate that involves dividing by n. In fact, we shall see in the next chapter that this estimate is related to a popular more general method for choosing smoothing level called *generalized cross-validation.*

One reasonable strategy for choosing K is to add basis functions until s^2 fails to decrease substantially. Figure 4.6 shows how s decreases to a value of about 0.56 degrees Celsius by the time we use 109 Fourier basis functions for smoothing the Montreal temperature data shown in Figure 4.1. There are places where s^2 is even lower, but we worried that the minimum at 240 basis functions corresponded to over-fitting the data.

A common strategy for estimating at least a limited number of covariances in $\mathbf{\Sigma}_e$ given a small N, or even $N = 1$, is to assume an autoregressive (AR) structure for the residuals. This is often realistic, since adjacent residuals are frequently correlated because they are mutually influenced by unobserved variables. For example, the weather on one day is naturally likely to be related to the weather on the previous day because of the influence of large slow-moving low or high pressure zones. An intermediate level text on regression analysis such as Draper and Smith (1998) can be consulted for details on how to estimate AR structures among residuals.

When a substantial number N of replicated curves are available, as in the growth curve data and Canadian weather data, we can attempt more sophisticated and detailed estimates of $\mathbf{\Sigma}_e$. For example, we may opt for

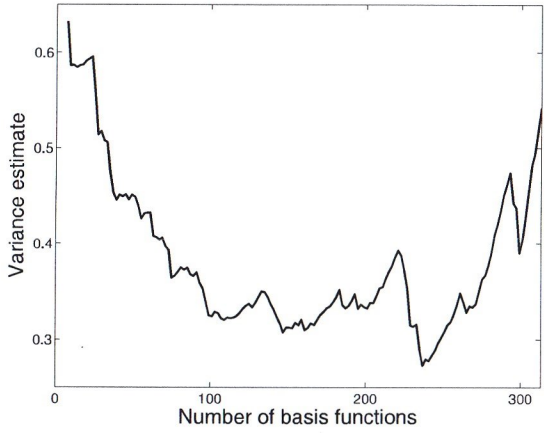

Figure 4.6. The relation between the number of Fourier basis functions and the unbiased estimate of the residual variance (4.22) in fitting the Montreal temperature data.

estimating the entire variance-covariance matrix from the N by n matrix $\mathbf{E}$ of residuals by

$$\hat{\mathbf{\Sigma}}_e = (N-1)^{-1}\mathbf{E}'\mathbf{E}.$$

However, even then, an estimate of a completely unrestricted $\mathbf{\Sigma}_e$ requires the estimation of $n(n-1)/2$ variances and covariances from N replications, and it is unlikely that data with the complexity of the daily weather records would ever have N sufficiently large to do this accurately.

4.6.3 Confidence limits

Confidence limits are typically computed by adding and subtracting a multiple of the standard errors, that is, the square root of the sampling variances, to the actual fit. For example, 95% limits correspond to about two standard errors up and down from a smooth fit. These standard errors are estimated using (4.21). Confidence limits on fits computed in this way are called *point-wise* because they reflect confidence regions for *fixed* values of t rather than regions for the curve as a whole.

Figure 4.7 shows the temperatures during the 16 days over which the January thaw takes place in Montreal, along with the smooth to the data and 95% point-wise confidence limits on the fit. The standard error of the estimated fit was 0.26 degrees Celsius.

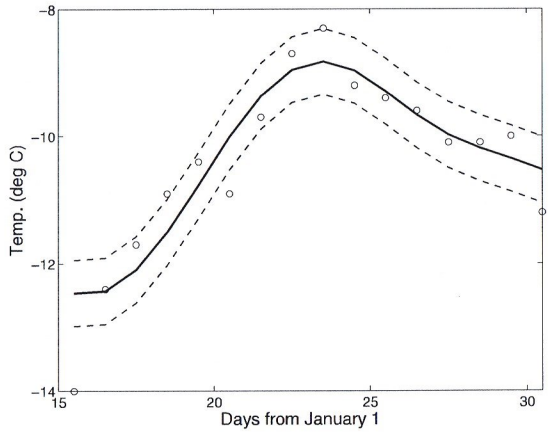

Figure 4.7. The temperatures over the mid-winter thaw for the Montreal temperature data. The solid line is the smooth curve estimated in Figure 4.1 and the lower and upper dashed lines are estimated 95% point-wise confidence limits for this fit.

We will have much to say in the next chapter and elsewhere about the hazards of placing too much faith in sampling variances and confidence limits estimated in these ways. But we should at least note two important ways in which confidence limits computed in this way may be problematic. First, it is implicitly assumed that K is a fixed constant, but the reality is that K for smoothing problems is more like a parameter estimated from the data, and consequently the size of these confidence limits does not reflect the uncertainty in our knowledge of K. Secondly, the smooth curve to which we add and subtract multiples of the standard error to get point-wise limits is itself subject to bias, and especially in regions of high curvature. We can bet, for example, that the solid curve in Figure 4.7 is too low on January 23rd, the center of the January thaw. Thus, the confidence limits calculated in this way are themselves biassed, and the region covered by them may not be quite as advertised.

4.7 Fitting data by localized least squares

For a smoothing method to make any sense at all, the value of the function estimate at a point t must be influenced mostly by the observations near t. This feature is an implicit property of the estimators we have considered

so far. In this section, we consider estimators where the local dependence is made more explicit by means of local weight functions.

Keeping within the domain of linear smoothing means that our estimate of the value of function x at argument t_j is of the form

$$x(t_j) = \sum_{\ell}^{n} w_\ell y_\ell \ .$$

It seems intuitively reasonable that the weights w_ℓ will only be relatively large for sampling values t_ℓ fairly close to the target value t_j. And, indeed, this tends to hold for the basis function smoothers (4.10) and (4.11).

We now look at smoothing methods that make this *localized weighting principle* explicit. The localizing weights w_j are simply constructed by a location and scale change of a *kernel* function with values $\texttt{Kern}(u)$. This kernel function is designed to have most of its mass concentrated close to 0, and to either decay rapidly or disappear entirely for $|u| \geq 1$. Three commonly used kernels are

Uniform:	$\texttt{Kern}(u) = 0.5$ for $\lvert u\rvert \leq 1$,	0 otherwise
Quadratic:	$\texttt{Kern}(u) = 0.75(1-u^2)$ for $\lvert u\rvert \leq 1$,	0 otherwise
Gaussian:	$\texttt{Kern}(u) = (2\pi)^{-1/2}\exp(-u^2/2)$.	

If we then define weight values to be

$$w_\ell(t) = \texttt{Kern}\left(\frac{t_\ell - t_j}{h}\right) \ , \tag{4.23}$$

then substantially large values $w_\ell(t)$ as a function of ℓ are now concentrated for t_ℓ in the vicinity of t_j. The degree of concentration is controlled by the size of h. The concentration parameter h is usually called the *bandwidth* parameter, and small values imply that only observations close to t receive any weight, while large h means that a wide-sweeping average uses values that are a considerable distance from t.

4.7.1 Kernel smoothing

The simplest and classic case of an estimator that makes use of local weights is the *kernel estimator*. The estimate at a given point is a linear combination of local observations,

$$\hat{x}(t) = \sum_{j}^{n} S_j(t) y_j \tag{4.24}$$

for some suitably defined weight functions S_j. Probably the most popular kernel estimator the Nadaraya-Watson estimator (Nadaraya, 1964; Watson,

8

Principal components analysis for functional data

8.1 Introduction

For many reasons, principal components analysis (PCA) of functional data is a key technique to consider. First, our own experience is that, after the preliminary steps of registering and displaying the data, the user wants to explore that data to see the features characterizing typical functions. Some of these features are expected to be there, for example the sinusoidal nature of temperature curves, but other aspects may be surprising. Some indication of the complexity of the data is also required, in the sense of how many types of curves and characteristics are to be found. Principal components analysis serves these ends admirably, and it is perhaps also for these reasons that it was the first method to be considered in the early literature on FDA.

Just as for the corresponding matrices in the classical multivariate case, the variance-covariance and correlation functions can be difficult to interpret, and do not always give a fully comprehensible presentation of the structure of the variability in the observed data directly. The same is true, of course, for variance-covariance and correlation matrices in classical multivariate analysis. A principal components analysis provides a way of looking at covariance structure that can be much more informative and can complement, or even replace altogether, a direct examination of the variance-covariance function.

PCA also offers an opportunity to consider some issues that reappear in subsequent chapters. For example, we consider immediately how PCA is

defined by the notion of a linear combination of function values, and why this notion, at the heart of most of multivariate data analysis, requires some care in a functional context. A second issue is that of *regularization*; for many data sets, PCA of functional data is more revealing if some type of smoothness is required of the principal components themselves. We consider this topic in detail in Chapter 9.

8.2 Defining functional PCA

8.2.1 PCA for multivariate data

The central concept exploited over and over again in multivariate statistics is that of taking a linear combination of variable values,

$$f_i = \sum_{j=1}^{p} \beta_j x_{ij}, \ i = 1, \ldots, N, \tag{8.1}$$

where β_j is a weighting coefficient applied to the observed values x_{ij} of the jth variable. We can express (8.1) as

$$f_i = \beta' x_i, \ i = 1, \ldots, N, \tag{8.2}$$

where β is the vector $(\beta_1, \ldots, \beta_p)'$ and x_i is the vector $(x_{i1}, \ldots, x_{ip})'$.

In the multivariate situation, we choose the weights so as to highlight or display types of variation that are very strongly represented in the data. Principal components analysis can be defined in terms of the following stepwise procedure, which defines sets of normalized weights that maximize variation in the f_i's:

1. Find the weight vector $\boldsymbol{\xi}_1 = (\xi_{11}, \ldots, \xi_{p1})'$ for which the linear combination values
$$f_{i1} = \sum_j \xi_{j1} x_{ij} = \boldsymbol{\xi}_1' x_i$$
have the largest possible mean square $N^{-1} \sum_i f_{i1}^2$ subject to the constraint
$$\sum_j \xi_{j1}^2 = \|\boldsymbol{\xi}_1\|^2 = 1.$$
2. Carry out second and subsequent steps, possibly up to a limit of the number of variables p. On the mth step, compute a new weight vector $\boldsymbol{\xi}_m$ with components ξ_{jm} and new values $f_{im} = \boldsymbol{\xi}_m' x_i$. Thus, the values f_{im} have maximum mean square, subject to the constraint $\|\boldsymbol{\xi}_m\|^2 = 1$ and the $m-1$ additional constraint(s)
$$\sum_j \xi_{jk} \xi_{jm} = \boldsymbol{\xi}_k' \boldsymbol{\xi}_m = 0, \ k < m.$$

The motivation for the first step is that by maximizing the mean square, we are identifying the strongest and most important mode of variation in the variables. The unit sum of squares constraint on the weights is essential to make the problem well defined; without it, the mean squares of the linear combination values could be made arbitrarily large. On second and subsequent steps, we seek the most important modes of variation again, but require the weights defining them to be orthogonal to those identified previously, so that they are indicating something new. Of course, the amount of variation measured in terms of $N^{-1}\sum_i f_{im}^2$ will decline on each step. At some point, usually well short of the maximum index p, we expect to lose interest in modes of variation thus defined.

The definition of principal components analysis does not actually specify the weights uniquely; for example, it is always possible to change the signs of all the values in any vector $\boldsymbol{\xi}_m$ without changing the value of the variance that it defines.

The values of the linear combinations f_{im} are called *principal component scores* and are often of great help in describing what these important components of variation mean in terms of the characteristics of specific cases or replicates.

To be sure, the mean is a very important aspect of the data, but we already have an easy technique for identifying it. Therefore, we usually subtract the mean for each variable from corresponding variable values before doing PCA. When this is done, maximizing the mean square of the principal component scores corresponds to maximizing their sample variance.

8.2.2 *Defining PCA for functional data*

How does PCA work in the functional context? The counterparts of variable values are function values $x_i(s)$, so that the discrete index j in the multivariate context has been replaced by the continuous index s. When we were considering vectors, the appropriate way of combining a weight vector β with a data vector x was to calculate the inner product

$$\beta' x = \sum_j \beta_j x_j.$$

When β and x are functions $\beta(s)$ and $x(s)$, summations over j are replaced by integrations over s to define the inner product

$$\int \beta x = \int \beta(s)x(s)\,ds. \tag{8.3}$$

Within the principal components analysis, the weights β_j now become functions with values $\beta_j(s)$. Using the notation (8.3), the principal

component scores corresponding to weight β are now

$$f_i = \int \beta x_i = \int \beta(s)x_i(s)\,ds. \tag{8.4}$$

For the rest of our discussion, we will often use the short form $\int \beta x_i$ for integrals in order to minimize notational clutter.

In the first functional PCA step, the weight function $\xi_1(s)$ is chosen to maximize $N^{-1}\sum_i f_{i1}^2 = N^{-1}\sum_i(\int \xi_1 x_i)^2$ subject to the continuous analogue $\int \xi_1(s)^2 ds = 1$ of the unit sum of squares constraint. This time, the notation $\|\xi_1\|^2$ is used to mean the squared norm $\int \xi_1(s)^2 ds = \int \xi_1^2$ of the *function* ξ_1.

Postponing computational details until Section 8.4, now consider as an illustration in the upper left panel in Figure 8.1. This displays the weight function ξ_1 for the Canadian temperature data after the mean across all 35 weather stations has been removed from each station's monthly temperature record. Although ξ_1 is positive throughout the year, the weight placed on the winter temperatures is about four times that placed on summer temperatures. This means that the greatest variability between weather stations will be found by heavily weighting winter temperatures, with only a light contribution from the summer months; Canadian weather is most variable in the wintertime, in short. Moreover, the percentage 89.3% at the top of the panel indicates that this type of variation strongly dominates all other types of variation. Weather stations for which the score f_{i1} is high will have much warmer than average winters combined with warm summers, and the two highest scores are in fact assigned to Vancouver and Victoria on the Pacific Coast. To no one's surprise, the largest negative score goes to Resolute in the High Arctic.

As for multivariate PCA, the weight function ξ_m is also required to satisfy the orthogonality constraint(s) $\int \xi_k \xi_m = 0$, $k < m$ on subsequent steps. Each weight function has the task of defining the most important mode of variation in the curves subject to each mode being orthogonal to all modes defined on previous steps. Note again that the weight functions are defined only to within a sign change.

The weight function ξ_2 for the temperature data is displayed in the upper right panel of Figure 8.1. Because it must be orthogonal to ξ_1, we cannot expect that it will define a mode of variation in the temperature functions that will be as important as the first. In fact, this second mode accounts for only 8.3% of the total variation, and consists of a positive contribution for the winter months and a negative contribution for the summer months, therefore corresponding to a measure of uniformity of temperature through the year. On this component, one of the highest scores f_{i2} goes to Prince Rupert, also on the Pacific coast, for which there is comparatively low discrepancy between winter and summer. Prairie stations such as Winnipeg, on the other hand, have hot summers and very cold winters, and receive large negative second component scores.

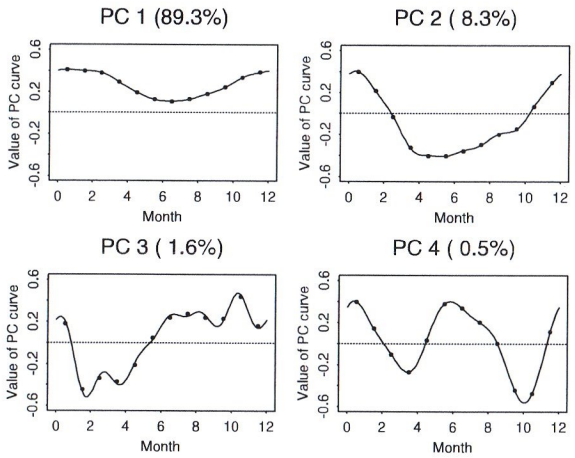

Figure 8.1. The first four principal component curves of the Canadian temperature data estimated by two techniques. The points are the estimates from the discretization approach, and the curves are the estimates from the expansion of the data in terms of a 12-term Fourier series. The percentages indicate the amount of total variation accounted for by each principal component.

The third and fourth components account for small proportions of the variation, since they are required to be orthogonal to the first two as well as to each other. At this point they are difficult to interpret, but we look at techniques for understanding them in Section 8.3.

Displays such as Figure 8.1 can remind one of the diagrams of modes of vibration in a string fixed at both ends always found in introductory physics texts. The first and dominant type is simple in structure and resembles a single cycle of a sine wave. Subdominant or higher order components are also roughly sinusoidal, but with more and more cycles. With this analogy in mind, we find the term *harmonics* evocative in referring to principal components of variation in curves in general.

8.2.3 *Defining an optimal empirical orthonormal basis*

There are several other ways to motivate PCA, and one is to define the following problem: We want to find a set of exactly K orthonormal functions ξ_m so that the expansion of each curve in terms of these basis functions approximates the curve as closely as possible. Since these basis functions

are orthonormal, it follows that the expansion will be of the form

$$\hat{x}_i(t) = \sum_{k=1}^{K} f_{ik}\xi_k(t),$$

where f_{ik} is the principal component value $\int x_i\xi_k$. As a fitting criterion for an individual curve, consider the integrated squared error

$$\|x_i - \hat{x}_i\|^2 = \int [x(s) - \hat{x}(s)]^2\, ds$$

and as a global measure of approximation,

$$\text{PCASSE} = \sum_{i=1}^{N} \|x_i - \hat{x}_i\|^2. \tag{8.5}$$

The problem is then, more precisely, what choice of basis will minimize the error criterion (8.5)?

The answer, it turns out, is precisely the same set of principal component weight functions that maximize variance components as defined above. For this reason, these functions ξ_m are referred to in some fields as *empirical orthonormal functions*, because they are determined by the data they are used to expand.

8.2.4 *PCA and eigenanalysis*

In this section, we investigate another characterization of PCA, in terms of the eigenanalysis of the variance-covariance function or operator.

Assume for this section that our observed values, x_{ij} in the multivariate context and $x_i(t)$ in the functional situation, result from subtracting the mean variable or function values, so that their sample means $N^{-1}\sum_i x_{ij}$, or cross-sectional means $N^{-1}\sum_i x_i(t)$, respectively, are zero.

Texts on multivariate data analysis tend to define principal components analysis as the task of finding the eigenvalues and eigenvectors of the covariance or correlation matrix. The logic for this is as follows. Let the $N \times p$ matrix $\mathbf{X}$ contain the values x_{ij} and the vector $\boldsymbol{\xi}$ of length p contain the weights for a linear combination. Then the mean square criterion for finding the first principal component weight vector can be written as

$$\max_{\boldsymbol{\xi}'\boldsymbol{\xi}=1} N^{-1}\boldsymbol{\xi}'\mathbf{X}'\mathbf{X}\boldsymbol{\xi} \tag{8.6}$$

since the vector of principal component scores f_i can be written as $\mathbf{X}\boldsymbol{\xi}$.

Use the $p \times p$ matrix $\mathbf{V}$ to indicate the sample variance-covariance matrix $\mathbf{V} = N^{-1}\mathbf{X}'\mathbf{X}$. (One may prefer to use a divisor of $N-1$ to N since the means have been estimated, but it makes no essential difference to the principal components analysis.) The criterion (8.6) can now be expressed

as

$$\max_{\boldsymbol{\xi}'\boldsymbol{\xi}=1} \boldsymbol{\xi}'\mathbf{V}\boldsymbol{\xi}.$$

As explained in Section A.5, this maximization problem is now solved by finding the solution with largest eigenvalue ρ of the eigenvector problem or *eigenequation*

$$\mathbf{V}\boldsymbol{\xi} = \rho\boldsymbol{\xi}. \tag{8.7}$$

There is a sequence of different eigenvalue-eigenvector pairs $(\rho_j, \boldsymbol{\xi}_j)$ satisfying this equation, and the eigenvectors $\boldsymbol{\xi}_j$ are orthogonal. Because the mean of each column of $\mathbf{X}$ is usually subtracted from all values in that column as a preliminary to principal components analysis, the rank of $\mathbf{X}$ is $N-1$ at most, and hence the $p \times p$ matrix $\mathbf{V}$ has, at most, $\min\{p, N-1\}$ nonzero eigenvalues ρ_j. For each j, the eigenvector $\boldsymbol{\xi}_j$ satisfies the maximization problem (8.6) subject to the additional constraint of being orthogonal to all the eigenvectors $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_{j-1}$ found so far. This is precisely what was required of the principal components in the second step laid out in Section 8.2.1. Therefore, as we have defined it, the multivariate PCA problem is equivalent to the algebraic and numerical problem of solving the eigenequation (8.7). Of course, there are standard computer algorithms for doing this.

Now consider the functional version of PCA. Define the covariance function $v(s,t)$ by

$$v(s,t) = N^{-1}\sum_{i=1}^{N} x_i(s)x_i(t). \tag{8.8}$$

Again, note that we may prefer to use $N-1$ to define the variance-covariance function v; nothing discussed here changes in any essential way.

The more general results set out in Section A.5.2 can be applied, to find the principal component weight functions $\xi_j(s)$. Each of these satisfies the equation

$$\int v(s,t)\xi(t)\,dt = \rho\xi(s) \tag{8.9}$$

for an appropriate eigenvalue ρ. The left side of (8.9) is an *integral transform* V of the weight function ξ defined by

$$V\xi = \int v(\cdot,t)\xi(t)\,dt. \tag{8.10}$$

This integral transform is called the *covariance operator* V. Therefore we may also express the eigenequation directly as

$$V\xi = \rho\xi, \tag{8.11}$$

where ξ is now an eigenfunction rather than an eigenvector. By suitable choice of notation, the equation (8.11) for functional PCA now looks the same as the eigenequation (8.7) relevant to conventional PCA.

There is an important difference between the multivariate and functional eigenanalysis problems, concerning the maximum number of different eigenvalue-eigenfunction pairs. The counterpart of the number of variables p in the multivariate case is the number of function values in the functional case, and thus infinity. However, provided the functions x_i are not linearly dependent, the operator V will have rank $N-1$, and there will be only $N-1$ nonzero eigenvalues.

To summarize, in this section we find that principal components analysis is defined as the search for a set of mutually orthogonal and normalized weight functions ξ_m. Functional PCA can be expressed as the problem of the eigenanalysis of the covariance operator V. By suitable choice of notation, the formal steps to be carried out are the same, whether the data are multivariate or functional.

In Section 8.4 we discuss practical methods for actually computing the eigenfunctions ξ_m, but first we consider some aspects of the display of principal components once they have been found.

8.3 Visualizing the results

The fact that interpreting the components is not always an entirely straightforward matter is common to most functional PCA problems. We now consider some techniques that may aid their interpretation.

8.3.1 Plotting components as perturbations of the mean

A method found to be helpful is to examine plots of the overall mean function and the functions obtained by adding and subtracting a suitable multiple of the principal component function in question. Figure 8.2 shows such a plot for the temperature data. In each case, the solid curve is the overall mean temperature, and the dotted and dashed curves show the effects of adding and subtracting a multiple of each principal component curve. This considerably clarifies the effects of the first two components. We can now see that the third principal component corresponds to a time shift effect combined with an overall increase in temperature and in range between winter and summer. The fourth corresponds to an effect whereby the onset of spring is later and autumn ends earlier.

In constructing this plot, it is necessary to choose which multiple of the principal component function to use. Define a constant C to be the root-mean-square difference between $\hat{\mu}$ and its overall time average,

$$C^2 = T^{-1}\|\hat{\mu} - \bar{\mu}\|^2; \tag{8.12}$$

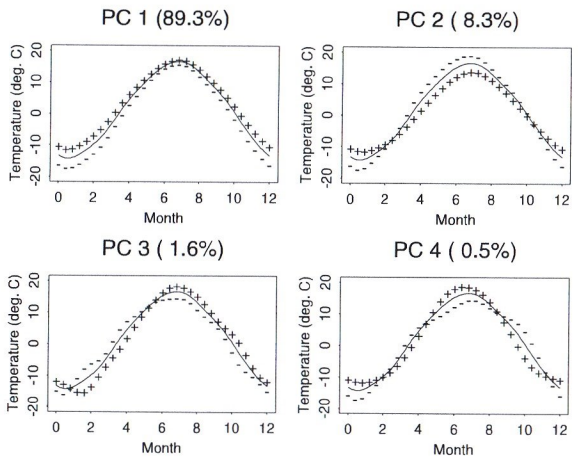

Figure 8.2. The mean temperature curves and the effects of adding (+) and subtracting (−) a suitable multiple of each PC curve.

where

$$\bar{\mu} = T^{-1} \int \hat{\mu}(t)\, dt.$$

It is then appropriate to plot $\hat{\mu}$ and $\hat{\mu} \pm 0.2C\hat{\gamma}_j$, where we have chosen the constant 0.2 to give easily interpretable results. Depending on the overall behavior of $\hat{\mu}$, it may be helpful to adjust the value 0.2 subjectively. But for ease of comparison between the various modes of variability, it is best to use the same constant for all the principal component functions plotted in any particular case.

In Figure 8.3, we consider the hip angles observed during the gait of 39 children, as plotted in Figure 1.8. The angles for a single cycle are shown, along with the results of a functional PCA of these data. The effect of the first principal component of variation is approximately to add or subtract a constant to the angle throughout the gait cycle. The second component corresponds roughly to a time shift effect, which is not constant through the cycle. The third component corresponds to a variation in the overall amplitude of the angle traced out during the cycle.

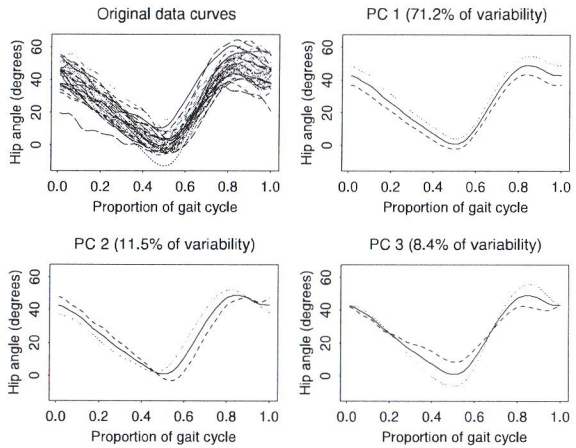

Figure 8.3. The hip angle observed in the gait cycles of 39 children, and the effect on the overall mean of adding and subtracting a suitable multiple of each of the first three principal component functions.

8.3.2 Plotting principal component scores

An important aspect of PCA is the examination of the scores f_{im} of each curve on each component. In Figure 8.4, each weather station is identified by a four-letter abbreviation of its name given in Table 8.1. The strings are positioned roughly according to the scores on the first two principal components; some positions have been adjusted slightly to improve legibility. The West Coast stations Vancouver (VANC), Victoria (VICT) and Prince Rupert (PRUP) are in the upper right corner because they have warmer winters than most stations (high on PC 1) and less summer-winter temperature variation (high on PC 2). Resolute (RESO), on the other hand, has an extremely cold winter, but does resemble the Pacific weather stations in having less summer/winter variation than some Arctic cousins, such as Inuvik (INUV).

8.3.3 Rotating principal components

In Section 8.2 we observed that the weight functions ξ_m can be viewed as defining an orthonormal set of K functions for expanding the curves to minimize a summed integrated squared error criterion (8.5). For the

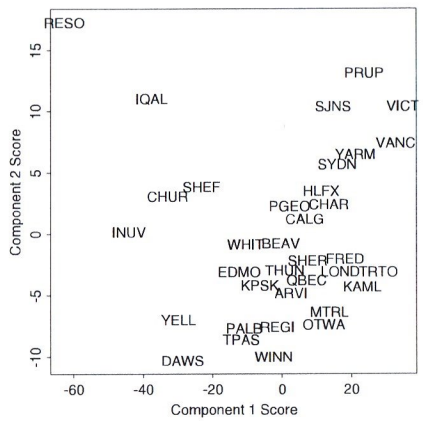

Figure 8.4. The scores of the weather stations on the first two principal components of temperature variation. The location of each weather station is shown by the four-letter abbreviation of its name assigned in Table 8.1.

Table 8.1. The Canadian Weather Stations

Arvida, Que.	Kapuskasing, Ont.	St. John's, Nfld
Beaverlodge, B.C.	London, Ont.	Sydney, N.S.
Calgary, Alta.	Montreal, Que.	The Pas, Man.
Charlottetown, P.E.I.	Ottawa, Ont.	Thunder Bay, Ont.
Churchill, Man.	Prince Albert, Sask.	Toronto, Ont.
Dawson, Yukon	Prince George, B.C.	Vancouver, B.C.
Edmonton, Alta.	Prince Rupert, B.C.	Victoria, B.C.
Fredericton, N.B.	Quebec City, Que.	Whitehorse, Yukon
Halifax, N.S.	Regina, Sask.	Winnipeg, Man.
Inuvik, N.W.T.	Resolute, N.W.T.	Yarmouth, N.S.
Iqualuit, N.W.T.	Schefferville, Que.	Yellowknife, N.W.T.
Kamloops, B.C.	Sherbrooke, Que.	

temperature data, for example, no set of four orthonormal functions will do a better job of approximating the curves than those displayed in Figure 8.1.

This does not mean, however, that there aren't other orthonormal sets that will do just as well. In fact, if we now use ξ to refer to the vector-valued

function $(\xi_1, \ldots, \xi_K)'$, then an equally good orthonormal set is defined by

$$\psi = \mathbf{T}\xi, \tag{8.13}$$

where $\mathbf{T}$ is any orthonormal matrix of order K, meaning that $\mathbf{T}'\mathbf{T} = \mathbf{T}\mathbf{T}' = \mathbf{I}$. From a geometrical perspective, the vector of functions ψ is a rigid rotation of ξ. Of course, after rotation, we can no longer expect that ψ_1 will define the largest component of variation. But the point is that the orthonormal basis functions $\psi_1, \ldots, \psi_K$ are just as effective at approximating the original curves in K dimensions as their unrotated counterparts.

Can we find some rotated functions that are perhaps a little easier to interpret? Here again, we can borrow a tool that has been invaluable in multivariate analysis, VARIMAX rotation. Let $\mathbf{B}$ be a $K \times n$ matrix representing the first K principal component functions $\xi_1, \ldots, \xi_K$. For the moment, suppose that $\mathbf{B}$ has, as row m, the values $\xi_m(t_1), \ldots, \xi_m(t_n)$ for n equally spaced argument values in the interval $\mathcal{T}$. The corresponding matrix $\mathbf{A}$ of values of the rotated basis functions $\psi = \mathbf{T}\xi$ will be given by

$$\mathbf{A} = \mathbf{T}\mathbf{B}. \tag{8.14}$$

The VARIMAX strategy for choosing the orthonormal rotation matrix $\mathbf{T}$ is to maximize the variation in the values a_{mj}^2 strung out as a single vector. Since $\mathbf{T}$ is a rotation matrix, the overall sum of these squared values will remain the same no matter what rotation we perform. In algebraic terms,

$$\sum_m \sum_j a_{mj}^2 = \text{trace}\,\mathbf{A}'\mathbf{A} = \text{trace}\,\mathbf{B}'\mathbf{T}'\mathbf{T}\mathbf{B} = \text{trace}\,\mathbf{B}'\mathbf{B}.$$

Therefore, maximizing the variance of the a_{mj}^2 can happen only if these values tend either to be relatively large or relatively near zero. The values a_{mj} themselves are encouraged to be either strongly positive, near zero, or strongly negative; in-between values are suppressed. This clustering of information tends to make the components of variation easier to interpret.

There are fast and stable computational techniques for computing the rotation matrix $\mathbf{T}$ that maximizes the VARIMAX criterion. A C function for computing the VARIMAX rotation can be found through the book's world-wide web page described in Section 1.9.

Figure 8.5 displays the VARIMAX rotation of the four principal components for the temperature data. There, $n = 12$ equally spaced time points t_j were used, and the variance of the squared values $\psi_m^2(t_j)$ was maximized with respect to $\mathbf{T}$. The resulting rotated functions ψ_m, along with the percentages of variances that they account for, are now quite different.

Collectively, the rotated functions ψ_m still account for a total of 99.7% of the variation, but they divide this variation in different proportions. The VARIMAX rotation has suppressed medium-sized values of ψ_m while preserving orthonormality. (Note that the rotated component scores are no longer uncorrelated; however, the sum of their variances is still the same, because $\mathbf{T}$ is a rotation matrix, and so they may still be considered to

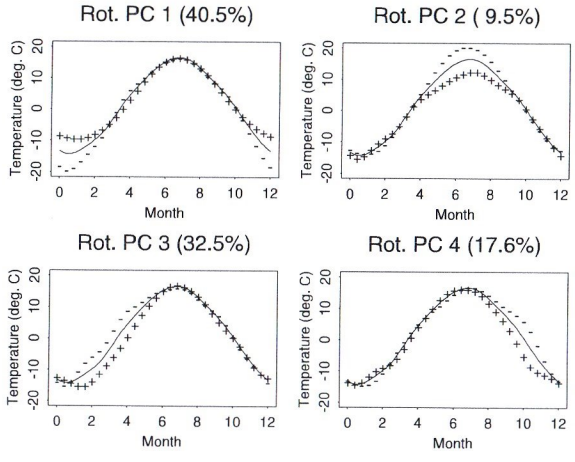

Figure 8.5. Weight functions rotated by applying the VARIMAX rotation criterion to weight function values, and plotted as positive and negative perturbations of the mean function.

partition the variability in the original data.) The result is four functions that account for local variation in the winter, summer, spring and autumn, respectively. Not only are these functions much easier to interpret, but we see something new: although winter variation remains extremely important, now spring variation is clearly almost as important, about twice as important as autumn variation and over three times as important as summer variation.

Another way of using the VARIMAX idea is to let $\mathbf{B}$ contain the coefficients for the expansion of each ξ_m in terms of a basis ϕ of n functions. Thus we rotate the coefficients of the basis expansion of each ξ_m rather than rotating the values of the ξ_m themselves. Figure 8.6 shows the results using a Fourier series expansion of the principal components. The results are much more similar to the original principal components displayed in Figure 8.2. The main difference is in the first two components. The first rotated component function in Figure 8.6 is much more constant than the original first principal component, and corresponds almost entirely to a constant temperature effect throughout the year. The general shape of the second component is not changed very much, but it accounts for more of the variability, having essentially taken on part of the variability in the first unrotated component. Because the first component originally accounted for such a large proportion, 89.3%, of the variability, it is not surprising that a

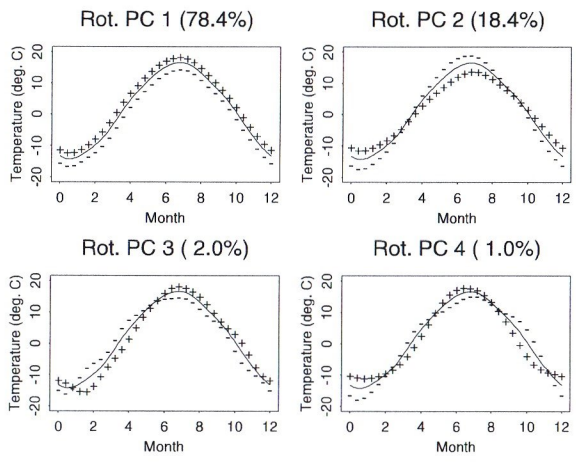

Figure 8.6. Weight functions rotated by applying the VARIMAX rotation criterion to weight function coefficients, and plotted as positive and negative perturbations of the mean function.

fairly small change in the shape of the second component results in moving about 10% of the total variability from the first to the second component. The third and fourth components are not enormously affected by the VARIMAX rotation in the Fourier domain.

By no means is the VARIMAX criterion the only rotation criterion available. References on factor analysis and multivariate statistics such as Basilevsky (1994), Johnson and Wichern (1988), Mulaik (1972) and Seber (1984) offer a number of other possibilities. Even from the relatively brief discussion in this section, it is clear that much research remains to be done on rotation schemes tailored more directly to the functional context.

8.4 Computational methods for functional PCA

Now suppose that we have a set of N curves x_i, and that preliminary steps such as curve registration and the possible subtraction of the mean curve from each (curve centering) have been completed. Let $v(s,t)$ be the sample covariance function of the observed data. In this section, we consider possible strategies for approaching the eigenanalysis problem in (8.9). In all cases, we convert the continuous functional eigenanalysis problem to an approximately equivalent matrix eigenanalysis task.

7

The registration and display of functional data

7.1 Introduction

We can now assume that our observations are in functional form, and want to proceed to consider methods for their analysis. We are not quite ready, however; a problem of critical importance to functional data needs a solution. We see often that variation in functional observations involves both phase and amplitude, and that confounding these two leads to many problems. Our main emphasis is on *registration* of the data, involving transformations of the argument t rather than the values $x(t)$.

Figure 1.2 illustrates a problem that can frustrate even the simplest analyses of replicated curves. Ten records of the acceleration in children's height show individually the salient features of growth: the large deceleration during infancy is followed by a rather complex but small-sized acceleration phase during late childhood. Then the dramatic acceleration-deceleration pulses of the pubertal growth spurt finally give way to zero acceleration in adulthood. But the timing of these salient features obviously varies from child to child, and ignoring this timing variation in computing a cross-sectional mean function, shown by the heavy dashed line in Figure 1.2, can result in a estimate of average acceleration that does not resemble any of the observed curves. In this case, the mean curve has less variation during the pubertal phase than any single curve, and the duration of the mean pubertal growth spurt is rather larger than that of any individual curve.

The problem is that the growth curves exhibit two types of variability. *Amplitude variability* pertains to the sizes of particular features such as the

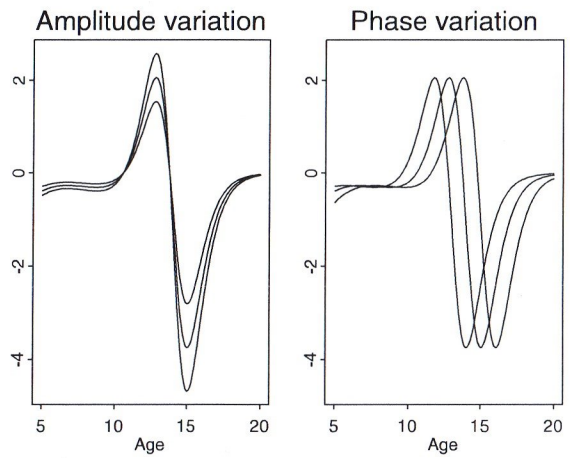

Figure 7.1. The left panel shows three height acceleration curves varying only in amplitude. The right panel shows three curves varying only in phase.

velocity peak in the pubertal growth spurt, ignoring their timings. *Phase variability* is variation in the timings of the features without considering their sizes. Before we can get a useful measure of a typical growth curve, we must separate these two types of variation, so that features such as the pubertal spurt occur at roughly the same "times" for all girls. The problem is expressed in schematic terms in Figure 7.1, where we see in the left panel two acceleration curves that differ only in amplitude, and in the right panel two curves with the same amplitude, but differing in phase.

The need to transform curves by transforming their arguments, which we call *curve registration*, can be motivated as follows. The rigid metric of physical time may not be directly relevant to the internal dynamics of many real-life systems. Rather, there can be a sort of biological or meteorological time scale that can be nonlinearly related to physical time, and can vary from case to case.

Human growth, for example, is the consequence of a complex sequence of hormonal events that do not happen at the same rate for every child. The intensity of the pubertal growth spurts of two children should be compared at their respective ages of peak velocity rather than at any fixed age. A colleague with a musical turn of mind refers to this as differences in the *tempo* of growth.

Similarly, weather is driven by ocean currents, reflectance changes for land surfaces, and other factors that are timed differently for different spatial locations and different years. Winter comes early in some years, and

late in others, and typically arrives later at some weather stations than others. We need to assess how cold the average winter is at the time the average temperature bottoms out rather than at any fixed time.

Put more abstractly, the values of two or more function values $x_i(t_i)$ can in principle differ because of two types of variation. The first is the more familiar vertical variation, or *amplitude variation*, due to the fact that $x_1(t)$ and $x_2(t)$ may simply differ at points of time t at which they are compared, but otherwise exhibit the same shape features at that time. But they may also exhibit *phase variation* in the sense that functions x_1 and x_2 should not be compared at the same time t because they are not exhibiting the same behavior. Instead, in order to compare the two functions, the time scale itself has to be distorted or transformed.

We now look at several types of curve registration problems, beginning first with the problem of simply translating or shifting the values of t by a constant amount δ. Then we discuss landmark registration, which involves transforming t nonlinearly in order to line up important features or landmarks for all curves. Finally, we look at a more general method for curve registration.

7.2 Shift registration

Many of the issues involved in registration can be illustrated by considering the simplest case, a simple shift in the time scale. The pinch force data illustrated in Figure 1.11 are an example of a set of functional observations that must be aligned by moving each curve horizontally before any meaningful cross-curve analysis is possible. This often happens because the time at which the recording process begins is arbitrary, and is unrelated to the beginning of the interesting segment of the data, in this case the period over which the measured squeeze actually takes place.

Let the interval $\mathcal{T}$ over which the functions are to be registered be $[T_1, T_2]$. We also need to assume that each sample function x_i is available for some region beyond each end of $\mathcal{T}$. The pinch force data, for example, are observed for substantial periods both before and after the force pulse that we wish to study. In the case of periodic data such as the Canadian temperature records, this requirement is easily met since one can wrap the function around by using the function's behavior at the opposing end of the interval.

We are actually interested in the values

$$x_i^*(t) = x_i(t + \delta_i),$$

where the shift parameter δ_i is chosen in order to appropriately align the curves. For the pinch force data, the size of δ_i is of no real interest, since it merely measures the gap between the initialization of recording and the beginning of a squeeze. Silverman (1995) refers to this situation, in which

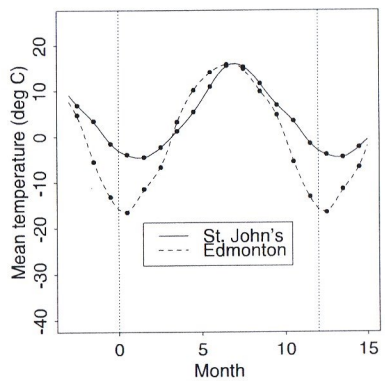

Figure 7.2. Temperature records for two weather stations where in the timing of the seasons differs by a roughly constant shift.

a shift parameter must be accounted for but is of no real interest, as a *nuisance effects* problem.

The Canadian temperature data present a curve alignment problem of a somewhat different nature. As Figure 7.2 indicates, two temperature records, such as those for St. John's, Newfoundland, and Edmonton, Alberta, can differ noticeably in terms of the phase or timing of key events, such as the lowest mean temperature and the timing of spring and autumn. In this case, the shifts that would align these two curves vertically are of intrinsic interest, and should be viewed as a component of variation that needs careful description. It turns out that continental stations such as Edmonton have earlier seasons than marine stations such as St. John's, because of the capacity of oceans to store heat and to release it slowly. In fact, either station's weather would have to be shifted by about three weeks to align the two.

When, as in the temperature data case, the shift is an important feature of each curve, we characterize its estimation as a *random effects* problem. Silverman (1995) also distinguishes a third and intermediate *fixed effects* case in which the shift must be carried out initially, and while not being discarded completely once the functions x_i^* have been constructed, is nevertheless only of tangential interest.

7.2.1 *The least squares criterion for shift alignment*

The basic mechanics of estimating the shifts δ_i are the same, whether they are considered as nuisance or random effects. The differences become important when we consider the analysis in subsequent chapters, because in the random effects case (and, to some extent, the fixed effects case) the δ_i enter the analysis. However, for present purposes we concentrate on the pinch force data as an example.

The estimation of a shift or an alignment requires a criterion that defines when several curves are properly registered. One possibility is to identify a specific feature or *landmark* for a curve, and shift each curve so that this feature occurs at a fixed point in time. The time of the maximum of the smoothed pinch force is an obvious landmark. Note that this might also be expressed as the time at which the first derivative crosses zero with negative slope, and landmarks are often more easily identifiable at the level of some derivative.

However, the registration by landmark or feature alignment has some potentially undesirable aspects: The location of the feature may be ambiguous for certain curves, and if the alignment is only of a single point, variations in other regions may be ignored. If, for example, we were to register the two temperature curves by aligning the midsummers, the midwinters might still remain seriously out of phase.

Instead, we can define a global registration criterion for identifying a shift δ_i for curve i as follows. First we estimate an overall mean function $\hat{\mu}(t)$ for t in $\mathcal{T}$. If the individual functional observations x_i are smooth, it usually suffices to estimate $\hat{\mu}$ by the sample average $\bar{x}$. However, we wish to be able to evaluate derivatives of $\hat{\mu}$, and so more generally we want to smooth the overall estimate using one of the methods described in Chapters 4 and 5. We can now define our global registration criterion by

$$\begin{aligned} \texttt{REGSSE} &= \sum_{i=1}^{N} \int_{\mathcal{T}} [x_i(t+\delta_i) - \hat{\mu}(t)]^2 \, ds \\ &= \sum_{i=1}^{N} \int_{\mathcal{T}} [x_i^*(t) - \hat{\mu}(t)]^2 \, ds. \end{aligned} \tag{7.1}$$

Thus, our measure of curve alignment is the integrated or global sum of squared vertical discrepancies between the shifted curves and the sample mean curve.

The target function for transformation in (7.1) is the unregistered cross-sectional estimated mean $\hat{\mu}$. But of course one of the goals of registration is to produce a better estimate of this same mean function. We therefore expect to proceed iteratively: beginning with the unregistered cross-sectional estimated mean, argument values for each curve are shifted so as to minimize REGSSE, then the estimated mean $\hat{\mu}$ is updated by re-estimating it from the *registered* curves x_i^*, and a new iteration is then undertaken using this revised target. This procedure of estimating a transformation by transforming to an iteratively updated average is often referred to as the *Procrustes method*. In practice, we have found that the process usually converges within one or two iterations.

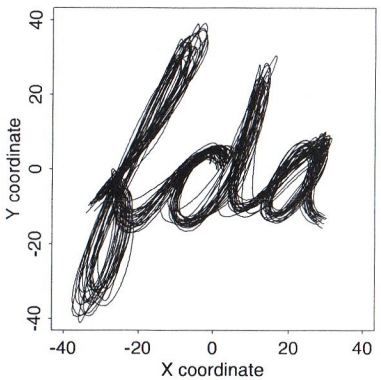

Figure 7.3. Twenty replications of "fda" written by one of the authors.

7.3 Feature or landmark registration

A landmark or a feature of a curve is some characteristic that one can associate with a specific argument value t. These are typically maxima, minima, or zero crossings of curves, and may be identified at the level of some derivatives as well as at the level of the curves themselves.

We now turn to the more general problem of estimating a possibly nonlinear transformation h_i of t, and indicate how we can use landmarks to estimate this transformation. Coincidentally, the illustrative example we use shows how vector-valued functional data can be handled by obvious extensions of methods for scalar-valued functions.

The landmark registration process requires for each curve x_i the identification of the argument values $t_{if}, f = 1, \ldots, F$ associated with each of F features. The goal is to construct a transformation h_i for each curve such that the registered curves with values

$$x^*(t) = x_i[h_i(t)]$$

have more or less identical argument values for any given landmark.

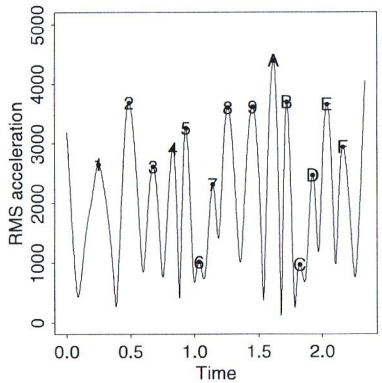

Figure 7.4. The average length of the acceleration vector for the 20 handwriting samples. The characters identify the 15 features used for landmark registration.

Consider, for example, the 20 replications of the letters "fda" in Figure 7.3. Each sample of handwriting was obtained by recording the position of a pen at a sampling rate of 600 times per second. There was some preprocessing to make each script begin and end at times 0 and 2.3 seconds, and to compute coordinates at the same 1,401 equally-spaced time-values. Each curve x_i in this situation is vector-valued, since two spatial coordinates are involved, and we use $\texttt{ScriptX}_i$ and $\texttt{ScriptY}_i$ to designate the X- and Y-coordinates, respectively.

Not surprisingly, there is some variation from observation to observation, and one goal is to explore the nature of this variation. But we want to take into account that, for example, variation in the "f" can be of two sorts. There is temporal variation due to the fact that timing of the top of the upper loop, for example, is variable. While this type of variation would not show up in the plots in Figure 7.3, it may still be an important aspect of how these curves vary. On the other hand, there is variation in the way the shape of each letter is formed, and this is obvious in the figure.

We estimated the accelerations or second derivatives of the two coordinate functions $D^2\texttt{ScriptX}_i$ and $D^2\texttt{ScriptY}_i$ by the local polynomial method described in Chapter 4. Figure 7.4 displays the average length of the acceleration vector

$$\sqrt{(D^2\texttt{ScriptX}_i)^2 + (D^2\texttt{ScriptY}_i)^2}$$

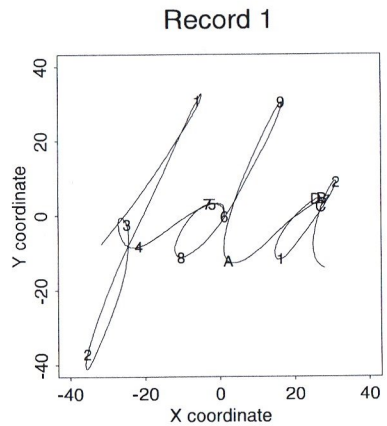

Figure 7.5. The first handwriting curve with the location of the 15 landmarks indicated by the characters used in Figure 7.4.

and we note that there are 15 clearly identified maxima, indicating points where the pen is changing direction. We also found that these maxima were easily identifiable in each record, and we were able to determine the values of t_{if} corresponding to them by just clicking on the appropriate points in a plot. Figure 7.5 shows the first curve with these 15 features labelled, and we can see that landmarks labelled "4" and "A" mark the boundaries between letters. Figure 7.6 plots the values of the landmark timings t_{if} against the corresponding timings for the mean function, t_{0f}. We were interested to see that the variability of the landmark timings was rather larger for the initial landmarks than for the later ones, and we were surprised by how small the variability was for all of them.

The identification of landmarks enabled us to compare the X- and Y-coordinate values for the 20 curves at the landmark times, but of course we also wanted to make comparisons at arbitrary points between landmarks. This required the computation of a function h_i for each curve, called a *time-warping function* in the engineering literature, with the properties

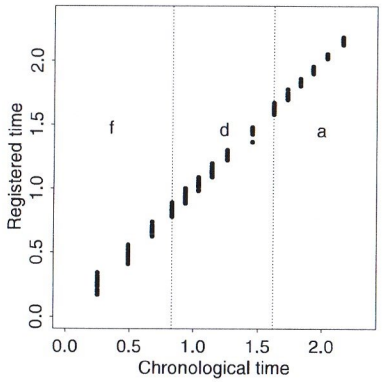

Figure 7.6. The timings of the landmarks for all 20 scripts plotted against the corresponding timings for the mean curve.

- $h_i(0) = 0$
- $h_i(2.3) = 2.3$
- $h_i(t_{0f}) = t_{if}, f = 1, \ldots, 15$
- h_i is strictly monotonic: $s < t$ implies that $h_i(t) < h_i(t)$.

The values of the adjusted curves at time t are $\texttt{ScriptX}[h_i(t)]$ and $\texttt{ScriptY}[h_i(t)]$. In all the adjusted curves, the landmarks each occur at the same time as in the mean function. In addition, the adjusted curves are also more or less aligned between landmarks. In this application, we merely used linear interpolation for time values between the points (t_{0f}, t_{if}) (as well as (0,0) and (2.3,2.3)) to define the time warping function h_i for each curve. We introduce more sophisticated notions in the next section. Figure 7.7 shows the warping function computed in this manner for the first script record. Because h_1 is below the diagonal line in the region of "f," the aligned time $h_1(t)$ is earlier than the actual time of features, and hence the actual times for curve 1 are retarded with respect to the mean curve.

We can now re-compute the mean curve by averaging the registered curves. The result is in Figure 7.8, shown along with the mean for the unregistered data. Although the differences are not dramatic, as we might expect given the mild curvature in h_1, we do see that the upper and lower loops of the "f" are now more pronounced, and in fact do represent the original curves substantially better.

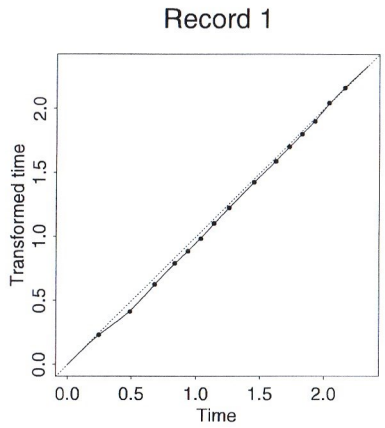

Figure 7.7. The time warping function h_1 estimated for the first record that registers its features with respect to the mean curve.

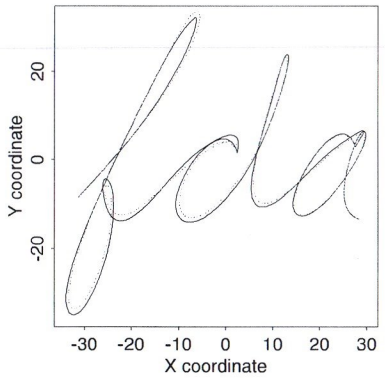

Figure 7.8. The solid line is the mean of the registered "fda" curves, and the dashed line is the mean of the unregistered curves.

7.4 Using the warping function h to register x

Now that a warping function h has been estimated from landmark registration, or by using the continuous method described in a later section, you will want to calculate the registered function values $x^*(t) = x[h(t)]$. This requires two steps.

First, estimate the *inverse warping function* $h^{-1}(t)$ with the property $h^{-1}[h(t)] = t$. Note that this is not an inverse in the sense of the reciprocal. Instead, $h^{-1}(t)$ is computed by smoothing or interpolating the relationship between $h(t)$ plotted on the horizontal axis and t plotted on the vertical axis. You can then use simple interpolation to get the values of this inverse function at an equally spaced set of values of t if required. Note that it will be essential that this smoothing or interpolation function be strictly monotonic, so you may have to use lots of values of t and/or employ monotone smoothing described in Chapter 6.

The second step is to smooth or interpolate the relationship between $h^{-1}(t)$ plotted on the abscissa and $x(t)$ plotted on the ordinate. You can then use simple interpolation to get the values of this registered function at an equally spaced set of values of t if required.

7.5 A more general warping function h

The linear interpolation scheme that we used on the handwriting data to estimate the time-warping function h has two limitations. First, if we want to compute higher order derivatives of the curves with respect to warped time, the warping function must also be differentiable to the same order, a linear interpolation would not carry us beyond the first derivative. Secondly, we will shortly consider *continuous registration* methods that do not use landmarks and where the idea of interpolating a sequence of points will not be helpful.

Time is itself a growth process, and thus can be linked to our discussion in Chapter 6 on how to model the children's growth curves. That is, we can use the formulation

$$h(t) = C_0 + C_1 \int_0^t \exp W(u)\, du \tag{7.2}$$

that we used in (6.9). Here the constants C_0 and C_1 are fixed by the requirement that $h(t) = t$ at the lower and upper limits of the interval over which we model the data. Or, if shift registration is a possibility, the constant term C_0 can be allowed to pick any constant phase shift that is required.

Physical or clock time grows linearly, of course, and thus corresponds to $W(u) = 0$. If $W(u)$ is *positive*, then $h(t) > t$, warped time is growing faster than clock time, and this is what we want if our observed process

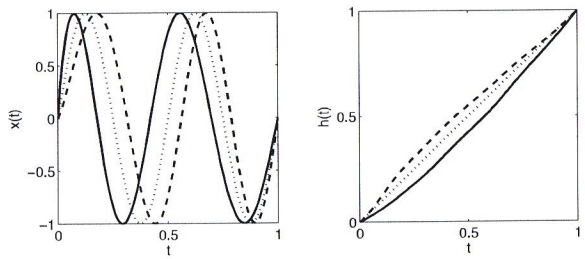

Figure 7.9. The left panel shows the target function, $x_0(t) = \sin(4\pi t)$, as a dotted line; an early function, $x_E(t) = \sin(4\pi t^{0.8})$, as a solid line; and a late function, $x_L(t) = \sin(4\pi t^{1.2})$, as a dashed line. The corresponding warping functions that register the early and late curves to the target are shown in the right panel.

is running *late*. Similarly, for negative values of $W(u)$, $h(t) < t$, and clock time is being slowed down for a process that is running ahead of some target.

The left panel of Figure 7.9 displays two examples. Here the target or standard function is $x_0(t) = \sin(4\pi t)$, the early function is $x_E(t) = \sin(4\pi t^{0.8})$ and the late function is $x_L(t) = \sin(4\pi t^{1.2})$. Warping $h_E(t) = t^{0.125}$ will register the first example since $\sin[4\pi(t^{0.8})^{1.25}] = \sin(4\pi t)$, and similarly $h_L(t) = t^{0.833}$. Approximations to the two warping functions by a method to be described below are presented in the right panel, and we can see there how early functions are associated with time-decelerating warpings, and late functions with time-accelerating warpings.

The use of (7.2) as a representation of a warping function has a very handy bonus. Providing that the warp h is reasonably smooth and mild, the inverse warp h^{-1} is achieved to a close approximation by merely replacing W in the equation by $-W$.

7.6 A continuous fitting criterion for registration

The least squares criterion (7.1) worked well for simple shift registration, but gets us into trouble for more general warping functions. The lower panel in Figure 7.10 shows why. When two functions differ in terms of amplitude as well as phase, the least squares criterion uses time warping to also minimize amplitude differences by trying to squeeze out of existence regions where amplitudes differ. Put another way, the least squares fitting criterion is intrinsically designed to assess differences in amplitude rather

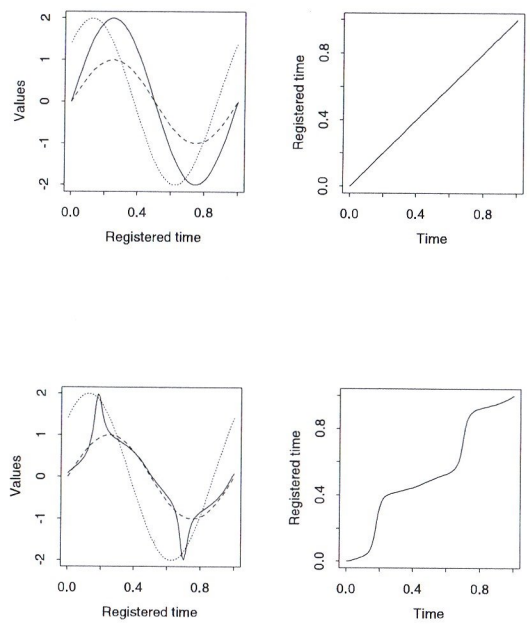

Figure 7.10. The upper two panels show results for an artificial registration problem using the minimum eigenvalue criterion. The dotted curve in the upper-left panel is the curve to be registered to the curve indicated by the dashed line. The solid line is the registered curve. The upper-right panel contains the warping function for this case, $h(t) = t$. The lower panels show the same results using the least squares criterion.

than phase. This wasn't a problem when only time shifts were involved since such simple time warps cannot affect amplitude differences.

Suppose two curves x_0 and x_1 differ only in amplitude but not in phase, such as in the left panel of Figure 7.10. Then, if we plot the function values $x_0(t)$ and $x_1(t)$ against each other, we will see a straight line. Amplitude differences will then be reflected in the slope of the line, a line at 45^o corresponding to no amplitude differences.

Now thinking about a line as a one-dimensional set of points on a plane, we can turn to principal components analysis as just the right technique for assessing how many dimensions are required to represent the distribution of these points. This technique will yield only one positive eigenvalue if the

point spread is, in fact, one-dimensional. That is, the size of the smallest eigenvalue measures departures from unidimensionality.

Let us consider now evaluating both the target function x_0 and the registered function x^* at a fine mesh of n values of t to obtain the pairs of values $(x_0(t), x[h(t)])$. Let the n by two matrix $\mathbf{X}$ contain these pairs of values. Then the two-by-two cross-product matrix $\mathbf{X}'\mathbf{X}$ would be what we would analyze by principal components.

The following order two matrix is the functional analogue of the cross-product matrix $\mathbf{X}'\mathbf{X}$.

$$\mathbf{T}(h) = \begin{bmatrix} \int \{x_0(t)\}^2\, dt & \int x_0(t)x[h(t)]\, dt \\ \int x_0(t)x[h(t)]\, dt & \int \{x[h(t)]\}^2\, dt \end{bmatrix} \tag{7.3}$$

We see that the summations over points implied by the expression $\mathbf{X}'\mathbf{X}$ have here been replaced by integrals. Otherwise this is the same matrix. We have expressed the matrix as a function of warping function h to remind ourselves that it does depend on h.

Consequently, we can now express our fitting criterion for assessing the degree to which two functions are registered as follows:

$$\texttt{MINEIG}(h) = \mu_2[\mathbf{T}(h)], \tag{7.4}$$

where the function μ_2 is the size of the second eigenvalue of its argument, which is an order two symmetric matrix. When $\texttt{MINEIG}(h) = 0$, we have achieved registration, and h is the warping function that does the job.

As is now routine, we will want to apply some regularization now and then to impose smoothness on h, so we extend our criterion to

$$\texttt{MINEIG}_\lambda(h) = \texttt{MINEIG}(h) + \lambda \int \{W^{(m)}(t)\}^2\, dt. \tag{7.5}$$

Here we are assuming that h is of the form (7.2), and that we achieve smoothness in h by smoothing the function W that defines it.

The results in Figure 7.9 were achieved by expanding W in terms of 13 B-splines with equally spaced knots, and penalizing the size of its second derivative using a smoothing parameter of $\lambda = 10^6$.

7.7 Registering the height acceleration curves

The 10 acceleration functions in Figure 1.2 were registered by the Procrustes method and the regularized basis expansion method set out in Section 7.6. The interval $\mathcal{T}$ was taken to be $[4, 18]$ with time measured in years. The break-values τ_k defining the monotone transformation family (7.2) were 4, 7, 10, 12, 14, 16 and 18 years, and the curves were registered over the interval [4,18] using criterion (7.5) with $\lambda = 0.001$. A single Procrustes iteration produced the results displayed in Figure 7.11. The left panel displays the 10 warping functions h_i, and the right panel shows

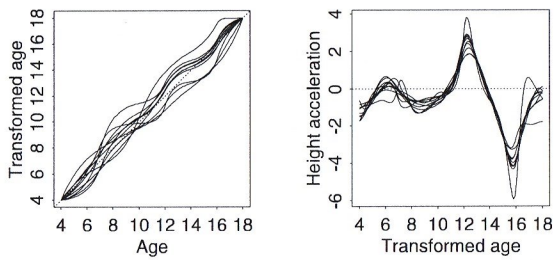

Figure 7.11. The left panel contains the estimated time warping functions h_i for the 10 height acceleration curves in Figure 1.2. The right panel displays the registered curves.

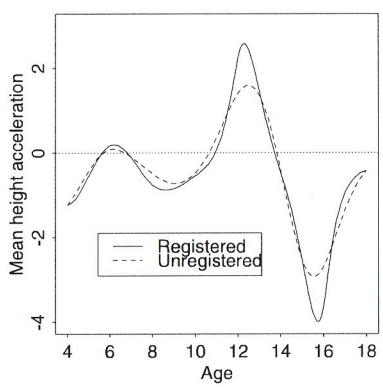

Figure 7.12. The cross-sectional means of the registered and unregistered height acceleration curves displayed in Figure 1.2.

the curve values $x_i[h_i(t)]$. Figure 7.12 compares the unregistered and registered cross-sectional means. We see that the differences are substantial, and moreover that the mean of the registered function tends to resemble much more closely most of the sample curves.

7.8 Some practical advice

Before registration, remove amplitude effects that can be accounted for by vertical shifts or scale changes, by centering and possibly rescaling the curves. This is standard advice in data analysis; deal with obvious effects in a simple way before moving on to more sophisticated procedures.

In general it is not clear that variation in the amplitude of curves can be cleanly separated from the variation that the registration process aims to account for. It is easy to construct examples where a registration function h that is allowed to be highly nonlinear can remove variation that is clearly of an amplitude nature, and the lower panels of Figure 7.10.

This problem of lack of identifiability of the two types of variation, horizontal and vertical, is perhaps less of a concern if only linear transformations are permitted, and is also not acute for landmark registration, where the role of the transformation is to only align curve features.

However, there is one situation that implies relatively unambiguous separation of the two types of variation. This happens with curves that cross zero at a number of points. At and near these zero crossings, only phase variation is possible. In effect, zero crossings are landmarks that should be aligned. Consequently, it may be wise to consider registering a derivative of a curve rather than the curve itself, since derivatives often cross zero. This is why we registered the acceleration curves above rather than the height or velocity curves.

If flexible families of monotone transformations such as those described above are used in conjunction with a global fitting criterion such as `MINEIG`, allow transformations to differ from linear only with caution by careful application of regularization.

In general, we have found it wise to first register on any landmarks that are clearly identifiable before using the continuous registration procedure. For example, in our work with the growth data we first register the curves using the zero-crossing in the middle of the pubertal growth spurt as a single landmark. Then we use the curves resulting from this preliminary registration as inputs to a continuous registration. If we use the notation h_L and $h_{C|L}$ to refer to the landmark warps and the continuous warps after landmark registration, respectively, then the final composite warping function is $h(t) = h_{C|L}[h_L(t)]$ or $h = h_{C|L} \circ h_L$.

7.9 Computational details

7.9.1 Shift registration by the Newton-Raphson algorithm

We can estimate a specific shift parameter δ_i iteratively by using a modified Newton-Raphson algorithm for minimizing `REGSSE`. This procedure requires derivatives of `REGSSE` with respect to the δ_i. If we assume that the

differences between x_i^* and $\hat{\mu}$ at the ends of the interval can be ignored (this is exactly true in the periodic case, and often approximately true in the non-periodic case if the effects of real interest are concentrated in the middle of the interval), then we have

$$\begin{aligned}\frac{\partial}{\partial\delta_i}\texttt{REGSSE} &= 2\int_{\mathcal{T}}\{x_i(t+\delta_i)-\hat{\mu}(t)\}Dx_i(t)\,dt \\ \frac{\partial^2}{\partial\delta_i^2}\texttt{REGSSE} &= 2\int_{\mathcal{T}}\{x_i(t+\delta_i)-\hat{\mu}(t)\}D^2x_i(t)\,dt \\ &+ 2\int_{\mathcal{T}}\{Dx_i(t)\}^2\,dt. \qquad (7.6)\end{aligned}$$

The modified Newton-Raphson algorithm works as follows:

Step 0: Begin with some initial shift estimates $\delta_i^{(0)}$, perhaps by aligning with respect to some feature, or even $\delta_i^{(0)} = 0$. But the better the initial estimate, the faster and more reliably the algorithm converges. Complete this step by estimating the average $\hat{\mu}$ of the shifted curves, using a method that allows the first two derivatives of $\hat{\mu}$ to give good estimates of the corresponding derivatives of the population mean, such as local polynomial regression of degree 4, or roughness penalty smoothing with an integrated squared fourth derivative penalty.

Step ν, for $\nu = 1, 2, \ldots$: Modify the estimate $\delta_i^{(\nu-1)}$ on the previous iteration by

$$\delta_i^{(\nu)} = \delta_i^{(\nu-1)} - \alpha\frac{(\partial/\partial\delta_i)\texttt{REGSSE}}{(\partial^2/\partial\delta_i^2)\texttt{REGSSE}},$$

where α is a step-size parameter that can sometimes simply be set to one. It is usual to drop the first term (7.6) in the second derivative of REGSSE since it vanishes at the minimizing values, and convergence without this term tends to be more reliable when current estimates are substantially far from the minimizing values. Once the new shifts are estimated, recompute the estimated average $\hat{\mu}$ of the shifted curves.

Although the algorithm can in principle be iterated to convergence, and although convergence is generally fast, we have found that a single iteration is often sufficient with reasonable initial estimates. For the pinch force data, we began by aligning the smoothed curves by setting the location of the maximum of each curve at 0.1 seconds. The shifts involved ranged from -20 to 50 milliseconds. We then carried out a single Newton-Raphson update ($\nu = 1$ above) where the range $\mathcal{T}$ of integration was from 23 to 251 milliseconds. The changes in the δ_i ranged from -3 to 2 milliseconds, and after this update, a second iteration did not yield any changes larger than a millisecond. The aligned curves are shown in Figure 7.13.

As part of a technique that they call *self-modelling nonlinear regression*, which attempts to estimate both parametric and nonparametric components of variation among several curves, Kneip and Gasser (1988) use linear

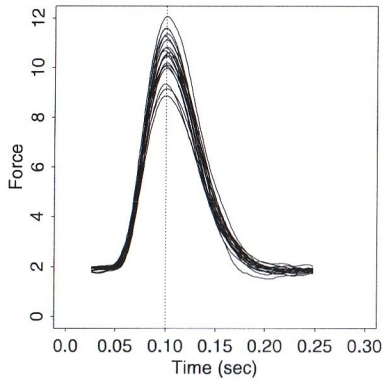

Figure 7.13. The pinch force curves aligned by minimizing the Procrustes criterion REGSSE.

transformations of t, that is both shift and scale changes. Kneip and Engel (1995) use such shift-scale transformations to identify "shape invariant features" of curves, which remain unaltered by these changes in t.

7.10 Further reading and notes

The classic paper on the estimation of time warping functions is Sakoe and Chiba (1978), who used dynamic programming to estimate the warping function in a context where there was no need for the warping function to be smooth.

Landmark registration has been studied in depth by Kneip and Gasser (1992) and Gasser and Kneip (1995), who refer to a landmark as a *structural feature*, its location as a *structural point*, to the distribution of landmark locations along the t axis as *structural intensity*, and to the process of averaging a set of curves after registration as *structural averaging*. Their papers contain various technical details on the asymptotic behavior of landmark estimates and warping functions estimated from them. Their papers on growth curves (Gasser et al., 1990, 1991a,b) are applications of this process. Another source of much information on the study of landmarks and their use in registration is Bookstein (1991).

Ramsay (1996b) and Ramsay and Li (1996) developed the fitting of a general and flexible family of warping functions h_i making use of a regular-

ization technique. Their work used a piecewise linear basis for function W in order to avoid numerical integration, but our subsequent work has found numerical integration to be easy to apply here as well as elsewhere, and consequently W may now be expanded in terms of any basis. Kneip, Li, MacGibbon and Ramsay (2000) developed a method that is rather analogous to local polynomial smoothing for identifying warping functions that register a sample of curves.

Wang and Gasser (1997, 1998, 1999) and Gervini and Gasser (2004) have evolved registration technology that does not use landmarks in a number of useful ways, and consider some important theoretical issues. Liu and Müller (2004) advanced their theoretical framework by discussing curve registration in the context of a model for random or stochastic functions where time is itself transformed in a random manner. They propose the operation of taking a *functional convex sum* as a way of computing convex sums of unregistered functions. This operation defines a type of mean that preserves the locations and shapes of features. See also Rønn (2001) for a model-based approach to shift registration.

The functional two-sample functional testing problem considered by Munoz, Maldonado, Staniswalis, Irwin and Byers (2002) uses landmark registration of some image density curves as a pre-processing step.