Springer Series in Statistics

James Ramsay
Giles Hooker

# Dynamic Data Analysis

## Modeling Data with Differential Equations

Springer

# Springer Series in Statistics

More information about this series at http://www.springer.com/series/692

James Ramsay · Giles Hooker

# Dynamic Data Analysis

## Modeling Data with Differential Equations

Springer

James Ramsay
Department of Psychology
McGill University
Ottawa, ON
Canada

Giles Hooker
Department of Biological Statistics and
Computational Biology
Cornell University
Ithaca, NY
USA

ISSN 0172-7397 ISSN 2197-568X (electronic)
Springer Series in Statistics
ISBN 978-1-4939-7188-6 ISBN 978-1-4939-7190-9 (eBook)
DOI 10.1007/978-1-4939-7190-9

Library of Congress Control Number: 2017941073



Printed on acid-free paper

This Springer imprint is published by Springer Nature
The registered company is Springer Science+Business Media LLC
The registered company address is: 233 Spring Street, New York, NY 10013, U.S.A.

# Preface

Getting pregnant is usually easy and fun, but the gestation and delivery may be another story; messy and painful perhaps, but instructive nevertheless. So it is with this book, which began with enthusiasm and confidence, but ten or so years later the twists and turns along the way emerge as a key part of the story.

Functional data analysis leads inevitably to dynamic systems. Ramsay and Silverman (2005) emphasized the reduction in bias and sampling variance that could be achieved by incorporating even an only approximately correct model into the penalty term by using a linear differential operator, thereby extending the more usual practice of defining roughness by the size of a high-order derivative. It was a natural next step to consider how one or more parameters that were needed to define such an operator might be estimated from data. Data associating the incidence of melanoma with solar activity became a prototype problem.

Principal differential analysis—a specific case of what we here term *gradient matching*—and its close resemblance to principal components analysis was the subject of a later chapter in Ramsay and Silverman (2005). The availability of high-resolution replicated data resulted in some rather successful applications, notably to data recording complex physical motion: handwriting and juggling. These analyses estimated low-dimensional basis systems that could be used to define the linear differential operators whose kernels were spanned by these systems which largely captured the variation in the data. This leads to the somewhat clumsy attempt, judging by limited attention that it received, to introduce the reader to Wronskians and Green's functions, not to mention such ethereal topics as reproducing kernel Hilbert spaces (we still regard the later chapters of Ramsay and Silverman (2005) as highly instructive for the so-inclined reader). In contrast, chapters on function linear regression, which did seem to appeal to readers, were for us only a partial success. Our attempts to use functions as covariates in a regression equation to approximate other functions seemed to us only marginally successful and, as a premonition of trouble to come, we were struck by how hard it was to find data to illustrate the methodologies involved.

The first serious attention given to parameter estimation for differential equations began with a collaboration with N. Heckman (Ramsay and Heckman 2000) where

we noted that the large number of parameters in the smoothing function relative to the few defining the operator tended to lead to overfitting of the data and bias in the estimates of the parameters of interest. The idea of parameter cascading described in Chap. 9 came from realizing that the implicit function theorem provided a way out of this dilemma by replacing the unrestricted coefficients of the basis function expansion by a smooth function of the parameters being estimated.

We benefited enormously by a close collaboration and friendship with Kim McAuley and Jim McLellan in the Department of Chemical Engineering at Queen's University, who were able to steer us to the large literature in that field on the nonlinear least squares estimation strategy described in Chap. 7, and to pass along the nylon and refinery data used in various chapters.

We discovered again just how hard it was to find data that we could use in demonstration and test analyses in the engineering world, where data are owned by industrial concerns protected against access by competitors. When we turned to the large literature on dynamics in various fields like biology, epidemiology and physiology, we found almost no use or display of data. For example, nearly every text used the spread of disease (SIR) or the closely related Lotka-Volterra equations as a first illustration of a nonlinear system; but, if data were available at all, it was only on infected cases or predator abundances, where data fits were essentially only smooths of the data and therefore uninformative. Only recently have we discovered the invaluable archive of dynamic systems with data assembled by Klaus Schittkowski (Schittkowski 2002), which has been a great help in completing this book, and we are most grateful for his cooperation.

Why, we asked, were data-based estimations of dynamical system so hard to find? One answer seemed to be the rather restricted set of parameters yielding the solution characteristics that motivated such systems as the Lotka–Volterra, SIR, tank reactor and the FitzHugh-Nagumo, which are featured in this volume. Parameter estimation strategies, including our own, were prone to bouncing parameters into regions where they generated completely inappropriate solutions.

But another possible explanation for the paucity of data is that, in many fields, the differential equation is viewed, effectively, as data itself. That is, if there are solutions of a proposed system that exhibits the shape characteristics seen in experiments and natural settings, such as oscillations in predator-prey abundances, then these systems are considered to be demonstrations that a scientific understanding of the live system has been gained. We observed that papers on dynamics without any display of fits to data often appeared in journals like *Nature* and *Science*. Even the Hodgkin-Huxley papers, themselves exceptional examples of data-based science, were quickly followed by downgrades of their model to those like the FitzHugh–Nagumo that retained the general shape features but were not intended as data models.

Ironically, the idea of the equation as data, although rather difficult for an information scientist to warm to, dovetails beautifully with the parameter cascading algorithm that we discuss in Chaps. 9 and 10, where we allow a smooth and continuous set of compromises between data-fitting and equation-solving. This

tension between data and equation is everywhere in evidence in these pages as well as the dynamical system modeling literature. It is, in fact, why we wrote this book.

Our central concept of a dynamical system as a buffer that translates sudden changes in input into smooth controlled output responses has led us to applications to data that we have previously analyzed, such as the daily Canadian weather data and the Chinese handwriting data (Ramsay 2000). We hope to have opened up entirely new opportunities for dynamical systems where none were envisaged before, which involve extensions of the functional linear model and what we call *dynamic smoothing*.

Dear reader, if you have survived to this point in this Preface, you must be wondering how much you need to know to read further. Take heart! We have worked hard to keep the technical level as low as possible, and our first goal is to bring those with little or even no exposure to differential equations as modeling objects into this fabulous data analysis landscape. Our emphasis on linear systems reflects a belief that nature is a tough place where only rugged and stable systems exhibit adaptive behavior over a wide range of environmental conditions survive.

We thank our former graduate students David Campbell and Jiguo Cao for their own versions of dynamic systems analyses. Cornell University contributed, besides the Tennebaum and Pollard team, our colleagues Steve Ellner and John Guckenheimer whose writing, experimentation and mathematical analysis continue to inform and inspire our work. We thank our Queen's collaborators Jim McClellan and Kim McAuley, as well as their students and colleagues for their guidance through the fascinating chemical engineering world and their hospitality and support.

Ottawa, Canada James Ramsay
March 2017

# Contents

# Symbols

| | |
|---|---|
| $t$ | A time value |
| $i$ | Usually, the index of a variable in a system of differential equations |
| $d$ | The total number of variables in a system |
| $j$ | Usually, the index of an observation |
| $n$ or $n_i$ | The number of observations for variable $i$ |
| $t$ | A vector of time values of length $n$ |
| $x_i$ | Used to designate a single variable among a set of $d$ variables |
| x($t$) | Used to represent the state vector of the system at time $t$. In the case of the SIR models above we have *x(t) = (S(t), I(t), R(t))* and we will continue to refer to the constituent parts of the state vector by other names or by using subscripts as in $\mathbf{x}(t) = (x_S(t), x_I(t), x_R(t))$. When the state vector x is viewed as a function of time, it will be described as the state *trajectory*. |
| $\omega_i$ | A positive weight to be applied to the fitting terms of the $i$th variable |
| **D** | The derivative operator that transforms a function $x$ into its time derivative $dx/dt$. $\mathrm{D}^m$ generates the derivative of order $m$. $\mathrm{D}^{-1}$ generates the antiderivative. Dx($t$) is the vector of time derivatives of x($t$). |
| $\beta$ | A coefficient or rate function in the homogeneous portion of a linear differential equation. This may be a function of time or the values of external variables, but may not be a function of the values of variables in the system. |
| **L** | A linear differential operator that transforms a function $x$ into a linear combination of time derivatives $D^j x$. The coefficients in the linear combination can be functions of time, but may not be functions of values of variables. |
| $\theta$ | A vector of parameters that require estimation from data |

| | |
|---|---|
| $\boldsymbol{\vartheta}$ | A vector of parameters that require estimation from data augmented by initial values that must also be estimated from data |
| $x_0$ | Refers to the starting point of the trajectory. We distinguish this from $x(0)$ because it will often need to be estimated as additional parameters. When this is the case, we will use the augmented parameter vector $\vartheta = (\theta, x_0)$. The initial time 0 can be replaced by an suitable real number. |
| f(x\|$\theta$) | Represents a vector-valued function of x that depends on a parameter vector $\theta$. Generally this is used to represent the right-hand side of a differential equation. |
| $\alpha$ | A coefficient or rate function in the forcing portion of a linear differential equation that multiplies an additive external input $u$ into a linear differential equation |
| u($t$) | Describes a vector of external or forcing functions that are additive inputs into an autonomous or homogeneous system and therefore render it nonautonomous or nonhomogeneous. An individual forcing function is $u_\ell$. |
| L | The total number of forcing functions in a linear differential equation |
| $y_{ij}$ | An observation or measurement value at time $t_j$ for variable $I$ |
| $y_j$ | Represents the vector of measurements of length $d$ of a set of $d$ variables at time $t_j$. Sometimes it will be useful to write down a matrix of observation vectors over time. For this we will use $Y_j$ in which rows denote time points and columns the dimension of the observation vector. |
| $\phi\kappa$ | A known basis function used to approximate a trajectory $x$ in a linear expansion |
| $K$ | The total number of basis functions in a linear expansion |
| $\phi$ | A vector of basis functions of length $K$ |
| $c_k$ or $c_{ik}$ | Real numbers that are the coefficients in a basis function expansion of a trajectory |
| **c or C** | Either a vector or a matrix of coefficients for one or more basis function expansions |
| $\rho$ | A smoothing parameter in the half-open interval [0, 1) used the analysis of linear differential equations by program Data2LD |
| J | The inner fitting criterion in a parameter cascading analysis in Chaps. 9 and 10 |
| H | The outer fitting criterion in a parameter cascading analysis in Chaps. 9 and 10 |
| $\lambda$ | A non-negative smoothing parameter in used the analysis of linear or nonlinear differential equations by program CollocInfer |
| **I** | An identity matrix |

| | |
|---|---|
| **B** | A matrix of rate values for a system of linear stationary differential equations |
| **R** | The matrix defining the penalty term for the homogeneous part of a linear differential equation, defined and used in Chap. 9 |
| **S** | The matrix defining the penalty term for the forcing part of a linear differential equation, defined and used in Chap. 9 |

Additional symbols, primarily for variables in specific dynamical systems, are found in the index of the book.

# Chapter 1
# Introduction to Dynamic Models

## 1.1 Six Examples of Input/Output Dynamics

Science and baseball have much in common; what counts is not where the ball is but where it's going and how fast it's getting there. This places the focus on modelling change, and change is more complex than stasis. The trajectory of a baseball is greatly affected by spin, so that the three dimensions of motion combine with three spin coordinates to produce a six-dimensional system for change distributed over the single dimension of time.

It is inevitable, then, that derivatives with respect to time and space appear often in mathematical models as measures of instantaneous change, and in this book we consider how these *differential equations* can be estimated from data.

The book is also an introduction to a new horizon in statistics: an equation as a model for data, where a differential equation is only one possible sub-class, and where data-fitting, parameter estimation, model evaluation, inference and so on do not rely on finding an algebraic solution to the equation or even an accurate numerical approximation to a solution.

This chapter lets data from various settings speak to us about situations where it is the statistical description of *change* that is the primary focus. Simple dynamical models are proposed in each setting, but with no attempt to describe what we must do in subsequent chapters: estimate model parameters from noisy data, display the uncertainty in these estimates, and assess fits to the data from competing models.

### *1.1.1 Smallpox in Montreal*

Nothing leads to insight faster than facing a gruesome death. In April of 1885, when a man knocked on the door of the Hotel Dieu hospital in Montreal hoping to see a doctor, the 167,000 citizens had for a few years neglected vaccination against smallpox. Within a few months 3234 would die, and about 9600 would be infected,



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_1

many of whom would be permanently disfigured. Figure 1.1 tells the story with data. We see the consequences of the introduction of compulsory vaccination in 1876 by Sir William Hingston, the Irish Roman Catholic mayor of Montreal and its chief surgeon. Working with Bishop Bourget to overcome the fear by French Canadian catholics that the real purpose of vaccination was the decimation their population by spreading the disease, it took a couple of years before most of Montreal's children were immunized, and we see in the figure the decline over the next 2 years in smallpox fatalities from around 600 per year to zero.

But by 1885 noncompliance had returned, partly due to fear-mongering by a quack physician from New York. Smallpox arrived in Montreal on the newly constructed Canadian Pacific Railroad with soldiers returning from the suppression of the Riel Rebellion in the prairies, and spread with the speed that the figure makes only too obvious. The insight that vaccination was essential spread nearly as quickly.

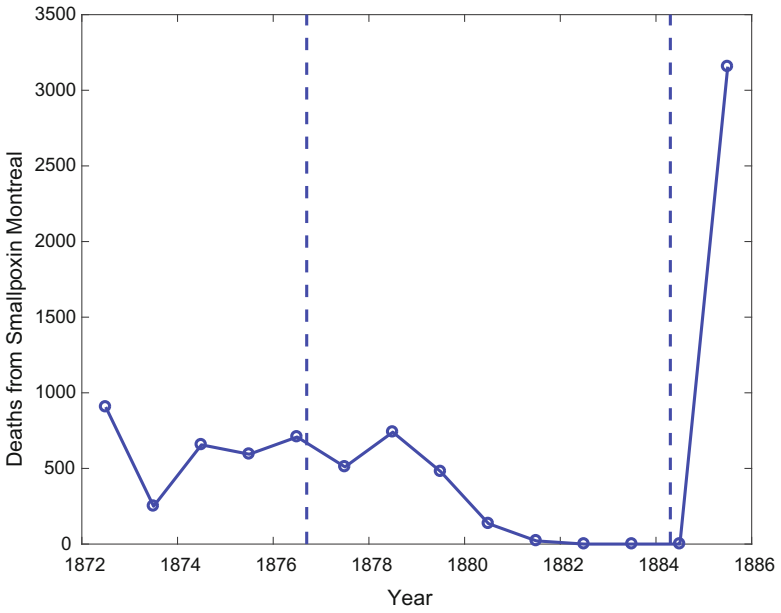


**Fig. 1.1** The number of smallpox deaths per year in Montreal, Quebec. The *first vertical line* indicates the introduction of compulsory vaccination, and the second the re-introduction of vaccination following the last smallpox epidemic in North America

We are, naturally, impressed by the three levels of small pox fatalities, 600, 0 and 3000, but the data are more dramatic in terms of the two rates of change: the disappearance of the disease in 2 years or so, and its explosive return. Thus we see the structure in the data on two time scales, namely the two long-term mortality levels, and the two short-term transitions. We need to model both the levels and the rates of change. That is, we want a model that relates $x(t)$ to $\mathrm{D}x(t)$, where $x(t)$ is deaths per year and we use the notation $\mathrm{D}x$ for the first derivative of $x$ instead of $\mathrm{d}x/\mathrm{d}t$.

Focussing first on the decline in deaths in the 1876–1879 period, we see that, when the death rate is high, the downward slope is negative and large, but as the death rate approaches zero, the slope remains negative but declines to zero. It seems plausible that the proportional relation

$$\mathrm{D}x(t) = -\beta x(t) \text{ or } \tau \mathrm{D}x(t) = -x(t) \tag{1.1}$$

could get us somewhere. What well-known function has a derivative value negatively proportional to a function value? Introductory calculus and a little reflection leads to $x(t) = C\exp(-\beta t)$ for some arbitrary constant $C$ as the solution to (1.1), and from there to the nonlinear model equation

$$x(t) = Ce^{-\beta t} + e(t) \text{ or } x(t) = Ce^{-t/\tau} + e(t) \tag{1.2}$$

where an appropriate positive constant $C$ must be estimated and the residual $e(t)$ takes care of small and ignorable effects. Of these two equations, (1.1) is simpler and tells the story with the greater clarity, and gives us our first lesson in constructing or parsing a *differential equation* that links rate of change to level.

Parameters $\beta$ or $1/\tau$ are, in effect, regression coefficients in a simple single-predictor linear relation between "covariate" $x$ and "response" $\mathrm{D}x$. A useful rule of thumb is that $x(t)$ will decline from a level of $C$ to practically zero in $4\tau$ time units, leading to a rough-and-ready estimate derived from the 3-year disappearance of smallpox, so that the *time constant* $\tau \approx 3/4$ years and $\beta \approx 4/3$. That is, $\beta$ controls the *speed* with which the level changes to its new level, and consequently represents what we mean by the *dynamics* of the system. Highly dynamic systems have large $\beta$'s and small $\tau$'s, and sluggish systems have the respective converses.

This simple dynamic equation models a *buffering* process in which an impact of an event, here the introduction of compulsory vaccination, is spread over time. This concept of a differential equating defining a buffer will play a central role in this book.

We need a statistical method for using the smallpox deaths reported to the city in April of each year to estimate parameter $\beta$, along with confidence limits. The useful data are those from 1876 to 1883, with the three values over 1878–1880 being especially informative about the rate of decline in smallpox mortality. The single report in 1885 is evocative, but by itself insufficient for describing the time course of the infection.

### *1.1.2 Spread of Disease Equations*

But what can be said about the onset of smallpox? Viewing vaccination as itself a disease, as many Montrealers did, smallpox by comparison is far more dynamic. Indeed, it is only surpassed by measles as the most communicable disease in humans. We have only a single point to record the time course of smallpox infection, but here

is a system of two equations that is often used to model the infection process:

$$\begin{aligned} \mathrm{D}S &= -\beta SI \\ \mathrm{D}I &= \beta SI - \delta I \ . \end{aligned} \tag{1.3}$$

The $S$ variable stands for the number of individuals susceptible to infection, roughly all of the 167,000 citizens less those with immunity because of a prior infection. Variable $I$ represents the number of infected people. Focussing on the first equation, here we have again the same relation as before between D$S$ and $S$. But with a difference! The speed of the impact of the infection is now $\beta I$, since we use $\beta$ here as the probability of infection of a single individual. This means that the population rate of infection gathers speed in proportion to the number of infected individuals, so that the number of uninfected persons goes down faster and faster as the epidemic spreads. Left to itself, the entire population would be infected in a year or so according to this model. But Mayor Hingston now had no trouble re-introducing compulsory vaccination, in spite of a riot or two, and this combined with severe quarantining arrested the disease by the end of the summer.

The second term in the left of the $I$ equation also has this proportionality of D$I$ to $I$, and this is due to infected persons becoming noninfective due to either recovery or death, with a speed represented by $\delta$. But of course it also has the first term, representing the transfer of the new infections into the infected pool. As a whole, the $SI$ equations seem to represent an *anti-buffer*, but only because the reaction speed of the buffer itself changes over time, in this case for the worse. We will return to simple first order buffers like these when we come to the chapters on how to use the data to estimate a buffer's characteristics.

### *1.1.3 Filling a Container*

Figure 1.2 shows the level of a fluid in a tray within an oil refinery distillation column before and after a valve setting is changed. This is a simple input–output system that resembles the structure of the smallpox data following the onset of vaccination. In this case, however, we have much more detailed data on both the input (V) and output (T) processes. We see that a change in valve setting at time point 67 results in a change in fluid level in Tray 47, and this change moves virtually instantaneously. The slope of the tray level then rises steadily from this initial large value to find a new equilibrium. This is what we would expect if we increase the flow into a tank with a constant pressure-driven outflow: a rapid rise which levels off as the level rises and the pressure on the outflow increases until it matches the new inflow rate.

We will see that the simple differential equation

$$\mathrm{D}T = -\beta T + \alpha V \tag{1.4}$$

does a fine job of representing these changes. Using our trick of estimating $\beta$ as the reciprocal of the time for 2/3 of the change to be realized, we see that $\tau$ is roughly 50, implying the value $\beta = 0.02$. The total change is about 4.5 level units, and we call this the *gain* of the system. It turns out that the gain, denoted by $K$, is $K = \alpha/\beta$, suggesting that $\alpha = K\beta$, or about 0.09. We will return to these data in a number of chapters as simple dynamic system furnished with abundant data.

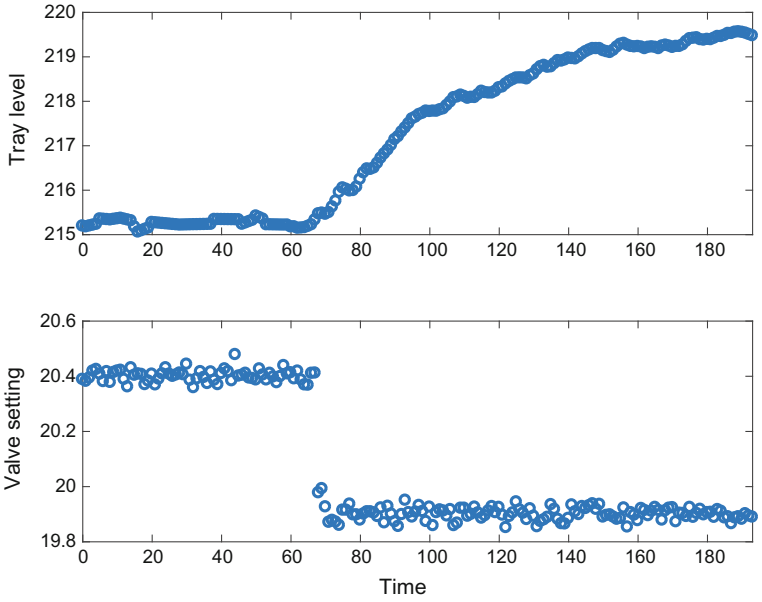


**Fig. 1.2** The *upper panel* shows the fluid level in a tray that receives input from another container when a valve is opened. The *lower level* shows the setting of a valve before and after being opened

### *1.1.4 Head Impact and Brain Acceleration*

The data in Fig. 1.3 were collected in a study to measure the effects of motorcycle accident on the driver's brain tissue. They were measurements of acceleration of brain tissue within the cranium of a corpse before and after being struck by a blunt object with a force typical in a collision by a motorcycle driver's head with a hard surface. Analyses of the data have been reported by many authors, including Härdle (1990) and Silverman (1985). This, too, is an input/output system, although we had to construct the input ourselves by representing it as an abrupt but brief pulse of force positioned at the point where the hammer encountered the cranium.

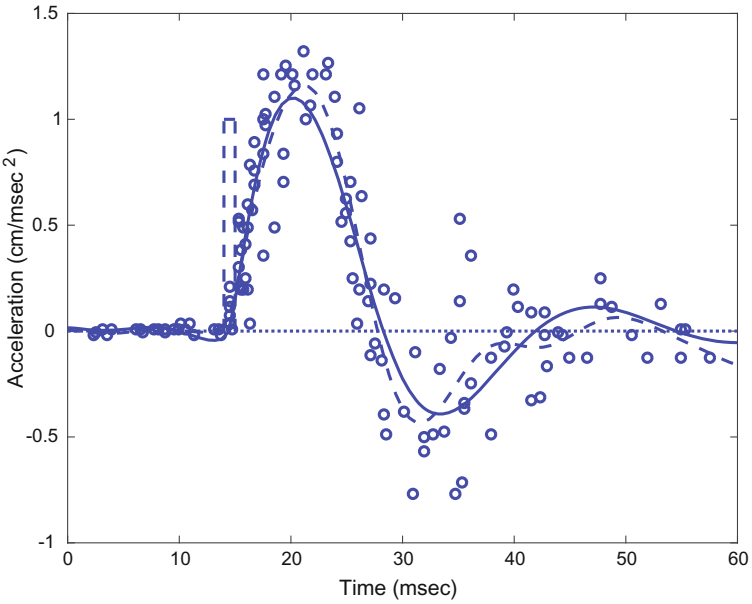


**Fig. 1.3** The *circles* indicate observations of acceleration of brain tissue from five replications of an experiment involving striking the cranium of a corpse with a blunt object. The *box* was constructed to represent the impact itself spread over one time unit. The *solid line* is the fit from the data for $\rho = 0.998$ and the *dashed line* is the fit for $\rho = 0.5$

The shape of the data indicates that there is a rapidly decaying oscillation in acceleration after the impact. Anyone who dipped a paddle into a still lake early in the morning knows that this is natural viscous fluids like brain tissue that cannot be compressed, but can dissipate energy by wave action. We will learn that this requires a system that uses the second derivative. We therefore analyzed the data using a three-parameter damped harmonic equation

$$\mathrm{D}^2 x(t) = -\beta_0 x(t) - \beta_1 \mathrm{D}x(t) + \alpha u(t) \tag{1.5}$$

where $u$ is the box function located at the impact time and with unit height and width which is displayed in the figure. Positive parameter $\beta_0$ defines the period or frequency of oscillation, and is often called the *stiffness* of the system. Parameter $\beta_1$ determines the rate of decay of the oscillations, and thus plays a role similar to that of the parameter in the smallpox equation (1.1). Impact pulse $u$ is an external input to the system described by the first two terms on the right, and is often referred to as a *forcing* term. Parameter $\alpha$ is essentially a regression coefficient that determines the amplitude of the oscillations, sometimes called the *gain* in the system.

### 1.1.5 Compartment Models and Pharmacokinetics

The decline in smallpox deaths can thought of as having the properties of a bath tub, where deaths are like water going down its drain. Perhaps you have noticed that when the plug is pulled, the water level at first goes down rather quickly, but that it drains out more and more slowly as the water level drops, so that you wind up waiting with shivering impatience for the dirty water to get out so that you can begin the cleanup. This happens because the pressure at the drain opening is proportional to the weight of the water above it, so that change in water level is proportional to level. First order dynamics, in short.

The uptake of drugs by the body is often a series of compartments, with the drug passing along from one to another. An injected substances passes from the tissue to the blood stream, from the blood stream to a target site such as the brain, from brain back into the blood stream, from the blood into the liver and so on. Each of these containers has its own elimination rate or drain size; and in each either all of the drug may be passed on, or some fraction may be absorbed or broken down and the balance eliminated. Simple first order dynamics seem to serve well to describe most of these systems, and pharmacologists find systems of first order constant coefficient differential equations highly useful in the discipline of pharmacokinetics (Fig. 1.4).

Suppose that at time 0 a drug is injected into a tissue site with concentration $C_0$, and that its concentration $C_a(t)$ in the blood stream declines at a rate proportional to its level, that is

$$\mathrm{D}C_a(t) = -\beta_a C_a(t) \ .$$

The decline in blood concentration is due to the drug being taken up by the target organs, and then eliminated by the liver with speed $\beta_b$, so that

$$\mathrm{D}C_b(t) = -\beta_b C_b(t) + \beta_a C_a(t) \ .$$

Note that $C_a(t)$ is a *forcing function* for the second equation.

It is typical that the absorption dynamics are rather faster than the elimination dynamics, and Fig. 1.5 shows how blood concentrations in fifteen patients increase rapidly at first, and then decline more slowly as the elimination process begins to dominate the absorption process. The Figure also illustrates the possibility that differential equations can vary randomly. We expect that constants such as $\beta_a$ and $\beta_b$ will vary from patient to patient, and so will the initial dose $C_0$ to at least some extent. That is, we often see differential equations in *multilevel* data structures where we have to model variation across levels (patients in this case) as well as within-level variation. Differential equations having randomly varying aspects are referred to as *stochastic*, although we have to be careful with the term *stochastic differential equation* since the phase has been co-opted by financial analysts and probability theorists to refer to something quite specialized.

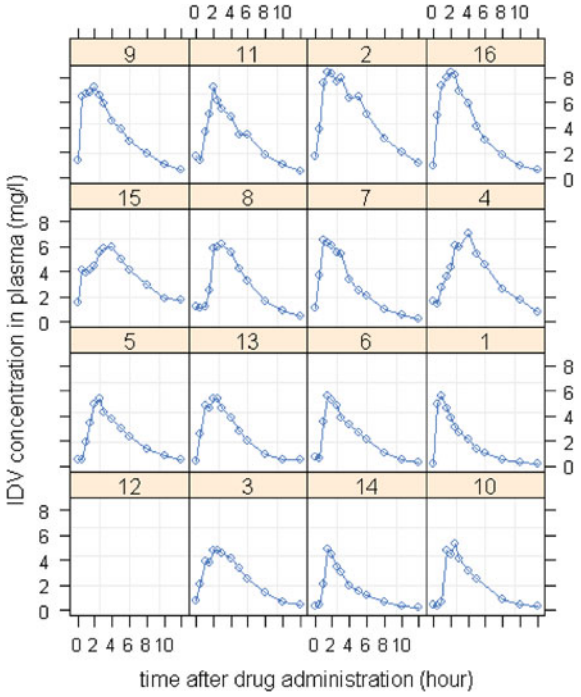


**Fig. 1.4** The uptake and elimination of a drug for 15 subjects as reflected in its concentration in the blood stream

These equations as well as the SIR equations are examples of *mass balance* systems where something is *conserved* in the system, so that each equation has to account for what happens to various fractions of that which remains constant. The central place occupied by differential equations in astronomy, chemistry, biology and physics is due primarily to conservation principles that imply that what goes in must come out somewhere, and which enforce smoothness of the time-varying aspects of the system. Conservation of momentum in mechanics, heat in thermodynamics, mass of chemical species in chemistry, volume in fluid dynamics and energy in all fields are examples. Even in phenomena as seemingly ephemeral as stock prices, total wealth is conserved as long as any stock can be sold for its paper value.

### *1.1.6 Chinese Handwriting*

The dynamics of the human body are, for most of us, endlessly fascinating. We are far from the fastest or the most skilled creatures on the planet, but when we watch ourselves drive home a goal on the soccer field or produce an arpeggio on a piano, the universe seems for those few seconds to collapse. How do we do this?

When we learn a movement, such as, say, handwriting, we at first make a few clumsy strokes, and then, again and again, until the shapes that we have in mind begin to emerge. Perhaps faster and with more precision for some of us than others, but by secondary school the production of script has been mastered, and one does it without needing any appreciable concentration.

Consider the Chinese script displayed in Fig. 1.6, which is the four characters for "statistics". This is the first of 50 replications of this script on a horizontal surface by Dr. Xiachun Li, a postdoctoral fellow working with one of us in 1996–1997. A small infra-red emitting diode was attached to the tip of the pen, and three cameras mounted on the recorded pen position 400 times a second with a typical error of about 0.5 mm. The total time for the production of the script varied slightly around 6 s. The experiment and some analyses were reported in Ramsay (2000).

The script includes these elements:

- strokes, curved or straight and in contact with the paper
- cusps, when the pen comes to a near standstill
- lifts, a stroke while the pen is more than 2.5 mm off the paper

On a hunch, we plotted the numbers from 1 to 46 at the times in the middles of 46 equal-sized time intervals, each 130.5 ms in length. It is striking that each numbered point corresponds closely to an event time, whether in the center of a curved stroke or a lift, or precisely at a cusp.

This constancy of event times for a well-learned sequence of movements has become quite familiar to us in our various investigations of human motion, although they are usually around 120–125 ms. The slight increase is no doubt due to the fact that the writing was executed while standing and on a surface substantially larger than would be used in normal circumstances.

We have conjectured that the brain achieves synchronized activation of many muscles by using a stable clock cycle of about this duration to fire a discrete pulse of neural activity over the many channels involved once each cycle. At the muscle end, the arrival of bundle of spikes resets the tension in the muscle fibre. It is well known (Kandel 2000) that the natural motion of the limb affected will be harmonic with little damping, and that the period of this oscillation will depend on the tensions in the attached muscles.

What would happen, then, if we saw the brain/arm/hand system as a single harmonic oscillator, each coordinate of which is defined by the equation $\mathrm{D}^2 x(t) = -\beta x(t)$, where $\mathrm{D}^2 x$ is the second derivative of variable $x$ and $\beta$ defines the period of oscillation along the coordinate direction? We would need two oscillators, one for the horizontal $X$ direction and the other for the vertical $Y$ direction. We could add one for $Z$ where the lifts play out, but let's not, given our two-dimensional viewing surface. This starts us off with two positive parameters, $\beta_X$ and $\beta_Y$, to estimate.

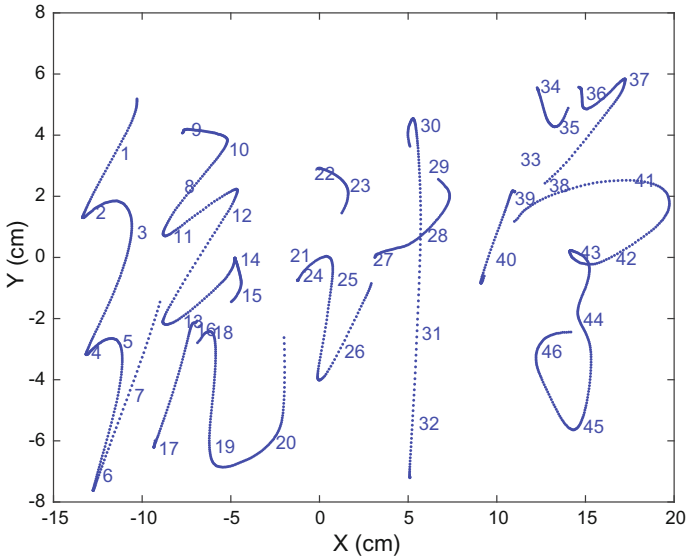


**Fig. 1.5** The *solid line* segments show the position of a pen while on a writing surface during a single writing of "statistics" in Chinese. The writing took 6 s, and the *numbers* are displayed at the centerpoints of 46 equally spaced time intervals

Now, suppose that this bundled neural activation of this harmonic system could be represented by two functions, $\alpha_X(t)$ and $\alpha_Y(t)$, each with 46 steps, and each step having the same 130.5 ms duration. We add these step functions to the right sides of the respective equations. We treat the height of each of the 46 steps as a parameter to be estimated, so that our total parameter load is $2 + 2 * 46 = 94$. This might sound like a lot of parameters, but we do have 2401 highly accurate observations per coordinate, and there are only two parameters per feature, with cusps, for example, requiring precise synchronization.

Using the estimating machinery that we will lay out in Chap. 9, Fig. 1.6 shows the estimated pen trajectory defined by the pair of dynamic equations

$$\begin{aligned} \mathrm{D}^2 X(t) &= -\beta_X X(t) + \alpha_X(t) \\ \mathrm{D}^2 Y(t) &= -\beta_Y Y(t) + \alpha_Y(t) \; . \end{aligned} \tag{1.6}$$

The reproduction of the actual script is remarkable, right down to reproduction of the cusps. The two estimated $\beta$ parameters correspond to 3/8 of a complete rotation over a subinterval, which is the amount of turning noted in the most curved segments. Figure 1.6 shows in each panel the estimated step functions, $\alpha_X(t)$ and $\alpha_Y(t)$, respectively.

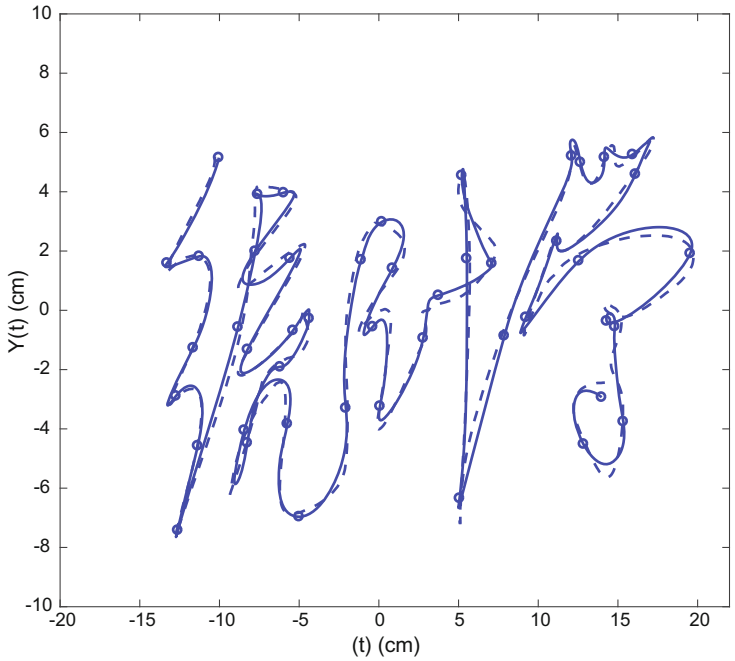


**Fig. 1.6** The *solid line* shows the estimated position on the writing surface of a pen whose motion is defined by the dynamic Eq. 1.6. The *dashed line* shows the actual pen position for this script. The *circles* are placed at the points where the step function input changes level

It would be easy matter to program a robot arm to carry out a writing of this script. Moreover, if we have in storage the roughly dozen steps required to produce any character, then one can see how a sequence of text could be input to the robot, and translated as rapidly as needed to the corresponding written script. More generally, we wonder if this isn't general recipe for any form of highly practiced human motion (Fig. 1.7).

### *1.1.7 Where to Go for More Dynamical Systems*

Although we have tried to show how differential equations can be used as models for observed processes, there is no substitute for seeing dynamic systems in action within the reader's favorite field. Fortunately, there are many texts available that highlight the modelling side of dynamics. Barnes and Fulford (2009), Blanchard et al. (2006) and Borrelli and Coleman (2004) are three readable introductions to dynamic modelling that cover a wide selection of areas. Ellner (2006) and a number of other texts are available as surveys of dynamics systems in biology, and Wilson (1999) does a great job for neuroscience. Don't, however, expect a great deal of data to be in evidence.

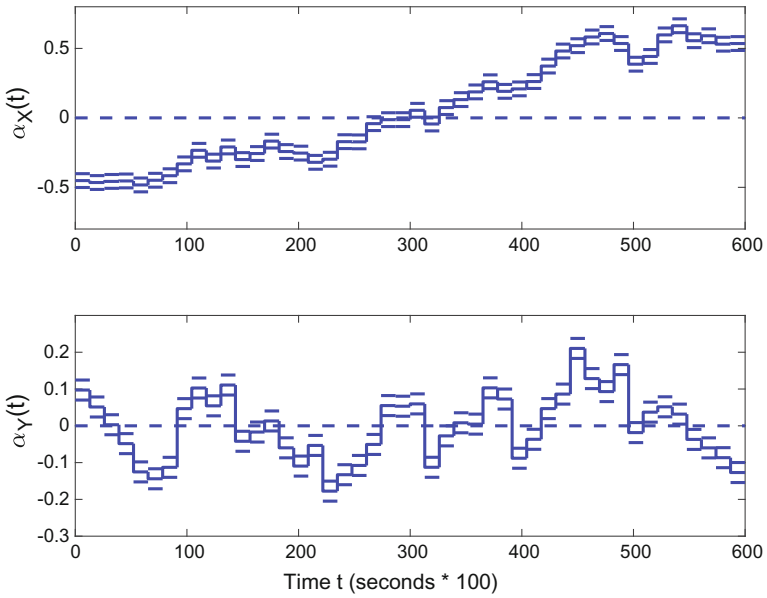


**Fig. 1.7** The *top panel* displays the estimated step function $\alpha_X(t)$ driving the horizontal component of the writing, and the *bottom panel* the corresponding input $\alpha_Y(t)$ to the vertical harmonic oscillator

For live data, our sources have been data collected by our colleagues as well our own collections. But Schittkowski (2002) has a remarkable collection of data, both simulated and real, along with literature citations and analyses of the type that we call trajectory matching in Chap. 7. A computer application, EASYFIT, is also available from the author that both gives access to these data and provides the analyses. We hope that this book provides a link between the statistics community and application-oriented mathematicians like this author.

Those who prefer an historical approach will have to begin with the origins of calculus itself; James Bernoulli, only three years after the publication of Newton's *Principia Mathematica*, modelled pendulum dynamics (Edwards 1979). The chemical engineering and process control literature is a great way to see dynamic models used in feedback systems (Marlin 2000). The first chapter of Deuflhard and Bornemann (2000) surveys a wide range of scientific applications of dynamics. Keeling (2008) can be consulted for more on modelling infectious diseases.

## 1.2 What This Book Undertakes

There is a large literature on parameter estimation methods in engineering and sciences, and the chemical engineering community has probably made the greatest progress. Our Chaps. 7 and 8 review the two main approaches: *trajectory matching*

involving alternating between numerical approximation of an initial value problem and parameter optimization, and the other, *gradient matching*, in which data smoothing methods are used to estimate the derivatives that the equation requires, followed by nonlinear least squares regression. Unfortunately both involve of these strategies have severe limitations that at times require large computational overhead and in others lead to serious instability and inaccuracy issues.

Our approach is based on Ramsay et al. (2007), and represents a synthesis of these two previous strategies. Called either *generalized profiling* or *parameter cascading*, it represents a solution as a basis function expansion, typically by spline functions, and involves an inner optimization process in which the coefficients defining these expansions are optimized for fixed parameter values, and an outer optimization of the fit to the defined by the parameter values. The first of the two chapters, 9, considers linear differential equations such as head impact, compartment and handwriting examples. Here there is a closed form solution to the inner optimization, so that, helped by a few tricks, rapid and stable optimization with minimal user input is achieved. More generally, Chap. 10 requires more setup, including functions for evaluating the right sides of the equations and various derivatives, but can tackle nonlinear problems.

## 1.3 Mathematical Requirements

This book may interest a wide range of readers, going well beyond the usual statistical communities. For example, applied mathematicians wanting to check their models against data may have a limited exposure to statistical methods for estimating confidence limits on parameters and inferential tools for deciding among competing models. We have tried to be generous with background material and literature resources to ease their way into the data sciences.

The reader will notice that these examples all involved a more in-depth description of the physical processes underlying data than is common in more standard statistical methods and that the models are developed in a fair amount of detail before the data are examined. This *mechanistic* approach to modelling is common in the physical sciences and applied mathematics, but is rarely even mentioned in statistics curricula. We regard this disconnect as unfortunate: statisticians miss out on a large class of powerful modelling tools while those who work with differential equation models have largely developed *ad hoc* (though by no means misguided) procedures to deal with data. This book is intended for both audiences: we devote the first half of the book to describing ordinary differential equations and how to work with them mathematically, and then turn to statistical techniques for estimating parameters, quantifying uncertainty in them, assessing the adequacy of models, and planning experiments.

## 1.4 Overview

The first half of the book covers the mathematical framework of ordinary differential equations—their interpretation, their solution and their properties. Much of this material is found in applied mathematics texts and we do not provide a comprehensive treatment here. Our intention is an accessible introduction for those not familiar with dynamic systems as models and enough background to be able to work with them statistically. In the next chapter we outline the basic mathematical notation, terminology and the types of models that we deal with. Chapter 3 will provide a framework for solving linear ordinary differentials. Chapter 4 takes up nonlinear differential equations, and will be devoted to the types of models that are commonly encountered, their origins, and heuristics for understanding how the model structure determines the shape aspects of the solutions. Chapter 5 focuses on numerical methods for approximating solutions for *ordinary differential equations* (ODEs) or *initial value problems*, where enough information is available at at the beginning of the trajectory to define the solution. This chapter reviews commonly used Runge–Kutta methods as well as collocation methods and some of the numerical trouble that you can run into. Chapter 6 presents a survey of qualitative aspects of the solutions to differential equations. While nonlinear ODEs cannot be solved, it is possible to describe limit points or cyclic behavior. We present a brief review of the basics of these tools for understanding the system.

The second half of the text is given over to statistical methods for fitting differential equation models to data. In each of these we describe means of estimating parameters, the numerical issues involved and ways to quantify statistical uncertainty. Chapter 7 describes the most direct means of working with ODEs and data: varying parameters and initial conditions so as to minimize the distance between ODE solutions and the data. We have titled this *Trajectory Matching*. Chapter 8 deals with *Gradient Matching* where, rather than solving ODEs numerically, we obtain non-parametric estimates of $\mathbf{x}$ and it's derivative D, and choose parameters to make the ODE fit these as well as possible. Besides some numerical advantages, this also allows for ways to assess how the dynamic system may be extended and improved, and these can also be used with the profiling techniques in the next chapters. Chapters 9 and 10 introduce a trade-off between trajectory and gradient matching, which we have titled *profiling*. These methods combine some of the advantages of each. We work through them in detail for linear systems in Chap. 9 and then present the extension to general ODE models in Chap. 10.

Throughout these chapters we have provided explanations for the statistical methods used and the reader need not be familiar with nonlinear regression or statistical smoothing methods. We acknowledge, however, that as with the first part of the book, our treatment of these subjects is necessarily brief and throughout we refer the reader to useful texts for more details.

While the models in this book are more complicated than the linear regression framework that makes up much of introductory statistics, we have focussed on making the material as mathematically accessible as possible. We intend this book to

be useful for first year graduate students in statistics or applied mathematics as well as to researchers in applied fields with some mathematical or statistical knowledge. We do assume the reader is familiar with the techniques of calculus and linear algebra along with the common statistical paradigms of estimation, testing, confidence intervals and, in places, Bayesian analysis. We hope, however, that little beyond this background will be required.

An important area that we do not treat is stochastic models, or models in which the system evolves probabilistically. This is a large and important area with a substantial statistical and mathematical literature. Indeed, one way to view the lack of fit in differential equation models is as being due to random disturbances of the system. We note, however, that mechanistically-inspired models of probabilistic evolution frequently do not contain sufficient variability to adequately match real-world data and further flexibility is still often required; see for example He et al. (2010). Statistical methods for partially observed Markov processes have generally employed a set of computational tools that are different from those we describe in this book and we have therefore not pursued these here. We note, though, that there have been important developments for these models, see Ionides et al. (2006), Beskos et al. (2006), Andrieu et al. (2010), Wood (2010), Golightly and Wilkinson (2011) and Fearnhead et al. (2014) among a substantial literature. Developing the intersection between these two literatures is an important and largely unexplored topic.

Similarly, we do not treat *partial differential equations*: systems which evolve over both space and time. These have received very little attention, with only a few references devoting to fitting them to data (Ramsay 2002; Azzimonti et al. 2014). In broad outline, the methods from Chaps. 9 and 10 can be applied to these types of models, but much work needs to be done to fill these in. This framework is also quite different from the ideas in *data assimilation* (Kennedy and O'Hagan 2001; Kaufman et al. 2011) in which large-scale computer models for systems like global climate–often based on partial differential equations–can be run only a very small number of times and a compromise between their solutions and observed data must be made to produce forecasts or to obtain rough parameter estimates.

# Chapter 2
# Differential Equations: Notation and Architecture

## 2.1 Introduction and Chapter Overview

This and the next three chapters are resources, designed to get the neophyte started, to recall and extend the skills of the initiated, and to call the expert's attention to the relative importance of what is already known. In this chapter we first introduce and motivate the notation that we use throughout the book, and specifically the *operator notation* D$x$ for the derivative of a function $x$. That is, we view a derivative as a *transformation* of a function into a new function whose value at any location $t$ is the slope of $x$ at that point.

Differential equations are divided into various classes, and we first describe the autonomous/forced dichotomy defined by whether or not the system has an external or exogenous input. Then we consider linear versus nonlinear systems, a distinction that is already familiar to statistical readers in regression analysis.

The data are assumed to be distributed over the whole interval of observation, and to be subject to measurement error to other types of random disturbances. This contrasts with the errorless initial and boundary value data configurations that apply to so many models.

## 2.2 Notation for Dynamical Systems

### *2.2.1 Dynamical System Variables*

Dynamical systems concern observations distributed over a substrate, such as time, space, space/time, wavelength, frequency and so forth. We mostly focus on continuous dynamical systems, where the substrate has the properties of a line, plane or higher dimensional vector space. We tend to index positions on the substate by $t$ for derivatives with respect to one-dimensional continua such as time, and we reserve $s$ for planes and volumes and other higher dimensional structures. Thus, we leave out



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_2

systems defined explicitly over discrete countable and equally spaced points such as days, quarters, years and etc.; that is, systems dealt with in the time series literature. We are aiming to keep the mathematical level as low as possible, and we therefore assume that few readers will mind that we also ignore data distributed over curved surfaces and other manifolds, but it is worth stating here that such situations are both important and not rare.

These chapters are also restricted to the description of *deterministic* dynamic systems where all the elements in the system are nonrandom entities. By contrast *stochastic* dynamic systems explicitly incorporate random elements, such as randomly varying functional inputs, randomly varying parameters or random initial values. But stochastic systems are typically modifications of systems that have a deterministic core, so that it is wise to get a firm grip on deterministic systems first.

A variable $x$ in a dynamical system is a function of position on the substrate, and the corresponding values of the functions at specific substrate values are $x(t)$ and so on. We will also refer to these functions as "states", "paths" and "trajectories." But we must also remember that certain dynamic systems are so familiar in particular fields that specialized letter allocations are used. An example is the SIR system in epidemiology where the letters $S$, $I$ and $R$ refer to "susceptible", "infected" and "recovered" populations, respectively. Although we will define these symbols before using them, readers will have to be adaptable.

Most situations in applied statistical settings involve more than one variable, and we follow the usual practice of using bold face, as in $\mathbf{x}$, to refer to a vector of functions. Occasionally the data configurations and the dynamic models themselves call for rectangular and higher dimensional arrays, in which case capital boldface letters such as $\mathbf{X}$ are employed.

Statisticians are familiar with linear and nonlinear regression models where some variables are designated as covariate, independent, causal or input, and one or more others as dependent or output. Consequently, we will use tend to use $u$, $v$ and $w$ as well as their boldface versions for these input variables, which may vary over the substrate $t$ or $s$. Typically the dynamical systems literature refers to these variables as "forcing" functions, and especially when they affect the dependent variable in an additive manner. Additive forcing tends to change the location of $x$ in function space but not alter its fundamental behavior. By contrast, inputs which affect output variables in a multiplicative way or have more complex effects will usually change more drastically the shape characteristics of an output trajectory.

### *2.2.2 Dynamical System Parameters*

Models almost always involve parameters that define the dynamical system, and which typically must be estimated from the data at hand or from data collected in collateral experiments. Here we follow standard practice in using the Greek alphabet for parameters, with boldface versions where vectors of parameters are assumed, and $\boldsymbol{\theta}$ will be the usual choice.

A fundamental goal is to estimate the values of these parameters from data, and also estimate the precision of these estimates given the noise level in the data. We may also conjecture that parameter values will vary randomly, typically in some structured way, from one realization of the system to another, and such parameters are called *random effects*. We will naturally then want to estimate characteristics of their variation, such as their means, variances and covariances.

### 2.2.3 Dynamical System Data Configurations

Since we approach dynamic systems as models for data, we will consider data configurations that are seldom considered in textbooks on the mathematics of dynamical systems, which tend to focus heavily on the *initial value* problems arising in science and engineering where data are only available at time $t = 0$, and are assumed to be virtually free of error. They also consider *boundary value* problems, where the system is also constrained to be in a specific state at some other fixed time $t_1$. By contrast, the statistician is seldom willing to assume error-free data, and we focus more on *distributed data* problems where data are available at a considerable number of time points for at least some subset of the variables involved in a dynamic system.

Data configurations are apt to vary from one variable to another. For example, some measurements are inexpensive and available on line, such as temperature, but others such as chemical assays are expensive and available only some time after the experiment is completed. It is typical that some variables will not be observed at all, so that these variables have some of the characteristics of the latent variables often used in various areas of statistics to model known but unobservable sources of variation.

### 2.2.4 Mathematical Background

We do need to know some of the fundamental theorems to be found in mathematical texts, and we shall try to state these as clearly as we can, but leave the reader to turn to textbooks for the proofs of these results. Moreover, a fair part of a college text on differential equations is devoted to techniques for calculating algebraic solutions to certain classic equations and classes of equations. But these constitute a tiny fraction of the differential equations in actual use. Thus, although it certainly worthwhile to derive solutions in a few simple cases, much more reliance will be placed in this book on numerical methods for the accurate approximation of solution values given both the values of parameters defining the system and, in most cases, initial states of the systems. Chapter 5 is devoted to this approximation technology.

Fortunately for both the readers and writers of this book, some fine texts have been written for first-timers and more advanced students of dynamical systems. We especially like Tennenbaum and Pollard (1963), a classic text that is now available as

an inexpensive paperback, and which has received the highest reader rating among the texts distributed by a well-known on-line bookseller. It is organized into 65 lessons, each corresponding roughly to a single lecture in a two-semester course, and each lesson's topic is illustrated with examples of historical and practical interest. Another source now in paperback is Coddington (1961). Engineering texts often have readable accounts of dynamical systems in the context of specific topics such a electrical circuit theory or chemical process control. Textbooks such as Barnes and Fulford (2009), Blanchard et al. (2006) and Borrelli and Coleman (2004) emphasize how differential equation systems are constructed in order to capture the behaviour of specific observed systems.

## 2.3 The Architecture of Dynamic Systems

### *2.3.1 Elementary Autonomous or Unforced Systems*

Dynamical systems are defined by equations involving one or more derivatives. The most elementary of these define polynomials; the solution to $\mathrm{D}^m x = 0$ is the space of all polynomials of degree $m - 1$. But most differential equations have expressions on the right side, and we will give considerable attention to the prototypical

$$\text{Diffusion equation or first order buffer: } \mathrm{D}x = -\beta x \ , \tag{2.1}$$

where $\beta$ is a nonzero positive constant. The solution to this system is $x(t) = C\exp(-\beta t)$, constant $C$ being arbitrary. Notice here that (2.1) hides the role of $t$, which we could bring back in by expressing the equation as $\mathrm{D}x(t) = -\beta x(t)$ or, more obsessively, as $(\mathrm{D}x)(t) = -\beta x(t)$. The result of removing $t$ is a nice clean expression that works well when we already understand what the substrate variable $t$ is.

The *order* of a differential equation is the highest order of derivative involved, so that, for example,

$$\text{Harmonic equation: } \mathrm{D}^2 x = -\omega^2 x \ , \tag{2.2}$$

is of order two. Its solution is $x(t) = C_1 \sin(\omega t) + C_2 \cos(\omega t)$, where both $C_1$ and $C_2$ are arbitrary constants. The wavelength of the sinusoid is $2\pi\omega$ and $2\pi/\omega$ is its period. The harmonic equation in turn is a special case of

$$\text{Second order buffer: } \mathrm{D}^2 x = -\beta_0 x - \beta_1 \mathrm{D}x \ . \tag{2.3}$$

It is common and convenient notation to isolate the highest order derivative in this way on the left side of the equation without any multiplier, and to move all other elements of the equation to the right side. According to this convention, a single differential equation of order $m$ can be expressed as

$$\mathrm{D}^m x = f(x, \mathrm{D}x, \ldots, \mathrm{D}^{m-1}x) \ . \tag{2.4}$$

Models of practical value in the real world typically require an account of how several variables interact to produce the behavior of a system, as we saw for the SIR and handwriting models in Chap. 1. Hence we need to gather several variables $x_i$, $i = 1, ..., d$, into a vector of variables $\mathbf{x}$ of dimension $d$. Letter $d$ carries a heavy load in this book, but as a rule it will be clear from the context when we use $d$ to indicate the *dimension* of a system, rather than a differential $\mathrm{d}x$ or $\mathrm{d}t$. This another reason for preferring the operator notation $\mathrm{D}x$ as our standard notation for a derivative.

An equation of order $m$ can always be reduced to a set of first order equations by defining artificial variables. For example, if we define $y = \mathrm{D}x$, then the second order linear equation (2.3) is split into the two order one equations $\mathrm{D}x = y$ and $\mathrm{D}y = -\beta_0 x - \beta_1 y$. Of course, the result is often neither natural nor easy to read, so that keeping multiple orders explicitly in a single equation is often to be preferred for clarity. But this does allow us to express any dynamical system, whether of order 1 or of order $m > 1$, as the differential equation system

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}) \tag{2.5}$$

where right-hand function $\mathbf{f}$ has a vector-valued argument and maps the values of $\mathbf{x}$ into a vector of dimension $d$, and where, moreover, some of the variables in $\mathbf{x}$ may be dummy variables coding one or more derivatives of one or more state variables. However, while this level of abstraction may have some appeal and utility in describing theoretical results, in practice we prefer to express dynamical systems in a manner that keeps the special structure of the system visible.

Parameters, like $\beta$ in the simple first order diffusion system above, $\omega$ in the harmonic equation and $(\beta_0, \beta_1)$ in the second order buffer equation play a prominent role in dynamic models, and we often like to keep them up front in our expressions. But if we opt for abstraction, then we can collect all the $n$ parameters into a parameter vector $\boldsymbol{\theta}$ in the expression

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}|\boldsymbol{\theta}) \ . \tag{2.6}$$

Occasionally, there may even be remaining time-variant aspects to a system, in which case we can add a $t$ itself as an additional argument in $f$ so that

$$\mathrm{D}\mathbf{x}(t) = \mathbf{f}(t, \mathbf{x}|\boldsymbol{\theta}). \tag{2.7}$$

### 2.3.2 *Forced or Non-autonomous Systems*

Now we turn to the critical role of input or forcing variables, such as force $F$ in

$$\text{Newton's second law: } \mathrm{D}^2 x = F/M \ , \tag{2.8}$$

where $x$ is position and $M$ is the mass of the body on which the force is acting. Dynamical systems model change, and are essential where the response of an system to a change in an input is not direct or straight-through. Often, for example, inputs can change abruptly, or arrive as short but intense bursts, and the system does not have the sensitivity or the energy resources to respond immediately to these nearly instantaneous changes. Cars can't stop instantly, so that the yellow traffic light permits the driver to spread the stopping response over an acceptable time interval. We use the term *buffer* to describe how a dynamical system spreads its response over a longer time interval than that over which the input changes.

Let $\mathbf{u}$ be a vector of $\ell$ external "forces" that change the behavior of a system in some way. We want to distinguish the behavior of a system isolated from any such inputs from one where they apply. Forced dynamic systems are expressed as

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}|\boldsymbol{\theta}, \mathbf{u}) \ . \tag{2.9}$$

An important special case, that is often assumed in the term "force", is forcing of an additive nature, expressible as

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}|\boldsymbol{\theta}) + \mathbf{g}(\mathbf{u}) \ . \tag{2.10}$$

If a system is relatively linear in the sense defined in Sect. 2.4.1, and the input is additive, the system will tend to respond in a more or less mild and predictable manner. For example, a step change in input will produce a more gradual approach to a new level in the output. A simple example that we will explore in more detail is the equation defining the concentration $C$ of a chemical species in the output of a chemical reactor with volume $V(m^3)$ and input flow rate $F(V/min)$

$$\mathrm{D}C = -\beta C + FC_0/V \tag{2.11}$$

where the speed-of-reaction parameter $\beta = (F + Vk)/V$ and $k$ is the rate constant for the reaction. Here $C_0$ in the second forcing term on the right side is a fixed equilibrium concentration, and we will see in the next chapter that $C$ will approach $C_0$ at an exponential rate no matter what the initial concentration $C(0)$.

But external values can also affect other aspects of the system, and specifically an external variable may change one or more parameter values of an otherwise autonomous system. When this happens, and it does often, the results can be much more dramatic. Stable systems can become unstable, relatively predictable systems can exhibit surprising variation, or even the seemingly unpredictable shapes called *chaotic* in the dynamical systems literature.

### 2.3.3 Differential Operator Notation

We can go further in emphasizing the input/output nature of a specific dynamical system. As an example, we can re-express the diffusion equation with forcing by input $u$, $\mathrm{D}x = -\beta x + \alpha u$, as

$$\mathrm{D}x + \beta x = \alpha u \ , \tag{2.12}$$

where parameter $\alpha$ modulates the impact of changes in forcing function $u$ on the system. Here we have put the autonomous part of the equation on the left side, and have effectively recast the model into regression format familiar to statisticians.

We use the notation $Lx = \beta x + \mathrm{D}x$, or more generally, $Lx = f(x|\theta) + \mathrm{D}x$ for (2.6), and call $L$ the *differential operator* associated with the dynamic system. In this operator notation, the differential operator for the forced diffusion equation is $L = \mathrm{D} + \beta I$, and for the second order buffer equation it would be $L = \mathrm{D}^2 + \beta_0 I + \beta_1 \mathrm{D}^1$. It is useful to think of $L$ as a generalization of the notion of a derivative, and especially since the Taylor series machinery for approximating functions at specific values of $t$, as well as other concepts associated with derivatives in elementary calculus courses, have versions in this more general setting. The autonomous differential equation in differential operator notation becomes

$$Lx = 0 \ . \tag{2.13}$$

In the next chapters, and elsewhere in the book, we explore how systems defined by specific differential operators respond to various prototypical inputs such as localized spikes, step functions, oscillating functions and random random noise processes.

Other differential operators can play a large role in scientific systems. For example, the curl and divergence operators, `curl` and `div`, are used as a compact way of expressing the four fundamental equations of electromagnetism, and are now used in many other contexts.

## 2.4 Types of Differential Equations

Between the simple diffusion equation and the general formulation (2.10) are a whole range of specialized classes of differential equations, each with their own special properties. It is essential at this point to learn the terminology for these classes before we take up their special characteristics in the next chapter.

### 2.4.1 Linear Differential Equations

Statisticians will not be surprised that linearity in differential equations, as in regression models, enables a host of specialized mathematical tools and results. We have

already seen three linear equations: the diffusion, harmonic and second order equations.

Here by *linear* we mean the linearity of the right side function $f$ with respect to its arguments $x$ and $x$'s derivatives. The single unforced linear equation of order $m$ is

$$\text{Order } m \text{ buffer: } \mathrm{D}^m x = \sum_{j=0}^{m-1} \beta_j \mathrm{D}^j x \quad \text{or} \quad L = \mathrm{D}^m - \sum_{j=0}^{m-1} \beta_j \mathrm{D}^j \ . \tag{2.14}$$

The *coefficient functions* $\beta_j$ may themselves vary over $t$, but must not depend on the values of $x$ or any of its derivatives.

Equation (2.14) may be extended in various ways that still preserve linearity. The highest order term $\mathrm{D}^m$ may also be multiplied by a coefficient function $\beta_m$, and if this coefficient goes to zero at a specific value $t$ over an interval, the equation then loses dimensionality as a system but retains linearity as a property.

Specific linear systems, such as the diffusion and harmonic equations, often have specific signs associated with their terms, such as the minus sign for the $x$-coefficient in the diffusion and harmonic equations. When this happens, it is often assumed that the sign of the coefficient does not change and, usually, the coefficient preceded by a minus sign is assumed positive. For example, in the second order buffer equation (2.3) it is common in many applications to assume that $\beta_0$, but not necessarily $\beta_1$, is always positive.

When all the coefficients functions $\beta_j$ are constants, we call the system *stationary* as well as linear, because that the fundamental structure of the dynamic system does not vary over $t$. Stationary linear systems have additional useful analytical techniques, especially, as we shall see later, the Laplace transform, and stationary systems are useful for a great many applications where the characteristics of a process do not substantially change within the time scale over which the system is observed. An example is the dynamics of electrical circuits where their physical characteristics have been rendered stable by cooling systems.

Be warned that many treatments of differential equations, and especially in the engineering literature, use the term "linear" to mean by default stationary as well. This unfortunate lack of clarity is also common in the statistical time series literature.

### 2.4.2 Nonlinear Dynamical Systems

Even seemingly minor departures from the linearity of the relation of right side function $f$ to its arguments can have dramatic consequences. For example,

$$\text{Catalytic equation: } \mathrm{D}x = \beta x(K - x) \ , \tag{2.15}$$

where $\beta$ is a positive constant has a solution

$$x(t) = K/\left[1 + \frac{K - x_0}{x_0}\exp(-\beta t)\right]$$

where $x_0$ is the value of $x$ at time zero and $0 < x_0 < K$. Solutions of this form are widely used in statistics for constraining function values to be within the interval $[0, K]$. We see that this happens because the slope of the function goes to zero as the value of $x$ approaches either zero or $K$.

A common source of nonlinearity is the appearance of products of variables in systems of two or more equations. For example, the equations of the SIR model for the spread of disease are

$$\begin{aligned} \mathrm{D}S &= -\beta SI \\ \mathrm{D}I &= \beta SI - \nu I \\ \mathrm{D}R &= \gamma I \end{aligned} \tag{2.16}$$

where $S$ is the number of susceptible but uninfected members, $I$ is the number of infected members, and $R$ is the number of recovered (including dead) of a population. Here the appearance of the product $SI$ renders these equations nonlinear, even though the equations are linear in their parameters $\beta$ and $\nu$. Equations of this type can be properly called quadratic in their variables, and constitute the largest subclass of nonlinear systems in applications. In our terminology for forcing, $S$ is rate-forced by $I$, $I$ in turn is rate-forced by $S$ (but with the opposite sign), and $R$ is add-forced by $I$. In the last $R$ equation, there is no term involving $R$, so that the change in $R$ is simply proportional to input $I$.

If the susceptible population $S$ is large, the number of infected individuals will exhibit exponential increase, so that the $I$-equation behaves as an *anti-buffer* that turns a small increase in infected individuals into an epidemic, as we saw with the smallpox data in Chap. 1. The great importance of the SIR model and its many variants demands that we return often to this example.

### *2.4.3 Partial Differential Equations*

Partial differential equations arise when more than one substrate is involved or when a substrate is multidimensional. The most common examples involve variation over space or over space and time. Partial differential equations play a huge role in climate modelling, the spatial aspects of spread of disease and pollution, fluid and airflow dynamics, two- and three-dimensional mechanical systems and a great many other areas in engineering and science.

Methods for working with these equations are beyond the scope of this book, since the mathematical technology involved can be substantially more advanced than for the ordinary differential equations that we consider. Nevertheless, they can retain aspects of ordinary linear systems. The Laplace equation

$$\frac{\partial x}{\partial t} = -\left(\frac{\partial^2 x}{\partial s_1^2} + \frac{\partial^2 x}{\partial s_2^2}\right) = -\triangle x \tag{2.17}$$

describes how spatial irregularities in the concentration of a solution diffuse over time to finally reach a state of equilibrium. The equation be viewed as a first order buffer in time forced by the negative spatial curvature of function $x(t, s)$ with respect to the two spatial coordinates $s_1$ and $s_2$. This forcing implies that over time hills (negative curvature) will be levelled and valleys (positive curvature) will be filled in.

### *2.4.4 Algebraic and Other Equations*

Mixtures of conventional algebraic equations, expressed as $g(x, u) = 0$, and differential equations arise in many contexts. These equations often define constraints on the solutions, of which a most common example is the initial value constraints $\mathbf{x}(0) = \xi$ where $\xi$ is a vector of real-valued constants.

In addition to algebraic equations, it is not unusual to have integral equations applying to the system as well. Moreover, delay-differential equations often are required where $\mathrm{D}x(t) = f[x(t - \delta)]$ for some nonzero lag $\delta$.

## 2.5 Data Configurations

The data available for estimating both the parameters defining a dynamic system and a function $\hat{x_i}$ can come from many different sources, and be related to the values $x_i(t)$ in many different ways. We specify some of the data configurations that are commonly encountered. We use $y$ as a rule to refer to data, and $y_{ij}, i = 1, \ldots, d; j = 1 \ldots, n_i$ to indicate data specific to variable $x_i$ and substrate value $t_{ij}$. Additional subscripts may also be needed to identify replicated data for specific pairs $(i, j)$.

### *2.5.1 Initial and Boundary Value Configurations*

By far the largest amount of mathematical material on dynamical systems is for *initial value problems* where the state of system $\mathbf{x}(t_0)$ at an initial time $t_0$ is known and the goal is to predict the behavior of a system at times $t > t_0$. Also included in initial value systems are those in which values of certain derivatives are also known, but we have seen that the use of dummy variables to label derivatives can convert these to the simpler initial state formulation.

Initial value systems are also of great interest to statisticians when the initial values are random, so that the trajectories or space curves with values $\mathbf{x}(t)$ are themselves random objects, possibly because initial values are measured with substantial

imprecision. In these cases, statisticians and probabilists find themselves teaming up with numerical analysts specializing in approximating solutions to the system given specific initial states and differential geometers bring their expertise on the analysis of structures on manifolds.

To initial value systems we can add boundary value systems where something is also known about the system at terminal time $t_1$. For example, a Brownian bridge is a trajectory that begins and ends at the same fixed value, such as a periodic process with fixed period.

### 2.5.2 Distributed Data Configurations

As a rule, we hope to work with measurements that are distributed over the complete range of observation.

This raises the critical question, "How many observations do I need?" What matters is not so much the number of measurements as how they are allocated along the line. Solutions to differential equations can have sharp localized features, often as a consequence of being forced by functions $u$ with abrupt changes. By a "feature" one means something like a change in level, a change in slope, a peak or valley, a crossing of some fixed threshold, or even a point of inflection. Using peak as an example, which needs to be defined in terms of its location, its amplitude and its curvature at the peak, three points carefully placed are the minimal configuration required to convey this information. But if the measurements are subject to any error, an accurate characterization can require anywhere from five to eleven points within the range of the peak depending on the amount of imprecision involved.

We call this the *resolution* of the data. Of course over regions in which nothing of interest happens, rather fewer observations will do, with the number depending on whether these regions are flat, tilted or subject to mild curvature. The refinery data in Sect. 1.1.3 are high resolution data by any standard, in contrast to the smallpox data where there are only a couple of observations defining the rate of change after vaccination.

### 2.5.3 Unobserved or Lightly Observed Variables

It is quite common to have some variables that are not observed or only sporadically measured. The spread of disease SIR model is a prototypical example. Fairly comprehensive records are often kept of the number of cases presenting with an infection, with daily or weekly observation times for variable $I$ being typical. On other hand, the number of susceptibles $S$ in a population may be available only annually, and then only approximately since the property may not apply to the whole population. A common situation is that $S$ is only observed once, at a time close to the beginning of the outbreak. The recovered variable $R$ may not be measured at all.

Having none or only a few observations for a variable can have a great effect on what can be estimated in a model. Certain parameters will typically be only determined by information on one or two variables, and therefore may be inestimable if these are not measured. The relationship holding between a measured and an unmeasured variable can also matter. If a variable is causal with respect to another, such as $S$ with respect to $I$, it will matter greatly that it is observed; but a downstream variable like $R$ can be well defined by a well defined upstream variable like $I$. If only $S(0)$ is observed, it can matter a great deal that the measurement error is small.

### *2.5.4 Observational Data and Measurement Models*

Whether data are available at only time $t_0$, or distributed over a range of $t$-values, it may be that the relationship of the data to a function value $x_i(t)$ may be nontrivial, rather than the usual straight-up additive error relation $E[y_{ij}] = x_i(t_j)$. For example, a set of 0's and 1's can indicate a set of binary outcomes occurring for variable $i$ with a probability $P[x_i(t)]$, in which case a logistic link generalized linear model is a likely choice for estimating function $P$ that connects the model to the data. Another example occurs when the data available relate to the sum, or some other combination, of two or more of the variables that cannot in practice be separated at the observational level. We may see this when we conceptually differentiate between reproducing organisms and the same organism in a non-reproductive phase of their life cycle, and it only the reproducing individuals that are affecting other variables.

It may happen, too, that the data for various variables differ enormously in their scale, as can happen for population and spread of disease models. In such cases we may consider either a model on the log scale with an exponential transformation of model values to fit the data, or logging the data prior to an analysis, which also requires a log-scale model, but with a straightforward connection to the data. That is, transforming variables can be important. In the log case, for example $\mathrm{D}x = f(x)$ is equivalent to $\mathrm{D}\ln x = f(x)/x$ or, if $y = \ln x$, $\mathrm{D}y = f(\exp y)\exp(-y)$.

The nature of a measurement can impact the data resolution in a distributed data configuration. For example binary variables convey much less information per measurement than directly connected continuous measurements with a high-signal-to noise ratio. A useful rule of thumb is that it takes at least five binary measurements at or near a location to yield the same resolution as a single low-noise continuous measure.

It happens often that the data do not provide enough information about the values of certain parameters to yield a useful estimate. For example, Chap. 10 looks at an example where a parameter specifying an upper limit cannot be estimated because there are no observations anywhere near such a limit. Similarly, estimating what a variable does near a lower limit such as zero requires that there be observations in that zone. Again, if a pair of parameters always appear in a product, then they are not individually identifiable.

The term *experimental design* in statistics refers to the the relationship between the data and elements of a model designed to fit them. Good experimental designs ensure that there are data informing the estimation of all elements of a model. Bad designs should usually be seen as an imperative to simplify the model before attempting any analysis.

## 2.6 Differential Equation Transformations

We noted above that variables can often vary over large scales, suggesting that it would be better to work with a logarithm of the data, whether with base $e$ or 10. Of course, that would imply that it would be convenient to know the differential equation in the log scale that is equivalent to the original. Letting $y = \log x$, we see that

$$\mathrm{D}\log x = \mathrm{D}y = \frac{f(x|\theta)}{x} \ .$$

That is, substitute $x \to e^y$ into the raw-scale equation and divide by $e^y$ to obtain the equivalent equation in the $Y$ or log-scale.

Similarly, a linearization of a differential model for a variable bounded below by 0 and above by $K$ would imply the transformation $y = \log[x/(K - x)]$ and therefore the multiplication of the right side function by the factor $K/(K - x)$.

## 2.7 A Notation Glossary

Now that we have introduced the basic set of models that will be employed in this book, we will collate our notation before continuing. The reader will have already seen some of these, but we will formalize it here. This book will largely be concerned with the combination of two models, although we have so far only formally discussed one. The first of these is the *process model* which describes the way a system changes over time. Formally, we will use

$x_i$ is used to designate a single variable among a set of $d$ variables.

$\mathbf{x}(t)$ is used to represent the state vector of the system at time $t$. In the case of the SIR models above we have $\mathbf{x}(t) = (S(t), I(t), R(t))$ and we will continue to refer to the constituent parts of the state vector by other names or by using subscripts as in $\mathbf{x}(t) = (x_S(t), x_I(t), x_R(t))$. When the state vector $\mathbf{x}$ is viewed as a function of time, it will be described as the state *trajectory*.

$\mathrm{D}\mathbf{x}(t)$ is the vector of time derivatives of $\mathbf{x}(t)$. The differential operator D will be used exclusively for time.

$\mathbf{f}(\mathbf{x}|\boldsymbol{\theta})$ represents a vector-valued function of $\mathbf{x}$ that depends on a parameter vector $\boldsymbol{\theta}$. Generally this is used to represent the right hand side of a differential equation so that the SIR differential equation can be expressed as $\mathrm{D}\mathbf{x} =$

$\mathbf{f}(\mathbf{x}|\boldsymbol{\theta})$. Again, the components of $\mathbf{f}$ can be indexed by named subscripts. In the SIR model we might have $\mathbf{f} = (f_S, f_I, f_R)$.

$\mathbf{u}(t)$ will be used to describe a vector of external inputs into the system that are not affected by system dynamics. If we imagine an external source of new susceptibles in the SIR spread of disease differential equation which increases $S$ with rate $u(t)$, the $S$ equation could be modified to $\mathrm{D}S = u(t) - \beta SI$. In a generic ODE model, we will include these by writing $\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}|\boldsymbol{\theta}, \mathbf{u})$.

$\mathbf{x}_0$ will refer to the starting point of the trajectory. We distinguish this from $\mathbf{x}(0)$ because it will often need to be estimated as additional parameters. When this is the case, we will use the augmented parameter vector $\boldsymbol{\vartheta} = (\boldsymbol{\theta}, \mathbf{x}_0)$.

Along with these notations, we need a second model to describe the measurement of the system, or the *observation model*. We may observed several quantities about a system and so describe

$\mathbf{y}_j$ will be used to represent the vector of measurements of length $d$ of a set of $d$ variables at time $t_j$. Sometimes it will be useful to write down a matrix of observation vectors over time. For this we will use $\mathbf{Y}_j$ in which rows denote time points and columns the dimension of the observation vector.

$P(\mathbf{y}_t|\mathbf{x}(t), \theta)$ gives a probabilistic model for the observations. Frequently, we will model $y_{it} = x_i(t) + \varepsilon_{it}$ — that is we will measure at least some components of the state vector directly and assume additive noise. However, it will often be useful to be more complex than this. In the case of the SIR models above, we might assume binomial sampling of the infected individuals, for example. Other times, the error variance will scale with the observations. There are also occasions in which the sum of two state variables is observed. This model can, of course, depend on $\boldsymbol{\theta}$, although we will mostly be concerned with parameters that govern the process model.

Finally, in order to avoid confusion, we have used D to represent solely a derivative with respect to time. In many places, we will need to differentiate a likelihood with respect to parameters or other quantities and we will refer to the vector of derivatives by $\partial_{\boldsymbol{\theta}}$ where the subscript indicates which derivatives are being taken.

# Chapter 3
# Linear Differential Equations and Systems

## 3.1 Introduction and Chapter Overview

The importance of linear systems of differential equations in real world modelling cannot be underestimated, constrained though it may be from a mathematical perspective. We shall see in the next chapter that most nonlinear equation systems involve relatively elementary nonlinear elements, such as products or powers of variables, or rate functions that have mildly nonlinear dependencies on the variables. Linear approximations to nonlinear differential equations can be quite useful as approximations to the actual nonlinear system over limited time intervals.

When an investigator has no particular differential equation in mind, but would still like to explore the dynamics of an observed system, linear equations are an excellent and powerful exploratory data analysis tool. The explorations of the Canadian weather data and Chinese handwriting in Chap. 1 are examples where we *induced* a dynamic model directly from the observed data, rather than entering the analysis with a pre-chosen system in mind.

The chapter begins with the simplest of linear differential equations, a model that involves only a single parameter. It extends to this to include a set of simple forcing function multiplied by a constant and added to the unforced equation. The result is a simple *buffer* that acts to soften the impact of sharp changes in input to the unforced system. This scenario is far from being unrealistic, since a very large number applications begin with this concept and then achieve the required modelling horsepower by making relatively elementary extensions. Two of these modifications are (1) to increase the order the derivative on the left side and (2) to allow the constant parameters to vary over time. The mathematical analysis of such systems involves only elementary algebra and is a valuable aid to understanding what these systems do.

The single variable model is then expanded to allow multiple variables, each of which may act as an additive forcing function to one or more of the other variables. In this way, surprisingly complex system features can emerge, even when all parameters are constants. Multi-variable systems are especially important in *feedback* systems



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_3

in process control designed to keep the controlled variable from wandering too far from a desired target.

The completely general linear case, in which any variable in an additive equation can be multiplied by a possibly time-varying factor that we call a *rate function*, in which the highest orders of the derivatives appearing in the equation are arbitrary, and where any number of additive forcing terms may appear in any equation, is a large topic. We can only sketch some results here, with some interesting examples to whet the reader's appetite for knowing more.

It might be helpful to have at the outset a clear idea of what is meant by a linear differential equation. A single such equation involves these elements:

- The left side of the equation involves a derivative of order $m \geq 0$, $\mathrm{D}^m x(t)$.
- The right side is a sum of *terms*, where each term is the product of:
  - either the derivative $\mathrm{D}^j x$ of a variable, where $j < m$, or an external function $u$, called a *forcing function*
  - a coefficient $\beta$ (for $x$) or $\alpha$ (for $u$) that may be either constant or varying over time, but may not be a function of $x$
  - a known constant $b$, which is most often either 1 or $-1$.

That is, a the value of a single linear differential equation of order $m$ at time $t$ is defined to be

$$\mathrm{D}^m x(t) = \sum_{j=0}^{m-1} b_j \beta_j(t|\boldsymbol{\theta}) \mathrm{D}^j x(t) + \sum_{\ell}^{L} a_\ell \alpha_\ell(t) u_\ell(t), \quad m \geq 0 \,. \tag{3.1}$$

Notice that the rate function $\beta_j$ can vary over time, are expected to be determined by values in a parameter vector $\boldsymbol{\theta}$, and it is assumed that at least some of these parameters will be need to be estimated from data. Where multiple linear equations are involved, called a *linear system*, a term can also involve any permissible derivative $\mathrm{D}^j x_i$, $j < m_i$) of any variable $x_i$. More generally, too, the left side $\mathrm{D}^m x$ may also be multiplied by a rate function $\beta_m(t|\boldsymbol{\theta})$, but we will, for simplicity of exposition, assume that $\beta_m(t|\boldsymbol{\theta}) > 0$ and that such a differential equation has been reformulated by dividing through by $\beta_m$.

The case of $m = 0$ is important and useful, and such an equation is called *algebraic*. In this case no terms involving variable on the left side can be on the right side; but other variables and their derivatives can, as well as any number of forcing terms. Algebraic equations permit the use of observed variables that are linear combinations of other variables in the system or of forcing functions.

We appreciate that the general form (3.1) can intimidate the reader, but in the next section we begin much more simply by first looking at a single differential equation with the simplest possible structure. In subsequent sections we will introduce forcing terms, higher order equations, and systems of equations.

## 3.2 The First Order Stationary Linear Buffer

We consider first the simplest differential equation that has practical applications. Aside from the wisdom of starting slowly, a thorough understanding of the implications of this equation is vital since it is the nucleus or motif that is clearly visible in many dynamic systems. We open this discussion by looking at the equation in its autonomous or unforced version, and then proceed to viewing the equation as a buffer that modifies four important types of inputs in its forced or non–autonomous role.

Here is the stripped-down autonomous version of the first order stationary linear buffer, expressed in both differential equation and differential operator format,

$$\mathrm{D}x = -\beta x \;\text{ or }\; \mathrm{L} = \mathrm{D} + \beta I \;. \tag{3.2}$$

The single parameter $\beta$ is a constant, and the minus sign precedes $\beta$ because in the majority of applications this constant will be taken to be positive.

This equation has a great deal in common with a linear regression model. We are, effectively, regressing change in $x$, expressed as $\mathrm{D}x$, on $x$ itself; if the value of $x$ is a position on some continuum, than we are regressing velocity on position. In this sense $\beta$ can be viewed as a regression coefficient.

However, for differential equations the variable appears on both sides of the equation, whereas this is not so for regression equations. Moreover, we shall see that the term "coefficient" has many roles in this monograph, so that "regression coefficient" can be confusing as a term.

On reflection, it seemed to the authors that a different term was required, and we settled on *rate function* for coefficient $\beta$ since it determines the rate of change in $x$ or the speed with which it reacts to a unit change in input. We add "function" to this system since we will often want $\beta$ to change over time.

It will take most readers no time at all to work out that the solution to (3.2) is

$$x(t) = Ce^{-\beta t} \;, \tag{3.3}$$

where $C$ is any nonzero constant. That is, this equation, like most differential equations, has an infinity of solutions, in this case corresponding to the values possible for $C$. We remind ourselves, too, that the constant $e$, said to be the third most important constant in mathematics after 0 and 1, is not essential; any positive constant not equal to one will do. For example, 10 is often a friendlier option when talking to non-mathematicians, and 2 is useful in the theories of probability and information. The solution exhibits exponential decay when $\beta > 0$, and exponential growth when $\beta < 0$; so that zero is a critical point for $\beta$, dividing a stable decay process that converges to zero from a growth process that diverges to infinity.

The corresponding operator equation $\mathrm{L}x = \beta x + \mathrm{D}x = 0$ transforms the exponential function to the zero function, and we say that the solution space is the *kernel* of L.

But what happens to an arbitrary input function when it passes through the operator L? The first order forced stationary linear equation is

$$\mathrm{D}x = -\beta x + \alpha u \text{ or } \mathrm{L}x = \alpha u \tag{3.4}$$

where $\alpha$ is a nonzero constant modulating the impact of an external explanatory variable $u$ on D$x$. The refinery data in Sect. 1.1.3 is well approximated by a solution of this equation.

This dynamic equation has, over an interval beginning at zero, the solution

$$x(t) = x(0)e^{-\beta t} + \alpha \int_0^t e^{-\beta(t-s)} u(s)\mathrm{d}s \ . \tag{3.5}$$

The integral in the second term is the *convolution* of $u$ with the *kernel function* $e^{-\beta(t-s)}$. The convolution operation is often used in mathematics to represent a smoothing or blurring operation, and the output $x$ is therefore smoother than the input $u$. We can say that $x$ is a *buffered* version of $u$ in the sense that $u$'s sharp variation will show up in $x$'s variation as being spread forward in time.

The ratio $K = \alpha/\beta$ is called the *gain* in system, being the total change in $x(t)$ resulting from a step change in input $u(t)$. Obviously the gain will be proportional to the forcing rate function $\alpha$, notice that is inversely proportional to the rate parameter $\beta$. This is because, the faster the system reacts, the shorter the time interval over which $u$ has an impact.

The top panels of Fig. 3.1 show four types of $u$'s varying over the interval [0,10]:

- a step change,
- a point input surrounded by zero,
- a sinusoidal input and
- a white noise random input.

The middle and bottom panels display the corresponding outputs $x$ for two values of the rate function $\beta$, a moderate level 0.5 and a fast level 2.0, respectively.

A step change becomes an exponentially smooth approach to the new level, and the larger $\beta$, the faster the new level is reached. But the gain in the bottom panel is only a quarter of what it is in the top panel, because the time over which $u(t)$ in (3.5) is effectively integrated in the convolution is likewise a quarter of what it is in the middle panel. A point input converts to an initial step increase followed by an exponential decay to baseline, again we see the gain is inversely proportional to the speed of the reaction. A sinusoidal input displays two effects: The amplitude is attenuated, and the output's crossing of zero, it's phase, lags to a later value in each oscillation. Both the attenuation and the lag are greater for the slower reaction in the middle panel. White noise remains random on output, but we note a slower and more wave-like variation around zero, a consequence of being positively autocorrelated. The slower the reaction, the smoother the output is.

Parameter $\beta$ is the reciprocal of time, and $\tau = 1/\beta$ is the time required to achieve about 2/3 of the complete response, and $4\tau$ time units suffices for 95% of response

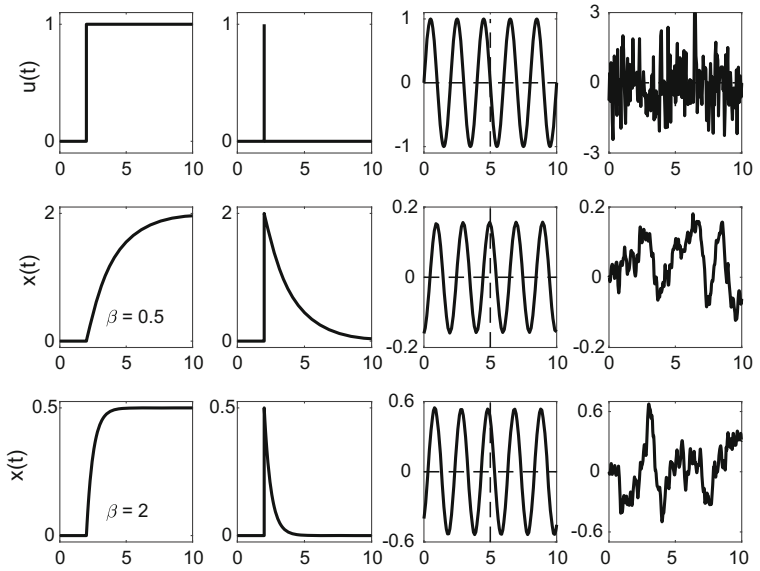


**Fig. 3.1** The *top panels* display four types of forcing functions $u(t)$, the *middle panels* indicate the consequences of using these as inputs to a first order stationary linear differential equation $\mathrm{D}x(t) = \beta x(t)$ with the moderate value $\beta = 0.5$, and the *bottom panels* display the output with a higher rate function value $\beta = 2.0$

change. These relations allow us to get a quick visual estimate of $\beta$. We note that a parameterization of (3.4) in terms of $\tau$ rather than $\beta$ might seem rather more natural, and in fact the equation $\tau \mathrm{D}x = x + \alpha u$ does often appear in texts.

Almost any system that we can imagine has a buffer to protect it against sudden shocks. It does this by spreading the response over a longer time interval than was required for the shock, and in so doing conserves its most precious resource: *energy*. The energy required to respond proportionately and on the same time scale to an arbitrarily large and arbitrarily instantaneous shock would be much larger than most systems are designed to have available, and would in the course of a system's normal mode of operation mostly sit in idle reserve. Consequently, response patterns characterizable by first order buffers are found everywhere in nature and the works of man.

Much can be learned about the dynamics of a system by plotting one derivative against another, a technique that is put to work in functional data analysis (Ramsay and Silverman 2005) where it is referred to as *phase-plane* plotting. First and higher order derivatives can often be estimated directly from noisy data by using data smoothing techniques such as spline smoothing, provided that the noise level is not too high and the number of observations or the resolution of the data is sufficiently high. If we plot the the values of the first derivative of a first order unforced system

against the corresponding values of the function itself, it will be clear from (3.2) that the result will be a straight line passing through the origin. Over regions where forcing by either a step or point function has taken place, this plot will be piecewise linear.

## 3.3 The Second Order Stationary Linear Equation

If systems need to be protected against shocks, don't they also need to be shielded from sudden changes in velocity, implying infinite acceleration? The concept of *energy*, defined as the capacity to do work, which in turn is defined as force acting through distance, plays a key role here. Newton's second law implies that an instantaneous change in velocity would require infinite force and therefore infinite energy to achieve. Figure 3.1 indicates that a first order system is unrealistic if that is so, since the slope goes from 0 to a positive value at the moment that the input changes level.

The second order equation

$$\mathrm{D}^2x = -\beta_0 x - \beta_1 \mathrm{D}x \quad \mathrm{L}x = \mathrm{D}^2x + \beta_0 x + \beta_1 \mathrm{D}x = 0 \tag{3.6}$$

might be what we need since its second derivative will have a smooth relationship to $x$ and $\mathrm{D}x$.

The solution to a special case of (3.6) is easy to obtain. When $-\beta_1 = 0$ and $\beta_0$ is positive, this is the *harmonic* equation, repeated here as

$$\mathrm{D}^2x = -(2\pi\omega)^2 x \quad \mathrm{L} = \mathrm{D}^2 + (2\pi\omega)^2 I \tag{3.7}$$

where $\beta_0 = (2\pi\omega)^2$. Because the second derivatives of $\sin(2\pi\omega t)$ and $\cos(2\pi\omega t)$ are $-(2\pi\omega)^2 \sin(2\pi\omega t)$ and $-(2\pi\omega)^2 \cos(2\pi\omega t)$, respectively, linear combinations of these two functions satisfy the differential equation; and $\omega$ is the frequency of the sinusoid per unit time and $1/\omega$ is its period or wavelength. The solution space is the set of functions of the form $x(t) = C_1 \sin(2\pi\omega t) + C_2 \cos(2\pi\omega t)$, and consequently is two-dimensional corresponding to the possible constants $C_1$ and $C_2$. If, on the other hand, the sign in front $\beta_0$ in (3.6) is positive, the two solution functions are linear combinations of the hyperbolic counterparts sinh and cosh, respectively.

More generally, the solutions of the second order stationary linear equation (3.6) take the form

$$x(t) = C_1 e^{z_1 t} + C_2 e^{z_2 t} \tag{3.8}$$

where $z_1$ and $z_2$ are roots of the *characteristic equation*

$$z^2 + \beta_1 z + \beta_0 = 0 \text{ or } z = \frac{-\beta_1 \pm \sqrt{\beta_1^2 - 4\beta_0}}{2},$$

and $C_1$ and $C_2$ are arbitrary constants. If the roots of the equation are complex, they will appear in conjugate pairs in the form $a \pm ib$. In particular, for the harmonic equation where $\beta_1 = 0$, the roots are $\pm i\sqrt{\beta_0}$, and the exponentials of these values define the functions sin and cos.

You will be correct if, at this point, you induce two rules: (1) solutions to $m$th order linear stationary differential equations are linear combinations of $m$ functions, and (2) these $m$ functions are all exponential functions, but with different rate constants. Below you will also find out that both rules apply even when the rate functions are not constants, but rather rate functions that vary over time. But then we will need a slightly more general notion of "exponential". What is important here is that the higher order linear systems behave like a cascade of first order buffers acting on the input signal and combining their effects, but where each first order buffer has its own rate or time constant.

An especially important case for applications arises when $\beta_0 = (2\pi\omega)^2 > 0$ and $\beta_1 \neq 0$. The solution family in this case is

$$x(t) = e^{\beta_1 t/2}[C_1 \sin(\sqrt{\beta_0 - \beta_1^2/2}t) + C_2 \cos(\sqrt{\beta_0 - \beta_1^2/2}t)] \,. \tag{3.9}$$

In this case of $\beta_0 > 0$, the second order linear station equation is that for *damped harmonic motion*, and we can see that $\beta_1 > 0$ will imply that the sinusoidal variation will diminish in amplitude. But for $\beta_1 < 0$, on the other hand, the amplitude will increase exponentially, while for $\beta_1 = 0$ we return to undamped harmonic motion.

Figure 3.2 shows damped harmonic motion forced by three of the four inputs in Fig. 3.1. The rate function $\beta_0 = 5$ in both figures. In the panels in column two of the figure, the damping coefficient $\beta_1$ is relatively small, and we see that the sinusoidal variation that would see for $\beta_1 = 0$ is very much in evidence, but declining to near zero amplitude over one time unit. One is reminded of how a first-timer behind the wheel drives a car, and on the whole we'd prefer to eliminate the oscillations. The third column of figures has a $\beta_1$ value that achieves exactly that, and, moreover, there are two improvements over the equivalent first order buffer: the new level is achieved much more quickly, and and the transition to the steep slope is smooth at the step-up time. Hence, a more efficient machine with less wear and tear on the mechanism (and on the nerves of the driving instructor). The final column shows the disastrous consequences of negative damping.

Plotting the second derivative against the first derivative can be especially revealing for second order systems because of the nature of *energy*. A mechanical system in free motion, such as the earth in its rotation around the sun, conserves energy, and as such, exhibits either a constant velocity linear trajectory, or, if tethered by gravity or some other bond to another body that is viewed as fixed, an elliptical orbit. The energy of a body of mass $m$ exists in two exchangeable forms:

Kinetic: $E = m(\mathrm{D}x)^2/2$ when in motion, or
Potential: $E = m\mathrm{D}^2 x$ when momentarily at rest, as in a pendulum at the end of its swing or a bungee jumper at the point of being launched.

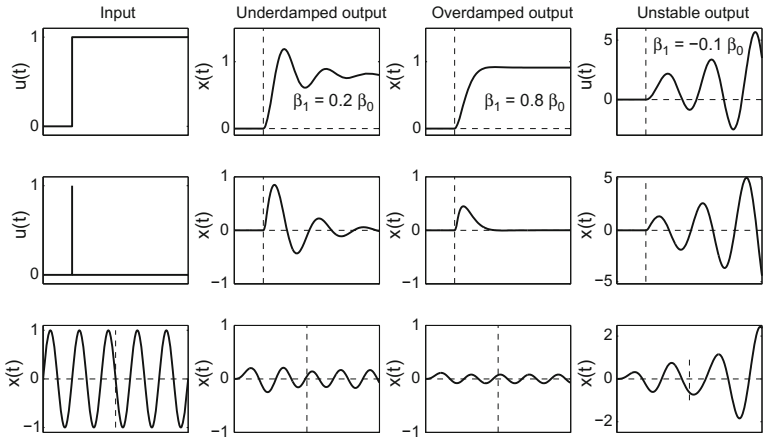


**Fig. 3.2** The *first panels* in each row display three types of functional input patterns, and the *remaining panels* in each row indicate the consequences of using these as inputs to the second order stationary differential equation $\mathrm{D}^2x(t) = -\beta_0 x(t) - \beta_1 \mathrm{D}x(t)$, $\beta_0 > 0$. Rate function $\beta_0 = 5$ for each panel

By plotting acceleration $\mathrm{D}^2x$ against velocity $\mathrm{D}x$, we see the interplay between potential energy and the signed square root of potential energy. This plot will be elliptical, as we can see from the solution to the harmonic equation (3.7), and the area contained with the ellipse will reflect the total energy in the system.

*Phase-plane plots* like this also work well when the process is only partially harmonic. The first panel of Fig. 3.3 shows the growth of the lower leg of a newborn baby over the its first 40 days, and we see that growth is a step-wise process. The daily data were smoothed by a strictly monotonic function, and the smooth is shown as a solid curve. In the second panel we see that the phase plane plot shows a series of loops corresponding to each of the growth steps. The smallest loop is the first, and each successive loop is larger as the growth process gains energy. The loops are all associated with centers with positive velocity since there is a positive nearly linear trend in the data. Ramsay and Silverman (2005) report that this harmonic component in growth is still visible at the age of ten, but with a much reduced energy level.

We obtained the harmonic equation by dropping $\beta_1$ from the general equation, but what happens when we drop $\beta_0$? We can see when we view the equation as a first order equation in *velocity*. The velocity decays or grow exponentially, but is alway either positive or negative. When we allow $\beta_1(t)$ to be a function of time, we have a recipe any strictly monotonic twice-differentiable function, and (Ramsay and Silverman, 2005) use this fact in many ways to describe and analyze *phase variation* in functional data, where time always runs forward, but with a speed that can vary over from one time to another, but with $\beta_1(t) = 0$ representing constant rate physical time. This monotonicity differential equation was used to fit the data in Fig. 3.3.

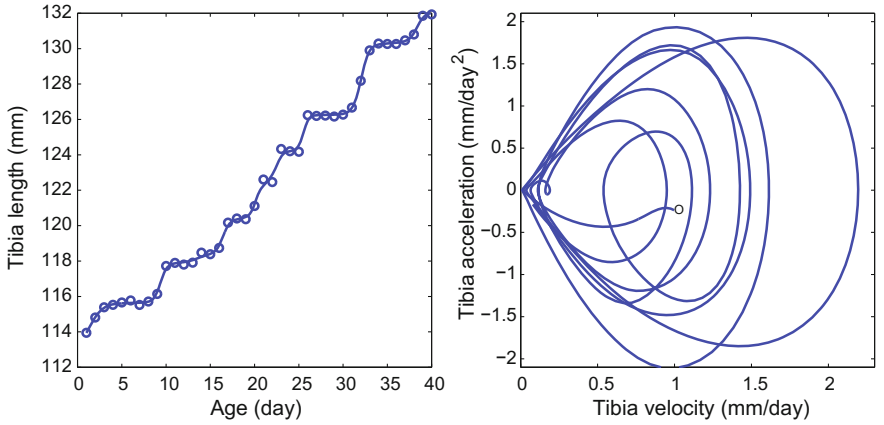


**Fig. 3.3** The *first panel* plots the daily lengths of the lower leg of a newborn baby over the first 40 days of its life, along with a smooth monotonic smooth of these data. The *second panel* plots the acceleration of smoothed leg length against its velocity. The initial point is marked by an "O"

## 3.4 The $m$th Order Stationary Linear Buffer

Now we consider the equation

$$\beta_m \mathrm{D}^m x = \sum_{j=0}^{m-1} b_j \beta_j \mathrm{D}^j x \ , \tag{3.10}$$

where $\beta_m \neq 0$ and the known constants $b_j$ are often assumed to be $-1$. Notice that the operator L version of the equation is a polynomial of degree $m$ in the symbol D. That is, by factoring this polynomial, it can be represented as

$$\mathrm{L} = \beta_m (z_{m-1} I + \mathrm{D}) \ldots (z_0 I + \mathrm{D})$$

where $z_j$, $j = 0, \ldots, m$ are roots of the polynomial. That is, an $m$th order linear stationary system is actually a cascade of first order buffers working in series. Because the roots are constants, the order of the factors in the series does not matter. It can be seen, then, that if a solution of the equation is of the form

$$x(t) = \sum_{j=0}^{m-1} C_{j+1} e^{z_j t} \ ,$$

then the first order operator $z_j I + \mathrm{D}$ will knock out the term involving $e^{z_j t}$, but on the other hand only change the multiplier for each of the other terms. Once this cascade of first order buffers has passed through the solution, all terms will be reduced to 0.

In the harmonic equation case, $\mathrm{L} = (2\pi\omega)^2 I + \mathrm{D}$ implies that the roots are $\pm 2i\pi\omega$, and these in turn imply the equivalent basis functions $(\sin(2\pi t), \cos(2\pi t))$.

To summarize, stationarity and linearity imply the possibility of solving the differential equation explicitly, even when a forcing term is added to the basic autonomous equation. We will bypass in this book the invaluable machinery of *transfer functions*, rational polynomials that result from the application of the *Laplace transform* to the differential equation, but readable accounts of this topic can be found in many sources, including Chap. 5 of Tennenbaum and Pollard (1963) and Marlin (2000).

## 3.5 Systems of Linear Stationary Equations

More generally, a multi-equation first order stationary system with $d$ variables will allow for each variable to force or input each other variable, so that

$$\mathrm{D}\mathbf{x} = \mathbf{B}\mathbf{x} \tag{3.11}$$

where the square matrix $\mathbf{B}$ will have arbitrary real-valued entries.

The solution to the Eq. (3.11) can be expressed as a simple extension of that for a single first order equation using the *matrix exponential function*

$$\mathbf{x}(t) = \mathbf{C}e^{\mathbf{B}t} \tag{3.12}$$

where $\mathbf{C}$ is a vector of initial value defining the state of the system at time 0. If a forcing term $\mathbf{u}$ is added to this equation to make $\mathrm{D}\mathbf{x} = \mathbf{B}\mathbf{x} + \mathbf{u}$, then a multivariate counterpart of (3.5) holds,

$$\mathbf{x}(t) = \mathbf{C}\exp(\mathbf{B}t) + \int_0^t \exp(\mathbf{B}(t-s)\mathbf{u}(s)\mathrm{d}s \ . \tag{3.13}$$

This result is nice on a mathematical level, but not necessarily any more useful than the differential equation itself for computing purposes. Rather than using the matrix exponential directly, we can obtain a system of $d$ basis functions $\phi_k, k = 1, \ldots, d$ spanning the space of solutions $\mathrm{L}x = \mathrm{D}x + \mathbf{B}x = 0$ directly from the eigenvalue/eigenvector decomposition of $\mathbf{B}$, provided that the eigenvalues $\lambda_k$ are distinct. That is,

$$\mathbf{B} = \sum_1^d \lambda_k \boldsymbol{\xi}_k \ ,$$

where $\lambda_k$ and $\boldsymbol{\xi}_k$ are the $k$th eigenvalue and eigenvector of $\mathbf{B}$, respectively. Note that $\mathbf{B}$ does not have to satisfy the conditions familiar to statisticians of being symmetric and positive definite, and as a consequence, some eigenvalues may be complex. But

if so, they come in complex conjugate pairs, case situation that we have already seen in Sect. 3.3.

The basis functions that we need are

$$\phi_k(t) = \exp(\lambda_k t)\boldsymbol{\xi}_k \ .$$

This is because

$$\mathrm{D}\exp(\lambda_k t)\boldsymbol{\xi}_k = \lambda_k \exp(\lambda_k t)\boldsymbol{\xi}_k = \mathbf{B}\exp(\lambda_k t)\boldsymbol{\xi}_t$$

so that $\exp(\lambda_k t)\boldsymbol{\xi}_k$ satisfies (3.12) and is therefore a function within the kernel of $\mathrm{L} = \mathrm{D} - \mathbf{BI}$.

In the case of complex conjugate pairs, the corresponding basis functions can be converted to sine and cosine functions with a period defined by the complex eigenvalue; and these, therefore, represent periodic components of variation in the solutions.

If the eigenvalues $\lambda_k$ are not distinct, which can certainly happen in practice, there exists an alternative decomposition of $\mathbf{B}$ called its *Jordan canonical form* that also yields a set of $d$ linearly independent basis functions. See references such as Coddington and Carlson (1997) for more details.

We can now re-interpret (3.11) with an arbitrary real coefficient matrix $\mathbf{B}$ as producing a linearly independent *basis* of functions $\boldsymbol{\Phi}(t)$ (taking complex and repeated eigenvalues into account as needed) and re-write the equation as

$$\mathbf{x}(t) = \boldsymbol{\Phi}(t)\mathbf{c} \ .$$

Here $\boldsymbol{\Phi}(t)$ is a $d$ by $d$ matrix and it can be shown to be invertible at any time $t$. This will be useful later on when examining forced systems.

## 3.6 A Linear System Example: Feedback Control

At this point we might need an example of a multi-variable linear differential equation that is often found in the literature, and which illustrates some of the ways in which variables in linear systems communicate with one another. We will return to this example in Chap. 9.

Feedback loops are an essential aspect of *process control*, the science and engineering of the automatic control of the outputs from industrial manufacturing processes. We illustrate the concept by this simple example. Suppose that an engineer in the diesel electric locomotive of a passenger train wants to change the speed. Advancing the throttle will do the job, of course, but the change is speed is hardly instantaneous. We capture this by the first order forced equation

$$\mathrm{D}S = -\beta_S S + \alpha C \tag{3.14}$$

where $S$ is speed measured in, say, miles per hour and time in minutes. Variable $C$ stands for whatever action is taken to determine the speed, which in this case is throttle movement. Each term must be a ratio of mph to time, so that reaction speed $\beta_S$ is in $\text{min}^{-1}$ and rate function $\alpha$ is a ratio with mph in the numerator and the product of min and throttle units in the denominator.

The engineer acts as a feedback system by using the speedometer to change the throttle position. Let the target speed, called a *set point*, be denoted by $S_0$. We can model throttle changes with a second linear equation

$$\text{D}C = \gamma(S_0 - S) \tag{3.15}$$

where $\gamma$ converts mph to throttle units. The equation says that the throttle position increases in proportion to the extent that the current speed falls short of the desired speed. Technically the equation is forced by two inputs, each with units of speed, but with rate functions equal in magnitude and opposite in sign. However, the critical difference is that $-S$ is an input from the other linear equation in the system, while input $S_0$ is external to the system; so that we prefer to say that the two-variable system $(S, C)$ is forced by the single input $S_0$. Note, too, that we have here a system in which one $\beta$ multiplies two quantities. This warns us that linear equation systems can have rate functions shared among two or more terms.

Of course the control process also has its own reaction time, but here we consider this to be so much shorter than the reaction time of the train that we can ignore it. When the set point speed is reached, $C$ no longer increases and the train settles into the desired speed at a rate determined by $\beta_S$.

The set point $S_0$ can also change over time, and Fig. 3.4 displays a set point in the upper panel by a dashed line that that indicates a target speed of 60 mph for the first 5 min, a decrease in speed to 40 mph for the next 5 min, perhaps because of passing through a village, an increase to 80 mph and a final return to 60 mph. The train's actual speed, beginning at 0 mph, is shown in the upper panel as a solid line, given that the reaction rate is $\beta_S = 2\,\text{min}^{-1}$ and $\alpha = 4\,\text{mph}/(\text{min} \times \text{C-units})$ The bottom panel displays the size of the forcing function $C$, defined in part by the rate function $\gamma$. We see a nice smooth change to target speed that takes about 3 min to realize.

We have three parameters in this model, and we want to know what will happen if these are changed. Parameter $\gamma$ in effect transforms speed units into controller units, so that changing its value is effectively the same as changing the size of the set point function $S_0$. The speed reaction rate $\beta$ defines the reaction time $\tau = 4/\beta$ that the train takes to respond to a step change in input $C$. For example, if coaches are added to the train, we would expect $\beta_S$ to drop.

This leaves the influence of parameter $\alpha$ to consider, and this turns out to be interesting. Figure 3.5 shows what happens if we either triple (solid line) or take one-third (dotted line) this parameter. Level twelve results in a faster reaction to set point change, but with an overshoot in speed. This might not get a car driver a ticket,

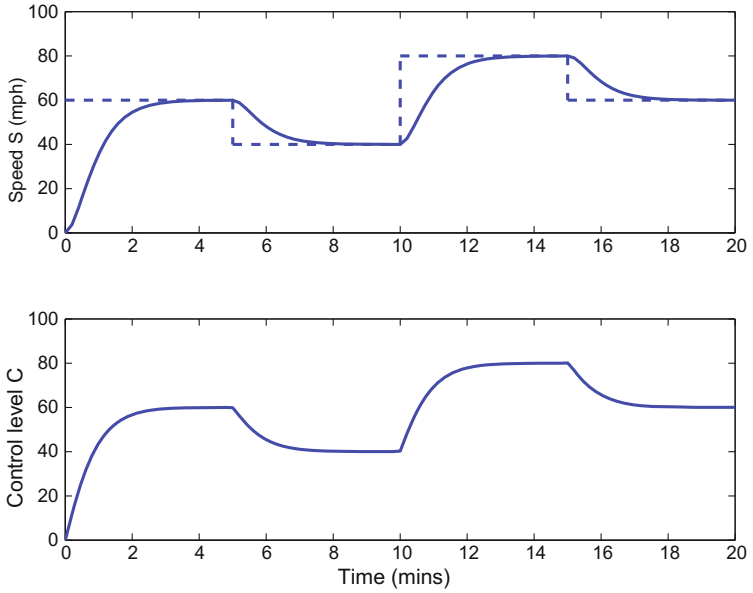


**Fig. 3.4** The *dashed line* in the *upper panel* indicates the desired speed of the train, and the *solid line* shows the actual speed determined by the feedback Eqs. (3.14) and (3.15). The throttle control position is shown in the *bottom panel*

but would be a serious matter in the tightly regulated environment of train operation. Level 4/3, however, decreases the reaction rate to a level likely to leave the train behind schedule.

How does this work? When $\alpha$ is increased, two things happen. The change in speed D$S$ increases so that less time (about 30 s) is taken for $S$ to reach $S_0$. This is also reduced time for $C$ to increase, so that it cannot in net terms increase by as much as it would if the baseline $\alpha = 4$ were used. But increased acceleration of the train, given its large mass and the $\beta_S$ that this implies, also means that once at the set speed, it cannot decelerate fast enough to avoid over-shooting the target. The control contribution switches sign and brings the speed back under control to reach a stable set speed in about 1 min. The same thing happens, but in reverse, at the speed reduction phases.

We can also understand the impact of feedback by combining these two equations. Solving the controller equation (3.15) with constant of integration 0 gives

$$C(t) = \gamma \int_0^t [S_0(u) - S(u)]\, du$$

and, when this is inserted into (3.14) and both sides are then differentiated, we obtain the single forced second order linear equation

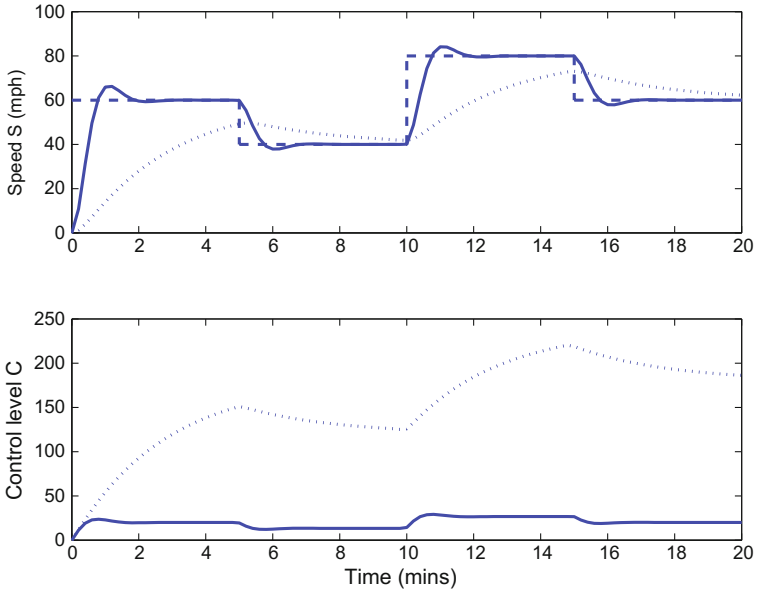


**Fig. 3.5** The *dashed line* in the upper panel indicates the desired speed of the train. The *solid line* shows the actual speed when the rate parameter in Fig. 3.4 is tripled, and the *dotted line* shows what happens when it is reduced by 1/3. The corresponding throttle control positions are shown in the bottom panel

$$\mathrm{D}^2 S = -\alpha\gamma S - \beta_S \mathrm{D}S + \alpha\gamma S_0.$$

We now see that the parameter $\gamma$ is superfluous since it always multiplies another parameter, $\alpha$, and so may be set to whatever we please, including 1. Consequently, the forced second order equation has two instead of three parameters since the $S$-rate function plus the forcing rate function multiplying the set point are constrained to add to zero.

Referring back to Fig. 3.2, we see that $\alpha = 4$ results in the response being critically damped, but that $\alpha = 12$ results in an underdamped system. More generally in process control, a first order system with feedback is effectively a single second order system, and as such can exhibit all the features of second order responses, including instability if the system is forced by a periodic input with its resonant frequency. In particular increasing $\alpha$ increases the rate function multiplying $S$ and therefore increases the natural resonant frequency of the system.

Process control (Marlin 2000) is a fascinating landscape of applications of first order stationary linear systems. The engineers who are experts in this area design control systems that keep chemical reactors, automobiles, aircraft, and a wide variety of devices on their output targets, even when unexpected external forces temporarily take them off course. The type of control loop described above is an example of *proportional* control, but *integrative* control systems can be preferable in terms

of stability and accuracy at the price of taking longer to return to target, and for really fast, but potentially unstable, reaction *derivative* control systems may be better. Collectively, this is the *PID* control family, and these three control strategies are often blended together for optimal performance. For applications of control theory to human behavior, see Jag (2003).

## 3.7 Nonstationary Linear Equations and Systems

It must be admitted that the statistical community has rather abused the concept of stationarity, this being the assumption that the equation itself or the model that it represents applies unchanged everywhere in time. The ARIMA difference equations and the conventional Kalman filter in classical time series analysis assume stationarity, but introductory texts are full of examples where even a casual graphical display will reveal that the dynamics of system itself also evolves over time or space. The defence for assuming stationarity is, of course, that if the evolution is mild, then much can still be learned from a model that is too simple. Moreover, a variety of nonlinear approaches to discrete time series have evolved in the last decades (Tong 1990; Fan and Yao 2005).

There comes a point, however, where either non-stationarity or nonlinearity, or both, are so obvious in the data and so well supported by solid theory that the analyst must face up to the additional problems that come with abandoning these twin pillars of much of dynamic systems theory. Of the two, non-stationarity is much the easier to deal with.

In this section we first have a look at a single first order linear equation where $\beta(t)$ varies over time. Here we see the fundamental structure that characterizes not only higher order equations or systems of first order equations, but a considerable portion of nonlinear dynamics as well. After a brief look at a single second order equation, we review the important theorems about first order linear systems of equations. In principle, this theory also describes higher order single equations and systems of first order linear equations as well, but higher order equations deserve a closer look, and will conclude this section.

### *3.7.1 The First Order Nonstationary Linear Buffer*

The family of solutions to the autonomous first order equation

$$\mathrm{D}x = -\beta x \quad \text{or} \quad \mathrm{L} = \beta I + \mathrm{D} \tag{3.16}$$

where rate function $\beta$ varies over time is

$$x(t) = x(t_0) \exp\left[ -\int_{t_0}^{t} \beta(v)\, dv \right] \tag{3.17}$$

where $t_0$ is the left boundary of the interval of integration, including $t_0 = -\infty$ if needed. We see that the value of $x$ is everywhere on one side of 0, as was the case for the stationary version. Indeed, (3.16) can be viewed as the universal equation for single-signed nonzero differentiable functions, and consequently we have that $\beta(t) = -D \log[x(t)/x(t_0)]$.

Notice that exponential decay or growth is intimately tied to the concept of linearity in the differential equation world. The only thing that changes in going from (3.2) to (3.16) is the constancy of the rate of change. Moreover, solutions to linear equations, as illustrated here, are one order more smooth than the rate function $\beta$ in the sense that they are guaranteed to have a first derivative, but the existence of higher order derivatives depends on the differentiability of $\beta$. Finally, we see that first order equation solutions lie within a one-dimensional family indexed by an arbitrary constant $C$. All of these features are a consequence of linearity, as we shall see below.

Figure 3.6 plots in the top panel the weekly number of infections by cholera in Bombay in the 1906 epidemic. We see an initial exponential growth phase up to about 12 weeks, followed by a tapering off of growth and then an exponential decay phase. The smooth curve is the result of fitting (3.16) to the data by estimating time-varying function $\beta$ using seven B-spline basis functions. The bottom panel shows $\beta$. We see that, apart the first week or so, the values of $\beta$ are negative up to about week 17. The negative sign on $\beta$ cancels the negative sign in the definition of the differential equation, and therefore implies exponential growth, that eventually becomes linear at the point where $\beta$ crosses zero. Over its positive phase, however, exponential decline takes place until about week 28. The fit to the data, based on seven B-spline coefficients, seems fine. We will return to these data in the next chapter. The fitting process is described in Chap. 9.

For a statistically-oriented example, we need go no further for an illustration than the univariate density function $f(t)$ as defined over the interval over which it is positive. Constant $C$ is that which sets its definite integral to one. Equation (3.16), where $\beta(t) = P(t)/Q(t)$ and $P$ and $Q$ are low-order polynomials, was the inspiration of the Pearson family of density functions (Pearson 1895), which contains many of the familiar textbook univariate probability densities.

The general first order non-stationary autonomous linear equation of order $m$

$$\mathrm{D}^m x = \sum_{j=0}^{m-1} b_j \beta_j \mathrm{D}^j x \;\text{ or }\; \mathrm{L} = \mathrm{D}^m x - \sum_{j=0}^{m-1} b_j \beta_j \mathrm{D}^j x \tag{3.18}$$

is the subject of a large literature in mathematics. This literature is heavily dependent on result for matrices that will be familiar to most statisticians. A readable review is Coddington and Carlson (1997).

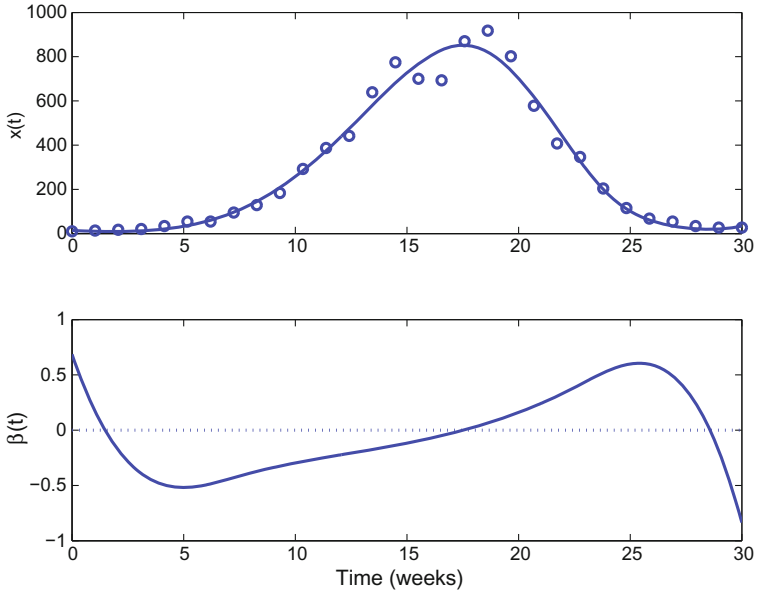


**Fig. 3.6** The *circles* in the *top panel* show the number of new infections by cholera during an outbreak in Bombay in 1906. The *solid curve* is a fit to the data by a solution to the first order nonstationary linear differential equation (3.16) defined by representing $\beta$ in terms of seven B-spline basis functions. The estimated $\beta$ is displayed in the *bottom panel*

### *3.7.2 First Order Nonstationary Linear Systems*

As with the single first order equation, we first consider the unforced case for a system of $d$ first order nonstationary linear equations

$$\mathrm{D}\mathbf{x}(t) = -\mathbf{B}(t)\mathbf{x}(t) \text{ or } \mathrm{L} = \mathbf{B}I + \mathrm{D} = 0\,. \tag{3.19}$$

We do this because the solution to a forced equation has a relatively simple relationship to the solution for the corresponding unforced equation, as we saw above for a single equation. As above, all of the rate functions in $\mathbf{B}$ are assumed to be continuous over the interval of $t$-values of interest, and $t$ is an element of a compact set.

The first result concerns the properties of the solution space. Just as in the stationary case, (3.11), the set of solutions of (3.19) is a linear function subspace of dimension $d$; and, consequently, given a set of initial values $s(t_0)$ there exists a unique solution within that space satisfying this initial value condition. We can, in fact, write down the analogue of (3.17) for this case:

$$\mathrm{D}\mathbf{x}(t) = \mathbf{C}\exp[-\int_{t_0}^{t}\mathbf{B}(\nu)\mathrm{d}\nu]\,.$$

Unlike in stationary systems, however, in this case there is no analytic expression for the matrix exponential.

The general forced linear system that corresponds to (3.18) can be expressed in matrix terms as

$$\mathrm{D}\mathbf{x}(t) = -\mathbf{B}(t)\mathbf{x}(t) + \mathbf{A}(t)\mathbf{u}(t) \text{ or } \mathrm{L} = \mathbf{B}I + \mathrm{D} = \mathbf{A}\mathbf{u} \tag{3.20}$$

where $\mathbf{A}$ is a matrix function with as many columns as the length of vector $\mathbf{u}$. Under the conditions of compactness of the domain $T$ and continuity of $\mathbf{B}$, and for any specified time value ($\tau \in T$, $\mathbf{x}_0$) any vector $\boldsymbol{\xi}$ of length $d$, there exists a unique solution to (3.19) satisfying $\mathbf{x}(\tau) = \boldsymbol{\xi}$. (See p. 226 in Coddington and Carlson 1997). This a powerful existence and uniqueness result, and one that does not hold for nonlinear systems. Additional results are available for isolated singularities in the rate functions.

Let $\mathbf{x}_0$ be a solution of the unforced linear system equations and let $\mathbf{x}_u$ be a particular fixed solution of the forced system equations. Then, as a consequence of the fact that $\mathrm{L}\mathbf{x}_0 = 0$, the function $\mathbf{x}_u + \mathbf{x}_0$ is also a solution of the forced equations. Thus, the space of all solutions of a given forced linear equation is an *affine* function space of dimension $d$ rather than a linear function space. That is, it is a space closed under both translations by particular solutions and by nonsingular linear transformations. Moreover, if $\boldsymbol{\Phi}$ is a basis for the unforced system, we can write an explicit equation for the solution of the forced linear system satisfying $\mathbf{x}(t_0) = \mathbf{x}_0$:

$$\mathbf{x}(t) = \boldsymbol{\Phi}(t)\mathbf{x}_0 + \boldsymbol{\Phi}(t) \int_{t_0}^{t} \boldsymbol{\Phi}^{-1}(u)\mathbf{A}(u) \,.\mathbf{u}(u)\, du. \tag{3.21}$$

We have seen that an $m$th order linear single equation can be re-expressed as a first order system of $m$ linear equations, and hence is within the family (3.16). Consequently the existence and uniqueness theorem for such equations or systems of equations can be expressed in terms of values at a fixed time point $t_0$ for the successive derivatives $\mathrm{D}^0 x, \ldots, \mathrm{D}^{m-1} x$ of $x$, since each derivative corresponds to an equation the re-expression of the equation as a first order linear system. Results analogous to those above for linear first order systems are in some cases, such as (3.21), are simplified.

Linear systems are important because they often capture the fundamentals of the dynamics, but also because mildly nonlinear systems can be approximated over suitably short intervals of $t$ by linear systems. Moreover, although solutions to forced linear systems cannot often be expressed in neat algebraic terms, they can usually be easily approximated, as we shall see below where we consider Green's functions. In summary, linearity conveys simple well-ordered behavior in dynamics just as it does in multivariate statistical models.

Often the interval of interest is the whole real line, as would be the case of characterizing a probability density function $f$ such as the Gaussian. In such cases, $\tau$ can be set at any finite value, such as at a mode of the density with the corresponding fixed value $\mathrm{D}f = 0$. The entire function can be constructed by solving the equation both forward and backward from this fixed value.

## 3.8 Linear Differential Equations Corresponding to Sets of Functions

In this section we see how a linear differential equation can be defined that annihilates a set of $m$ known and linearly independent functions, and explain why one would want such a thing. Dealing with this problem also allows us to introduce the *Wronskian* matrix function that will allow us to solve another important problem in the next section.

We often see in functional data some trends that can be well described by a small number of elementary functions. This insight sometimes comes from what we know by common sense, and it sometimes comes by preliminary explorations of functional data. Suppose, then, that we are given a set of $m$ linearly independent functions $\boldsymbol{\phi}_j$, $j = 1, \ldots, d$. Here's an example.

Psychometricians develop models for the performance of a set of examinees on a test composed of a set of multiple choice items. The central construct is a probability function $P(\theta)$ that models the probability that examinees at an ability level $\theta$ will answer a specific item correctly. You might think that such a function, if $\theta$ can be any real number, would have an ogival shape with a left asymptote of zero and a right asymptote of one. Were that the case, the very useful log-odds ratio $W(\theta) = \log P(\theta)/[1 - P(\theta)]$ would be close to a set of straight lines with varying slopes and intercepts, and a two-dimensional basis function system would work fine. But multiple choice items can be successfully answered just by guessing or some other strategy equally dumb, so that the lower asymptote of the $P$ functions is larger than zero, and this shows up in the function $W$ as curves that are nearly linear but often concave upward. Rossi et al. (2002) added a third function $\log(\exp(\theta) + 1)$ to the basis system which increases linearly on the right but becomes constant on the left.

The complementary problem of identifying a linear $m$th order linear differential operator L such that

$$\mathrm{L}\boldsymbol{\phi}_j = 0, \, j = 1, \ldots, d \tag{3.22}$$

plays an important role in spline smoothing, since if it is known a priori that a set of curves tend to strongly resemble linear combinations of these known null-space or kernel functions, a great improvement in smoothing bias and sampling variance can be achieved by using L to define the roughness penalty $\int [\mathrm{L}x(t)]^2 \, \mathrm{d}t$.

There are other iconic higher order linear equations, too. Consider as an example the linear *Hermite equation* for the Gaussian density function, here expressed in it's standardized form:

$$p(z) = Ce^{-\frac{z^2}{2}} \;\text{ or }\; \mathrm{D}^2 p(z) = (z^2 - 1)p(z) \; . \tag{3.23}$$

This suggests the idea of plotting a standardized approximation $\hat{p}$ of a density function based on a sample in terms of the transform $\mathrm{L}\hat{p}(z) = \mathrm{D}^2\hat{p}(z) - (z^2 - 1)\hat{p}(z)$ as an interesting way of noticing non-Gaussian behavior in the approximation. That is, the operator version of a linear differential equation can act as a *filter*, as we

have already seen in the previous subsection, to remove certain known shapes from candidate functions so as to see more clearly their unique features.

The *Wronskian* matrix

$$\mathbf{W}(t) = \begin{bmatrix} \phi_1(t) & \mathrm{D}\phi_1(t) & \dots & \mathrm{D}^{m-1}\phi_1(t) \\ \vdots & \vdots & \vdots & \vdots \\ \phi_m(t) & \mathrm{D}\phi_m(t) & \dots & \mathrm{D}^{m-1}\phi_m(t) \end{bmatrix}. \tag{3.24}$$

and the rate function vector $\boldsymbol{\beta} = (\beta_0, \dots, \beta_{m-1})^T$ defining the right side of the linear equation define the functional matrix equation

$$\mathbf{W}\boldsymbol{\beta} = -\mathrm{D}^m\boldsymbol{\phi} \ ,$$

so that, with a little help perhaps from symbolic computation software to solve the equation for $\boldsymbol{\beta}$, an explicit representation of the equation can be realized.

If we try this out on the harmonic basis system, $\sin(\omega t)$ and $\cos(\omega t)$, for example, get that

$$\begin{bmatrix} \sin(\omega t) & \omega\cos(\omega t) \\ \cos(\omega t) & -\omega\sin(\omega t) \end{bmatrix} \begin{bmatrix} -\beta_0 \\ -\beta_1 \end{bmatrix} = \begin{bmatrix} -\omega^2\sin(\omega t) \\ -\omega^2\cos(\omega t) \end{bmatrix} \tag{3.25}$$

and the solution $\beta_0 = \omega^2$ and $\beta_1 = 0$ pops out.

The consideration of phase variation is a great way of generating new problems. Suppose we have a harmonic oscillator, but time is running fast and slow, so that there is a strictly monotonic function $s = h(t)$ such that oscillator applies to warped time $s$. The reader can work out that the change in the harmonic equation defined by phase variation is quite benign:

$$\mathrm{L} = \mathrm{D}^2 + \omega^2(\mathrm{D}h)^2 I \ ,$$

since $\mathrm{D}h > 0$.

## 3.9 Green's Functions for Forcing Function Inputs

Our focus in this book is on input/output systems. At the beginning this chapter we explored how the simple buffer $\mathrm{D}x = -\beta x$ or the linear differential operator $\mathrm{L}x = \mathrm{D}x + \beta x$ transforms various inputs into smoother outputs, and Figs. 3.1 and 3.2 were designed to provide some insights. But we want some more general tools allow us to explore the operator equation

$$\mathrm{L}x = u$$

where L is an arbitrary order $m$ linear differential operator. That is, how can we, given input $u$, get at the output $x$?

Of course, any part of $x$ satisfying $\mathrm{L}x = 0$ is not going to be defined by $u$ alone. But as a rule we have auxiliary constraints that can determine that part of $x$. A classic example is knowledge of the value $x$ and its first $m-1$ derivatives at time 0. But we could also use observational data to do this, as we will see in Chaps. 7–10.

So let's focus on the "inversion" of $\mathrm{L}x = u$ where we only ask for a result that does *not* satisfy $\mathrm{L}x = 0$. Here our intuition suggests that the "inverse" of a differential operator will involve integration, although it is a tad mathematically naive to see integration as merely the reverse of differentiation. But intuition is often enough right to be trusted, and the answer to the inverse question is indeed defined by

$$x(t) = \sum_{1}^{m} c_k \phi_k(t) + \int_0^t G(t;s)\mathrm{L}u(s)\,\mathrm{d}s \ . \tag{3.26}$$

The first half of the equation captures the part of $x$ satisfying $\mathrm{L}x = 0$ where the $m$ basis functions $\phi_k(t)$ span this kernel space. We'll see in Chap. 9 an easy method for finding such a basis. We can also choose $m$ linearly independent vectors of initial values $[x(0), \mathrm{D}x(0), \ldots, \mathrm{D}^{m-1}x(0)]$ and use the initial value solution approximation methods described in Chap. 5 to set up $m$ functions $\phi_k$.

The second half is a linear integral operator with a bivariate function $G(t;s)$ acting as a weight function (unfortunately, often called the "kernel" of the integral operator). Function $G$ is called the *Green's function* associated with L and the set linear constraints identifying the differential equation. What these constraints are doesn't matter for our purpose, and the Green's functions that we use are associated with the initial value constraints.

Actually, we've already seen a Green's function in (3.5) for the first order stationary operator. There

$$G(t;s) = \exp[-\beta(t-s)] \ .$$

The solution to the general case of L of order $m$ and nonstationary is also relatively easy to get, and the key is the Wronskian matrix (3.24). Let $\eta_k$, $k = 1, \ldots, m$ be the functions in the last row of $W^{-1}$. Then

$$G(t;s) = \sum_{k}^{m} \phi_k(t)\eta_k(s) \ . \tag{3.27}$$

For example, using $\sin(\omega t)$ and $\cos(\omega t)$ to span the kernel of the harmonic operator, we obtain

$$G(t:s) = [\sin(\omega t)\cos(\omega s) - \cos(\omega t)\sin(\omega s)]/\omega \ .$$

We can put this Green's function to work, for example, for the Mandarin handwriting data in Chap. 1 where we can observe the consequences of *any* input to the two harmonic oscillators that we will identify in Chap. 9.

# Chapter 4
# Nonlinear Differential Equations and Systems

## 4.1 Introduction and Chapter Overview

Autonomous nonlinear first order systems or single higher order nonlinear equations are expressible as

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}|\boldsymbol{\theta}) \text{ or } \mathrm{D}^m x = f\left(x, \mathrm{D}x, \ldots, \mathrm{D}^{m-1}x|\boldsymbol{\theta}\right) , \tag{4.1}$$

respectively. A great deal depends on how the right side functions $\mathbf{f}$ or $f$ vary with respect to their arguments, which are the values or $\mathbf{x}$ or, in the case of a single $x$, a certain number of the derivatives of $x$. This is a very large modelling landscape, but fortunately the large majority of nonlinear models actually do not get very far from the linear comfort zone.

In this chapter, we see how nonlinear dynamical systems often start out as the kinds of linear equations and systems that are discussed in Chap. 3, but which are subsequently modified in fairly simple ways to tune the model to better fit the data. Seeing the linearity behind the nonlinear system can help to understand how the system works, and even suggest further modifications. We will show that many of the modifications arise from some sort of mild dependency of the rate functions $\beta$ on the values of one or more internal or external variables. We call this dependency *rate-forcing*. We look at some important case studies in this chapter, and the chapter also acts as a kind of repository for the dynamic systems that are taken up in later chapters.

## 4.2 The Soft Landing Modification

For example, the level of water $x(t)$ in a bathtub with an open drain should in principle satisfy our simple model $\mathrm{D}x = -\beta x$. But we all know that the last drops of water take forever to get down the drain, and often evaporate before that happens, suggesting



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_4

that $\beta$ declines as $x$ approaches zero. Or, when a really big tub is really full, and the drain is small, the high pressure in the drain can cause turbulence and back pressure that seems to indicate that $\beta$ also goes down.

What happens when droplets remain in the bottom of the tub is that viscosity, frictional and adhesive processes slow the movement of the fluid down to the point where these factors become more important than gravitational pull. This is a complex process, but it can be caricatured by noting that the speed parameter $\beta$ seems to decrease when the value of $x$ approaches zero, with a consequent increase in the time parameter $\tau$. Were the Montreal smallpox epidemic allowed to go unchecked, it would burn itself out in part because there would be soon so few susceptible citizens that the chances that they would encounter an infected individual would be slim. Which would be good news for the variola virus since it would be certain that a few susceptible individuals would still be around to rebuild the population, and therefore restart the plague.

This automatic lowering of $\beta$ can be modelled by replacing a constant $\beta$ by the ratio

$$\beta(x) = \frac{\beta_{max} x}{(K + x)}. \tag{4.2}$$

When $x(t)$ is large with respect to threshold parameter $K$, $x/(K + x) \approx 1$ and the decay is at the fixed exponential rate $\beta_{max}$. But as $x(t) \to 0$, the effective reaction speed also goes to zero and the whole system slows down. Constant $K$ is often called the "half-saturation" because when $x$ reaches this level, the reaction speed is $\beta_{max}/2$. In population dynamics, this leads to the predator population declining to the point that the prey, who usually have a larger birth rate, and can consequently bounce back in the interregnum when there are few prey left. This is why the two populations tend to exhibit the cycles of abundance and scarcity that we see in nature. The rate function (4.2) is often called a "hill" function because of its shape, but it goes by the more formal name of the *Michaelis/Menton* function due to its originators who worked in enzyme kinetic models.

The resulting differential equation $\mathrm{D}x = -\beta(x)x$ is no longer linear. But, we note that if we allow $\beta$ to be time-varying, and we can estimate this functional version of $\beta$ using the methods in Chap. 9; then a plot of this coefficient versus $x$ will reveal the nonlinear structure. That is, some nonlinear differential equations are not consistent with only *stationary* linear equations.

## 4.3 Existence and Uniqueness Results

While we can happily write down equations of the form (4.1), it isn't clear that there is always a solution $\mathbf{x}(t)$ that satisfies it. In the linear ODE models in Chap. 3, this could be observed by inspection, but we generally cannot analytically write down solutions to nonlinear ODE models.

We now briefly review what can said about the existence and uniqueness of what are described as *first order initial value problems*, where the the "data" consist only of the state $\mathbf{x}_\tau$ of the system (4.1) at time $\tau$ or, in the case of a single $m$th order equation consist of the values of $x$ and its first $m-1$ derivatives at time $\tau$. Like linear differential equations, most, but not all, nonlinear equations have a $d$-dimensional solution space corresponding to the $d$-dimensional space of initial values $\mathbf{x}_0$. First we define a type of smoothness or regularity of the right-side function $\mathbf{f}$ with respect to variation of $\mathbf{x}$ over an interval $|t-\tau| <= T$ and which is continuous in the usual sense with respect to both $t$ and $x$.

Let the $\mathbf{f}$ be continuous in $t$ and $\mathbf{y}$ over a $d+1$-dimensional rectangle $S$ be defined as containing $t$-values $|t-\tau| \leq T$ and $\mathbf{y}$-values $|\mathbf{y}-\mathbf{a}| \leq \mathbf{b}$ for some $\mathbf{a}$ and $\mathbf{b}$. The right-side function $\mathbf{f}$ is *Lipschitz-continuous* over $S$ if for every $t$ and every pair $(\mathbf{y}_1, \mathbf{y}_2)$ in $S$ there exists a constant $K$ such that $|\mathbf{f}(t, \mathbf{y}_1) - \mathbf{f}(t, \mathbf{y}_2)| < K|\mathbf{y}_1 - \mathbf{y}_2|$. A useful sufficient condition for Lipschitz-continuity is that $\mathbf{f}$ have a continuous partial derivative with respect of $\mathbf{y}$ and the order $d$ matrix of absolute values of partial $y$-derivatives satisfies $|\partial \mathbf{f}/\partial \mathbf{y}| < K$. This means that there is a limit on how quickly D$x$ can change when $x$ is varied.

There are a variety of existence and uniqueness results that hold given Lipschitz continuity of $\mathbf{f}$ over $S$. A *local* result is that there exists a constant $\delta > 0$ such that the initial value problem has a unique solution over $|t-\tau| < \delta$.

This says that if we start at time $\tau$, there is an $\mathbf{x}(t)$ that satisfies (4.1), at least for some time ahead. The restriction to *local* existence is because we can write down equations with $\mathbf{x}(t)$ which blow up to infinity in finite time. A good example of this is D$x = x^2$ which has solutions $x(t) = 1/(c-t)$ where the constant $c$ is determined by $x_0$.

The representation of (4.1) is natural in the study of ODEs as mathematical objects: they are uniquely determined, given a set of initial conditions; and this will also be a useful viewpoint in Chap. 5 where $\mathbf{x}_0$ gives us a place from which to start approximating $\mathbf{x}(0)$ when we can't do so analytically. We remind ourselves, however, that data often do not come as errorless observations of the variables at a specific time point, and we make some statistically-minded observations about using the initial value formulation for data fitting:

1. Given $\mathbf{x}_0$ and $\boldsymbol{\theta}$, we completely determine solutions to (4.1) and can write these as $\mathbf{x}(t, \boldsymbol{\theta}, \mathbf{x}_0)$. Thus, fitting these solutions to data can simply be thought of in terms of nonlinear least squares. This approach will be described in Chap. 7; there are several good references for this, for example Bates and Watts (1988); Seber and Wild (2003).
2. While we expect $\boldsymbol{\theta}$ to be a relevant object of statistical estimation, the initial conditions $\mathbf{x}_0$ must be either known *a priori* or must also be estimated.
3. A search over parameters can lead to values in which (4.1) does not have solutions that are defined over the entire time-interval in the data. This happens rarely, but can cause optimizers to break down.
4. Except in special circumstances, while we know $\mathbf{x}(t, \boldsymbol{\theta}, \mathbf{x}_0)$ is a unique function of time, analytic expressions for it are not available and we must approximate it

numerically. Besides additional computational cost, this can also pose problems when the numerical approximation is poor at some values of $\boldsymbol{\theta}$.

In Chap. 5 we will examine means of approximating solutions $\mathbf{x}(t)$ to (4.1) while Chap. 6 presents some means of analyzing the way these solutions behave.

## 4.4 Higher Order Equations

A further way of building differential equation models is to use more derivatives. The most prominent example of this is Newton's law of motion, stating that an object's acceleration is proportional to the force applied to it. Using $\mathbf{x}$ as the object's position, this can be stated mathematically as

$$\mathrm{D}^2\mathbf{x} = \mathbf{f}(t) \ .$$

There are then many ways in which force can be modelled. Returning to the gradient field, we could imagine a ball on a surface. The force on the ball is due to gravity, but since it cannot move through the surface, it's acceleration depends on the local gradient. If the surface is given by $G(\mathbf{x})$ we have that

$$\mathrm{D}^2\mathbf{x} = -g\nabla_{\mathbf{x}}G(\mathbf{x})$$

where the negative sign is because the ball rolls downhill and $g$ is the gravitational constant. We can further complicate this by adding friction terms, forces in the opposite direction to the ball's velocity, giving

$$\mathrm{D}^2\mathbf{x} = -\alpha\mathrm{D}\mathbf{x} - g\nabla_{\mathbf{x}}G(\mathbf{x}) \ . \tag{4.3}$$

Here we have a second-order equation expressed in terms of three derivatives $\mathrm{D}^2$, $\mathrm{D}^1$ and $\mathrm{D}^0$ of $\mathbf{x}$. As we have seen in Chaps. 3 and 4 it is always possible to reduce a higher-order ODE to a first order ODE by expanding the state vector to include derivatives. In the case of the ball with friction in (4.3) we can use $\mathbf{z} = \mathrm{D}\mathbf{x}$ as another set of state vector components and write

$$\begin{aligned}\mathrm{D}\mathbf{x} &= \mathbf{z} \\ \mathrm{D}\mathbf{z} &= -\alpha\mathbf{z} - g\nabla_{\mathbf{x}}G(\mathbf{x}) \ .\end{aligned}$$

The first of these equations essentially just defines $\mathbf{z}$ and the second replaces $\mathrm{D}\mathbf{x}$ with $\mathbf{z}$ everywhere it appears. This also helps to make clear that for this system, in addition to providing the initial position of our ball, $\mathbf{x}(0)$, we also need to specify its starting velocity $\mathbf{z}(0)$ before we can obtain a solution.

This method of substitution can be carried out for any higher-order ODE, with a new set of state variables for each order of derivative that appears. We should also

note that while this approach in principle reduces the problem to first-order systems, the practical implications are that we have doubled the size of the state vector and this can make such a re-formulated model more numerically challenging to work with. In practice it is rare for real-world systems to employ more than second order derivatives, although when they do, the terms "jerk," "pop," "snapple," and "crack" have been known for the values of orders three to six.

## 4.5 Input/Output Systems

Another important source of modeling terms comes from external inputs into a system. Here again, the simplest such expression was given by Newton:

$$\mathrm{D}v = u(t)$$

where $v$ is the velocity of an object and $u(t)$ is a force applied to it. In this case, $u$ is *external* to the system which responds to it dynamically. These can be viewed as input/output systems when the response of the system to $u$ is relevant. While much of our analysis in the next few chapters will apply to *autonomous* (i.e. self-contained) systems, input/output systems are the main focus in a number of fields, particularly in engineering where the response of chemical systems to temperature, for example, is of primary interest.

Chapter 3 introduced the response of linear differential equation systems to inputs where we observed the tendency of dynamic models to smooth out changes in $u$. More generally, non-linear systems can also be viewed as responding to stimuli. In Chap. 6 we will examine models for neural firing behavior where neurons respond to an electric current by producing "action potentials" or sequences of sharp peaks in voltage across the neural membrane. Understanding mathematically how this occurs is the motivation behind the study of *bifurcations* which we overview there.

## 4.6 Case Studies

### *4.6.1 Bounded Variation: The Catalytic Equation*

We saw in Chap. 3 that some simple linear equations define useful families of functions, so that $\mathrm{D}^m x = -\beta \mathrm{D}^{m-1} x$ defines positive, monotonic and concave functions for $m = 1, 2, 3$, respectively. In a sense, these equations are identified with transformations. That is, $\mathrm{D}x = \beta x$, equivalent to $x = C\exp(\mathrm{D}^{-1}\beta)$ where we use the notation $\mathrm{D}^{-1}x$ for the indefinite integral of $x$ plus an arbitrary constant. This suggests the logarithmic transformation of the solution as a means of linearizing the

family, and likewise the log-derivative and log-second-derivative transforms for the monotonic and concave families, respectively.

A simple nonlinear equation corresponds to functions constrained to lie between two boundaries:

$$\mathrm{D}x = \beta x(K - x), \quad \beta > 0 \tag{4.4}$$

has the solution for constant $\beta$:

$$x = \frac{Ke^{\beta}t}{1 + e^{\beta}t} \quad \text{or} \quad \log \frac{x}{K - x} = \int_0^t \beta \mathrm{d}t \ . \tag{4.5}$$

For statisticians, this is the familiar logistic function and the system is also described as the *logistic equation*.

Strictly speaking, we haven't included a starting point $x_0$ here, but it is fairly easy to see that starting at a particular point just involves replacing $t$ with $t - c$ where $c = \log[x_0/(K - x_0)]$ is chosen so that $x(0) = x_0$.

Again, we see (4.4) as a nonlinear equation where the rate parameter depends on the value of $x$ and is $\beta^* = \beta(K - x)$. When $x \to 0$ the equation becomes one of exponential growth, but as $x \to K$, the contribution of $x$ moves $\beta^*(x)$ to 0 and $x$ approaches the asymptotic value of $K$. D$x$ becomes negative if $x > K$, but we don't generally consider the system ever starting in this range, and of course it won't get there from below $K$. In Chap. 6, the values $x = K$ and $x = 0$ are called *stable points* for the system, as illustrated in Fig. 4.1.

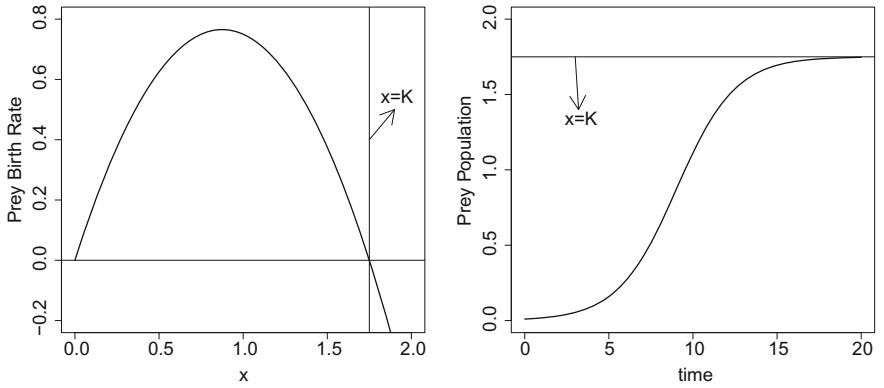


**Fig. 4.1** Catalytic growth. *Left* D$x$ plotted as a function of $x$ in (4.4). *Right* the resulting solution given as $x(t)$

More generally, if $\beta$ is time-varying, and then

$$x = \frac{Ke^{\mathrm{D}^{-1}\beta}}{1 + e^{\mathrm{D}^{-1}\beta}} \quad \text{or} \quad \log \frac{x}{K - x} = \mathrm{D}^{-1}\beta \ . \tag{4.6}$$

The model derives its name from chemical applications where a catalyst promotes a reaction up to the point where it is exhausted. This equation is often a motif in more complex equations in the same manner as the first order linear buffer is, as we shall see in Sect. 4.6.2. Moreover, the equation is also equivalent to a linear equation with time-varying $\beta$.

In Chap. 7 we will use the Rosenzweig-MacArthur ODE model (Rosenzweig and MacArthur 1963) to model the population dynamics of a predator and prey in a chemostat. These equations are

$$\begin{aligned} \mathrm{D}C &= \rho C(\kappa - C) - \frac{\gamma \beta C B}{\chi + C} \\ \mathrm{D}B &= \frac{\beta C B}{\chi + C} - \delta B \end{aligned} \tag{4.7}$$

and we note that they make use of both an upper boundary parameter $\kappa$ and of a soft landing parameter $\chi$.

### 4.6.2 Rate Forcing: The SIR Spread of Disease System

The SIR class of models were designed to model the spread of a disease in a population (Kermack and McKendrick 1927). A recent survey of infective disease modeling is Keeling (2008).

We will examine how the model's structure affects the nature of the system. We have chosen the SIR model because it contains a common type of nonlinearity in which one variable changes the dynamics by rate-forcing another variable. A population of susceptible individuals of size $S(t)$ at time $t$ is exposed to a number $I(t)$ of infected individuals. Those infected pass into a post-infected or "recovered" group of size $R(t)$ who cannot be infected again either because they have acquired immunity or have expired. Here are the equations:

$$\begin{aligned} \mathrm{D}S(t) &= -\beta I(t) S(t) \\ \mathrm{D}I(t) &= \beta I(t) S(t) - \gamma I(t) \\ \mathrm{D}R(t) &= \gamma I(t) \, . \end{aligned} \tag{4.8}$$

Models of this nature are often called *compartment models* because a total mass moves between a number of compartments, and the dynamic system describes the transport process. We see that the susceptibles succumb to the disease according to a first order linear buffer with a time-varying rate $\beta^*(t) = \beta I(t)$, which makes sense since the more infected cases, the more likely one is to be infected. Infected cases, according to the first term in the $I$ equation, increase exponentially at a time-varying rate $\beta S(t)$, capturing the obvious fact that as the number of susceptibles declines, the rate of increase will decline proportionately. This contribution to $\mathrm{D}I$ is offset by

the cure (or death) decline modulated by fixed rate parameter $\gamma$. They pass into the $R$ population as a forcing term $u(t) = I(t)$ modulated by the same rate parameter that tends to empty the infected compartment. The critical parameter is $\beta$, which controls the rate at which the infection spreads. Specifically, if $N = S(0)$, the initial size of the population before anyone is infected, than $\beta N$ is the average number of individuals infected by a single infected case per unit time. A natural unit of time for the model is $\tau = 1/\gamma$, the recovery time for 2/3 of the infected individuals.

We can see the effect of all this for a fixed $\gamma = 1$, an initial $S$ population of 500, a single infected individual, and a range of $\beta$ values in Fig. 4.2. The $\beta$ values vary from 0.002 to 0.004, or 1/2 to 2 infections per $I$ and per $\tau$. The larger $\beta$ the faster the infection spreads, and the more susceptibles are infected. For example, if our single infected person infects 2 persons per $\tau$-day, in about 6 days 80 cases are infected, and ultimately about 80% of the population is infected and then recovered. But for $\beta = 0.002$, or 1/2 an infection per $I\tau$, only about 25 cases of infection are ultimately recorded.

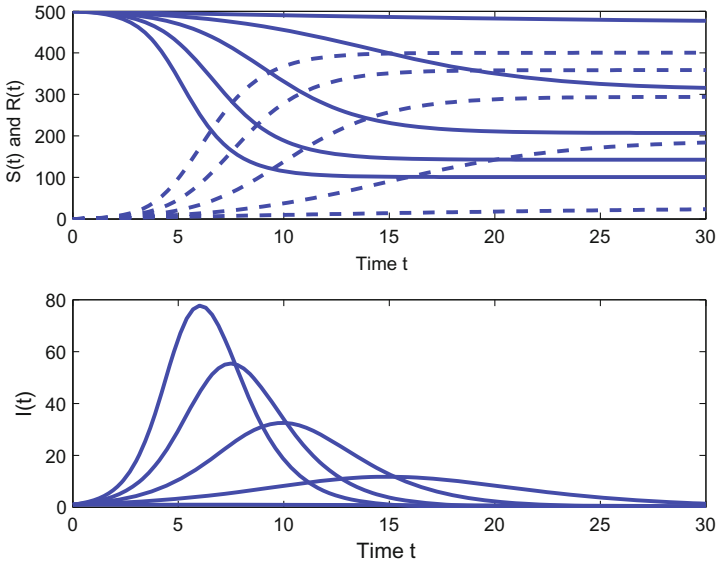


**Fig. 4.2** The time course of infection in an SIR model (4.8) where $\gamma = 1$, $S(0) = 499$, $I(0) = 1$, $R(0) = 0$ and $\beta$ takes on values 0.002, 0.0025, 0.003, 0.0035 and 0.004. The *upper panel* shows the number of susceptible individuals in a population of fixed size 500 (*solid curves*) and the number of recovered cases (*dashed curves*). The *lower panel* contains the number of infected individuals. Higher values of $\beta$ produce larger numbers of infections but a shorter epidemic. The final number of infected individuals in order of size of $\beta$ is 23, 184, 294, 358 and 400

Ultimately, of course, the disease burns itself out as the number of susceptibles declines to the point where an infected individual is unlikely to come into contact with a susceptible individual before recovering. The quantity $R_0 = \beta N/\gamma$, ranging here

from 1/2 to 2, is called the "basic reproductive rate," and plays a central role in the characterization of communicable diseases. In our experiments, when $R_0$ reaches 5, the entire population is infected. Smallpox has $R_0$ values of 3.5–6 and one of the most infectious human diseases, measles, has values of 16 or higher in an unvaccinated population.

Notice that the sum of the three variables at any time $t$ is equal to the entire population size $N$. Among the many extensions of this model used in actual disease modelling, we can introduce a background birth and death rate, both of which can have a large effect on the disease dynamics since the spread of infection depends critically on the number of susceptibles available. Also implicitly assumed is that each individual freely mixes with the entire population, as might be the case in a boarding school, for example. In fact, contact with infected cases where childhood diseases are involved greatly increases at the beginning of the school year, and this can be modelled by allowing $\beta$ itself to vary over time. The SEIR version of the model allows for an intermediate "exposed" period of infection where the infected person is infective, but not yet symptomatic.

The SIR model and its many variants are often fit to data where there are observations only on $I$. When there is no information on $S$, the $I$-equation becomes equivalent to the first order non-stationary linear equation (4.1), and consequently the model can be fit to any data on $I$ whatever, and therefore offers no insight into the process. The Bombay cholera data analyzed in Chap. 3 are a case in point. On the other hand, it happens often that the initial size $S(0)$ is roughly known before the outbreak of the disease, and even this tiny bit of extra information can lead to some reasonable estimates of the SIR model parameters and the shape of the $S$ function. If the hill function is employed instead of a constant $\beta$, an estimate of the threshold parameter $K$ requires data for $S$-values close to zero.

The SIR model is designed to fit count data where the scales of the $S$ and $R$ variables can be orders of magnitude larger than the scale of $I$. For this reason, the logarithmic version of the model, the setup of which is considered in Sect. 2.6 is often used to fit the logarithm of the data, where the base of the logarithm is often 10 rather than $e$ in order express results in a form that is more comprehensible to clients.

In summary the SIR model is constructed from three first order buffers, but derives its nonlinearity from the fact that $S$ and $I$ rate-force each other's dynamics. Formally, we may think of the SIR structure as *quadratic* rather than linear since products of variables are involved. The impact of one variable modifying another variable's dynamics can be far more dramatic than the effect of additive forcing, and the result in this case is a system that is a kind of "anti-buffer" in the sense that a small additive perturbation of a single infected individual can have an explosive impact on the system, although in this case the impact is comparatively short-lived. A more elaborate quadratic system of ten differential equations used to model a chemical reaction is taken up in Chap. 10.

### 4.6.3 *From Linear to Nonlinear: The FitzHugh-Nagumo Equations*

Dynamical systems exhibiting oscillations with fixed a period are one of the most important classes, primarily because they have a particular type of *stable* variation that we often see in nature. Of course, the harmonic oscillator discussed in Chap. 3 is one such stably periodic system. Here we look at a two-variable system that is made nonlinear in order to have asymmetric cycles.

First we modify the two-variable first order linear system version of $\mathrm{D}^2x = -x - \mathrm{D}x$ slightly by defining $\mathrm{D}x = -y$:

$$\begin{aligned} \mathrm{D}x &= -y \\ \mathrm{D}y &= -x + y \ . \end{aligned} \tag{4.9}$$

Next we generalize this system into what is known as the *Liénard* equation, used in electrical circuit theory (Hirsch and Devaney 2013):

$$\begin{aligned} \mathrm{D}x &= -y \\ \mathrm{D}y &= -f(x) + y \ . \end{aligned} \tag{4.10}$$

The *van der Pol* equation, where $f(x) = x^3 - x$, is:

$$\begin{aligned} \mathrm{D}x &= -y \\ \mathrm{D}y &= x - x^3 + y \ . \end{aligned} \tag{4.11}$$

Finally, we study a modified van der Pol system where three parameters are tossed in that don't have much influence on the shape of the periodic oscillations.

The FitzHugh-Nagumo (FitzHugh 1961) equations were intended to capture the essential dynamic properties of oscilloscope records of voltages observed in neurones when they fire. These equations were constructed to be a simplification of the Hodgkin-Huxley equations (Hodgkin and Huxley 1952). Variable $V$ represents the voltage across the membrane of a neurone, and $R$ is a sum of "recovery" ion currents.

$$\frac{d}{\mathrm{d}t}V = c(V - V^3/3 - R) \tag{4.12}$$

$$\frac{d}{\mathrm{d}t}R = (V - a + bR)/c \ . \tag{4.13}$$

Intuitively, $V$ represents the voltage across the membrane of a neurone, and $R$ is an aggregate of "recovery" ion currents. Figure 4.3 shows the solutions $V$ and $R$ corresponding to initial conditions $V(0) = -1$ and $R = 1$; and to parameter values $a = -0.8$, $b = -0.3$, and $c = 2.5$. The voltage trajectory resembles more or less a series of spike potentials in a firing nerve cell axon, and in particular has periodic asymmetric spikes separated by substantial recovery periods.

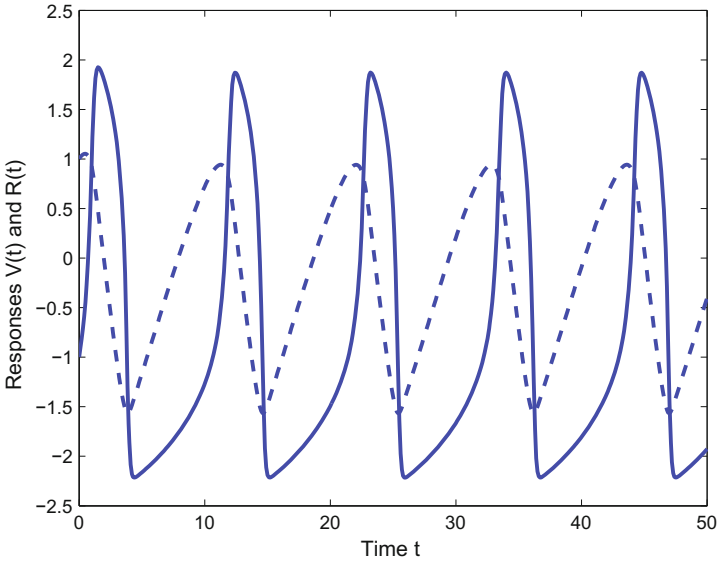


**Fig. 4.3** The *solid line* indicates the solution for voltage $V(t)$ to the FitzHugh-Nagumo system (4.12) and the *dashed line* shows the recovery variable $R(t)$. The voltage trajectory has roughly the shape of a series of spike potentials in the axon of a nerve cell

We see in the $R$ equation the first order stationary linear buffer with with speed of response $b/c$ and additive forcing by $V$ and constant $a/c$. Since $b/c$ is negative, this buffer exhibits exponential decay interrupted by the impact of a spike in $V$. There is a substantial delay before a $V$-spike produces an increase in $R$, as is characteristic of first order buffers.

On the other hand, the $V$ equation is nonlinear because of the influence of the $V^3$ term in the equation. Because $c$ is positive, the $V$ term contributes an exponential increase or decrease in $V$ depending on the current sign of $V$. On the other hand, the $V^3$ term has a negative sign, leading to an exponential decay or approach to an asymptote depending the current sign of $V$. When $|V| < 1$, the $V$ term dominates the $V^3$ term, but otherwise the influence of the positive $V$ term becomes less and less important. It is this effective flip in the dynamics that leads to the periodic behavior of $V$. The asymmetry of the spikes are determined by the cubic power and by the interaction with the recovery variable $R$. Thus, we can use what we know about first order buffers from Fig. 3.1 to puzzle out how why this system does what it does.

The FitzHugh-Nagumo system shares with many dynamic systems the data problem that, while measuring $V$ across an axon membrane is comparatively easy, the aggregated nature of $R$ renders difficult or impossible the collection of data on its values. We have to hope that we can achieve accurate estimates of the three parameters as well as on the shape of $R$ on the basis of information on $V$ alone.

A more elaborate two-variable neuronal response model that ties the recovery phase to experimental variables and more closely resembles actual spike potentials is the *Morris-Lecar* system (Morris and Lecar 1981)

$$\begin{aligned} C\mathrm{D}V &= I - g_L(V - V_L) - g_{Ca}M_{ss}(V)(V - V_{Ca}) - g_K R(V - V_K) \\ \mathrm{D}R &= \frac{R_{ss}(V) - R}{\tau_R(V)} \end{aligned} \tag{4.14}$$

where the additional functions of membrane potential $V$ are

$$\begin{aligned} M_{ss}(V) &= \frac{1}{2}\left[1 + \tanh(\frac{V - V_1}{V_2})\right] \\ R_{ss}(V) &= \frac{1}{2}\left[1 + \tanh(\frac{V - V_3}{V_4})\right] \\ \tau_R(V) &= 1/\left[\phi \cosh(\frac{V - V_3}{V_4})\right] . \end{aligned} \tag{4.15}$$

The observed or otherwise known constants are the applied current $I$, the membrane capacitance $C$, the leak conductance $g_L$, the calcium $Ca^{++}$ conductance $g_{Ca}$, the potassium $K^+$ conductance $g_K$ and the corresponding equilibrium potentials $V_L$, $V_{Ca}$ and $V_K$, and the reference frequency $\phi$. The tuning parameters that could be estimated from data are the steady state and time constant potentials $V_1, \ldots, V_4$.

Another equation that has similarities to the FitzHugh-Nagumo equation for $V$ is the *Duffing equation*:

$$\mathrm{D}^2 x = \beta_0 x - \beta_1 \mathrm{D}x - \beta_2 x^3 \tag{4.16}$$

where the parameters are usually taken to be positive. If the equation is forced by $\alpha \cos(\omega t)$, $\omega > 0$, it is called the *Duffing oscillator*. The Duffing oscillator can, depending on parameter values, exhibit surprisingly complex and varied behaviour, including the chaotic variation that is described in Chap. 6, and it has been used as a filter for EEG data, where it seems to do a good job of differentiating among emotions (Bhowmik and Konar 2010).

### 4.6.4 Nonlinear Mutual Forcing: The Tank Reactor Equations

We turn now to two case studies taken from Ramsay et al. (2007). The first of these is a nonlinear two-variable system that plays a central role in chemical engineering, and which illustrates many of the issues taken up in this chapter.

A continuously stirred tank reactor (CSTR) is a tank surrounded by a cooling jacket and containing an impeller which stirs its contents. A fluid containing a reagent with

concentration $C_{in}$ enters the tank at a flow rate $F_{in}$ and temperature $T_{in}$. A coolant in the cooling jacket has temperature $T_{co}$ and flow rate $F_{co}$. An exothermic reaction within the tank produces an output with changed concentration $C$ at temperature $T$.

The differential equations used to model a CSTR, taken from Marlin (2000) and somewhat simplified by changes like setting the volume of the tank to one, are

$$\begin{aligned} \mathrm{D}C &= -\beta(T)C + F_{in}(C_{in} - C) \\ \mathrm{D}T &= 130\beta(T)C + \frac{2aF_{co}^{b+1}}{(2F_{co} + aF_{co}^{b})}(T - T_{co}) + F_{in}(T_{in} - T) \ , \end{aligned} \tag{4.17}$$

where concentration is in moles per cubic metre, temperature is in degrees Kelvin and

$$\beta(T) = \kappa \exp\left[-10^4 \tau(1/T - 1/T_{ref})\right] \ . \tag{4.18}$$

Each variable is forced by the other, as well as by functions of known external inputs. The two composite rate functions in the homogeneous part of each equation involve a term defined by functions of known flow functions $F_{in}$ and $F_{co}$ and a term $\beta_T$. The latter is a nonlinear function of temperature, called an *Arrhenius* coefficient after the Swedish chemist who worked out the affect of temperature on chemical reactions in 1889. Temperature $T_{ref}$ is a baseline temperature inserted into the coefficient to normalize the parameters $\tau$, $\kappa$ and $a$ that must be estimated from data.

The plant engineer needs to understand the dynamics of the two output variables $C$ and $T$ as determined by the five inputs $C_{in}$, $F_{in}$, $T_{in}$, $T_{co}$ and $F_{co}$. A typical experiment designed to reveal these dynamics is illustrated in Fig. 4.4, where we see each input variable stepped up from a baseline level, stepped down, and then returned to baseline. Two baseline levels are presented for the most critical input, the coolant temperature $T_{co}$.

The values of output variables $C$ and $T$ under the two experimental regimes, given values 0.833, 0.461, 1.678 and 0.5 for parameters $\tau$, $\kappa$, $a$ and $b$, respectively, are shown in Fig. 4.5. When the reactor runs in the cool mode, where the baseline coolant temperature is 335 degrees K, the two outputs respond smoothly to the step changes in all inputs. However, an increase in baseline coolant temperature by 30 degrees K generates oscillations that come close to instability when the coolant temperature decreases. Oscillations in concentration are undesirable in an actual industrial process, and feedback loops are used to dampen them down to a tolerable level.

The output oscillations resemble rather closely the underdamped situation in Fig. 3.5 in the linear feedback model discussed in Chap. 3, and suggest that this CSTR system, in spite of being nonlinear, can be closely approximated by a linear system. Marlin (2000), which contains an excellent account of CSTR dynamics, shows that such is the case, and that linear dynamics can be sufficient in most cases. But this nonlinear system has surprises which, though seldom seen in practice, need to be considered. For example, under certain experimental regimes and parameter

combinations, the system can display the kind of periodic spikes that the FitzHugh-Nagumo model was designed to emulate.

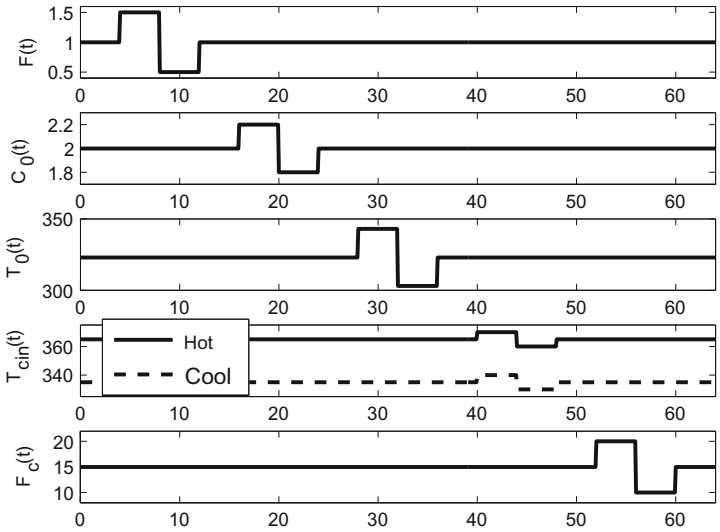


**Fig. 4.4** The five inputs to the chemical reactor modeled by the Eqs. (4.17) and (4.18): flow rate $F(t)$, input concentration $C_0(t)$, input temperature $T_0(t)$, coolant temperature $T_{co}(t)$ and coolant flow $F_0(t)$. Coolant temperature $T_{co}(t)$ was set at two baseline levels, cool and hot

The importance of the CSTR system is in part due to the difficulty and expense of measuring concentration. The mutual forcing nature of the systems means that measurements of temperature alone, which are accurate, cheap, online and can yield accurate estimates of concentration, whether in the same temperature mode or not (Ramsay et al. 2007).

### *4.6.5 Modeling Nylon Production*

If water ($W$) in the form of steam is bubbled through molten nylon ($L$) under high temperatures, nylon will split into amine ($A$) and carboxyl ($C$) groups. To produce nylon, on the other hand, $A$ and $C$ are mixed together under high temperatures, and their reaction produces $L$ and $W$, water then escaping as steam. These reverse reactions are depicted symbolically by $A + C \rightleftharpoons L + W$. The reaction dynamic equations are

$$\begin{aligned} -\mathrm{D}L = \mathrm{D}A = \mathrm{D}C &= -k_p * 10^{-3}(CA - LW/K_a) \\ \mathrm{D}W &= k_p * 10^{-3}(CA - LW/K_a) - k_m(W - W_{eq}) \end{aligned} \quad (4.19)$$

where

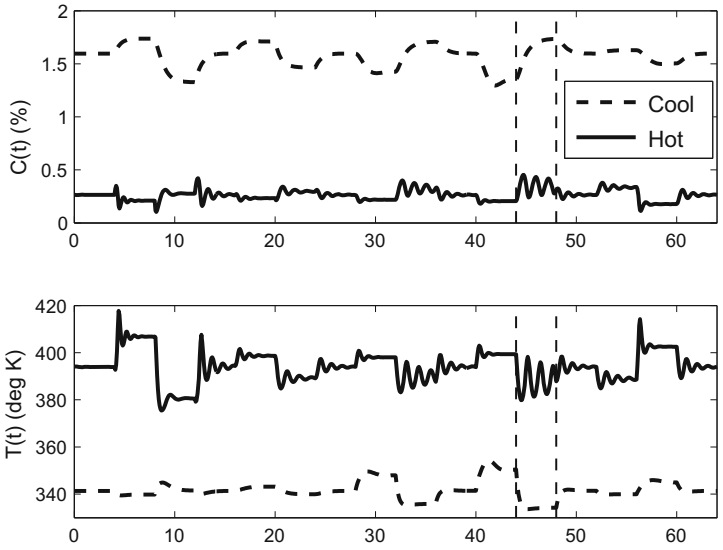


**Fig. 4.5** The two outputs, for each of baseline coolant temperatures $T_{co}$ of 335 and 365°. K, from the chemical reactor modeled by the two Eq. (4.17): concentration $C(t)$ and temperature $T(t)$. The input functions are shown in Fig. 4.4. Times at which an input variable $T_{co}(t)$ was stepped down and then up are shown as *vertical dotted lines*

$$K_a = \left[\left(1 + \frac{g}{1000} W_{eq}\right) C_T\right] K_{a0} \exp\left[-\frac{\Delta H}{R}\left(\frac{1}{T} - \frac{1}{T_0}\right)\right]$$

and $R = 8.3145 * 10^{-3}$, $C_T = 20.97 \exp[-9.624 + 3613/T]$ and a reference temperature $T_0 = 549.15$ was chosen to be in the middle of the range of experimentally manipulated temperatures. Rate parameter $k_m = 24.3$ was estimated in previous studies. Due to the reaction mass balance, if $A, C$ and $W$ are known then $L$ can be algebraically removed from the equations, so that we will only estimate those three components.

In an experiment described in Zheng et al. (2005), a mixture of steam and an inert gas was bubbled into molten nylon to maintain a constant $W$, causing $A, C, L$ and $W$ to move towards equilibrium concentrations. Within each of six experimental runs the steam pressure was stepped down from its initial level at times $\tau_{i1}, i = 1, \ldots, 6$, and then returned to its initial pressure at times $\tau_{i2}$. The temperature $T_i$ and concentration difference $A_i(t) - C_i(t)$ varied over runs but were constant within a run. Samples of the molten mixture were extracted at irregularly spaced intervals, and the $A$ and $C$ concentrations measured. The goal was to estimate the rate parameters $\boldsymbol{\theta} = [k_p, g, K_{a0}, \Delta H]$. Figure 4.6 shows the data for the runs aligned by experiment

within columns. Since concentrations of $A$ and $C$ are expected to differ only by a vertical shift, their plots within an experimental run are shifted versions of the same vertical spread. The temperature of each run is given above the plots for each set of components.

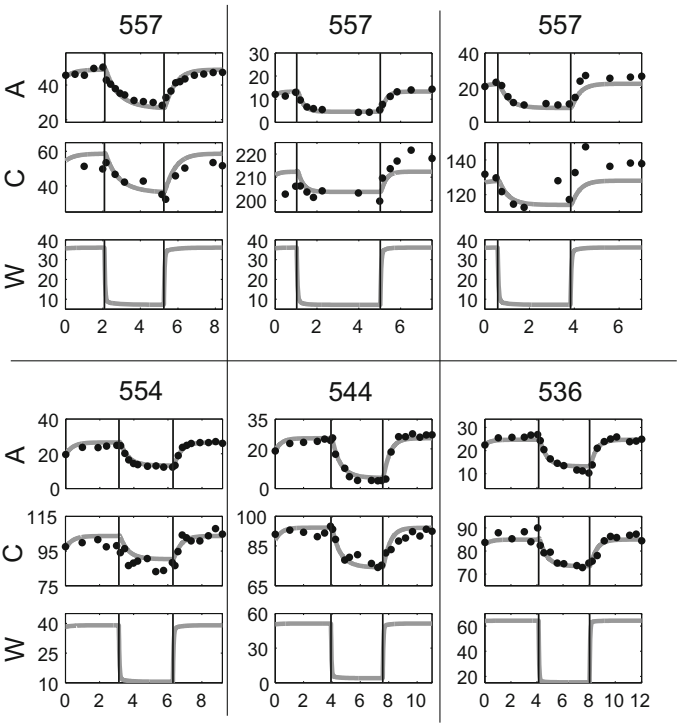


**Fig. 4.6** Nylon components $A$, $C$ and $W$ along with the solution to the differential equations using initial values estimated by the smooth for each of six experiments. The times of step change in input pressures are marked by thin *vertical lines*. *Horizontal axes* indicate time in hours, and *vertical axes* are concentrations in moles. The labels above each experiment indicate the constant temperature in degrees Kelvin

The profile estimation process was run initially with $\lambda = 10^{-4}$. Upon convergence of $\hat{\boldsymbol{\theta}}$, $\lambda$ was increased by a factor of ten and the estimation process rerun using the most recent estimates as the latest set of initial parameter guesses, increasing $\lambda$ up to $10^3$. Beginning with such a small value of $\lambda$ made the results robust to choice of initial parameter guesses. Further details concerning the data analysis are available in Campbell and Steele (2012).

The parameter estimates along with 95% limits were: $k_p = 20.59 \pm 3.26$, $g = 26.86 \pm 6.82$, $K_{a0} = 50.22 \pm 6.34$ and $\Delta H = -36.46 \pm 7.57$. The solutions to the differential equations using the final parameter estimates for $\hat{\boldsymbol{\theta}}$ and the initial system states estimated by the data smooth are shown in Fig. 4.6.

# Chapter 5
# Numerical Solutions

## 5.1 Introduction

As we have seen until now, ordinary differential equation models are a powerfully expressive class of models for describing the way systems evolve over time. However, outside a fairly narrow class of equations, we cannot obtain solutions to these ODEs analytically, and therefore directly understand the behavior that these equations produce. Methods of finding numerical solutions will be described here, while the analysis of dynamical behavior will be given in Chap. 6. Naturally, our treatment of both of these topics will be superficial and is no substitute for a text devoted to them, but we hope to provide enough background for the reader to understand numerical approaches to solving ODEs and the challenges that these pose for statistical inference.

We particularly want to note that most of the material presented in these two chapters does not discuss ODEs as input–output systems. In this chapter, we allow the right hand side function $\mathbf{f}(t, \mathbf{x}, \boldsymbol{\theta})$ to change over time–hence accounting for potential inputs–but Chap. 6 will mostly focus explicitly on *autonomous* systems in which the dynamic relations between $\mathbf{x}$ and its derivatives are fixed. The reasons for this are mathematical simplicity of two different forms: numerical methods for solving ODEs do not change a great deal when continuous inputs are added (although discontinuities must be accounted for); on the other hand, analyzing the qualitative behavior of ODEs in the presence of anything but slowly-varying inputs requires a great deal of mathematical machinery. That stated, we will provide some commentary about extensions to input–output models at the end of each chapter.

Chapter 4 discussed nonlinear differential equations as initial value problems, stated formally in (4.1), where we describe a system by its starting value $\mathbf{x}_0$ and the relationship between the rate of change, $D\mathbf{x}$, and its state $\mathbf{x}$. Section 4.3 also demonstrated that this was enough to uniquely identify $\mathbf{x}(t)$ as a function of $t$, which we will label $\mathbf{x}(t, \boldsymbol{\theta}, \mathbf{x}_0)$. However, except in special cases, it generally is not possible to write down an analytic expression for $\mathbf{x}(t, \boldsymbol{\theta}, \mathbf{x}_0)$. Instead, we must rely on computational methods to approximate it.



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_5

For the rest of this chapter we will describe numerical methods to approximate $\mathbf{x}(t, \boldsymbol{\theta}, \mathbf{x}_0)$ which we will then be able to use within estimation methods. As will be the case for much of this book, there is an extensive literature on this topic and we will provide a very incomplete survey of these methods, confining ourselves to the simplest and most commonly-employed methods. Full treatment of this problem can be found in Deuflhard and Bornemann (2000) and Kincaid and Cheney (2002) as well as more general texts on differential equations (Borrelli and Coleman 2004; Nagle et al. 2008).

## 5.2 Euler Methods

In this section we will start with the simplest of numerical ODE solvers. These are named Euler methods and date to 1768. Although these methods are generally too inaccurate to be useful, they are a good first step in developing more sophisticated approximations.

The differential equation states that $\mathbf{x}(t)$ is changing at rate $\mathbf{f}(t, \mathbf{x}, \boldsymbol{\theta})$, so that if we let time move forward $h$ units where $h$ is small, we should change $\mathbf{x}$ by $h\mathbf{f}(t, \mathbf{x}, \boldsymbol{\theta})$ units. Formally, we write

$$\mathbf{x}(t+h) \approx \mathbf{x}(t) + h\mathrm{D}\mathbf{x}(t) = \mathbf{x}(t) + h\mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta}).$$

If we iterate this, we can use the same procedure to start at the new value $x(t+h)$ and move to $x(t+2h)$ by incrementing by $h\mathbf{f}(t, \mathbf{x}(t+h), \boldsymbol{\theta})$. This scheme is given explicitly in the pseudo-code below:

1. Start $\hat{\mathbf{x}}(t_0) = \mathbf{x}_0$, $t = t_0$.
2. Do until $t > t_1$:
   a. $\hat{\mathbf{x}}(t+h) = \hat{\mathbf{x}}(t) + h\mathbf{f}(t, \hat{\mathbf{x}}(t), \boldsymbol{\theta})$,
   b. $t = t + h$.

This scheme is illustrated in the left hand plot of Fig. 5.1. The approximation initially moves in a direction tangential to the estimated trajectory; subsequent steps move tangentially to the trajectory they would have taken if they started from the last value. As the step size decreases we get closer and closer to the truth.

While this scheme is simple and intuitive, it tends to perform poorly for many systems. For example, consider the system

$$\mathrm{D}\mathbf{x} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \mathbf{x}, \quad \mathbf{x}_0 = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

which is solved for $\mathbf{x}(t) = (\cos(t), \sin(t))$ and for which $\|\mathbf{x}(t)\| = 1$ for all $t$. For the Euler method, however, it is easily checked that

$$\|\mathbf{x}(t) + h\mathrm{D}\mathbf{x}(t)\|^2 = (x_1(t) - hx_2(t))^2 + (x_2(t) + hx_1(t))^2 = 1 + h^2.$$

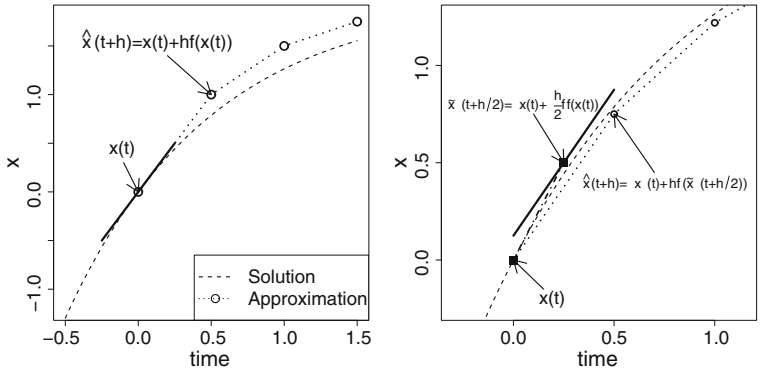


**Fig. 5.1** Illustrations of Euler (*left*) and second order Runge–Kutta methods for approximating the solution to an ordinary differential equation, in this case $\mathrm{D}x = -x + 2$. The Euler step moves tangentially to the true trajectory for $h$ time units. The Runge–Kutta method moves $h/2$ time units, calculates the derivative of the trajectory at this point and then moves $h$ steps in that direction starting from the original point. *Right-hand plot axes* are different from the *left-hand plot* to allow additional detail to be presented, but both plots use the same $h$

Thus, an Euler-step approximation to this model will always tend to spiral away from the origin even though the analytical solution is stable, as we see in Fig. 5.2. Euler methods can have similarly distorting effects on dynamics in other systems, too, as we will see in an analysis of the property of stiffness below.

One way to understand the success of a scheme is through an analysis of how well it approximates the true underlying function $\mathbf{x}(t)$. To do this, we can look at one step from $\mathbf{x}(t)$ to $\mathbf{x}(t + h)$. Here we can use Taylor's theorem to give us

$$\mathbf{x}(t + h) = \mathbf{x}(t) + h\mathrm{D}\mathbf{x}(t) + \frac{h^2}{2}\mathrm{D}^2\mathbf{x}(t^*),$$

where $t^*$ is some time between $t$ and $t + h$. When we substitute $\mathbf{f}(t, \mathbf{x}(t), \boldsymbol{\theta})$ for $\mathrm{D}\mathbf{x}(t)$ we can re-arrange the equation above to yield

$$\mathbf{x}(t + h) - \mathbf{x}(t) - \mathbf{f}(t, \mathbf{x}, \boldsymbol{\theta}) = O\left(h^2\right).$$

Now, of course, we want an error over the whole interval, and since there are $(t_1 - t_0)/h$ steps, we have an over-all error of $O(h)$ (assuming that the system behaves nicely). Here, to improve the accuracy of the over-all solution by an order of 10, we must do 10 times as much work. As we will see below, this can be substantially improved with Runge–Kutta methods.

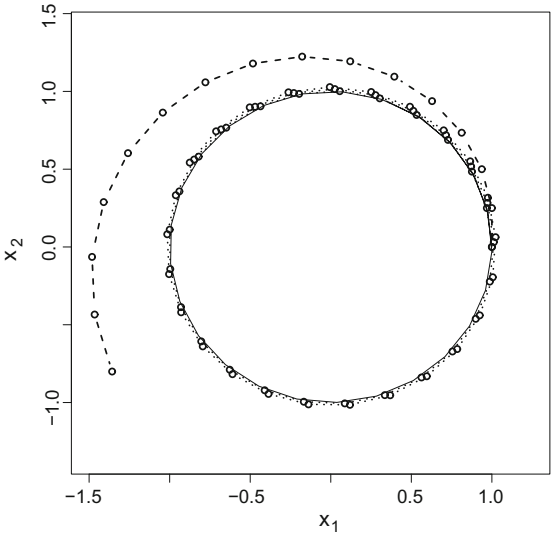


**Fig. 5.2** Approximations to circular motion by Euler (*dashed*) and second-order Runge–Kutta methods (*dotted*). Both methods spiral out from the true *circular* orbit, but the Euler method does so at a much faster rate

## 5.3 Runge–Kutta Methods

As we have seen above, we can think of Euler's methods as using first-order Taylor expansion of $\mathbf{x}(t+h)$ around $\mathbf{x}(t)$ and we have observed that it does not converge particularly quickly. It is tempting, therefore, to extend this to a higher-order expansion:

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\mathrm{D}\mathbf{x}(t) + \frac{h^2}{2}\mathrm{D}^2\mathbf{x}(t) + O\left(h^3\right),$$

and we could employ the relationship

$$\mathrm{D}^2\mathbf{x}(t) = \mathrm{D}\mathbf{f}(t,\, \mathbf{x}(t),\, \boldsymbol{\theta}) = \mathbf{f}(t,\, \mathbf{x}(t),\, \boldsymbol{\theta})\partial_{\mathbf{x}}\mathbf{f}(t,\, \mathbf{x}(t),\, \boldsymbol{\theta}) + \partial_t\mathbf{f}(t,\, \mathbf{x}(t),\, \boldsymbol{\theta})$$

for the third term in the expansion. This would require us to start evaluating partial derivatives of $\mathbf{f}$ with correspondingly more work on the users part and more potential for error. Instead, we can achieve the same level of accuracy by an evaluation of $\mathbf{f}$ at $t + h/2$. To do this, we first of all take an Euler step of size $h/2$ to approximate $\tilde{\mathbf{x}}(t + h/2)$ and then use the derivative at $h/2$ to take a step from $\mathbf{x}(t)$ to $\mathbf{x}(t+h)$. It is easiest to understand this in terms of pseudo-code:

1. Start $\hat{\mathbf{x}}(t_0) = \mathbf{x}_0,\ t = t_0$.
2. Do until $t > t_1$:
   a. $\tilde{\mathbf{x}}(t + h/2) = \hat{\mathbf{x}}(t) + \frac{1}{2}h\mathbf{f}(t,\, \hat{\mathbf{x}}(t),\, \boldsymbol{\theta})$
   b. $\hat{\mathbf{x}}(t+h) = \hat{\mathbf{x}}(t) + h\mathbf{f}(t + h/2,\, \tilde{\mathbf{x}}(t + h/2),\, \boldsymbol{\theta})$
   c. $t = t + h$

The right hand plot of Figs. 5.1 and 5.2 illustrates this graphically where the improvement on Euler schemes is readily apparent.

Intuitively, $\mathbf{f}(t + h/2,\ \tilde{\mathbf{x}}(t + h/2),\ \boldsymbol{\theta})$ should help account for how $\mathbf{f}$ changes between $t$ and $t + h$. In fact, if we put these two steps together we have

$$\hat{\mathbf{x}}(t + h) = \mathbf{x}(t) + h\mathbf{f}(t + h/2,\ \mathbf{x}(t) + h/2\mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta}),$$

and we can expand the second term to be

$$\begin{aligned} &h\mathbf{f}(t + h/2,\ \mathbf{x}(t) + h/2\mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta}),\ \boldsymbol{\theta}) \\ &\quad = h\mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta}) + \frac{h^2}{2}\left[\mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta})\partial_{\mathbf{x}}\mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta})\right] \\ &\qquad + \frac{h^2}{2}\left[\partial_t\mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta})\right] + O\left(h^3\right), \end{aligned}$$

in which we recognize the second term as being the second derivative we needed above. Since one step of this scheme has an error that is $O(h^3)$, the over-all error is of $O(h^2)$ meaning that ten times the work nets us 100 times the accuracy.

For the circular motion example employed above, it is fairly easy to calculate that the Runge–Kutta step is exactly

$$\begin{aligned} \hat{x}_1(t + h) &= x_1(t) + hx_2(t) - \frac{h^2}{2}x_1(t), \\ \hat{x}_2(t + h) &= x_2(t) - hx_1(t) - \frac{h^2}{2}x_2(t), \end{aligned}$$

with $\|\hat{\mathbf{x}}(t)\|^2 = 1 + h^4/4$, if we start on the unit circle. A small increase in the radius of the Runge–Kutta solutions is visible in Fig. 5.2, but this remains considerably more accurate than the Euler step.

The point of the intermediate step $\tilde{\mathbf{x}}$ here is to help conceptualize the algorithm. In particular, similar calculations can improve the error of the approximation up to any degree desired. Typically, a fourth-order method is used which can be written as

$$\hat{\mathbf{x}}(t + h) = \hat{\mathbf{x}}(t) + \frac{h}{6}\left(k_1 + 2k_2 + 2k_3 + k_4\right),$$

where the values $k_1, \ldots, k_4$ are the extension of $\tilde{\mathbf{x}}(t + h/2)$ above and are generated in sequence:

$$\begin{aligned} k_1 &= \mathbf{f}(t,\ \hat{\mathbf{x}}(t)) \\ k_2 &= \mathbf{f}\left(t + h/2,\ \hat{\mathbf{x}}(t) + hk_1/2\right) \\ k_3 &= \mathbf{f}\left(t + h/2,\ \hat{\mathbf{x}}(t) + hk_2/2\right) \\ k_4 &= \mathbf{f}\left(t + h,\ \hat{\mathbf{x}}(t) + hk_3\right). \end{aligned}$$

The over-all error of this method is $O(h^4)$. Although higher methods can also be derived, this is the most commonly used differential equation solver due to its efficiency and ease of implementation. Despite this, it does not always perform well, especially for "stiff" systems; and we will explore where it can break down below.

## 5.4 Collocation Methods

The methods so far describe provide approximations to $\mathbf{x}(t)$ at a set of points $t$, $t + h$, $t + 2h$, etc. If we wish to find a value of $\mathbf{x}$ between these points we must interpolate between them. Alternative means of solving ODEs are described as *collocation methods*. These methods approximate the solution to (4.1) explicitly as a combination of pre-defined functions. We will make use of these in Chap. 9 as a way of moving from initial value problems in which we start from $\mathbf{x}_0$ and approximate $\mathbf{x}(t)$ going forward in time to trying to choose a trajectory that matches data that are distributed over time as well as possible.

Collocation methods represent $\mathbf{x}(t)$ explicitly in terms of a linear combination of a set of basis functions:

$$\mathbf{x}(t) \approx \sum_{k=1}^{K} \mathbf{c}_k \phi_k(t) = \mathbf{C}\boldsymbol{\phi}(t), \tag{5.1}$$

where the $\phi_k(t)$ are a pre-chosen set of functions and the $\mathbf{c}_k$ are weights given to each function. Here $\boldsymbol{\phi}(t)$ is the vector $(\phi_1(t), \ldots, \phi_K(t))'$ and $\mathbf{C}$ is a matrix whose rows are the $\mathbf{c}_k$.

Most commonly, collocation solvers divide the interval $[t_0\, t_1]$ into sub-intervals of length $h$ and use a set of low-order polynomials within each sub-interval. In particular, suppose that we are interested in solving the equation from $t_0$ to $t_0 + h$. For the moment, it will be convenient for us to think of the derivative of $\mathbf{x}$ in terms of a basis rather than $\mathbf{x}$ itself:

$$\mathrm{D}\mathbf{x}(t) \approx \sum_{k=1}^{K} \mathbf{c}_k \mathrm{D}\phi_k(t).$$

We now try to choose the $\mathbf{c}_k$ so as to make the differential equation fit exactly at $K$ collocation points $s_1, \ldots, s_K$ on $[t_0\, t_0 + h]$. In particular, for each $s_l$

$$\mathbf{x}(s_l) = \mathbf{x}(t_0) + \int_{t_0}^{s_l} \mathrm{D}\mathbf{x}(u) = \mathbf{x}(t_0) + \sum_{k=1}^{K} a_{jl}\mathbf{c}_k$$

where

$$a_{jl} = \int_{t_0}^{s_l} \mathrm{D}\phi_k(t)\mathrm{d}t.$$

Here we have expressed $a_{jl}$ somewhat unusually as an integral of $\mathrm{D}\phi_k(t)$; this is so that the increment represents a step starting at $\mathbf{x}(t_0)$. We can now insist that

$$\mathrm{D}\mathbf{x}\left(s_l\right) = \sum_{k=1}^{K} \mathbf{c}_k \mathrm{D}\phi_k\left(s_l\right) = \mathbf{f}\left(t_0 + s_l,\ \mathbf{x}\left(t_0\right) + \sum_{k=1}^{K} a_{jl}\mathbf{c}_k,\ \boldsymbol{\theta}\right).$$

This produces dK equations for the dK variables $\mathbf{c}_k$ (remembering that there are $d$ state variables) that can be solved numerically. Terminologically, we refer to the $\phi_k(t)$ as *collocation functions* with the $s_k$ being *collocation points*.

The Lagrange polynomial basis is often used for collocation or approximation purposes in numerical methodology, and is defined by

$$\phi_k(t) = \frac{(t - t_0)\ldots(t - t_{k-1})(t - t_{k+1})\ldots(t - t_K)}{(t_k - t_0)\ldots(t_k - t_{k-1})(t_k - t_{k+1})\ldots(t_k - t_K)}. \tag{5.2}$$

Because $\mathrm{D}\phi_k(s_l) = \delta_{jl}$ for the Lagrange polynomial basis, these equations reduce to

$$c_{ij} = f_i\left(t_0 + s_l,\ x\left(t_0\right) + \sum_{k=1}^{K} a_{ik}c_{kj}\right), \quad i = 1, \ldots, K, \quad j = 1, \ldots, K.$$

In fact, this choice yields what is referred to as an implicit Runge–Kutta method ("implicit" because the $c_{ij}$ are not explicitly defined but rather numerical solutions to an equation). The solution then proceeds by setting $t_1 = t_0 + h$ and $x(t_1) = \mathbf{x}(t_0) + \sum_{k=1}^{K} a_{kK}$ and repeating the same process in in the next interval $[t_1,\ t_1 + h]$.

It is often useful to use Lagrange polynomials on a standard interval [0, 1] in this case, we use $\phi(ht + t_0)$ to obtain bases on the interval $[t_0,\ t_0 + h]$ and if the $s_l$ are also defined as a fixed set of points on [0, 1] with $a_{kl} = \int_0^{s_l} \phi_k(t)\mathrm{d}t$ we can re-scale to $\int_0^{hs_l} \mathrm{D}\phi(ht)\mathrm{d}t = ha_{kl}$. See Deuflhard and Bornemann (2000) for a more detailed description. Typically, $K$ is small (usually 3) and the error in the collocation approximation can be calculated to be $O(h^K)$.

A more general framework will be used below in which this idea is generalized in which we return to (5.1) with a basis that covers the interval $[t_0\, t_1]$ with $(s_1, \ldots, s_K)$ chosen over the same interval. Using (5.1), we can define the following equations for the $\mathbf{c}_k$:

$$\sum_{k=1}^{K} \mathbf{c}_k \mathrm{D}\phi_k\left(s_l\right) = \mathbf{f}\left(s_l,\ \sum_{k=1}^{K} \mathbf{c}_k\phi_k\left(s_l\right),\ \boldsymbol{\theta}\right), \quad k = 1, \ldots, K, \quad l = 1, \ldots, K. \tag{5.3}$$

This is a much more direct formulation of a collocation approach which is feasible because we have not incorporated a specific starting value $\mathbf{x}(t_0)$ and can express $\mathbf{x}(t)$ explicitly as a basis expansion and add the initial condition $\mathbf{x}(t_0) = \mathbf{x}_0$ as a further equation to be satisfied.

There are many potential choices for the bases $\phi_k$ here. We will discuss some of these—B-splines in particular—in Chap. 8, but the formulation is generic: wavelets, Fourier series and any other sensible choice could all be employed.

The basic idea is illustrated graphically in Fig. 5.3. Here we have illustrated a B-spline basis with 11 knots; the solution will be estimated by a linear combination of these functions. In particular, the right hand plot also illustrates the points at which we insist the ODE be matched along with the B-splines scaled by their coefficients. For this illustration, we have used the system $\mathrm{D}x = -x + 2$ so we can translate (5.3) into

$$\begin{bmatrix} \boldsymbol{\phi}(0) \\ \mathrm{D}\boldsymbol{\Phi} + \boldsymbol{\Phi} \end{bmatrix} \mathbf{c} = \begin{bmatrix} x_0 \\ 2 \end{bmatrix},$$

where $\boldsymbol{\phi}(0) = (\phi_1(0), \ldots, \phi_K(0))$ and $\boldsymbol{\Phi}$ is the matrix with $[\boldsymbol{\Phi}]_{ij} = \phi_k(s_i)$. In this case there is only one component of the trajectory, so only one vector of coefficients $\mathbf{c}$. Generally, the equations are non-linear and require concatenating the trajectory components. For the $s_l$ we have used the mid points between knots as well as the end points of the interval.

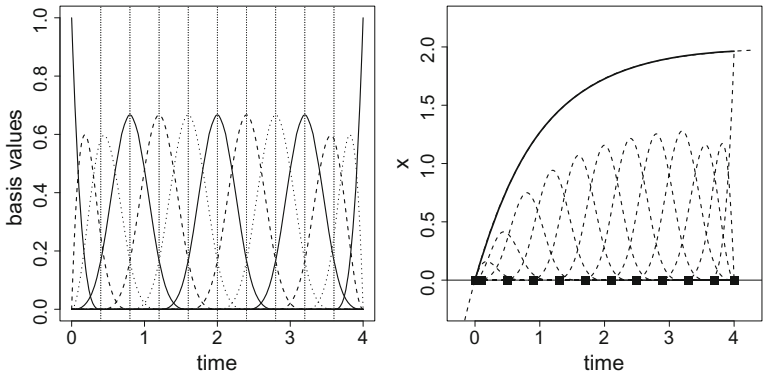


**Fig. 5.3** An illustration of a collocation method. *Left hand plots* the values of a B-spline basis of 13 basis functions defined to be polynomials between breaks (*vertical lines*) and to joint up smoothly at breaks. *Right* a collocation fit to the differential equation $\mathrm{D}x = -x + 2$; the basis functions are scaled by the values of their coefficients and the ODE is forced to be exactly correct at the collocation points, marked by *solid squares*. Visually, the collocation fit exactly obscures the true curve (*dashed lines*)

There are a number of reasons for preferring the local-formulation of using a small basis on a small interval $[t_0,\ t_0 + h]$: while it requires repeated solutions of nonlinear equations, they are of much lower dimension than if a global basis is defined and this

in turn results in a more stable algorithm. However, we will see that global basis such as the B-splines above can be more useful when estimating parameters. Here, we have data distributed over the whole range of $t$ rather than simply specifying an initial condition at $\mathbf{x}(t_0)$ from which to start approximating. This global formulation allows for both the $\mathbf{c}_k$ and the $\boldsymbol{\theta}$ to chosen simultaneously and this can provide substantial numerical advantages.

## 5.5 Numerical Problems

### *5.5.1 Stiffness*

Applying a Taylor expansion to the Euler method in Sect. 5.2 as well as higher-order Runge–Kutta methods suggests that numerical error, at least, in terms of individual time-steps, can be bounded by some derivative of the solution of the ODE. Unfortunately, this analysis does not always provide an adequate characterization of numerical problems associated with solving ODEs, partly because it does not address the propagation of this error through the rest of the solution. This can result in an ODE which has very smooth dynamics having an approximation with quite unstable results. ODEs which exhibit this phenomenon are described as "stiff" and may be illustrated by the one-dimensional system

$$\mathrm{D}x = -15x.$$

The solution of this system is $x(t) = e^{-15t}$ and it converges rapidly to a fixed value of 0. However, Euler and Runge–Kutta methods exhibit strong instability when trying to solve it unless very small step sizes are used—this is illustrated in Fig. 5.4. Euler systems oscillate around the fixed point of 0; if the step size is not small enough these oscillations actually increase. Runge–Kutta methods manage to cancel these oscillations out, but this yields an approximation that converges to the fixed point much more slowly than real solutions.

In the case of this equation, the cause of numerical instability can be described as being due to high curvature relative to step size as the system approaches the fixed point. The error associated with overshoot is then propagated as the solver continues forward, requiring a long time to settle down. An explicit definition of stiffness, however, remains elusive and we define it as requiring excessively small step sizes relative to the smoothness of the over-all solution.

There are numerous characterizations of stability of numerical methods. *A-stability*, of particular note, focusses on the requirement that numerical solutions to $\mathrm{D}x = -kx$ converge to zero for any positive k. For example, in the Euler scheme above with step-size h, it is easy to show that

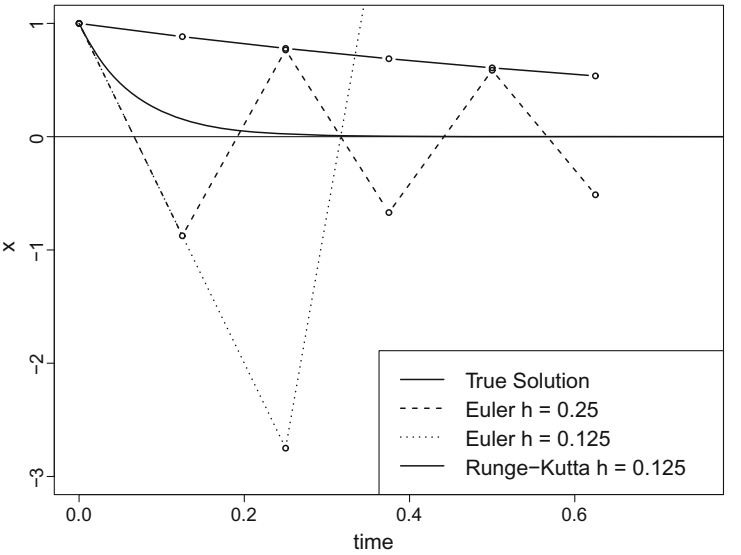


**Fig. 5.4** Example of a stiff system $\mathrm{D}x = -15x$. *Solid blue line* produces exact solution $x(t) = \exp(-15t)$; the *dashed* and *dotted lines* are solutions by Euler methods with $h = 0.125$ and 0.25, respectively and the *solid black line with circles* is the second-order Runge–Kutta solution

$$x(t + nh) = (1 - kh)^n,$$

and this will converge to zero only when $|1 - kh| < 1$ or $h < 2/k$, it is thus not A-stable (since for any $h$, we will find non-convergence for some $k$) and it is, in fact, possible to show that no explicit Runge–Kutta method is A-stable.

To combat stiffness, alternative solvers have been developed. These are either implicit, as in the collocation methods above, or rely on calculating the Jacobian $\partial_{\mathbf{x}}\mathbf{f}$, or both, hence reducing their appeal when stiffness is not a problem. Rosenbrock methods are particularly common among these and have been implemented in multiple software packages. The statistical modeller should be aware that such problems can come up, and that there are numerical methods to deal with them. Some solvers attempt to detect stiff systems and change the numerical method they employ automatically—the `lsoda` methods (for example in the `R` package `deSolve`) produce solutions that are almost identical to the truth.

### 5.5.2 Discontinuous Inputs

So far we have allowed our dynamics to vary with time: $\mathrm{D}\mathbf{x}(t) = \mathbf{f}(t,\ \mathbf{x}(t),\ \boldsymbol{\theta})$, which means that we implicitly allow our models to account for inputs to the system, but we have not described these explicitly. These methods can still be used if we explicitly

include an input $\mathbf{u}(t)$ into $\mathrm{D}\mathbf{x}(t) = \mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta}, \mathbf{u}(t))$, provided that $\mathbf{u}(t)$ is smooth enough.

In real world systems, however, it is very common that $\mathbf{u}(t)$ is discontinuous in, places. Systems which are either locally disturbed, as in the cranial deformation data, or where we see an abrupt change in level as in the refinery data (both in Chap. 3) do not satisfy this condition. These types of step changes are common in experimental or industrial systems.

The effect of this is easiest to see in Euler methods where we relied on the Taylor expansion:

$$\mathbf{x}(t+h) = \mathbf{x}(t) + h\mathrm{D}\mathbf{x}(t) + o\left(h^2\right),$$

to guarantee the accuracy of our method. If $\mathrm{D}\mathbf{x}(t)$ is discontinuous between $t$ and $t+h$, this expansion is no longer valid and we can expect locally poor convergence. The same issues apply to Runge–Kutta methods if $\mathbf{u}(t)$ has a discontinuous derivative (yielding errors in later terms in the Taylor series).

In these cases it is generally wise to split the solution into regions where $u(t)$ is continuous and differentiable between regions. We solve from the initial point up until the first discontinuity (at time $t_d$, say), obtain the final approximated $\mathbf{x}(t_d)$ and then re-start from this value until the next discontinuity.

This approach is efficient where there are few discontinuities to worry about (again, a common scenario in physical systems). However, it becomes more problematic when we let the number of step changes increase below the resolution $h$ of our solver.

### 5.5.3 Constraints and Transformations

In addition to stiff equations, numerical solutions to ordinary differential equations can also suffer when the state variables change over several orders of magnitude and when there are natural constraints on the state space. This occurs frequently in ecological or epidemiological models when the numbers of species (or infectives) can change dramatically and there is a natural boundary that negative population numbers are not allowable.

In these circumstances transforming the state variable $\mathbf{z}(t) = \mathbf{h}(\mathbf{x}(t))$ via an invertible transform can help improve the accuracy of a differential equation solver. For state variables that take positive values, the transformation $z_i(t) = \log x_i(t)$ is particularly useful. We can now express an equivalent differential equation for $\mathbf{z}$ using the chain rule:

$$\mathrm{D}\mathbf{z}(t) = \mathrm{D}\log\mathbf{x}(t) = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta})/\mathbf{x}(t) = \mathbf{f}\left(e^{\mathbf{z}(t)}, \boldsymbol{\theta}\right) * e^{-\mathbf{z}(t)},$$

where we have used $\mathbf{a}/\mathbf{b}$ and $\mathbf{a} * \mathbf{b}$ to indicate element-wise division and multiplication.

The effect of this transformation can most readily be seen for the SIR equations from Chap. 4. By setting $(\tilde{S},\ \tilde{I},\ \tilde{R}) = (\log S,\ \log I,\ \log R)$, then we can translate the equations from (4.8) on the log scale to be

$$\begin{bmatrix} \mathrm{D}\tilde{S} \\ \mathrm{D}\tilde{I} \\ \mathrm{D}\tilde{R} \end{bmatrix} = \begin{bmatrix} -\beta e^{\tilde{I}} \\ \beta e^{\tilde{S}} - \gamma \\ \gamma e^{\tilde{I}} - \delta \end{bmatrix}.$$

The effect of this transform can be seen in Fig. 5.5—the positivity constraints of the system are naturally enforced on the log scale. For log transforms, particularly of populations, there is a natural description of the rates as being *per population* since we divide by the population size.

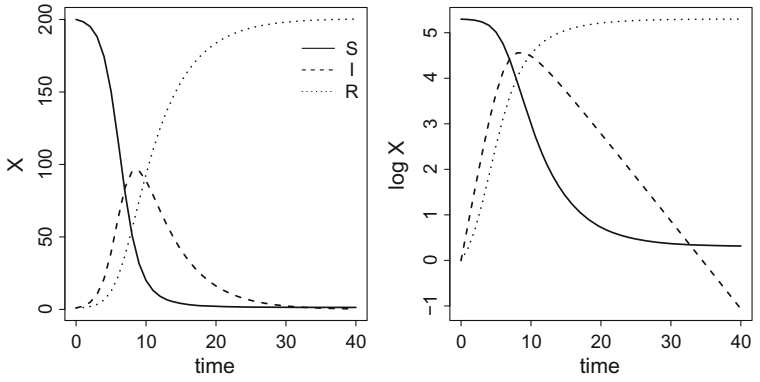


**Fig. 5.5** The SIR model on the scale of the original variables (*left*) and on the log scale (*right*)

Of course, the easiest way to see that this can be effective in improving numerical methods is to examine the stiff system we tried above in which we simply have $\mathrm{D}\log x = -15$ or $\log x(t) = \log x_0 - 15t$ which Euler's method will reproduce exactly.

The log transform is easily the most common transformation used to stabilize the solutions to ordinary differential equations; both because positive state variables occur fairly commonly, and because it allows the original variables to change over orders of magnitude relatively easily. For other constraints, alternative transformations may be appropriate; we may apply a logistic transform to a variable that takes values on [0, 1], for example. It may even be useful to transform two variables together; although we will not use such examples here. Generically, if $\mathbf{z}(t) = \mathbf{h}(\mathbf{x}(t))$ is invertible we can write

$$\mathrm{D}\mathbf{z}(t) = \partial_x \mathbf{h}\left(\mathbf{h}^{-1}(\mathbf{z}(t))\right)\mathbf{f}\left(\mathbf{h}^{-1}(\mathbf{z}(t)),\ \boldsymbol{\theta}\right),$$

where $\partial_x \mathbf{h}$ represents the Jacobian of $\mathbf{h}$ with respect to $\mathbf{x}$ and into which we substitute $\mathbf{h}^{-1}(\mathbf{z}(t))$ and the product is a matrix multiplication. While this formulation appears ugly, as we have seen above, it can result in quite simple dynamical systems models and can be worth employing. We will also use this idea to get rough parameter estimates in Chap. 7.

# Chapter 6
# Qualitative Behavior

## 6.1 Introduction

Within the discipline of applied mathematics, a large part of dynamical systems theory is concerned with the description of the qualitative behavior of dynamical systems. By this we mean whether the system converges to a fixed point, diverges to infinity, produces a consistent pattern of oscillations or does something more complicated. Frequently, these qualitative features are the main points of interest, especially if they change with parameters. For example, we want to know if a disease will become manageable, if oscillations between predators and prey will be maintained indefinitely or if long term weather patterns are predictable.

The analysis of qualitative behavior in ODEs is not a completely foreign concept for statisticians. In regression models, an important focus is on the interpretation of parameters and the effect of interaction terms on model output. In the context of dynamical systems, it is often qualitative features of solutions that are important to understand. Indeed, in applied mathematics modeling, the parameters themselves are often of secondary interest, as is quantitative agreement with data. Rather, the goal is to produce a plausible, if idealized, model that reproduces the *types* of behavior seen in nature and when they occur. This, it is hoped, will yield a better understanding of the mechanisms involved in the natural process.

This distinction – that qualitative rather than quantitative agreement with observations has been a focus of model development – is an important viewpoint. Firstly, it explains the observation that many ODE models – particularly those that produce complex dynamics – have poor quantitative agreement with observed data. This should not be surprising: ODE models assume an idealized system in which unmeasured, external influences are removed and populations are so large that random fluctuations can be ignored. It is only in these simplified contexts that a mathematical analysis of qualitative behavior can be made. We hope that the conclusions reached from this analysis can be applied, at least approximately, to more complex



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_6

and realistic systems where small perturbations of the system can have long-lasting effects on its quantitative behavior by, for example, shifting the timing of peaks in system that exhibits cycles.

Much of the interest in these systems arises through the theory of *bifurcations*: essentially when qualitative aspects of systems change when some parameter in the system is altered. It is not our purpose here to provide a comprehensive guide to bifurcation theory (there are numerous large books and a whole research field devoted to this; see Kutznetsov 2004; Nagle et al. 2008; Alligood et al. 1997; Borrelli and Coleman 2004 for example); we will instead describe some simple tools with which to understand system behavior at a basic level. Our intention is to provide an overview of some of the features that can be produced with these models, some basic mathematical analyses of them, and a window into a way of thinking about models that can be quite different from traditional statistical concerns.

The analysis of qualitative behavior can already be examined in the linear systems analyzed in Chap. 3 where solutions to linear ODEs can be written down explicitly. Recalling a linear first order differential equation in one dimension:

$$\mathrm{D}x = \beta x \Rightarrow x(t) = Ce^{\beta t} ,$$

we see that the behavior of the system transitions at $\beta = 0$ between tending to zero and blowing up to infinity. This change in behavior is termed a *bifurcation* which occurs at $\beta = 0$. In the general homogeneous first order differential equation for a vector-valued $\mathrm{D}\mathbf{x} = B\mathbf{x}$, the difference between converging to zero and diverging depends on the eigenvalues of $B$. For nonlinear systems, there is generally no algebraic expression for solutions that we can so easily inspect and we instead approximate their behavior locally by the behavior of a linear ODE. However, because of their nonlinear structure, they can exhibit a broader range of behaviors than we see in linear dynamics.

For the most part, this chapter will consider *autonomous* systems to simplify the analysis. By this we mean that $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ does not change over time. In particular, external inputs are not considered here. The reason for this is that this way we can ensure that the evolution of $\mathbf{x}$ will not depend on the time at which it is started. More explicitly, if $\mathbf{x}(t_1) = \mathbf{x}(t_0)$ at some time points $t_1 \neq t_0$, then $\mathbf{x}(t_1 + s) = \mathbf{x}(t_0 + s)$ at all times following them. If this occurs, we can guarantee that the system will continue to cycle. Clearly, this is a strong restriction that excludes many systems that are scientifically interesting. We will discuss ways in which this analysis can be extended in Sect. 6.6.

In this chapter, we will first examine the fixed points of a system – where the system stays at the same value forever in Sect. 6.2 where we will characterize whether they attract the system, or repel it. We'll then examine cyclic systems – described as having limit cycles in Sect. 6.3. Section 6.4 will describe some types of *bifurcations*, or parameter values at which the qualitative behavior of the system changes. We will discus some more complex features in Sect. 6.5 and non-autonomous systems in Sect. 6.6. The reader should be aware that this chapter is summarizing at least a course-worth of material in a very compressed fashion.

## 6.2 Fixed Points

The first point of analysis in any system is to understand its fixed points – that is, at what values of $\mathbf{x}$ will the system remain at $\mathbf{x}$? In the context of ODEs

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$$

the fixed points are given as solutions $\mathbf{x}^*$ to the equation

$$\mathbf{f}(\mathbf{x}^*) = 0$$

since this condition is equivalent to $\mathbf{x}(t)$ having no derivative in $t$, meaning that once the system hits $\mathbf{x}^*$ it will remain there forever. The simplest such system is a one dimensional linear ODE

$$\mathrm{D}x = \alpha - \beta x$$

in which at $x = \alpha/\beta$, $\mathrm{D}x = 0$.

As a two-dimensional example, consider the Lotka-Volterra equations discussed in Chap. 4 to describe the interaction of a predator $y$ with a prey species $x$:

$$\begin{aligned} \mathrm{D}x &= \alpha x - \beta x y \\ \mathrm{D}y &= \delta x y - \gamma y \ . \end{aligned} \tag{6.1}$$

Here it is easy to see that the solution $(x, y) = (0, 0)$ is a fixed point: with neither predators nor prey, the ecology will do nothing. There is another fixed point, however at $(x, y) = (\gamma/\delta, \alpha/\beta)$. However, as we will see below, neither fixed point tells us much about trajectories that are close to them in this case. For this we require some stability analysis.

### *6.2.1 Stability*

Fixed points indicate the values of $\mathbf{x}$ where the system will not change. However, this does not account for the behavior of the system near a fixed point. In some cases, if the system is close to a fixed point it will be attracted to it; a ball rolling to the bottom of a valley is a good, but imperfect, analogy. When this occurs, the fixed point is *stable*: small perturbations from it will be attracted back to it. In other cases, even a small perturbation will result in the system leaving the neighborhood of the fixed point – as in a ball balanced on the top of a peak – and in this case the fixed point is said to be *unstable*.

We can see this most readily in solutions to stationary linear differential equations. Recall from Chap. 3 that the second order linear Equation 3.6

$$\mathrm{D}^2 x = -\beta_1 \mathrm{D}x - \beta_0 x$$

has solutions of the form

$$x(t) = e^{\beta_1 t/2}\left[[C_1 \sin\left(\sqrt{\beta_0 - \beta_1^2/2}t\right) + C_2 \cos\left((\sqrt{\beta_0 - \beta_1^2/2}t\right)\right]$$

if $\beta_1^2 < 4\beta_0$ and

$$x(t) = e^{\beta_1 t/2}\left[C_1 \exp\left(\sqrt{\beta_1^2/2 - \beta_0}t\right) + C_2 \exp\left(-\sqrt{\beta_1^2/2 - \beta_0}t\right)\right]$$

if $\beta_1^2 > 4\beta_0$. From here we can readily read off some behavior. First, if $\beta_1 > 0$, $x(t)$ will tend to infinity at an exponential rate. Alternatively, if $\beta_1 < 0$ the system shrinks exponentially to zero. This makes zero either *unstable* or *stable* depending on the sign of $\beta_1$. Similarly, the system will exhibit periodic behavior (either shrinking or growing) if $4\beta_0 > \beta_1^2$ and will not oscillate otherwise.

Our solutions derived in Chap. 3 can be used more generally. Recall that the equation

$$\mathrm{D}\mathbf{x} = \mathbf{B}\mathbf{x}$$

has solutions given by

$$\mathbf{x}(t) = \sum_{i=1}^{d} c_i e^{u_i t} v_i$$

where the $u_i$ and $v_i$ are eigenvalues and eigenvectors of $\mathbf{B}$ and the $c_i$ are constants that depend on initial conditions. Now, recalling that $u_i = a_i + ib_i$ we have

$$e^{(a_i + ib_i)t} = e^{a_i t}(\cos(b_i t) + i \sin(b_i t)) \tag{6.2}$$

so that we can see that behavior is driven by the values of the $u_i$. This yields the following classification:

- If the real parts of the $u_i$ are all negative, the system converges to zero as $t$ grows. This makes the origin a *stable* fixed point.
- If the real parts of the $u_i$ are all positive, the system diverges from zero in all directions. This makes the origin *unstable*.
- If some of the real parts of $u_i$ are positive and some are negative, the system converges towards zero along eigen-directions associated with negative $Re(u_i)$ and diverges along eigen-directions associated with positive $Re(u_i)$. This is referred to as a *saddle point*. Note that unless a point lies exactly along an eigenvector, a solution starting from there will ultimately diverge.
- If the real parts of any pair of eigenvalues are exactly zero, but there are non-zero imaginary parts, the system produces circular behavior in the plane defined by their eigendirections.

This classification is illustrated in Fig. 6.1 where we show stable, unstable and saddle fixed points. Note that when there are non-zero complex parts, the system produces a spiral inwards or outwards associated with the trigonometric terms in (6.2). However, there are no spiral equivalents to a saddle node.

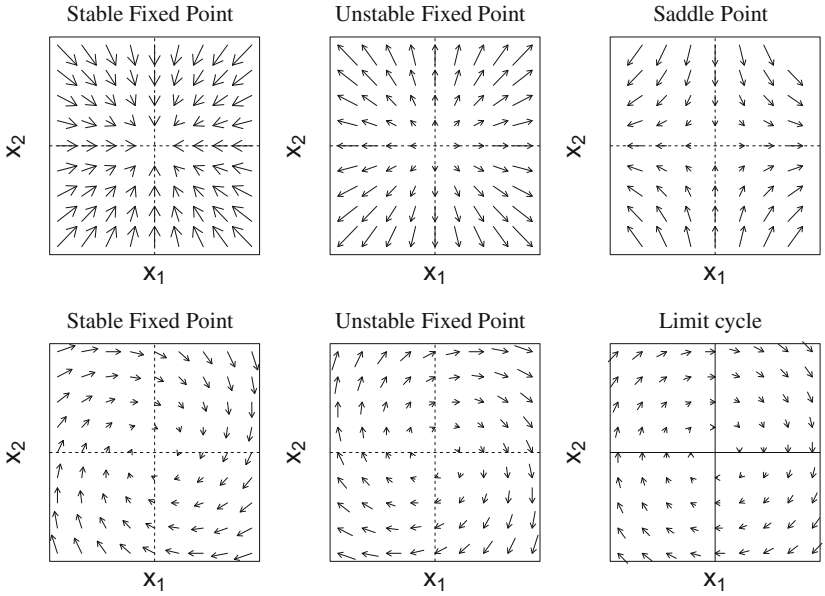


**Fig. 6.1** Types of stability in linear systems $\mathrm{D}\mathbf{x} = A\mathbf{x}$. *Top row* systems in which $A$ has no imaginary eigenvalues; a stable fixed point in which all trajectories converge along a straight line to the origin, an unstable fixed point in which all trajectories diverge from the origin in a straight line and a saddle point in which trajectories converge towards 0 in $x_2$ but diverge in $x_1$. *Bottom row* systems in which $A$ has imaginary eigenvalues resulting in spiral or circular motion; stable fixed point, unstable fixed point and a limit cycle corresponding to eigenvalues with no real part

This works well for linear systems for which we can write down the global classification. However, for nonlinear systems we can only indirectly infer behavior. As a way to try and understand this, we can consider the vector field nearby a fixed point $\mathbf{x}^*$. In particular, if we make a linear approximation to $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ (which will only hold near $\mathbf{x}^*$), we can use the behavior we observed above as a guide. In formal terms, we write

$$\mathrm{D}\mathbf{x} = \mathbf{f}(\mathbf{x}^*, \boldsymbol{\theta}) + \partial_{\mathbf{x}}\mathbf{f}(\mathbf{x}^*, \boldsymbol{\theta})(\mathbf{x} - \mathbf{x}^*) + o\left(\|\mathbf{x} - \mathbf{x}^*\|^2\right) \ .$$

Here the Jacobian $\mathbf{J} = \partial_{\mathbf{x}}\mathbf{f}(\mathbf{x}^*, \boldsymbol{\theta})$ is a square matrix with derivatives along columns and components of $\mathbf{f}$ down rows:

$$[\mathbf{J}]_{ij} = \frac{df_i(\mathbf{x}^*, \boldsymbol{\theta})}{dx_j}.$$

Substituting $\mathbf{z} = \mathbf{x} - \mathbf{x}^*$ yields

$$\mathrm{D}\mathbf{z} = \mathbf{J}\mathbf{z} + o\left(\|\mathbf{z}\|^2\right) .$$

Thus, close enough to $\mathbf{x}^*$ the system behaves like the linear equations

$$\mathrm{D}\mathbf{z} = \mathbf{J}\mathbf{z} . \tag{6.3}$$

and we will say that $\mathbf{x}^*$ is *stable* if the real parts of all the eigenvalues of $\mathbf{J}$ are negative, and *unstable* if the real part of one is positive because locally, at least, $\mathbf{x}^*$ attracts or repels trajectories respective.

We will illustrate this with three examples:

Example 1: We examine an *SIR* system from Chap. 4 with a constant birth rate $\mu$ and where we have dropped the recovered class $R$:

$$\begin{aligned} \mathrm{D}S &= \mu - \alpha SI \\ \mathrm{D}I &= \alpha SI - \beta I \end{aligned}$$

It is easy to see by first setting $\mathrm{D}I = 0$ that there is only one fixed point for these equations:

$$I^* = \frac{\mu}{\gamma}, \; S^* = \frac{\gamma}{\alpha} .$$

If we now turn to the Jacobian of the system, we arrive at

$$\mathbf{J}(S, I) = \begin{bmatrix} -\alpha I & -\alpha S \\ \alpha I & \alpha S - \gamma \end{bmatrix}, \; \mathbf{J}(S^*, I^*) = \begin{bmatrix} -\frac{\alpha\mu}{\gamma} & -\gamma \\ \frac{\alpha\mu}{\gamma} & 0 \end{bmatrix} .$$

The eigenvalues of $\mathbf{J}(S^*, I^*)$ are given by $-\alpha\mu/2\gamma \pm \sqrt{(\alpha\mu/2\gamma)^2 - \alpha\mu}$ which we can readily verify are always negative. So the fixed point is stable. For some values of the parameters, the eigenvalues may also be complex indicating a spiral towards the fixed point as in Fig. 6.2; as functions of time, $(S(t), I(t))$ will exhibit damped oscillations. We should note that if we start with $I = 0$, then we always have $\mathrm{D}I = 0$ but $\mathrm{D}S = \mu$ so the trajectory is simply described by $S = \mu t$ which is not attracted to the fixed point.

Example 2: Consider the van der Pol equations

$$\begin{aligned} \mathrm{D}w &= a(w - w^3/3 - v) \\ \mathrm{D}v &= w/a \end{aligned}$$

which are used to describe certain electrical circuits. We see that $(0, 0)$ is a fixed point of the system. If we examine the Jacobian

$$\mathbf{J}(w, v) = \begin{bmatrix} a - w^2 a & -a \\ 1/a & 0 \end{bmatrix}.$$

it is easy to see that at $(w, v) = (0, 0)$ the eigenvalues are $a/2 \pm \sqrt{a^2/4 - 1}$ which are always positive, so the origin is unstable. In fact, later on we will see that the system exhibits cycles, but near the origin it tends to grow away from it.
This provides at least an initial description of the system; however it cannot be guaranteed to hold outside of a local region. Global analysis of behavior can be considerably more difficult as we see below.

Example 3: To illustrate when this can be problematic, we will return to the Lotka-Volterra model, where

$$\mathbf{J}(x, y) = \begin{bmatrix} \alpha - \beta y & -\beta x \\ \delta y & \delta x - \gamma \end{bmatrix}.$$

Examining the fixed points we have

$$\mathbf{J}(0, 0) = \begin{bmatrix} \alpha & 0 \\ 0 & -\gamma \end{bmatrix}, \ \mathbf{J}\left(\frac{\gamma}{\delta}, \frac{\alpha}{\beta}\right) = \begin{bmatrix} 0 & -\beta\gamma/\delta \\ \alpha\delta/\beta & 0 \end{bmatrix}.$$

where we observe that $(0, 0)$ is a saddle node – it will be approached by predators when there is very little of either species, but the prey will have explosive growth – while the eigenvalues of $\mathbf{J}(\gamma/\delta, \alpha/\beta)$ are both purely complex. This indicates that a limit cycle is likely, but needs to be examined more closely.

We have plotted vector fields along with example solutions of these differential equations in Fig. 6.2. This figure also applies to the next section: the second two examples both have stable limit cycles and we provide a means of showing this formally below.

## 6.3 Global Analysis and Limit Cycles

While the stability of fixed points provides local information about the behavior of a system, having an unstable fixed point does not guarantee that the system will diverge to infinity. As we have seen empirically in the Lotka-Volterra system (6.1), the fixed points are a saddle node and a limit cycle, but general solutions to the equations, while periodic, are distinctly non-circular. In general, a global analysis of system behavior can be very challenging – indeed it occupies a considerable part of the research in applied mathematics – but we illustrate a few methods that will provide some insight into system behavior.

### 6.3.1 Use of Conservation Laws

We saw in the discussion of the continuously stirred tank reactor in Chap. 4 that there are quantities that cannot be changed by system dynamics. These are known as conservation laws; most famous among these are physical laws such as conservation of energy in a system. Alternatively, in a closed chemical reaction, the total number of atoms of the contents cannot be changed. In dynamical terms what this means is there is a fixed quantity $h(\mathbf{x}, \boldsymbol{\theta})$ that is a function of the state $\mathbf{x}$ and which does not change. Therefore, $\mathbf{x}$ must travel along level sets of $h$.

More explicitly for any ODE if we can find $h(\mathbf{x}, \boldsymbol{\theta})$ such that

$$\mathrm{D}h(\mathbf{x}) = [\partial_{\mathbf{x}} h(\mathbf{x})]^T \, \mathbf{f}(\mathbf{x}, \boldsymbol{\theta}) = 0 \, ,$$

then we know that if $h(\mathbf{x}_0, \boldsymbol{\theta}) = h^*$ then $\mathbf{x}(t)$ must travel on the $h^*$ contour of $h(\mathbf{x})$. In particular, if the contours of $h$ are closed, then we know that $\mathbf{x}(t)$ must either have a stable fixed point on the contour, or it will have a limit cycle.

Returning to the Lotka-Volterra equations (6.1), if we construct

$$h(x, y) = -\delta x + \gamma \log(x) - \beta y + \alpha \log(y) \tag{6.4}$$

we have that

$$\left[\partial_{(x,y)} h(x, y)\right]^T \mathbf{f}(x, y) = \left[-\delta + \frac{\gamma}{x}\right][\alpha x - \beta x y] + \left[-\beta + \frac{\alpha}{y}\right][\delta x y - \gamma y] = 0 \, ,$$

so that the system will stay on the contours of $h(x, y)$. Since we know that $\mathbf{x}(t)$ has only a fixed points $(0, 0)$ and $(\delta/\gamma, \alpha/\beta)$, the system must trace out trajectories that are contours of $h(x, y)$. These are shown in right panel in Fig. 6.2 where we see

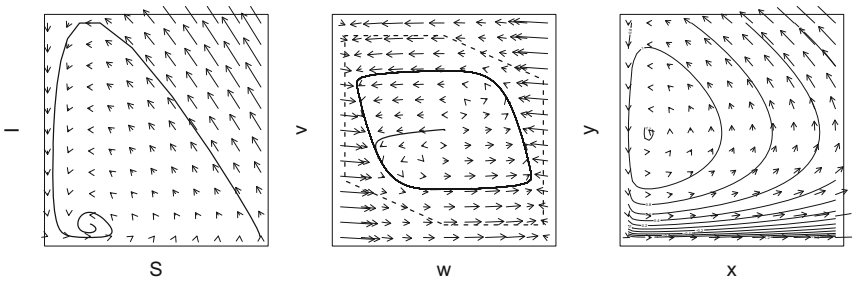


**Fig. 6.2** Vector fields for three example systems. *Left* an SI model with constant birth rate, *arrows* indicate the vector field and *solid lines* an example solution. *Center* phase plane and vector field of the van der Pol oscillator. *Solid lines* provide a solution of the ordinary differential equation, *dashed lines* outline a bounding box from which the movement of the system is always into the interior of the box. *Right* phase plane of the Lotka-Volterra model. *Arrows* indicate direction of motion, contours are of the level sets of $h(x, y)$ in (6.4). The system will stay on the contour on which it starts

that a limit cycle emerging from our analysis; but one which depends on the initial conditions.

This method, as with those below, cannot be applied to all systems and there is no formula for deciding that a conservation law can be found. However, when it can, this provides an elegant and useful method of describing system dynamics. We should note that there have been some attempts at discovering conservation laws from data (Schmidt and Lipson 2009; Schulte and Drew 2010). There are numerous difficulties with this, including the possibility that conservation laws may exist only when including unobservable state variables, and the means by which noise can be filtered out. We will not examine these ideas further below, but they remain an interesting area of open research.

### *6.3.2 Bounding Boxes*

Another way to establish the existence of a limit cycle, at least in planar systems, is to be able to bound the dynamics. Imagine that we can construct a box such that if we start $\mathbf{x}$ at any point on the edge of the box, it immediately moves into the interior. Then we know that once $\mathbf{x}(t)$ is inside the box, it can never escape.

Inside the box a number of behaviors can result. A stable fixed point will, of course, attract trajectories. The more interesting case is when an unstable fixed point can has a bounding box. In three or more dimensions, a stable limit cycle could exist, or the system may produce more complicated or chaotic, dynamics that are briefly described below. In two dimensions, however, a consequence of the Poincaré-Bendixson theorem is that such systems must have a stable fixed point or a limit cycle (e.g. Borrelli and Coleman 2004).

The middle panel in Fig. 6.2 illustrates this for the van der Pol system. We have drawn an irregular hexagon on which the derivatives $(\mathrm{D}x, \mathrm{D}y)$ all point into the interior. It is somewhat tedious, algebraically, to demonstrate that this is, in fact, the case.

There are further techniques for demonstrating the existence of stable limit cycles. It might be possible, for example, to find a quantity that is not conserved, but is bounded. Some of these can be obtained through the bifurcation analysis described below – especially for fast-slow systems – others require analysis specific to the system at hand. We will not go beyond these analyses here; the interested reader is directed to a large literature, for example Kutznetsov (2004).

## 6.4 Bifurcations

Bifurcation analysis refers to changes in qualitative behavior as parameters of the system change. While we will present some of the simpler forms of this analysis, there is an enormous literature describing complex forms of bifurcations. Our purpose here

is to provide the basic language to describe qualitative changes in systems and an understanding of the ideas and their application.

The very simplest bifurcation can be observed in the linear equation

$$\mathrm{D}x = \beta x \tag{6.5}$$

with solutions $x(t) = x_0 e^{\beta t}$ where we see that there is a fixed point at $x = 0$ which is stable if $\beta < 0$ and unstable if $\beta > 0$. In fact, for linear systems the only real changes in behavior are from stable to unstable fixed points and in the direction (or existence) of circular motion. However, nonlinear systems can exhibit a variety of behaviors as a parameter is changed.

At this point, we must answer the fundamental question: "Why should a statistician care? Aren't parameters fixed, anyway?" There are, we believe, three reasons that make the material in this section relevant:

1. The statistical properties of parameter estimates can change depending on system behavior. For a system with stable fixed points, for example, there are generally no consistent estimators of initial conditions if observations are taken over an increasing time domain.
2. Frequently, the parameter that is being varied can be controlled in an experiment. In the example below, we examine the change in the qualitative behavior of the FitzHugh-Nagumo equations for neural firing as the neuron is stimulated by an electric current.
3. In some systems it can be useful to think of a slow-moving state variable as a parameter and examine the behavior of the fast-moving variables with the slow-moving variable fixed. This can allow us to understand the global behavior of a system by breaking it down into lower-dimensional counterparts.

Through the rest of this section, we will briefly illustrate some of the most common forms of bifurcation. The analysis of these is not different to extending the analysis of stability given in the previous section to a range of possible parameter values. However, the forms of bifurcation will be useful to understand and, as we will see, we will sometimes be able to demonstrate that a limit cycle must exist close to a bifurcation point. For the the sake of presentation, we will join numerous mathematical texts in presenting only the simplest examples of these, confining ourselves to one dimension where possible. The same behaviors can be found in more realistic systems at the cost of more complex mathematical calculations.

### *6.4.1 Transcritical Bifurcations*

Transcritical bifurcations describe a system in which one or more fixed points change stability as a parameter changes. The linear system (6.5) is a simplest example: $x = 0$ is a fixed point which is stable for $\beta < 0$ and unstable for $\beta > 0$. In a more complex setting, we will consider the one-dimensional system

$$\mathrm{D}x = rx - x^2$$

where we consider the behavior of the system as $r$ changes sign. Here it is easy to see that $\mathrm{D}x = 0$ at $x = 0$ and at $x = r$. By looking at the Jacobian $r - 2x$ we see that if $r < 0$ then the fixed point at $x = 0$ is stable since locally $x(t) \approx e^{rt}$ and $x = r$ is unstable. However, as $r$ crosses 0, the $x = 0$ fixed point becomes unstable, while the $x = r$ fixed point becomes stable; effectively, the two fixed points swap stability.

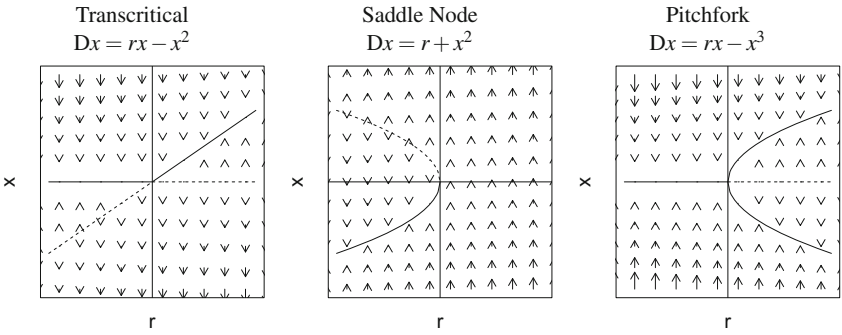


**Fig. 6.3** Three types of bifurcation. In each of the plots, the parameter $r$ is given along the $x$-axis and the fixed points – as they change with $r$ – are represented by *lines*; *solid lines* indicate stable fixed points while *dashed lines* are unstable. The directly of motion (vertical because there is one state variable $x$ at each value of $r$) is indicated by the *arrows*. In each case there is a bifurcation at $r = 0$ given by a *vertical line* where the behavior of the system changes

This can be illustrated on a bifurcation diagram in Fig. 6.3. We plot the fixed points on the $y$ axis against the parameter $r$ on the $x$ axis. Stable fixed points are depicted by solid lines while unstable fixed points are given dashed lines. To aid an understanding of behavior, we have also plotted arrows indicating the direction of motion from any initial point.

### *6.4.2 Saddle Node Bifurcations*

Saddle node bifurcations occur when an unstable system produces a pair of fixed points, one stable and one unstable. A simple system for this is

$$\mathrm{D}x = r + x^2$$

where if $r > 0$ there is no fixed point – $\mathrm{D}x$ is always positive and the system increases to infinity. On the other hand, as $r$ becomes negative, there are fixed points at $x = \pm\sqrt{-r}$ and since the Jacobian is $2x$ the negative fixed point is stable while the positive fixed point is unstable. This is depicted in the 2nd panel of Fig. 6.3.

### 6.4.3 Pitchfork Bifurcations

There is no need to produce only one pair of fixed points at any given time. In pitchfork bifurcations, one stable fixed points becomes three – two stable fixed points with an unstable point between them. Continuing our set of univariate examples, this is produced by the system

$$\mathrm{D}x = rx - x^3$$

Here, if $r < 0$ the only solution to $\mathrm{D}x = 0$ is $x = 0$. The Jacobian of the system is $r - 3x^2$ so the system is stable at the fixed point.

However, as we move to $r > 0$, we observe two more fixed points occurring at $\pm\sqrt{r}$. Moreover, since $r > 0$, the $x = 0$ fixed point has now become unstable, while the two new fixed points are both stable ($J(x) = -2r$); note that for univariate differential equations, pairs of stable fixed points *must* have an unstable fixed point between them. The third plot in Fig. 6.3 provides a graphical representation of this behavior.

It is easy to see that reversing the signs of this system (to give $\mathrm{D}x = x^3 - rx$), we can produce the opposite behavior – an unstable fixed point at $r < 0$ becomes stable, but surrounded by two unstable points at $\pm\sqrt{r}$. This creates a region of stability between the two unstable fixed points and is known as a *subcritical* pitchfork bifurcation (as opposed to the supercritical case above).

### 6.4.4 Hopf Bifurcations

Hopf bifurcations have a form that gives us some extra insight into system behavior, and tend to cause most excitement in applied mathematicians. These describe a stable fixed point becoming unstable and producing a stable limit cycle along the way.

To demonstrate this, we'll consider van der Pol system expressed in terms of two state variables (we need at least two dimensions for a system to cycle at all)

$$\begin{aligned}\mathrm{D}x &= -y + (\mu - y^2)x \\ \mathrm{D}y &= x\ .\end{aligned}$$

Here we see that the origin is a fixed point (this tends to happen in systems developed for mathematical analysis; it's less common in systems that describe the real world) and the jacobian is

$$J(x, y) = \begin{bmatrix} \mu & -1 - 2yx \\ 1 & 0 \end{bmatrix},\ J(0, 0) = \begin{bmatrix} \mu & -1 \\ 1 & 0 \end{bmatrix},$$

which has eigenvalues $\mu/2 \pm \sqrt{\mu^2/4 - 1}$ which are always of the same sign as $\mu$. Moreover, at $\mu = 0$ the eigenvalues are purely imaginary: $\pm i$. So as $\mu$ passes through

zero the origin goes from being a stable fixed point to an unstable fixed point via a pair of purely imaginary eigenvalues. This is illustrated in Fig. 6.4 where the limit cycle is depicted in the final figure by drawing it in centered at each value of $\mu$.

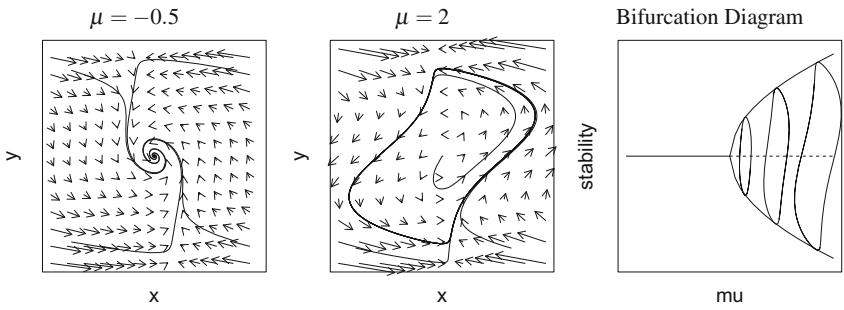


**Fig. 6.4** A Hopf bifurcation in the van der Pol system. *Left* at $\mu < 0$ the fixed point is stable and all trajectories spiral into it. *Middle* for $\mu > 0$, there is a stable limit cycle that attracts trajectories from both its interior and exterior. *Right* bifurcation diagram of the stability of the system as $\mu$ changes: the limit cycle around the unstable fixed point at the origin is depicted by plotting its maximum and minimum $y$ value and depicting the cycle as it changes while scaling $x$ and re-centering it at $x = \mu$

This behavior – a fixed point changing into a limit cycle – is called a "Hopf bifurcation" due to the Hopf bifurcation theorem which says, in a more general context, that if a fixed point moves with $\theta$ from being stable to unstable by having one pair of eigenvalues move to being purely imaginary $\lambda = \pm i\omega$ (while the rest have negative real parts) then close to the bifurcation, either the unstable fixed point must be surrounded by a stable limit cycle, or the stable fixed point must be surrounded by an unstable limit cycle. Close to the bifurcation, the cycle will have period $\omega/2\pi$ and its amplitude will shrink to zero at the bifurcation point.

The example here is described as a *supercritical* Hopf bifurcation; the opposite case (an unstable limit cycle surrounding a stable fixed point which becomes unstable) is described as *subcritical*. The Hopf bifurcation theorem is one of the easiest means of demonstrating the existence of a limit cycle in more than two dimensions and these are some of the most commonly studied features in dynamics. It is limited, though, in applying only very close to a bifurcation point and only when a single pair of eigenvalues has real part crossing zero.

## 6.5 Some Other Features

Besides bifurcations, differential equations can exhibit some other qualitative features. Chaos, of course, gets its share of publicity. It is also possible to understand systems when some states move much more slowly than others, or as input-output systems. As yet, we have seen little work on the effect that these features have on statistical methods; we've also seen little by the way of statistical procedures aimed at investigating them. Below, we provide brief, heuristic, introductions to each.

### 6.5.1 Chaos

The notion of chaos has been one of the most influential developments in 20th century mathematics. Few have not heard of the "butterfly effect," – that a butterfly flapping its wings in Brazil can cause a hurricane in Florida – coined by Edward Lorenz as a description of the consequences of chaotic systems. At an intuitive level, chaotic systems have complex, non-repeating trajectories that nevertheless remain within a closed volume and exhibit high sensitivity to initial conditions. That is, the system neither diverges, has fixed points or stable limit cycles. This book will not focus on chaotic systems – although there are some famous real-world examples – but it deserves some attention as one of the most interesting consequences of dynamics. Our discussion in this section is therefore deliberately informal; more sophisticated presentations of chaos are provided in Alligood et al. (1997).

The most famous example of this is the three-dimensional system[1] named for Lorenz himself:

$$\begin{aligned} \mathrm{D}x &= \sigma(y - x) \\ \mathrm{D}y &= x(\rho - z) - y \\ \mathrm{D}z &= xy - \beta z \ . \end{aligned} \tag{6.6}$$

This model was originally produced as a simplified system describing atmospheric dynamics, where Lorenz observed complex behavior with high sensitivity to starting values. An example of the trajectories produced by this system at $(\sigma, \rho, \beta) = (10, 28, 8/3)$ is given in Fig. 6.5. Here we have also plotted the divergence of $x$ from two points that are 0.01 apart – for a while they appear to track but we can see that they end up in different loops at different times after a short while.

In terms of dynamical analysis, for $\rho < 1$ there is a single fixed point which goes through a saddle node bifurcation at $\rho = 1$ creating two stable fixed points. These each then pass through a Hopf bifurcation, creating two stable limit cycles that increase in diameter quickly as $\rho$ increases. These start interacting so that at $\rho = 28$ the classical chaotic picture emerges. In the top right panel Fig. 6.5 solutions spiral out from an unstable fixed point until they hit a critical value at which point they switch to spiraling out from the other fixed point. The number of rotations around each fixed point between transitions appears random but in fact proceeds from the deterministic evolution of the Lorenz equation. The bifurcation behavior is depicted in the bottom right panel of Fig. 6.5 where we have indicated for the size of the limiting orbit by the maximum and minimum value of $x$. Because this is a chaotic system, this very approximate calculation can be quite inaccurate, hence the jagged-looking line.

[1] It is possible to show that differential equations must have at least a three-dimensional state variable to show chaos. This is because a trajectory cannot intersect itself; in two dimensions you either proceed to a fixed point or end up on a limit cycle.

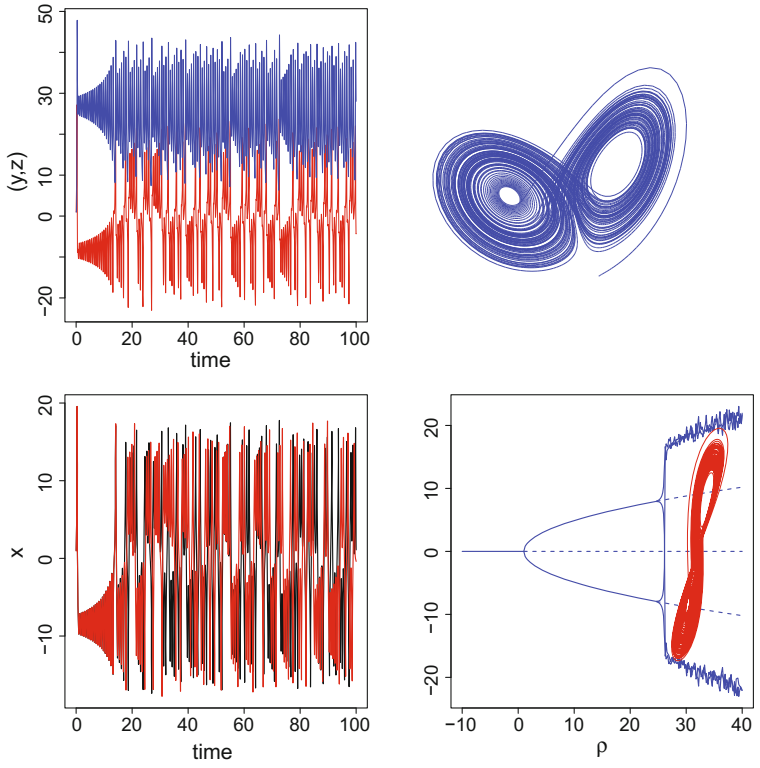


**Fig. 6.5** The Lorenz system (6.6). *Top left* a time-series plot of $y$ and $z$. *Bottom left* two time series of $x$ with starting points differing by 0.01 demonstrating the sensitivity of the system to initial conditions. *Top right* a three-dimensional phase portrait of the system. *Bottom right* a bifurcation diagram

Whether a system is chaotic is measured by it's Lyapunov exponents. At an intuitive level, these measure how quickly trajectories starting from $\mathbf{x}_0$ and $\mathbf{x}_0 + \boldsymbol{\varepsilon}$ diverge for small $\boldsymbol{\varepsilon}$. These are given on a log scale, so if any Lyapunov exponent is positive, the trajectories move away from each other exponentially quickly. This happens in systems that diverge but when we can also show that the system is contained in a finite region. The only way for this to occur is if the trajectories follow a complicated path that never intersects itself (hence chaos can only happen in three or more dimensions). The seeming contradiction between trajectories diverging from each other exponentially quickly and yet remaining within a finite region occurs because the divergence is measured in the limit as $\boldsymbol{\varepsilon} \to 0$ so we really observe that the derivative $\partial_{\mathbf{x}_0}\mathbf{x}(t, \mathbf{x}_0, \boldsymbol{\theta})$ grows exponentially fast as $t$ increases.

Besides producing appealing pictures, what are the implications of this behavior for statistics? The first is predictability; since we generally don't really know initial conditions perfectly, the sensitivity of chaotic systems to these conditions means that we can only predict very short time-periods ahead. Another consequence is for numerics: the introduction of numerical errors in solving differential equations can mean that we end up quite far from the true trajectory of the system. For statistics, the implications of this behavior have not been very carefully analyzed. On one hand, sensitivity to initial conditions should mean that we can estimate these very well indeed. On the other, the fact that our numerical solutions to these ODEs might not correspond to the initial conditions we think we are using may make us skeptical of this. We are not aware of much work about the implications of all these considerations for the problem of estimating parameters.

### 6.5.2 *Fast-Slow Systems*

The analysis of dynamical systems can be aided by separating time scales between fast-moving and slow-moving state variables. We assume that the fast-moving variables are always at their limiting behavior – treated as though the slow-moving variables are fixed – we can then break the analysis into lower dimensional and (hopefully) understandable components.

We will illustrate this with a system that is similar to the van der Pol equations above. In this case we take a two-dimensional system

$$\begin{aligned} \mathrm{D}x &= rx - x^3 + y \\ \mathrm{D}y &= \varepsilon x \end{aligned}$$

where $\varepsilon$ is intended to be small so that $y$ moves slowly relative to $x$. In this system we can then analyze the behavior of $x$ as though $y$ was fixed and only then examine $y$. Here we refer to $y$ as being the *slow manifold* and $y$ as the *fast manifold*. Although these correspond to individual state variables, in more complex systems similar analyses can be carried out in which the slow manifold is simply some lower dimensional subset of the state space.

The dynamics of the system are depicted in Fig. 6.6. We observe that for $y > 2r\sqrt{r}/3\sqrt{3}$, $\mathrm{D}x = 0$ only at a single point with $x < -r$ and in particular, the fixed point (for each given $y$) is stable. Similarly for $y < 2r\sqrt{r}/3\sqrt{3}$, there is only one fixed point at $x > r$. However, as $y$ moves towards zero, the system undergoes a saddle node bifurcation with a new pair of fixed points appearing at $x = \mp\sqrt{2r}$ when $y = \pm r\sqrt{r}/2\sqrt{2}$. For $y \in [-r\sqrt{r}/2\sqrt{2},\ r\sqrt{r}/2\sqrt{2}]$ the system has three fixed points with the outer-two being stable and the inner being unstable.

Now, consider the behavior of $y(t)$ if we assume that $x(t)$ is always at its fixed point for each value of $y$. Assume, to begin with, that $y \in [-r\sqrt{r}/2\sqrt{2},\ 2\sqrt{2r}]$ and $x$ is at its positive stable fixed point. Since $\mathrm{D}y = -\varepsilon x$, $y(t)$ will slowly decrease in value until it crosses $-r\sqrt{r}/2\sqrt{2}$; at this point the system in $x$ crosses the saddle-node bifucation so that the only fixed point is at $x < -\sqrt{r}2$. We can assume that $x$ moves to this point "instantaneously" as far as $y$ is concerned; $\mathrm{D}y$ is now positive and $y(t)$ will increase. Since $x(t)$ is at the negative stable fixed point, it will stay there until $y$ passes the saddle node bifurcation at $y = 2\sqrt{2r}$ at which point the negative fixed point disappears, $x$ moves to the positive fixed point and $y(t)$ starts to decrease again. The result of this is the limit cycle shown in Fig. 6.6; when plotted over time it exhibits a "square wave" with time spent on the fast manifold appearing as near-vertical transitions between slower dynamics.

## 6.6 Non-autonomous Systems

So far, this chapter has considered only systems who's dynamics do not change over time. This makes analysis easy because the future behavior of $\mathbf{x}$ just depends on its current value. However, we are frequently faced with systems in which this is not the case. These represent an *input-output* system in which an input function $u(t)$ is transformed into output $\mathbf{x}(t)$. Many of our initial examples in Chap. 1 took exactly this form.

In linear systems, explicit formulae for how $\mathbf{x}(t)$ transforms the input are given in Chap. 3 where we generally observe a smoothing out of inputs. In nonlinear systems we generally cannot be so explicit. For these problems, the most direct way to understand system responses to inputs is to assume that the inputs change much more slowly than $\mathbf{x}(t)$ responds to them. Just as in fast-slow systems, we can then assume that $\mathbf{x}(t)$ is always at its equilibrium behavior at each value of the input.

As an example, we will depart from this chapter's use of very simple models to examine the Hodgkin-Huxley model of neuron firing (Wilson 1999). This model, which earned its originators the Nobel Prize in 1963, has a four-dimensional state vector that describes the flow of a number of different chemical ions across the membrane of a neurone:

$$
\begin{aligned}
\mathrm{D}v &= C\left[I(t) - G_{Na}m^3h(v - E_{Na}) - G_K n^4(v - E_K) - G_L(v - E_L)\right] \\
\mathrm{D}m &= \frac{0.1(v+40)}{1 - e^{-0.1(v+40)}}(1-m) - 4e^{-(v+65)/18}m \\
\mathrm{D}h &= 0.07e^{-(v+65)/20}(1-h) - \frac{1}{1+e^{-(v+35)/10}}h \\
\mathrm{D}n &= \frac{0.01(v+55)}{1 - e^{-0.1(v+55)}}(1-n) - 0.125e^{-(v+65)/80}n\ .
\end{aligned}
$$

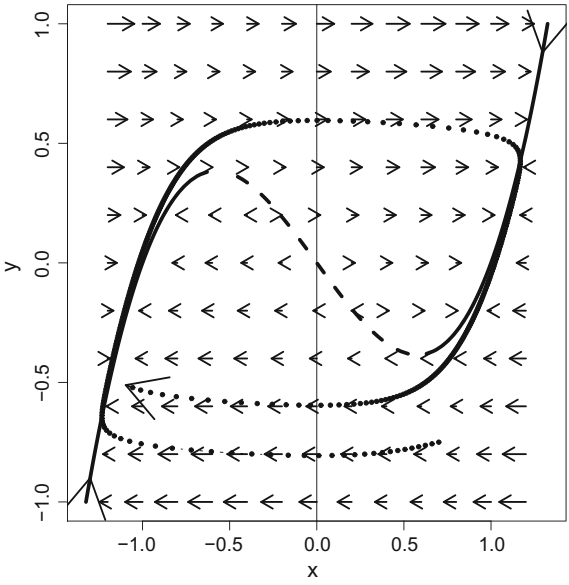


**Fig. 6.6** A fast-slow system in which the dynamics of $x$ are much faster than those of $y$. *Lines* indicate the slow manifold where D$x = 0$ for each value of $y$, *solid lines* indicate stable fixed points while *dashed lines* indicate unstable fixed points. The *arrows* indicate the vector field in $x$ for each y. *Dots* are equally-spaced values of an example trajectory. Where $x$ is negative, $y$ is increasing slowly, where it is positive, $y$ is decreasing slowly

In this model the voltage $v$ across the neurone membrane is measured – it is also the primary state variable of interest. The voltage consists of an input current $I(t)$ as well as currents associated with Sodium (Na) and Potassium (K) ion channels along with a generalized "leak" of current (L). The terms in the first equation are just Ohm's law stating that each of these voltages will tend towards their equilibrium value given by $E_{Na}$, $E_K$ and $E_L$, respectively.

The rate at which they do so, however, depends on the specific conductances of both the sodium and potassium ion channels. These are themselves parameterized by quantities $m$, $h$ and $n$, representing the fractions of potential channels that are able to allow ions to pass, and these variables themselves respond dynamically to the voltage. In the case of the potassium channel, for example, we may need four copies of a certain type of protein to be in the right configuration (hence the $n^4$ term occurring in Potassium conductance). The number of proteins in this configuration, however, changes depending on the voltage and the configurations of the proteins around it. The constant $C$ in the equation for $v$ is intended to indicate that $v$ moves on a "fast" time scale relative to the other variables. It should be noted that this system is quite difficult to analyze – fixed points must be found numerically, for example. However, several simplifications can be made that make the system two dimensional and hence more readily understood – see Wilson (1999) for a details.

These equations include an input current $I(t)$. When $I(t)$ is zero, the system can be shown to have a stable fixed point at the commonly used parameter values $(E_{Na}, E_K, E_L, G_{Na}, G_K, G_L, C) = (50, , -77, -54.4, 120, 36, 0.3, 1)$. However, as $I(t)$ increases, the system undergoes at Hopf bifurcation producing a stable limit cycle at around $I = 6.5$. This can be seen in Fig. 6.7. The input shown

in the left panel is typical of neuron experiments, an input voltage is turned on, causing the neuron to fire repeatedly, allowed to remain on for a while and then turned off at which point the stable fixed point takes-over the dynamics.

However, this interpretation should be treated with some caution: the plot on the right demonstrates behavior (over a longer experiment) when $I(t)$ is increased linearly up to the value 10. In this case, $v$ tracks the fixed point closely until it becomes unstable, when it takes a long period (in neural terms) to spiral out to the limit cycle. When the voltage is decreased linearly and the fixed point re-appears, however, it finds it fairly quickly. The distinction here is that the jump from $I = 0$ to $I = 10$ in the left panel leaves the system close to the limit cycle already. In fact, by increasing $I$ slowly enough, the spike train can be suppressed for very long periods.

The alternative, in which $I(t)$ changes as fast or faster than the dynamics of $\mathbf{x}(t)$ require specialized analysis and depends on the characteristics of $I(t)$ (sometimes, we can think of $I(t)$ as having its own dynamics).

## 6.7 Commentary

In this chapter we have provided an overview of mathematical methods to examine the qualitative features of dynamical behavior. There is an enormous literature on this and the material here is by no means comprehensive. This aspect of dynamical systems modeling has been enormously important in motivating and justifying many of the systems currently in use and we believe that anyone wishing to examine the

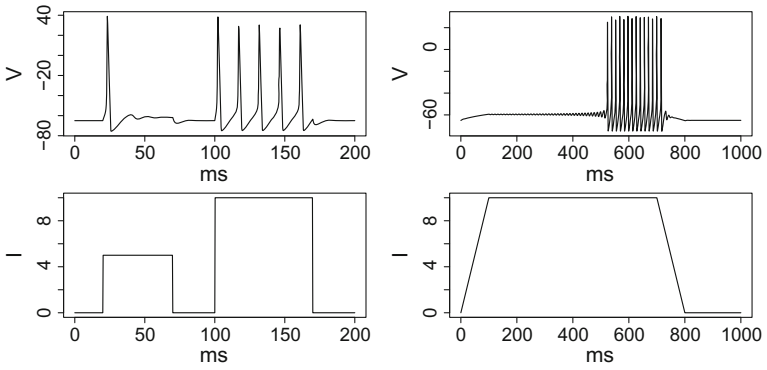


**Fig. 6.7** Two simulated single neuron experiments using the Hodgkin-Huxley model. In both cases the input current is given in the *lower plot*. The *left hand plot* presents a standard neural experiment in which the current is switched on, and then switched off again and the system responds with a stable fixed point or limit cycle according to the value of $I$. In the *right hand plot*, $I$ is increased linearly in time and the system tracks the fixed point and takes a long time to spiral out to the limit cycle after the bifurcation has been crossed

combination of differential equations and data should understand and appreciate the power of these approaches.

Nonetheless, the statistical modeler should also understand and appreciate their limitations, and these are numerous. First of all, the analysis only deals with the asymptotic properties of dynamical behavior; the long-run trajectory of the model. Most systems approach this trajectory with an exponential rate, so in some sense this ought to be reasonable. Nonetheless, we often find that *transients* – dynamical behaviors on the way to asymptopia – are often both informative and the main portion of a trajectory that we observe. Consider, once again, the simple linear system (6.3). If we let $t \to \infty$ it isn't difficult to see that observations will yield increasingly little information about $\beta$; indeed there are no consistent estimates of $\beta$ in this situation. It is the transient behavior as $x(t)$ tends to its fixed point that is informative about the parameters. A second important limitation of this analysis is that it often only applies very near to bifurcation points. The Hopf bifurcation theorem, for example, does not specify how far away (in parameter space) from the bifurcation a limit cycle will exist, nor how large around the now-unstable fixed point it can be. We can refine our analysis to examine the phase relations in the cycle (Cortez and Ellner 2010) but the application of this result to any particular set of parameters remains heuristic. In general, numerical methods, despite their potential flaws, must be used to investigate actual behavior.

The final critique of these methods is that they apply to very simplified systems that frequently do not fit the observed data very well. They often mimic behavior well in a qualitative sense, but a statistician would not want to plot both the ODE solution and real-world data on the same graph. This is simply because these systems tend to be highly simplified idealizations of a more-complex reality: they presume convenient forms, few state variables and isolation from the broader world. None of these hold in reality and while we can again hope that a tractably-analyzable ODE does, in fact, capture the essential behavior of a model, this again remains – at least initially – a heuristic argument.

# Chapter 7
# Nonlinear Least Squares or Trajectory Matching

## 7.1 Introduction

Chapters 3–4 in this book have focussed on an overview of differential equation models, the mechanistic thinking behind their development and ways to work with them mathematically. We can now move on to discussing how they may be reconciled with real-world data. Largely, our task will be obtaining estimates of parameters from data and providing confidence intervals for them, but we will also touch on diagnostics and model criticism as well as experimental design.

We assume that the reader is familiar with the statistical concepts of estimation, hypothesis testing and Bayesian analysis; we have attempted to briefly provide some background to these as they come up but the reader is also directed to the many text on statistical inference. For this chapter, texts on nonlinear regression analysis such as Bates and Watts (1988) or Seber and Wild (2003) are particularly useful resources.

In this chapter we take up the oldest and still the most popular method for parameter estimation, referred to in many texts as "nonlinear least squares" or by the acronym NLS. The modelling context is the standard ordinary differential equation initial value problem of the form

$$\mathrm{D}\mathbf{x}(t) = \mathbf{f}(t; \mathbf{x}(t), \boldsymbol{\theta}) \text{ and } \mathbf{x}(t_0) = \mathbf{x}_0 \tag{7.1}$$

where $\mathbf{x}(t)$ is a vector containing the state or trajectory values $x_i(t)$; $i = 1, \ldots, d$.

Although in many situations the initial state $\mathbf{x}_0$ of the system can be considered as known, we will face up to the challenge that this may not be so, and therefore that the initial state vector can itself be an additional set of parameters to be estimated. We therefore use the *var theta* symbol, $\boldsymbol{\vartheta}$, to indicate the subset of $(\boldsymbol{\theta}, \mathbf{x}_0)$ that must be estimated throughout this chapter and suppress any elements of $\boldsymbol{\theta}$ that are known, so that it is understood that $\mathbf{x}(t; \boldsymbol{\vartheta}) = \mathbf{x}(t; \boldsymbol{\theta}, \mathbf{x}_0)$ and that all the elements of $\boldsymbol{\vartheta}$ must be estimated. It will occasionally be important to distinguish between $\boldsymbol{\theta}$ and $\mathbf{x}_0$ and we return to using these separately where that is the case.



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_7

A great simplification in notation is achieved if we begin by considering only a single equation, and we will take advantage of this framework in this and the next two sections, and then in Sect. 7.4 discuss what needs to be considered in moving to systems of equations, where only a subset of the system may be observed. That is, for now we will replace $\mathbf{x}(t)$ by $x(t)$. Note, too, that we also assume a first order equation, so that only a single initial value $x_0$ is associated with the equation. That is, a single higher order differential equation will be treated later in the chapter as a system of first order equations.

Although the techniques in this chapter are generally described as nonlinear least squares (NLS), we feel that the term *trajectory matching* is more descriptive of this process as applied to differential equations; and we contrast it to *gradient matching* in the next chapter. In this chapter we provide an overview of numerical methods, and some of the hazards, involved in trajectory matching. We also examine ways to estimate uncertainty in parameters that are obtained from data.

The parameter estimation strategy in this chapter can be characterized as nonlinear regression. That is, the ordinary differential equation solution $x$ is approximated by $\hat{x}$ computed using the methods in Chap. 5 at each set of parameters $\boldsymbol{\theta}$ and initial conditions $x_0$. The parameters are manipulated so as to maximize agreement with the data. Here, as in most of the nonlinear regression literature, agreement with the data is expressed in terms of the sum of squared errors criterion

$$\mathtt{SSE}(\boldsymbol{\vartheta}) = \sum_{j=1}^{n} \left(y_j - x(t_j, \boldsymbol{\vartheta})\right)^2. \tag{7.2}$$

However, not much in the theory of nonlinear regression changes if we use other loss functions such as negative log likelihood, as we will do in Sect. 7.4 and Chap. 10 where we will bypass the numerical approximation entirely.

The least squares fitting criterion does bring some simplifications in terms of both computation and inference, and we will discuss these in the next two sections. The Gauss–Newton method is the most commonly used technique for optimization in non-linear least squares problems and we take this up in the next section. These methods can be used both for minimizing squared error or maximizing log likelihoods as well as obtaining confidence intervals for parameters. However, ordinary differential equation models can pose particular problems; likelihood surfaces can be highly multi-modal and often have discontinuities associated with bifurcation points. Under these circumstances, alternative numerical optimization methods can be employed; most numerical software packages provide a suite of options and the interested reader is referred to Kincaid and Cheney (2002). Ideas from the collocation methods reviewed in Chap. 5 can also be used to help to smooth out the roughness surface, and we overview these approaches and why they are helpful.

Below, Sects. 7.2–7.4.4 provide a description of nonlinear least squares as applied to differential equation models to estimate parameters and obtain confidence intervals first in a one-dimensional case and then for multi-dimensional observations. While this approach is mathematically elegant, it has several potential shortcomings.

These include optimization problems, finding good initial parameter guesses, and identifiability. The final sections deal with related methods: Bayesian approaches are undertaken in Sect. 7.5, specialized optimization approaches in Sect. 7.6 and fitting features of the data in Sect. 7.7. We end with the case study of head impacts presented in Chap. 1.

## 7.2 Least Squares with Gauss–Newton Methods

We start this chapter using the most readily-analyzable model for observations in nonlinear regression. In this section, we assume that we have a univariate observation $y$ corresponding to state variable $x$ at time $t_j$ along with observational error:

$$y_j = x(t_j, \boldsymbol{\theta}, x_0) + \varepsilon_j. \tag{7.3}$$

The minimization of error sum of squares (7.2) can be motivated by the assumption that the errors or residuals $\varepsilon_j$ are independent of each other as well as the trajectory $x$ itself. It is also implicitly assumed that the variance of the $\varepsilon_j$'s is finite. One admits, however, that the least squares criterion is usually chosen for much more mundane reasons, such as the ready availability of software.

Unfortunately, we can seldom find the minimizing $\boldsymbol{\vartheta}$ analytically, and therefore must resort to numerical optimization methods. The most common way to minimize $\mathtt{SSE}(\boldsymbol{\vartheta})$ is with a *Gauss–Newton algorithm* because of its reliability and reasonably fast convergence. The method also has the useful property of making it easy to obtain estimates of standard errors for $\boldsymbol{\vartheta}$.

We start with an initial guess $\boldsymbol{\vartheta}^0$. Taylor's theorem says that we can approximate $\mathtt{SSE}(\boldsymbol{\vartheta})$ by a quadratic function of $\boldsymbol{\vartheta}$ given in terms of its derivatives at $\boldsymbol{\vartheta}^0$:

$$\mathtt{SSE}(\boldsymbol{\vartheta}) \approx \mathtt{SSE}(\boldsymbol{\vartheta}^0) + \partial_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0)(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}^0) + \frac{1}{2}(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}^0)^T \partial^2_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0)(\boldsymbol{\vartheta} - \boldsymbol{\vartheta}^0), \tag{7.4}$$

where $\partial_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0)$ is the *gradient vector* at the initial guess and $\partial^2_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0)$ is the corresponding *Hessian matrix.*

This quadratic approximation to $\mathtt{SSE}(\boldsymbol{\vartheta})$ is minimized at

$$\boldsymbol{\vartheta}^1 = \boldsymbol{\vartheta}^0 - \left[\partial^2_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0)\right]^{-1} \partial_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0) \tag{7.5}$$

provided that the Hessian matrix is positive definite, meaning that it has positive eigenvalues. Thus, $\boldsymbol{\vartheta}^1$ is new guess for repeating the procedure. This is the *Newton–Raphson method* for minimizing a function (Nocedal and Wright 2006), and the basic idea is illustrated in Fig. 7.2 for an analysis of the refinery data in Fig. 7.1.

For the least squares criterion the derivatives are:

$$\begin{aligned}\partial_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0) &= -2\sum_{j=1}^{n} \partial_{\boldsymbol{\vartheta}} x(t_j, \boldsymbol{\vartheta}) \left(y_j - x(t_j, \boldsymbol{\vartheta})\right) \\ &= -2\partial_{\boldsymbol{\vartheta}} \mathbf{x}(\boldsymbol{\vartheta})^T (\mathbf{y} - \mathbf{x}(\boldsymbol{\vartheta}))\end{aligned}$$

where $\mathbf{y}$ is the vector of length $n$ containing observations $(y_j)$ and $\mathbf{x}(\boldsymbol{\vartheta})$ is the vector of $(x(t_j, \boldsymbol{\vartheta})$ values. The matrix

$$\mathbf{J}(\boldsymbol{\vartheta}) = [\mathbf{J}(\boldsymbol{\vartheta})]_{jl} = \frac{dx(t_j, \boldsymbol{\vartheta})}{d\vartheta_l},$$

having $n$ rows and a column for each parameter in $\boldsymbol{\vartheta}$, is often called the *Jacobian matrix*. We can write out the Hessian matrix using $\mathbf{J}(\boldsymbol{\vartheta})$ as

$$\partial^2_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta}^0) = -2\mathbf{J}(\boldsymbol{\vartheta})^T \mathbf{J}(\boldsymbol{\vartheta}) - 2\partial^2_{\boldsymbol{\vartheta}} x(\boldsymbol{\vartheta})^T (\mathbf{y} - \mathbf{x}(\boldsymbol{\vartheta})).$$

At the true $\boldsymbol{\vartheta}$, where $y_j - x(t_j, \boldsymbol{\vartheta}) = \varepsilon_j$, the second term is a weighted average of mean-zero random variables, and therefore should be small near the true $\boldsymbol{\vartheta}_0$. Consequently omitting this term will be a minor perturbation of the complete Hessian matrix. Moreover, the first term is a symmetric cross-product matrix, which by its nature is likely to be positive definite. The result of dropping the second term is a simple algorithm based only on first derivatives of $x(t; \boldsymbol{\vartheta})$ with respect to parameters. It's steps are the following:

1. Initialize $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^0$.
2. Do until convergence:

$$\begin{aligned}\mathbf{H}(\boldsymbol{\vartheta}^k) &= \mathbf{J}(\boldsymbol{\vartheta}^k)^T \mathbf{J}(\boldsymbol{\vartheta}^k) \\ \mathbf{g}(\boldsymbol{\vartheta}^k) &= \mathbf{J}(\boldsymbol{\vartheta}^k)^T (\mathbf{y} - x(\boldsymbol{\vartheta}^k)) \\ \boldsymbol{\vartheta}^{k+1} &= \boldsymbol{\vartheta}^k - \mathbf{H}(\boldsymbol{\vartheta}^k)^{-1} \mathbf{g}(\boldsymbol{\vartheta}^k).\end{aligned}$$

A good implementation of this algorithm should include checks that each step actually reduces $\mathtt{SSE}(\boldsymbol{\vartheta})$, and may include a search for a minimum along a line in the step direction $-\mathbf{H}(\boldsymbol{\vartheta}^k)^{-1}\mathbf{g}(\boldsymbol{\vartheta}^k)$. The Levenberg–Marquardt algorithm (Nocedal and Wright 2006) replaces $\mathbf{H}(\boldsymbol{\vartheta}^k)$ with $\mathbf{H}(\boldsymbol{\vartheta}^k) + \gamma_k \mathbf{I}$ where $\gamma_k$ is chosen each iteration to ensure both that each step improves the objective function as well as making sure that the matrix can be inverted. It is also advisable to add error-catching codes in case the algorithm moves parameter estimates to values for which numerical approximations to $x(t; \boldsymbol{\vartheta})$ fail.

This algorithm does not, of course, terminate at an exact minimum and we therefore need to decide when we have found estimates that are "good enough". We can judge this in a number of ways:

1. There has been little movement in parameter estimates: $\|\boldsymbol{\vartheta}^{k+1} - \boldsymbol{\vartheta}^k\| < \delta_1$,
2. The objective function has not improved much: $SSE(\boldsymbol{\vartheta}^k) - \mathtt{SSE}(\boldsymbol{\vartheta}^{k+1}) < \delta_2$.

3. The gradient of the objective function is small: $\|\partial_{\boldsymbol{\vartheta}} \mathtt{SSE}(\boldsymbol{\vartheta})\| < \delta_3$.

Generally, tolerances can be set for each one. Termination criteria for the algorithm can require one or more than one of these tolerances to be met. Nocedal and Wright (2006) recommend at least requiring a tolerance on the gradient.

Here we emphasize an important principle in nonlinear least squares: that parameters be transformed so that they have comparable magnitudes. Otherwise, the change in parameter estimates, or the size of the gradient can be dominated by one parameter having a large magnitude. Gauss–Newton methods can also experience numerical problems when gradients with respect to different components of $\boldsymbol{\vartheta}$ have very different scalings. Parameters can be made numerically comparable by linear re-scaling, or using $\log(\vartheta)$ in place of $\vartheta$ when the natural parameter should be positive. Likewise, the components of $\mathbf{x}$ themselves should be of comparable sizes. For example, when working with the SIR model, the susceptible population may be orders of magnitude larger than the infected populations, in which case transforming the equation to its logarithmic form is advisable.

An important point to note here is that this algorithm assumes that we can calculate $\mathbf{J}(\boldsymbol{\vartheta}) = \partial_{\boldsymbol{\vartheta}} \mathbf{x}(\boldsymbol{\vartheta})$. This is a non-trivial problem, which we examine in the next section.

### 7.2.1 Sensitivity Equations

To implement the Gauss–Newton algorithm, we need to calculate $\mathbf{J}(\boldsymbol{\vartheta})$. This is made difficult because we have only implicitly defined $x(t; \boldsymbol{\vartheta})$ through the initial value problem (7.1). We therefore do not, in general, have an algebraic expression to differentiate. As a first approach, it is possible to use a finite difference to numerically approximate

$$\frac{dx(t; \boldsymbol{\vartheta})}{d\vartheta_l} \approx \frac{x(t; \boldsymbol{\vartheta} + \delta \mathbf{e}_l) - x(t; \boldsymbol{\vartheta})}{\delta}$$

where $\mathbf{e}_l$ is a vector of zeros with a one in the $l$th position and $\delta$ is a small positive number. However, these approximations can be unstable, especially if the ODE solver you are using includes adaptive strategies to choose step sizes. In these cases, numerical artifacts can be a large part of the resulting gradient.

Instead, we define the *sensitivity equations* by considering the time-derivative of $\mathbf{J}(t; \boldsymbol{\vartheta})$. To do so, we re-expand $\boldsymbol{\vartheta}$ into $\boldsymbol{\theta}$ and $x_0$ since they must be treated differently. First we observe that by changing the order of differentiation and applying the chain rule

$$\mathrm{D}\left[\partial_{\boldsymbol{\theta}} x(t; \boldsymbol{\theta}, x_0)\right] = \partial_{\boldsymbol{\theta}} f(t; x(t; \boldsymbol{\theta}, x_0), \boldsymbol{\theta}) + \partial_x f(t; x(t; \boldsymbol{\theta}, x_0), \boldsymbol{\theta}) \partial_{\boldsymbol{\theta}} x(t; \boldsymbol{\theta}, x_0)$$

and similarly

$$\mathrm{D}\left[\partial_{x_0} x(t; \boldsymbol{\theta}, x_0)\right] = \partial_x f(t; x(t; \boldsymbol{\theta}, x_0), \boldsymbol{\theta}) \partial_{x_0} x(t; \boldsymbol{\theta}, x_0).$$

Here, we have written out differential equations that treat $\mathbf{J}$ as additional state variables whose solutions can be approximated alongside the original equations within a numerical solver. Before we can simply solve these extended equations, however, we need to specify initial conditions. This can be done a priori. Since $x(t_0) = x_0$, $x(t_0)$ does not depend on the parameters $\boldsymbol{\theta}$ so that

$$\partial_{\boldsymbol{\theta}} x(t_0; \boldsymbol{\theta}, x_0) = \mathbf{0}$$

where $\mathbf{0}$ is a matrix of zeros. Similarly,

$$\partial_{x_0} x(t_0; \boldsymbol{\theta}, x_0) = \mathbf{I}$$

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

Putting all these together we have the expanded set of differential equations

$$\mathrm{D}\begin{pmatrix} x \\ \partial_{\boldsymbol{\theta}} x \\ \partial_{x_0} x \end{pmatrix} = \begin{pmatrix} f(t; x, \boldsymbol{\theta}) \\ \partial_{\boldsymbol{\theta}} f(t; x, \boldsymbol{\theta}) + \partial_x f(t; x, \boldsymbol{\theta}) \partial_{\boldsymbol{\theta}} x \\ \partial_x f(t; x, \boldsymbol{\theta}) \partial_{x_0} x \end{pmatrix}$$

with corresponding initial conditions

$$\begin{pmatrix} x(t_0) \\ \partial_{\boldsymbol{\theta}} x(t_0) \\ \partial_{x_0} x(t_0) \end{pmatrix} = \begin{pmatrix} x_0 \\ \mathbf{0} \\ \mathbf{I} \end{pmatrix}.$$

We illustrate this process with one of the simplest systems we have available. Figure 7.1, illustrates data from an oil refinery, described in Chap. 1 and also displayed in Fig. 1.2. Here we measure level in a large tray. When the outflow is turned off, the tray fills to a new level, tending towards it exponentially. To describe this behavior, we write that

$$\mathrm{D}x = \beta_0 + \beta_1 x + \beta_2 u \tag{7.6}$$

in which $u$ is represented by the step change in the flow into the tank. We expect $\beta_1$ to be negative, so the fixed point for this equation is $(\beta_0 + \beta_2 u)/\beta_1$. In this system, $u$ is stepped from 96.5 down to zero at $t_0 = 66$ min. Formally, the solutions to (7.6) are given by

$$x(t) = x_0 e^{\beta_1 t} - \frac{\beta_0}{\beta_1}(1 - e^{\beta_1}) - 96.5 * \frac{\beta_2}{\beta_1} e^{\beta_1 t}(e^{-\beta_1 (t) \min(t, t_0)} - 1)$$

which can be differentiated with respect to the parameters and $x_0$. We can also obtain this by solving the sensitivity equations

$$\mathrm{D}\begin{pmatrix} x \\ \partial_{\beta_0} x \\ \partial_{\beta_1} x \\ \partial_{\beta_2} x \\ \partial_{x_0} x \end{pmatrix} = \begin{pmatrix} \beta_0 + \beta_1 x + \beta_2 u \\ 1 + \beta_1 \partial_{\beta_0} x \\ x + \beta_1 \partial_{\beta_1} x \\ u + \beta_1 \partial_{\beta_3} x \\ \beta_1 \partial_{x_0} x \end{pmatrix}$$

starting from initial conditions

$$\mathrm{D}\begin{pmatrix} x \\ \partial_{\beta_0} x \\ \partial_{\beta_1} x \\ \partial_{\beta_2} x \\ \partial_{x_0} x \end{pmatrix} = \begin{pmatrix} x_0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

(since this is a linear system, the reader can verify for themselves that we arrive at the same function either way). Figure 7.1 provides the estimates we get by minimizing squared error using these derivatives, along with solutions to the sensitivity equations. The plot in Fig. 7.2 was produced by changing the value of $\beta_1$ away from the optimum values found below.

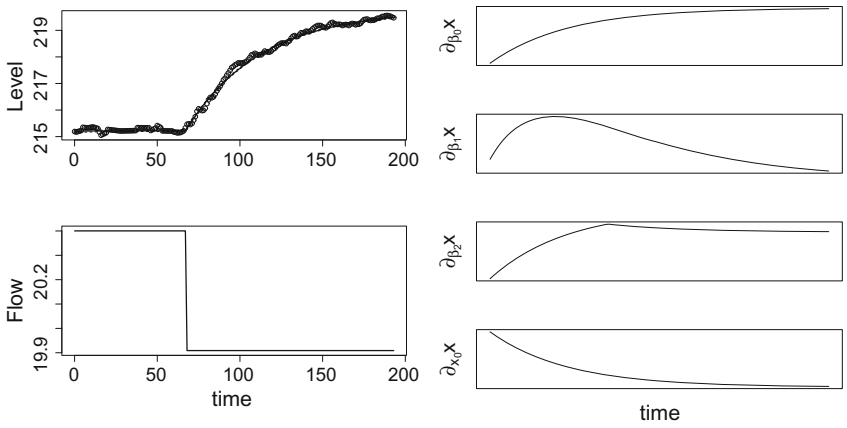


**Fig. 7.1** Fit of the first order forced linear differential equation to the refinery data. *Left* the data and fit at the estimated parameter values $\beta_0 = 9.24,\ \beta_1 = -0.0229, \beta_2 = -0.212,\ x_0 = 215.27$ found by minimizing squared error. The *lower panel* shows the forcing flow for the tray. *Right* solutions to the sensitivity equations with respect to each parameter and $x_0$

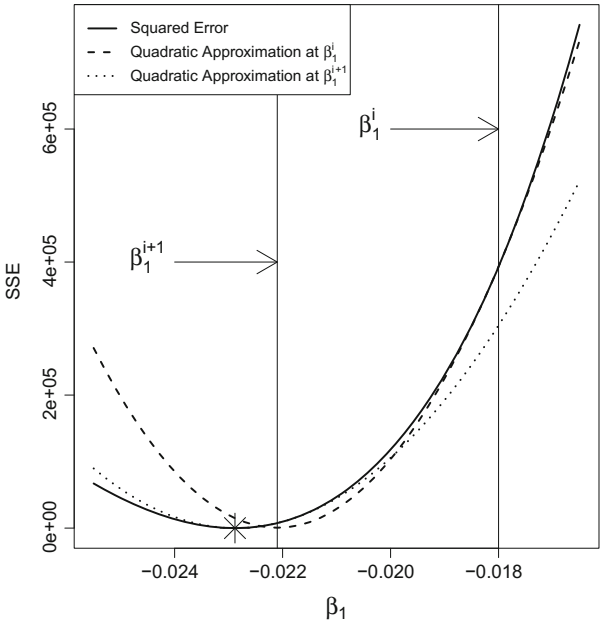


**Fig. 7.2** An illustration of the Newton–Raphson algorithm for estimating $\beta_1$ in the refinery example. The *solid line* gives squared error as $\beta_1$ is varied, with $x_0$, $\beta_0$ and $\beta_2$ held fixed. An initial guess is $\beta_j^0 = -0.0018$ and the *dashed line* provides a quadratic approximation to it. $\beta_1^{i+1}$ is at the minimum of this quadratic approximation and the *dotted line* gives a new approximation to $\texttt{SSE}(\beta_1)$. This successively approaches the point marked with *asterisks*: the actual minimizer

### 7.2.2 Automatic Differentiation

Sensitivity equations calculate derivatives of the theoretical process with respect to parameters and initial conditions. However, they come at the cost of increasing the dimension of the system that must be estimated. Further, they don't account for the numerical approximation of ODEs. We have also seen that employing finite differences directly can be inaccurate.

An alternative means of taking a derivative is to employ *Automatic Differentiation* (AD) (Neidinger 2010). The idea of AD is that, rather than using an analytical expression for the derivative, the function is evaluated in terms of the numerical operations performed by the computer. That is, if we write down an expression for the calculations that the computer makes in approximating $x(t; \boldsymbol{\vartheta})$, for example, then we can apply the chain rule to this expression to obtain its gradient with respect to $\boldsymbol{\vartheta}$. These ideas can be used to obtain sensitivities, but the reader will notice that our real target is $\partial_{\boldsymbol{\vartheta}} \texttt{SSE}(\boldsymbol{\vartheta})$ which was obtained from the sensitivity equations via the chain rule! It therefore makes sense just to obtain $\partial_{\boldsymbol{\vartheta}} \texttt{SSE}(\boldsymbol{\vartheta})$ directly using AD. In Tien and Guckenheimer (2008) they were used to fit observed qualitative features of dynamical systems—a method we will examine in Sect. 7.7.

AD requires additional functionality that tracks numerical operation in code. Packages to perform this exist in `R` and Matlab as well as many other programming environments.

## 7.3 Inference

It is, of course, not enough to simply estimate values of the parameters, we also want to find a way to describe their precision. In this book, we mostly take a frequentist approach to statistics, although the Bayesian framework will be illustrated in Sect. 7.5, below. While we assume a general familiarity with statistical concepts, we will review the basics framework for inference here, as we demonstrate it's application to trajectory matching.

Our quantification of uncertainty starts from the question *if we imagine conducting the experiment and obtaining new data many times, how far, on average, would our parameters be from the truth?*. We have described our data as being generated by (7.3). That is, in a hypothetical repeat data set we would have observations $y_j^*$ that satisfy

$$y_j^* = x(t_j, \boldsymbol{\theta}, x_0) + \varepsilon_j^*$$

where the $\varepsilon_j^*$ are different random values, but nothing else has changed. Since we have obtain estimates $\hat{\boldsymbol{\vartheta}}$ based on minimizing the distance between a solution to the ODE and our (random) data, we can think of these estimates as functions of the data $\hat{\boldsymbol{\vartheta}} = \boldsymbol{\vartheta}(y_1, \ldots, y_n)$ and thus our estimates inherit random variability from the data.

To describe our uncertainty about $\boldsymbol{\vartheta}$, therefore, we need to describe the distribution of the estimate $\hat{\boldsymbol{\vartheta}}$. For this, we appeal to the central limit theorem, which states that the distribution of a sum of independent random variables will tend to the normal distribution as the number of terms increases. Consequently, under appropriate conditions, if the true set of parameters are $\boldsymbol{\vartheta}$ and we estimate $\hat{\boldsymbol{\vartheta}}$ via nonlinear least squares, the parameter estimates are asymptotically normal, that is

$$\hat{\boldsymbol{\vartheta}} \sim N\left(\boldsymbol{\vartheta}, \sigma^2 \left[\mathbf{J}(\boldsymbol{\vartheta})^T \mathbf{J}(\hat{\boldsymbol{\vartheta}})\right]^{-1}\right) \tag{7.7}$$

where $\sigma^2$ is the population variance of the errors $\varepsilon$. This convergence holds under reasonable regularity conditions on $x(\boldsymbol{\vartheta})$ and $\varepsilon_j$. This is not always the case for dynamic systems even if the asymptotic assumptions are correct. However, lacking much alternative, we will largely rely on this result.

In order to apply this calculation, we need an estimate of $\sigma^2$. This is generally obtained from the empirical residuals at the estimated parameters:

$$\hat{\sigma}^2 = \frac{1}{n-p} \sum_{j=1}^{n} (y_j - x(t_j, \hat{\boldsymbol{\vartheta}}))^2. \tag{7.8}$$

Here we have followed the standard practice of dividing by $n - p$, where $p$ the number of estimated parameters, exactly as in linear regression, to allow for minimizing the very term that we are using as an estimate.

Having formed $\hat{\Sigma}_{\hat{\boldsymbol{\vartheta}}} = \hat{\sigma}^2 \left[\mathbf{J}(\hat{\boldsymbol{\vartheta}})^T \mathbf{J}(\hat{\boldsymbol{\vartheta}})\right]^{-1}$ as an estimate for the covariance of $\hat{\boldsymbol{\vartheta}}$, we still need some way of turning this into a statement about the generating parameters $\boldsymbol{\vartheta}_0$. To do this, we observe that the difference between $\hat{\boldsymbol{\vartheta}}$ and $\boldsymbol{\vartheta}_0$ is also normal:

$$\hat{\boldsymbol{\vartheta}} - \boldsymbol{\vartheta} \sim N\left(0, \sigma^2 \left[\mathbf{J}(\boldsymbol{\vartheta})^T \mathbf{J}(\hat{\boldsymbol{\vartheta}})\right]^{-1}\right).$$

Looking at a particular parameter $\vartheta_k$, this means that the estimated $\hat{\vartheta}_k$ is within $1.96\sigma_{\hat{\vartheta}_k}$ units of the true $\vartheta_k$ with 95% probability. Here $\sigma_{\hat{\vartheta}_k}$ is the square root of the $(k, k)$th entry in $\Sigma_{\hat{\boldsymbol{\vartheta}}}$. This is generally translated into a confidence interval

$$[\hat{\vartheta}_k - 1.96\sigma_{\hat{\vartheta}_k},\ \hat{\vartheta}_k + 1.96.\sigma_{\hat{\vartheta}_k}].$$

Since, by the central limit theory, $\hat{\vartheta}_k$ tends to be within $1.96\sigma_{\hat{\vartheta}_k}$ units of $\vartheta_k$ in 95% of hypothetical replicate experiments, this interval covers the true value with the same 0.95 probability. The value 1.96 results from the 0.975 critical value of the standard normal distribution: $P(Z < 1.96) = 0.975$ where probability 0.025 is allowed for being each of too large or too small.

This idea can be extended to confidence ellipses for the whole vector $\boldsymbol{\vartheta}$ or some sub-vector of it, although this is less easy to interpret than giving an interval for each element. More generally, we can replace 1.96 with any quantile $z^\alpha$ of the standard normal to control the probability of a confidence interval missing the true value at $\alpha$. In analogy to linear regression, employ a $t$ distribution with $n - k$ degrees of freedom to account for the variance associated with estimating $\hat{\sigma}^2$. Unlike linear regression, this is not exact, but it does provide a little extra conservatism.

We can also translate these into prediction intervals for $x(t)$ via a Taylor expansion of $x(t; \boldsymbol{\vartheta})$

$$x(t; \hat{\boldsymbol{\vartheta}}) = x(t; \boldsymbol{\vartheta}^0) + \mathbf{J}(t; \hat{\boldsymbol{\vartheta}})(\hat{\boldsymbol{\vartheta}} - \boldsymbol{\vartheta}^0)$$

where $\mathbf{J}(t; \hat{\boldsymbol{\vartheta}})$ is the single-row Jacobian evaluated at only $t$. We now have that

$$\text{var}(x(t; \hat{\boldsymbol{\vartheta}})) \approx \mathbf{J}(t; \hat{\boldsymbol{\vartheta}})^T \Sigma_{\hat{\boldsymbol{\vartheta}}} \mathbf{J}(t; \hat{\boldsymbol{\vartheta}}) = \Sigma_x.$$

Thus if we wish a prediction interval to cover a new point in any one dimension $y_j = x(t_j, \boldsymbol{\vartheta}) + \sigma\varepsilon$ we can employ the interval

$$[x(t; \hat{\boldsymbol{\vartheta}}) - 1.96(\sigma_x + \sigma),\ x(t; \hat{\boldsymbol{\vartheta}}) + 1.96(\sigma_x + \sigma)].$$

Note that these estimates only require solutions to the sensitivity equations at the estimates $\hat{\boldsymbol{\vartheta}}$. These are employed within a Gauss–Newton method and therefore they are essentially free.

## 7.4 Measurements on Multiple Variables

The analysis we have given so far only applies to measurements of only one state variable, which includes situations involving than one measurement at a time point. In this section, we will extend the discussion of (7.3) to $d > 1$ variables and where measurements are available for a only subset $\mathscr{M}$ of these. We must allow for the typical magnitudes of the measured variables to vary, sometimes by orders of magnitude. But even if the variable sizes are comparable, we can expect that some variables can be measured precisely, and other much less so. For example, physical variables like weight, length, temperature and so on are cheaply but precisely measurable, but the abundance species in an ecology, or concentration of components in a chemical reaction are much less so. Finally, it may be that both the location and number of times of observation may vary, so that $n_i$ denotes the number of observations for variable $i$. Therefore, our data model is now

$$y_{ji} = x_i(t_j, \boldsymbol{\vartheta}) + \varepsilon_{ji}, i \in \mathscr{M},\ j = 1, \ldots, n_i, \tag{7.9}$$

where $\varepsilon_{ji}$ has variance $\sigma_i^2$ but the errors are otherwise independent.

This model is still a simplification of many systems: we can expect that observation processes need not have Gaussian errors or that observations of different state variables at the same time point are dependent. Measurements may also correspond to some transformation of $x_i$. For example, if observations are made by sampling a small fraction of a system, we may record $ax_i$ for some fraction $a$. Alternatively, if measurement errors are proportional to $x_i$ it may be appropriate to think of $\log(y_{ji})$ as a measurement of $\log(x_i(t_j))$. In some systems we may also measure the sum of two state variables, or their ratio.

### *7.4.1 Multivariate Gauss–Newton Method*

When minimizing least squares, it will likely make sense to weight each state variable differently to account for different scales and different measurement precision. We therefore modify (7.4) to obtain a weighted sum of squares criterion, where the weight for variable $i$ is $\omega_i$:

$$\mathtt{SSE}(\boldsymbol{\vartheta}) = \sum_{i \in \mathscr{M}} \sum_{j=1}^{n_i} \omega_i \left(y_{ji} - x_i(t_j, \boldsymbol{\vartheta})\right)^2. \tag{7.10}$$

We will see below that choosing $\omega_i = 1/\sigma_i^2$ is a particularly sensible choice, but we will have to estimate the measurement error variance $\sigma_i^2$ from the data before this option can be employed. Where possible, then, variable transformations should be used to equilibrate variables, but otherwise the use of a priori weights $\omega_i$ is strongly encouraged.

Examining the derivatives of $\texttt{SSE}(\boldsymbol{\vartheta})$ we get the following expressions

$$\begin{aligned}\partial_{\boldsymbol{\vartheta}} \texttt{SSE}(\boldsymbol{\vartheta}^0) &= -2\sum_{i=1}^{d} \frac{1}{\sigma_i^2} \partial_{\boldsymbol{\vartheta}} \mathbf{x}_i(\boldsymbol{\vartheta})^T (\mathbf{y}_i - \mathbf{x}_i(\boldsymbol{\vartheta})) \\ &= -2\sum_{i=1}^{d} \frac{1}{\sigma_i^2} \mathbf{J}_i(\boldsymbol{\vartheta})^T (\mathbf{y}_i - \mathbf{x}_i(\boldsymbol{\vartheta}))\end{aligned}$$

where $\mathbf{J}_i(\boldsymbol{\vartheta})$ is the $n_i \times p$ Jacobian matrix for variable $i$. We can write out the second derivative matrix similarly:

$$\begin{aligned}\partial^2_{\boldsymbol{\vartheta}} \texttt{SSE}(\boldsymbol{\vartheta}^0) = -2\sum_{i=1}^{d} \omega_i \mathbf{J}_i(\boldsymbol{\vartheta})^T \mathbf{J}_i(\boldsymbol{\vartheta}) \\ - 2\sum_{i=1}^{d} \omega_i \partial^2_{\boldsymbol{\vartheta}} \mathbf{x}_i(\boldsymbol{\vartheta})^T (\mathbf{y}_i - \mathbf{x}_i(\boldsymbol{\vartheta})).\end{aligned}$$

As was the case for univariate observations, we expect that the second term should be small near the true $\boldsymbol{\vartheta}_0$ and this term can be generally omitted. This results in the following algorithm:

1. Initialize $\boldsymbol{\vartheta} = \boldsymbol{\vartheta}^0$.
2. Do until convergence:

$$\begin{aligned}\mathbf{H}(\boldsymbol{\vartheta}^k) &= \sum_{i=1}^{d} \omega_i \mathbf{J}_i(\boldsymbol{\vartheta}^k)^T \mathbf{J}_i(\boldsymbol{\vartheta}^k) \\ \mathbf{g}(\boldsymbol{\vartheta}^k) &= \sum_{i=1}^{d} \omega_i \mathbf{J}_i(\boldsymbol{\vartheta}^k)^T (\mathbf{y}_i - x_i(\boldsymbol{\vartheta}^k)) \\ \boldsymbol{\vartheta}^{k+1} &= \boldsymbol{\vartheta}^k - \mathbf{H}(\boldsymbol{\vartheta}^k)^{-1} \mathbf{g}(\boldsymbol{\vartheta}^k).\end{aligned}$$

This is iterated until convergence. We can again use sensitivity equations to obtain the derivatives in the Jacobian matrices $\mathbf{J}_i(\boldsymbol{\vartheta})$, and the convergence criteria discussed in Sect. 7.2 also apply.

Our discussion above has implicitly assumed that we have measurements on all the components of $\mathbf{x}$. However, this is often not the case—frequently, components of a system are unmeasured or unmeasurable. Fortunately, our calculations are largely unaffected by this. Unmeasured components are needed to solve the ODE system at

each parameter value, and they may also appear in the sensitivity equations for the measured components (particularly if their initial conditions are being estimated), but they otherwise do not change the methods of estimating parameters or of obtaining confidence intervals.

### 7.4.2 Variable Weighting Using Error Variance

While the methods above will yield consistent estimates $\hat{\boldsymbol{\vartheta}}$ for any choice of $\omega_i$, the choices that we make can affect the precision of our estimates. We rely on the result that

$$\hat{\boldsymbol{\vartheta}} \sim N\left(\boldsymbol{\vartheta}, \Sigma_{\hat{\boldsymbol{\vartheta}}}\right) \tag{7.11}$$

where in this case we have a more complicated expression

$$\Sigma_{\hat{\boldsymbol{\vartheta}}} = \mathbf{H}(\hat{\boldsymbol{\vartheta}})^{-1}\left[\sum_{i=1}^{d} \omega_i^2 \sigma_i^2 \mathbf{J}_i(\hat{\boldsymbol{\vartheta}})^T \mathbf{J}_i(\hat{\boldsymbol{\vartheta}})\right]\mathbf{H}(\hat{\boldsymbol{\vartheta}})^{-1} \tag{7.12}$$

that takes the classic "sandwich" form of a variance estimate. Using this, we can produce the same confidence and prediction intervals as we did above; after estimate $\sigma_i^2$.

This variance is correct for any choice of $\omega_i$. However, if we choose $\omega_i = 1/\sigma_i^2$, we can simplify this expression to the inverse of the information matrix:

$$\mathbf{I}_{\hat{\boldsymbol{\vartheta}}}^{-1} = \left[\sum_{i=1}^{d} \frac{1}{\sigma_i^2} \mathbf{J}_i(\hat{\boldsymbol{\vartheta}})^T \mathbf{J}_i(\hat{\boldsymbol{\vartheta}})\right]^{-1}. \tag{7.13}$$

Besides having a nice form, this is also the smallest variance that we can obtain. It also corresponds to weighting each state variable by the precision of its measurements, which seem pretty natural.

To do this, we have to have the $\sigma_i^2$'s before we estimate $\boldsymbol{\vartheta}$, but need an estimate of $\boldsymbol{\vartheta}$ before we can estimate $\sigma_i^2$. This suggests an iterative scheme outlined below.

### 7.4.3 Estimating $\sigma_i^2$

The weighted squared error objectives above assume that we know $\sigma_i^2$, the variance of the measurement errors in the $y_{ji}$. These can be estimated by (7.8), and we can, as before, employ critical values from a $t_{n-p}$ distribution rather than a normal to account for the uncertainty in these estimates.

We face the problem that for optimal precision, the $\sigma_i^2$ occur in the ordinary least squares estimates above through $\omega_i = 1/\sigma_i^2$, but we must have estimated $\boldsymbol{\vartheta}$ in order to obtain estimates for $\sigma_i^2$. In the case that we only measured one quantity (the case most commonly discussed in texts on nonlinear least squares), the minimizer of $\sigma^2 \mathrm{SSE}(\boldsymbol{\vartheta})$ is exactly the same as that of $\mathrm{SSE}(\boldsymbol{\vartheta})$. We can therefore start with setting $\sigma_i^2 = 1$ and simply re-estimate it once we have $\hat{\boldsymbol{\vartheta}}$.

However, when more than one state variable is measured, the $\sigma_i^2$'s control the relative weight give to each dimension. In this case, the relative values of $\sigma_i$ affect our parameter estimates $\hat{\boldsymbol{\vartheta}}$. A solution to this is to alternate estimating $\boldsymbol{\vartheta}$ and $\sigma_i^2$. This is a co-ordinate descent strategy for maximizing the log likelihood in a normal error model and can be generalized if the $\mathbf{y}_j \sim N(0, \Sigma)$ do not have independent errors, although with some additional work to find estimates of $\boldsymbol{\vartheta}$ (Bates and Watts 1988).

In order to execute this iterative procedure, we need initial starting points for $\sigma_i^2$. We can begin by minimizing the unweighted sum of squares—setting the $\sigma_i^2$ to be all equal—and obtain an initial $\boldsymbol{\vartheta}$ in this manner. Alternatively, when observations are frequent, relative to the speed of the dynamics, successive differences between observations can be used as an estimate:

$$\tilde{\sigma}_i^2 = \frac{1}{n-1} \sum_{i=2}^{n} \left( \frac{y_{ji} - y_{j(i-1)}}{2} \right)^2 .$$

although more sophisticated estimators are available (Eubank and Spiegelman 1990). In this estimate, we have divided the differences by 2 because they include two measurement errors and have not normalized by $n - p$ since the estimate occurs before fitting $\boldsymbol{\vartheta}$. This estimate will be inflated because it also includes the differences $x_i(t_j) - x_i(t_{i-1})$ and we assume that these are small relative to the measurement error. However, it does produce a consistent estimate of $\sigma_i^2$ as the observation times become more closely spaced and can be a good starting point if estimating $\boldsymbol{\vartheta}$ is computationally expensive.

The framework we have describe here is, of course, highly restrictive. Most obviously, we are assuming that two measurements taken at the same time are nonetheless independent. It might make more sense to assume that the *vector* of observations at time $t_j$ is multivariate normal:

$$\mathbf{y}_j \sim N(\mathbf{x}(t_j, \boldsymbol{\vartheta}), \Sigma)$$

for some covariance matrix $\Sigma$ which will then need to be estimated. Rather than go into detail here, we will cover the case of a general fitting criterion below and then provide the specifics for multivariate data.

### 7.4.4 Example: FitzHugh–Nagumo Models

We illustrate this with a simulated example from the FitzHugh–Nagumo equations. These equations, discussed in Chap. 4, have a voltage $V$ and a recovery variable $R$ with the following dynamics:

$$\begin{aligned} \mathrm{D}V &= c(V - V^3/3 + R) \\ \mathrm{D}R &= -(V - a - bR)/c. \end{aligned} \tag{7.14}$$

For our purposes, however, we will treat both states as being measured and add measurement error at intervals of 1/2 from 0 to 20; however, we add measurement errors with standard deviation 0.2 to $V$ whereas we make $R$ less precisely measured with standard deviation 0.5 so that our data are given by

$$y_{jv} = V(t_j) + 0.2\varepsilon_{jv}, \quad y_{jr} = R(t_j) + 0.5\varepsilon_{jr}.$$

The trajectories generating these data and the data themselves are in the first panel of Fig. 7.3.

To fit these data, we start from the true values of the parameters: $(a, b, c) = (0.2, 0.2, 3)$ and initial conditions $(V_0, R_0) = c(-1, 1)$. This is a luxury that is only available for simulated data; we have chosen to do so in order to find the closest local minimum to the truth for reasons that will become evident in Sect. 7.4.5. The first five iterations of the Gauss–Newton algorithm now yield the following values:

| Iteration | $a$ | $b$ | $c$ | $V_0$ | $R_0$ |
|---|---|---|---|---|---|
| 1 | 0.2000 | 0.2000 | 3.0000 | −1.0000 | 1.0000 |
| 2 | 0.1932 | 0.0942 | 3.0066 | −1.0893 | 1.0734 |
| 3 | 0.1925 | 0.0892 | 2.9964 | −1.0763 | 1.0793 |
| 4 | 0.1928 | 0.0834 | 2.9947 | −1.0761 | 1.0820 |
| 5 | 0.1928 | 0.0828 | 2.9946 | −1.0759 | 1.0828 |
| 6 | 0.1928 | 0.0828 | 2.9946 | −1.0759 | 1.0830 |

where we can see that all these parameters have very quickly stabilized to at least three decimal places; though parameter $b$ has moved a long way from its true value, due to being poorly defined by the data.

When we estimate the error standard deviations, we obtain 0.19 and 0.49 and applying the variance estimate in (7.12) gives us the following confidence intervals which all cover the true parameters.

| | Lower | Upper |
|---|---|---|
| $a$ | 0.154 | 0.232 |
| $b$ | −0.232 | 0.398 |
| $c$ | 2.918 | 3.071 |
| $V_0$ | −1.231 | −0.921 |
| $R_0$ | 0.890 | 1.276 |

However, giving equal weight to each of the $y_v$ and $y_r$ is clearly inefficient since we know that our measurements of $y_v$ have considerably greater precision. If we re-weight the data according to our estimated variance, we get new parameter estimates, and then new estimated variances. Iterating this a few times, however stabilizes our estimated measurement standard deviations to 0.18 and 0.50. When using these variances we obtain parameter estimates $(a, b, c) = (0.179, 0.258, 2.977)$ and initial conditions $(V_0, R_0) = (-1.076, 1.115)$. Applying the variance estimate (7.13) we obtain intervals which are considerably narrower than those we obtained from our first estimates. Note, too, that the estimate of $b$ as well as its confidence limits have substantially changed.

| | Lower | Upper |
|---|---|---|
| $a$ | 0.147 | 0.212 |
| $b$ | 0.048 | 0.469 |
| $c$ | 2.899 | 3.056 |
| $v_0$ | −1.204 | −0.947 |
| $w_0$ | 0.968 | 1.262 |

### 7.4.5 *Practical Problems: Local Minima*

One of the practical difficulties in fitting ODEs to data is that the shapes in the trajectories $x_i(t, \boldsymbol{\theta})$ can change with $\boldsymbol{\theta}$ in ways that make the squared error surface $\texttt{SSE}(\boldsymbol{\theta})$ quite complex. While we seek an over-all minimum, the surface can have many local minima in which a Gauss–Newton method can get trapped. This can result in poor fits to the data, and require alternative optimization methods which can significantly increase the computational burden involve.

Figure 7.3 provides an example of this. It was generated by solving the FitzHugh–Nagumo equations (7.14) used in Sect. 7.4.4. In this case, we obtained solutions at every 0.05 time point without adding noise. We then plotted the the squared error between this solution and solutions obtained for nearby values of $a$ and $b$. Two surfaces have been provided here, the left gives the results obtained by solving the FitzHugh–Nagumo equations with the `ode45` function in Matlab while the right is what results from repeating the experiment using the `lsoda` function in R. Here, both plots indicate breaks in the surface that correspond to bifurcation points in which a stable limit cycle becomes a fixed point; the various behaviors of these estimates are given in Fig. 7.4. However the left surface has considerably more complexity, which we speculate to be artifacts of the numerical settings in `ode45`. Thus both the ODE solver and initial guesses can have complex effects on the resulting estimates.

These types of artifacts create problems for gradient-based optimizers because they find *local* optimum; often (although certainly not always) one that is close to the initial guess. We can try many different starting values, and this is a common strategy. But, while this improves the chances of finding a *global* optimum, it is no guarantee.

There are also a collection of alternative methods for searching for the minimum of a function over parameters. Many of these methods are computationally challenging; simulated annealing, which is designed to overcome complex objective surfaces, can require solving the ODE thousands of times. The Bayesian methods in Sect. 7.5 take a quite different approach to statistical inference, but can be classed among these numerical methods. Some methods also try to avoid solving the ODE exactly. This strategy is outlined in Sect. 7.6, and we give a more comprehensive treatment of this idea in Chaps. 9 and 10.

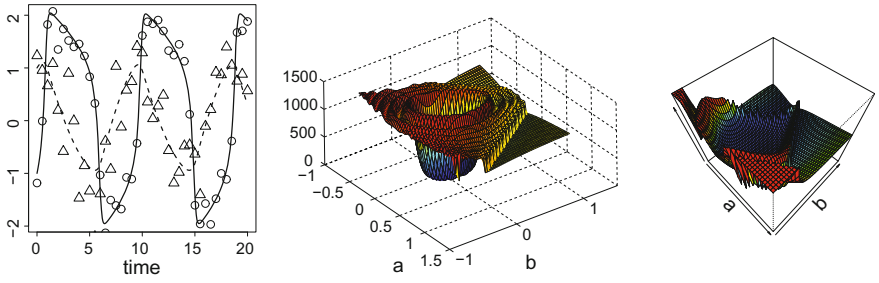


**Fig. 7.3** Squared error between solutions to the FitzHugh–Nagumo equations as parameters $a$ and $b$ are changed. *Left plot* gives trajectories at the parameters $a = b = 0.2$ along with simulated data that we will fit later: *circles* and *solid lines* give observations and true trajectories for $v$ while *triangles* and *dashed lines* give $r$. The center surface results from using Matlab's `ode45` to calculate the difference between these trajectories and those obtained at different values of $a$ and $b$. The *right plot* is the result of the same calculation using the R function `lsoda`. Extra detail in the Matlab surface results from the particular numerical settings use to approximate solutions

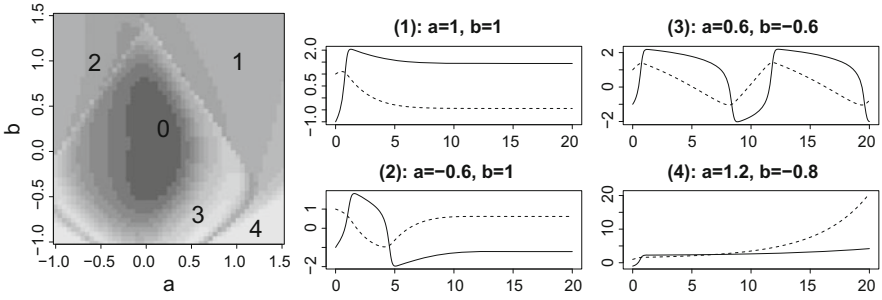


**Fig. 7.4** An explanation for the complex squared error surface in Fig. 7.3. *Left plot* provides a grey-scale representation of the *right-hand plot* in Fig. 7.3. The *numbered points* represent parameters values of $a$ and $b$ plotted in the corresponding plots in the *center* and on the *right*. The point labeled '0' gives the target values corresponding to the trajectories plotted in Fig. 7.3

### 7.4.6 Initial Parameter Values for the Chemostat Data

Given the difficult optimization problems illustrated above, it is important to start at reasonable parameter values. Finding good initial guesses is something of a dark art, made darker by the difficulty of publishing good techniques for this. The *gradient matching* methods in Chap. 8 have sometimes been justified as good initial guesses for trajectory matching, although we believe there are better justifications; gradient matching also cannot be employed on all systems.

In general, finding good initial guesses requires some intelligence on the part of the user, but some basic manipulations can be helpful. We begin by observing that most systems of interest have candidate parameter values that at least give the right qualitative dynamics. These can still yield trajectories that are a long way from the observed data, but we can often perform manual manipulations to obtain better estimates.

We illustrate this with some data from an ecological experiment described in Yoshida et al. (2007). These data come from a chemostat experiment in which algae are grown in a nitrogen-limited medium and serve as a prey species to rotifers, which are near-microscopic animals. The original purpose of this article was to illustrate the breakdown of ecological dynamics due to algal evolution, but we will focus on the initial periods where classical predator-prey dynamics appears to hold and fit these with a Rosenzweig-MacArthur ODE model:

$$\begin{aligned} \mathrm{D}C &= \rho C(\kappa - C) - \frac{\gamma \beta CB}{\chi + C} \\ \mathrm{D}B &= \frac{\beta CB}{\chi + C} - \delta B. \end{aligned} \tag{7.15}$$

In order to fit these, we need reasonable initial conditions. Here we begin by plotting both our data and solutions of the ODE at the parameters used in Chap. 8; see the left-hand panel in Fig. 7.5. These are carefully plotted on separate graphs (and on the logarithmic scale) because it is clear that both the amplitude and the period of the oscillations do not match. If we simply start from these parameters and attempt to fit the data, we find very poor fits. Many optimization methods also return errors as parameters are forced into regions where numerical solvers break down.

Observing the structure of (7.15), however, we see that there are ways to directly manipulate the parameters to at least obtain the right period and amplitude. First of all, we see that the period at our initial parameters is around 50 days, which we would like to bring down to a little over 10 days as observed in the data. To do this, we make use of the fact that

$$\mathrm{D}x(at) = a[\mathrm{D}x](at)$$

so that multiplying each equation in (7.15) by $a$ will "speed up" the period of the system by $a$ units. Here, we chose $a$ to be 4.54 and observe that this is equivalent to multiplying each of $\rho$, $\beta$ and $\delta$ by $a$.

Second, we observe that we can re-scale variables; if we set $C^* = a_1 C, B^* = a_2 B$, then we can again apply the chain rule to (7.15) to discover that $C^*$ and $B^*$ satisfy (7.15) with new parameters $\rho^* = \rho/a_1$, $\kappa^* = a_1\kappa$, $\chi^* = a_1\chi$ and $\gamma^* = (a_1/a_2)\gamma$. In order to estimate $a_1$ and $b_1$, we in fact added these as parameters. We minimized squared error on the log scale; representing our data as $(C_j, B_j)$ at time $t_j$, we transformed the parameters to have approximately the correct timing as above and then minimized

$$\sum_{j=1}^{30} (\log C_j - \log(a_1 C(t_j, \boldsymbol{\theta})))^2 + (\log B_j - \log(a_2 B(t_j, \boldsymbol{\theta})))^2$$

for all of $a_1$, $a_2$ and $\boldsymbol{\theta}$. While this minimization problem is not well defined (as we have seen above, we can change $a_1$ and $a_2$ arbitrarily and compensate by a change the $\boldsymbol{\theta}$) it is more flexible than starting from an initial guess.

Following this, we we updated the parameters as above to get rid of $a_1$ and $a_2$ and simply minimized squared error (on the log scale). Both minimizations were carried out using a Nelder–Meade method we then began from these solutions and used a Newton–Raphson-like method to find a convergent solution. This is plotted in the right-hand panel of Fig. 7.5.

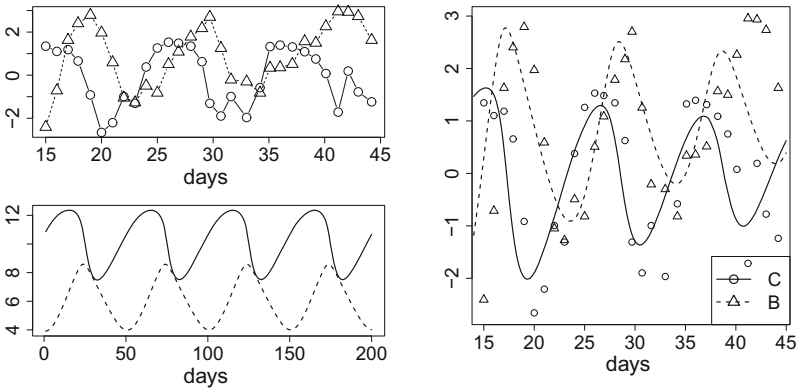


**Fig. 7.5** Fits of chemostat data to the Rosenzweig–MacArthur model. *Left* a comparison of the logged data and solutions to (7.15) with initial parameters before some basic transformation. *Right* data and solutions at the finalized objective function

We want to note that this particular sequence of optimizations and parameter transformations was obtained by trial and error. We found that using a Newton-type method at any earlier stages quickly resulted in errors, for example, that motivated trying alternative optimizers. A different model, or different data, might require quite different techniques; good fits may be found by optimizing early, on the other hand, it may be necessary to refine initial parameter guesses even further. However, there are some simple tricks that can be employed:

1. Shift the inter-time intervals of equations by multiplying equations by a constant. Often this translates to changes in parameter values.
2. Find a way to change state variables that can be mimicked by changes in parameters. Re-scaling state variables, as above, almost always has this property; in some cases shifting the level of the state variable will also be possible.
3. In some cases, good initial conditions can be found from data directly. In others, especially where there is stable cyclic behavior, solving the system for a length of time and then choosing initial conditions from an appropriate point in the cycle can be helpful.
4. Optimizing over only some parameters to begin with can help get to better values. We might, for example, have kept $\kappa$ and $\chi$ fixed and found good values for the other parameters first.
5. Alternatively, allowing some flexibility in the observation process—as in our estimation $a_1$ and $a_2$ above, can also be useful, even if we will ultimately re-absorb that into our parameter estimates.

All of these should be tried intelligently with an eye to the type of system under study, the behavior that is expected, and the observations that are available.

### *7.4.7 Identifiability*

Parameter identifiability is an important issue in fitting ODE models to data. The mechanistic perspective used to develop ODEs often yields an expression in terms of parameters that cannot be well-determined from data. A useful example of this is re-scaling state variables in the Rosenzweig-MacArthur model as described above; it's reasonable to assume that we sample some percentage $\alpha$ of an ecosystem, so that our observations are of the form $(\alpha C(t_j) + \varepsilon_c, \alpha B(t_j) + \varepsilon_B)$. However, we cannot estimate $\alpha$ and all the other parameters jointly, even though they all have interpretable meanings.

In some cases, this type of confounding is straightforward to detect. For example, parameters on the right hand side of the ODE may always appear together as a sum or a ratio. If the right hand side of the ODE doesn't change with some particular change of parameters, there is a problem of identifiability. In other cases, it can be more subtle, and often a result of the observation process and can also involve initial conditions. As a variation on the ecological example above, we can consider the Susceptible-Infected-Recovered ODE model for epidemics:

$$\begin{aligned} \mathrm{D}S &= -\alpha SI \\ \mathrm{D}I &= \alpha SI - \beta I \\ \mathrm{D}R &= \beta I. \end{aligned}$$

Notice here that there are only two parameters $\alpha$ and $\beta$, used in multiple equations, which should make them easier to identify. However, in most epidemic models, we

only measure the number of infected individuals and usually we only measure some percentage $p$ of them; $S$ and $R$ are unmeasured states. Let's assume that we observe $I^* = pI$ perfectly and at all time points, then we can write-out the same equations in terms of $I^*$ (dropping $R$ since we don't observe it, anyway):

$$\begin{aligned} \mathrm{D}S &= -\frac{\alpha}{p} S I^* \\ \mathrm{D}I^* &= \alpha S I^* - \frac{\beta}{p} I^*. \end{aligned}$$

Had we observed both $S$ and $I$, we should still be able to identify all three of $\alpha$, $\beta$ and $p$. However, we generally don't get to measure the Susceptible population and so would need to estimate at least its initial values, $S_0$ and we cannot distinguish this effect from others. To see that, let's write $S = S_0 S^*$, so that $S^*$ starts from the value 1, then we can write out an equation for $(S^*, I^*)$:

$$\begin{aligned} \mathrm{D}S^* &= -\frac{\alpha}{p} S^* I^* \\ \mathrm{D}I^* &= \frac{\alpha}{S_0} \alpha S^* I^* - \frac{\beta}{p} I^* \end{aligned} \tag{7.16}$$

where it should be clear that we cannot distinguish all of $\alpha$, $\beta$, $p$ and $S_0$ since they jointly define only three rates. In this case, however, we often have an estimate of the over-all at-risk population size to use as $S_0$.

For a general ODE model, it is not necessarily easy to work out whether all the parameters and initial conditions can be determined by the data. There has been some attention given to this problem, but many of the formal methods suggested involve rather cumbersome applications. A good review can be found in Miao et al. (2011). We join this paper in advocating some simple measures to identify whether parameters can be determined from a finite set of noisy data in practice:

1. Perform a simulation experiment to identify the precision of the parameter estimates.
2. Solve the sensitivity equations (at a plausible set of parameters) and test how linearly dependent they are.
3. Perform an eigen decomposition of the Hessian $\partial^2_{\boldsymbol{\vartheta}} l(\hat{\boldsymbol{\vartheta}})$; very small or zero eigenvalues may indicate that some set of parameters are poorly identified.

We can illustrate this with the SIR model (7.16) above, where we assume we only measure $I^*$ but had not recognized the problem of estimating $S_0$. Figure 7.6 presents the sensitivity of the $I^*$ equation in (7.16) with respect to initial conditions and parameters, using $\alpha = 0.005$, $\beta = 0.02$ and $p = 0.2$. Here we see strongly similar shapes in the sensitivity to $\alpha$, $p$ and $S_0$. If we perform a principal components analysis on these sensitivities, the final component has zero eigenvalue with a vector that contrasts the equations for $\alpha$, $p$ and $S_0$, revealing the analysis we presented above. If we remove the sensitivity to $S_0$—assuming it known and fixed when we estimate

parameters—the smallest component contrasts the sensitivities to $\beta$ and $p$ but is still greater than zero.

In addition to strict lack of identifiability, some parameters or parameter combinations may be only weakly determined by the data. This can be due to poor parameter scaling, as well as to a lack of sensitivity of the observed states to them. In early work, we discovered that the CSTR equations described in Chap. 4 have terms of the form $\kappa \exp(\tau x)$ where the effects of parameters $\kappa$ and $\tau$ are nearly indistinguishable unless $x$ covers a very broad range of values (Ramsay et al. 2007).

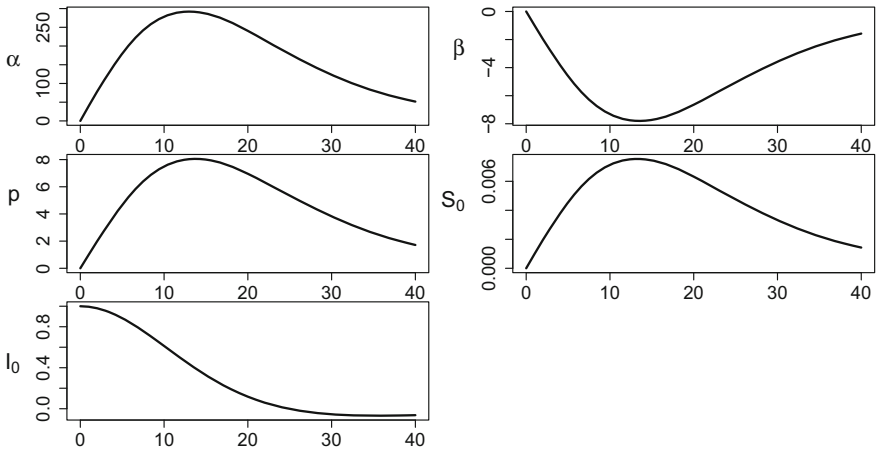


**Fig. 7.6** The derivative of $I^*(t)$ with respect to $\alpha, \beta, p, S_0$ and $I_0^*$ in the model (7.16). The sensitivities of $\alpha$, $\beta$ and $S_0$ are linearly dependent, meaning that if only $I^*$ is observed, these three parameters cannot all be estimated from data

These effects can be even stronger in high dimensional systems such as cell signaling or protein networks, even when the dynamics are linear. Brown et al. (2004) and later Gutenkunst et al. (2007) observed the phenomenon that in these systems, the eigenvalues of the Hessian matrix $\mathbf{H}(\boldsymbol{\vartheta})$ tend to show exponential decay, meaning that the data are very informative about some combinations of parameter values, while others are nearly unidentifiable from the observations. This also implies that, while it is important to pinpoint the values of some parameters, there are others that have very little impact on the outcomes we are interested in—whether that be observational data, or the system's reaction to inputs. It also means in that the squared error surface tends to have "gorges"—relatively thin but long regions in parameter space where good agreement with the data can be found. Such systems have been termed "sloppy" and have inspired specific, often Bayesian, methods to tackle them; see Girolami and Calderhead (2011), Transtrum et al. (2011).

## 7.5 Bayesian Methods and Markov Chain Monte Carlo

The methods that we have discussed so far are frequentist in conception—they focus on the question of how different our estimates could be if we imagine re-collecting the data with new random errors. Bayesian methods have a very different goal of updating our uncertainty about $\boldsymbol{\vartheta}$ in the light of new evidence. A great deal of the literature on nonlinear dynamical systems takes a Bayesian perspective, partly because the computational challenges of working with differential equations lend themselves to this. Here we will briefly review the framework for Bayesian estimation and go over some numerical techniques as applied to ordinary differential equation models. We present only the simplest of approaches here, but point to more sophisticated means of sampling from the posterior which have been developed specifically for ODE models.

The Bayesian framework starts with prior distribution $\pi(\boldsymbol{\vartheta})$ over the space of possible parameter values. This distribution is intended to represent our state of knowledge or belief about the values of $\boldsymbol{\vartheta}$ *before* observing data and may be derived from previous experiments, experience or theory; or may be based on a subjective understanding of the likely parameter values. This prior is the key to Bayesian analysis since we can now combine it with data to update our beliefs. Specifically, our model specification can now be extended to

$$\boldsymbol{\vartheta} \sim \pi(\boldsymbol{\vartheta}),\ y_{ji}|\boldsymbol{\vartheta} \sim N(\mathbf{x}(t_j, \boldsymbol{\vartheta}), \sigma_i^2).$$

Here we have written $y_{ij}|\boldsymbol{\vartheta}$ to indicate a conditional distribution: for a particular value of $\boldsymbol{\vartheta}$, we obtain a specific distribution for $y_{ij}$. Putting all the data together, the conditional density of the observation matrix $Y$ given $\boldsymbol{\vartheta}$ can now be seen to be numerically equal to the likelihood

$$\log P(Y|\boldsymbol{\vartheta}) = l(\boldsymbol{\theta}).$$

In this case

$$l(\boldsymbol{\vartheta}) = \sum_{i=1}^{d}\sum_{j=1}^{2} \frac{1}{\sqrt{2\pi}\sigma_i}(y_{ji} - x(t_j, \boldsymbol{\vartheta}))^2 - \frac{1}{2}\log 2\pi\sigma_i^2.$$

The central concept in Bayesian statistics is that given $\pi(\boldsymbol{\vartheta})$ and $P(Y|\boldsymbol{\vartheta})$, we can apply Bayes theorem to recover the a posteriori density of $\boldsymbol{\vartheta}$ conditional on $Y$:

$$P(\boldsymbol{\vartheta}|Y) = K^{-1}P(Y|\boldsymbol{\vartheta})\pi(\boldsymbol{\vartheta}) = K^{-1}e^{l(\boldsymbol{\vartheta})}\pi(\boldsymbol{\vartheta}), \tag{7.17}$$

where $K$ is a normalizing constant so that $P(\boldsymbol{\vartheta}|Y)$ integrates to 1. This represents the update to our knowledge of $\boldsymbol{\vartheta}$ in the light of evidence from $Y$. Interpretation uses a probability distribution—first $\pi(\boldsymbol{\vartheta})$ and then $P(\boldsymbol{\vartheta}|Y)$—to encapsulate our state of understanding before and after the experimental evidence. There is no frequentist

sense of *this is the distribution of parameters we would get from replicate experiments*. However, Bayesian methods can be shown to have good statistical properties from a frequentist perspective.

Technically, the posterior distribution gives us a complete description of our state of knowledge for $\boldsymbol{\vartheta}$, but it is still often convenient to summarize it. A single best estimate for parameters is usually given by the expected value of $\boldsymbol{\vartheta}$:

$$\tilde{\boldsymbol{\vartheta}} = \int \boldsymbol{\vartheta} P(\boldsymbol{\vartheta}|Y) d\boldsymbol{\vartheta},$$

otherwise referred to as the expected *a posteriori* estimate (EAP). To describe our remaining uncertainty about a particular $\boldsymbol{\vartheta}_k$, we can examine a *credible interval*: the central $(1-\alpha)\%$ region of its distribution. It's possible to define credible regions, but as with frequentist summaries, it's generally more interpretable just to examine one $\boldsymbol{\vartheta}_k$ at a time. From a frequentist perspective, both the EAP and credible intervals perform equivalently to estimates and confidence intervals when $n$ is large.

The Bayesian framework is appealing at a high level. However, neither credible intervals nor point estimates can be calculated without knowing the value of $K$; this is algebraically intractable in all but quite specialized settings. Instead, we use Markov Chain Monte Carlo methods to produce a sample of points $\vartheta_1, \ldots, \vartheta_M$ from the posterior distribution $P(\boldsymbol{\vartheta}|Y)$. We can then use these samples to obtain credible intervals and EAPs.

It is not our intention in this book to provide a comprehensive overview of Bayesian statistics and methods and the reader unfamiliar with these methods is referred to Albert (2009) and Brooks et al. (2011), among many others, for a practical guide. The central idea is to create a Markov chain whose stationary distribution is the desired $P(\boldsymbol{\vartheta}|Y)$. We can then run the Markov chain and obtain samples from the chain.

A simple version of MCMC takes a two-step form. First, we propose a move from $\boldsymbol{\vartheta}^k$ to $\boldsymbol{\vartheta}^{k+1}$, we then accept that step and make a move, or reject it and stay at $\boldsymbol{\vartheta}^k$. This algorithm can be given as:

1. Choose $\boldsymbol{\vartheta}^*$ with density $q(\boldsymbol{\vartheta}^*|\boldsymbol{\vartheta}^k)$ that depends on the current $\boldsymbol{\vartheta}^k$.
2. Calculate
$$\alpha = \frac{P(Y|\boldsymbol{\vartheta}^*)}{P(Y|\boldsymbol{\vartheta}^k)} \frac{\pi(\boldsymbol{\vartheta}^*)}{\pi(\boldsymbol{\vartheta}^k)} \frac{q(\boldsymbol{\vartheta}^k|\boldsymbol{\vartheta}^*)}{q(\boldsymbol{\vartheta}^*|\boldsymbol{\vartheta}^k)}. \tag{7.18}$$
3. Set $\boldsymbol{\vartheta}^{k+1} = \boldsymbol{\vartheta}^*$ with probability $\alpha$, otherwise $\boldsymbol{\vartheta}^{k+1} = \boldsymbol{\vartheta}^k$.

In the acceptance probability (7.18), the first ratio measures how the density of $\boldsymbol{\vartheta}^{k+1}$ to $\boldsymbol{\vartheta}^k$, while the second balances the the probability of moving from $\boldsymbol{\vartheta}^k$ to $\boldsymbol{\vartheta}^{k+1}$ with that of moving in the other direction. This turns out to be exactly what is required to achieve the desired stationary distribution (Albert 2009, see many references including).

This leaves us with the problem of choosing $q(\boldsymbol{\vartheta}^*|\boldsymbol{\vartheta}^k)$. The most common default choice is a symmetric random walk with normally distributed steps:

$$\boldsymbol{\vartheta}^* = \boldsymbol{\vartheta}^k + \boldsymbol{\varepsilon},\ \boldsymbol{\varepsilon} \sim N(0, \tau^2 \mathbf{I}).$$

This has the advantage that the second ratio in (7.18) becomes 1. This still requires the variance of the steps we take, $\tau^2$, be selected so as to allow us to move fairly frequently, but so that the correlation in the resulting values not be too large; usually we target having an average value of $\alpha$ to be around 0.3.

To make this concrete, we re-examine the FitzHugh–Nagumo data we generated above. Here, our parameters are $a, b, c, V_0, R_0$ as well as the measurement variances $\sigma_V^2$ and $\sigma_R^2$. Because the last two of these parameters are positive, we will in fact use their logarithms which we write as $s_V$ and $s_R$ resulting in a parameter vector $\boldsymbol{\vartheta} = (a, b, c, V_0, R_0, s_V, s_R)$. We give each element a $N(0, 10)$ prior, chosen to be uninformative. This gives us a model for the joint probability of our observations and parameters as

$$\begin{aligned} &P(y_{1V}, \ldots, y_{41V}, y_{1R}, \ldots, y_{41R}, \boldsymbol{\vartheta}) \\ &\qquad = \prod_{j=1}^{41} \left[\phi(y_{jV}, V(t_j; \boldsymbol{\vartheta}), e^{s_V})\phi(y_{jR}, R(t_j; \boldsymbol{\vartheta}), e^{s_R})\right] \prod_{k=1}^{7} \phi(\vartheta_k; 0, 10) \end{aligned}$$

where $\phi(x; \mu, \sigma^2)$ is the normal density with mean $\mu$ and variance $\sigma^2$.

We selected our proposal distribution for $\boldsymbol{\vartheta}$ to be Gaussian with independent dimensions. We determined the size of the steps by first maximizing

$$\log \ell(\boldsymbol{\vartheta}) = \log P(y_{V1}, \ldots, y_{V41}, y_{R1}, \ldots, y_{R41}, \boldsymbol{\vartheta})$$

over $\boldsymbol{\vartheta}$ and obtaining the Hessian $H$ at the optimum (here we benefitted from knowing the true parameters when we maximized, see comments above). We then used proposed moves with standard deviations of $0.5\sqrt{\text{diag}(H)}$—about 1/2 the sampling standard deviation from a Gaussian approximation. When making a proposed move from $\boldsymbol{\vartheta}$ to $\boldsymbol{\vartheta}'$ we calculated the acceptance probability as

$$\begin{aligned} P(\boldsymbol{\vartheta}, \boldsymbol{\vartheta}') = \exp[&\log P(y_{1V}, \ldots, y_{41V}, y_{1R}, \ldots, y_{41R}, \boldsymbol{\vartheta}') \\ &- \log P(y_{1V}, \ldots, y_{41V}, y_{1R}, \ldots, y_{41R}, \boldsymbol{\vartheta})] \end{aligned}$$

where calculating the probabilities on the log scale helps to stabilize the numerical calculation of their values.

We started from the true parameter values and ran MCMC for 5000 steps (this includes the 79% of times that we did not accept the proposed move). We have produced scatter plots of the resulting values in Fig. 7.7 along with a plot of the value of the log posterior as the chain ran.

There is some indication both of autocorrelation and that it took a little while for the Markov chain to reach a stationary distribution. We therefore removed the first 1,000 values that we generated, and we took every 5th value of what remains. That left us with 800 "samples" of the $P(\boldsymbol{\vartheta}|Y)$ distribution. We can now approximate the

EAP by averaging the 800 samples from $\vartheta_k|Y$. We can also directly compute credible intervals either by obtaining a variance from these samples and using Normal critical values as in Sect. 7.3, or by simply looking at the 0.025 and 0.095 quantiles in these samples.

These estimates and intervals are recorded in Table 7.1; note that we have re-exponentiated $s_V$ and $s_R$ to report (asymmetric) intervals for noise standard deviations. Here we see fairly good agreement between Bayesian intervals and the frequentist intervals computed in Sect. 7.4.4 and that these all cover the true values. This is to be expected when using a moderate amount of data and wide prior distributions; (Lele et al. 2010) used this correspondence to suggest ways to obtain exact frequentist intervals from MCMC results by "cloning" data sets to overwhelm the influence of the prior.

Nominally, MCMC results should converge from any starting value, if allowed to run for long enough even with a posterior that has multiple modes. We could imagine that they will eventually propose a step that goes from one local optimum to another, such as those seen in Fig. 7.3. However this could take a very long time. When we attempted this starting from all the true parameters but with $a$ and $b$ set to $(-0.6, 1)$ (point number 3 in Fig. 7.3) we ran into parameter values where our ODE solver

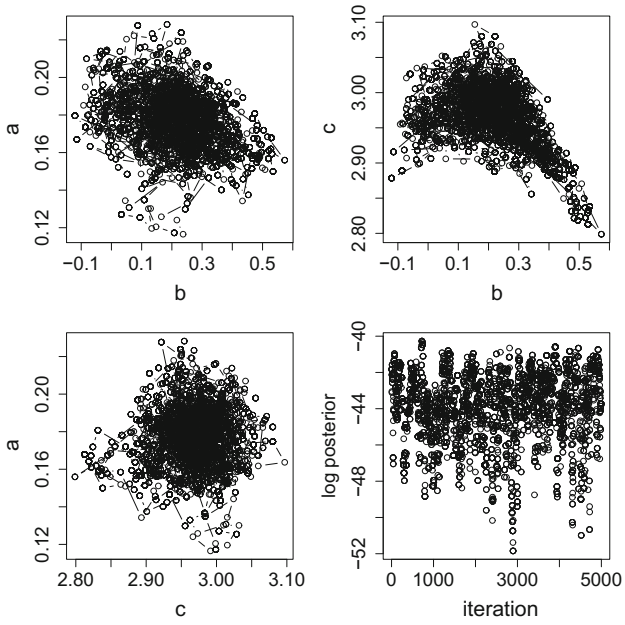


**Fig. 7.7** MCMC results for estimating parameters in the FitzHugh–Nagumo equations from noisy data. Graphs give scatter-pots of the moves in parameters $a$, $b$ and $c$ (we also included initial conditions and measurement error standard deviations in the chain). *Bottom right plot* gives the value of the log posterior over the 5000 iterations where we see that it takes some time to reach the range of posterior values

**Table 7.1** Bayesian expected a posteriori estimates and 95% credible intervals from MCMC results for the example in Sect. 7.4.4. These were obtained after running a Markov chain for 5000, steps, discarding the first 1000 and then taking every fifth value in the chain

| | $a$ | $b$ | $c$ | $V_0$ | $R_0$ | $\sigma_V^2$ | $\sigma_R^2$ |
|---|---|---|---|---|---|---|---|
| Truth | 0.200 | 0.200 | 3.000 | −1.000 | 1.000 | 0.200 | 0.500 |
| EAP | 0.178 | 0.212 | 2.965 | −1.099 | 1.133 | 0.197 | 0.510 |
| Lower | 0.146 | −0.022 | 2.871 | −1.244 | 0.983 | 0.161 | 0.405 |
| Upper | 0.210 | 0.463 | 3.044 | −0.966 | 1.275 | 0.244 | 0.655 |

produced errors before we found the optimum around the true values. The user is advised to be aware of these possibilities and to pay attention to starting values even in this case.

There are mathematical guarantees that the sequence $\boldsymbol{\vartheta}^1, \boldsymbol{\vartheta}^2, \ldots$ will eventually have the desired distribution, but not on how long it takes to get here. There are a number of diagnostics for whether the Markov chain has converged. This is a much larger topic than we can cover here; the reader is referred to Brooks et al. (2011).

The random-walk Metropolis algorithm that we have employed in these examples is simple to implement but can perform poorly, particularly when $P(\boldsymbol{\vartheta}|Y)$ has many local modes where it can get caught for many iterations. There is a large literature on MCMC methods to deal with this which we won't review in detail here. They include *tempering* methods which run parallel MCMC chains on increasingly smoothed-out versions of $P(\boldsymbol{\vartheta}|Y)$ so that the smoother chains can explore the space better (see Earl and Deem 2005, for an overview). Campbell and Steele (2012) employ methods similar to this in conjunction with the profiling methods described in Chap. 9. Others attempt to use the geometry of $P(\boldsymbol{\vartheta}|Y)$ to improve our ability to sample. Girolami and Calderhead (2011), Transtrum et al. (2011) both treat this problem with reference to nonlinear dynamics explicitly.

## 7.6 Constrained Optimization: Multiple Shooting and Collocation Methods

In addition to global optimization, a number of methods make use of the structure of collocation methods, discussed in Chap. 5, to overcome local structure in the likelihood surface. Recall that collocation methods represent $\mathbf{x}(t; \boldsymbol{\vartheta})$ via a basis expansion. The coefficients of this expansion are chosen so that the ODE is satisfied at a set of collocation points.

This representation has been used to combine solving the differential equation and fitting parameters to data into single constrained optimization. We choose parameters $\boldsymbol{\vartheta}$ as well as the coefficients of the basis expansion $\mathbf{c}_1, \ldots, \mathbf{c}_d$ to satisfy:

$$(\boldsymbol{\vartheta}, \mathbf{c}_1, \ldots, \mathbf{c}_d) = \operatorname{argmin} \sum_{i=1}^{d} \sum_{j=1}^{n} \left(y_{ji} - \boldsymbol{\phi}(t)^T \mathbf{c}_i\right)^2$$

subject to

$$\begin{aligned} \boldsymbol{\phi}(t_0)\mathbf{C} &= \mathbf{x}_0 \\ \mathrm{D}\boldsymbol{\phi}(t_l)\mathbf{C} &= \mathbf{f}(t_l, \boldsymbol{\phi}(t_l)\mathbf{C}, \boldsymbol{\theta}),\ l = 1, \ldots, K-1. \end{aligned} \tag{7.19}$$

Here the constraints require the basis-expansion to satisfy the differential equation—up to the approximation in the collocation equations (7.19)—while the objective provides the squared error. Note here that although $\boldsymbol{\vartheta}$ does not appear in the objective, its inclusion in the constraints results in these being estimated as well.

The advantage of this formulation is that it is given explicitly in terms of both coefficients for the basis representation of $\mathbf{x}$ and the parameters. In particular, the constraints (7.19) need not be satisfied while the algorithm is running. This means the algorithm has less tendency to get caught in local optima of the objective surface. This claim is difficult to justify mathematically. However, we can gain some insight by converting the problem into its equivalent Lagrange multipliers form. We extend the objective function to include a weighted term for how far each constraint is from being satisfied:

$$\begin{aligned} \Lambda(\mathbf{C}, \boldsymbol{\theta}, \boldsymbol{\lambda}) = &\sum_{i=1}^{d} \sum_{j=1}^{n} \left(y_{ji} - \mathbf{c}_i' \boldsymbol{\phi}(t)\right)^2 \\ &+ \boldsymbol{\lambda}_0^T [\boldsymbol{\phi}(t_0)\mathbf{C} - \mathbf{x}_0] + \sum_{l=1}^{K-1} \boldsymbol{\lambda}_l^T [\mathrm{D}\boldsymbol{\phi}(t_l)\mathbf{C} - \mathbf{f}(t_l, \boldsymbol{\phi}(t_l)\mathbf{C}, \boldsymbol{\theta})]. \end{aligned} \tag{7.20}$$

Here $\boldsymbol{\lambda}_0, \ldots, \boldsymbol{\lambda}_{K-1}$ are weighted values of the constraint called Lagrange multipliers. We now set the gradient with respect to all of $\mathbf{C}$, $\boldsymbol{\theta}$ and $\boldsymbol{\lambda}$ to zero

$$\partial_{(\mathbf{C}, \boldsymbol{\theta}, \boldsymbol{\lambda})} \Lambda(\mathbf{C}, \boldsymbol{\theta}, \boldsymbol{\lambda}) = 0,$$

and observe that the equation involving the gradient with respect to $\boldsymbol{\lambda}$ gets us back to the constraints. The purpose of putting the problem in this form (besides it being computationally convenient to work with) is that we can observe that the only non-quadratic term in (7.20) is

$$\mathrm{D}\boldsymbol{\phi}(t_l)\mathbf{C} - \mathbf{f}(t_l, \boldsymbol{\phi}(t_l)\mathbf{C}, \boldsymbol{\theta}),$$

and this is often more nearly convex in $\mathbf{C}$ and $\boldsymbol{\theta}$ than the squared error that results from solving the ordinary differential equation. If $\mathbf{f}$ is linear, then this is convex in either $\mathbf{C}$ or $\boldsymbol{\theta}$ separately. Fixing $\boldsymbol{\lambda}$, this results in a linear combination of gradients in $\mathbf{C}$ and $\boldsymbol{\theta}$; each of which is relatively easy to zero, and hence resulting in an easier

problem. Indeed, this serves as an important motivator for the derivative matching approaches in Chap. 8 and the profiling methods proposed in Chaps. 9 and 10. The methods examined here have been presented as pertaining to collocation methods; these have particularly been advocated by Tjoa and Biegler (1991) using interior point optimization schemes.

Similar methods can be more easily illustrated, graphically, using the *multiple shooting* methods proposed in Bock (1983). In these, the ODE is only solved over short periods—often from one observation to the next—then new initial conditions are allowed. We therefore have a large set of initial conditions—one for each break point. This means we start from one observation and solve the ODE to the next. However, we gradually constrain the difference between the solution over one period and the initial condition for the next.

As an experiment, we re-examined the chemostat data from Sect. 7.4.6 and tried to fit these using multiple shooting. The left-most plot gives our initial guesses: we have used the initial guess from Sect. 7.4.6 by re-scaling both time as well as $C$ and $B$ as parameters. The dotted lines then give the difference between the value after solving the ODE and the next initial conditions. We then calculated the sum of squared errors from the data, plus a scaled sum of squared errors between end points and new initial conditions:

$$\sum_{j=1}^{n} \|\mathbf{y}_j - \mathbf{x}(t_j; \boldsymbol{\theta}, \mathbf{x}_{t_{j-1}})\|^2 + \lambda \sum_{j=2}^{n} \|\mathbf{x}(t_j, \boldsymbol{\theta}, \mathbf{x}_{t_{j-1}}) - \mathbf{x}_{t_j}\|^2$$

where $\mathbf{x}_{t_{j-1}}$ indicates the "reset" initial conditions starting at time $t_{j-1}$. This was minimized over parameters and initial conditions. We then increased the penalty on matching-up ODE solutions to get final parameters with solutions that are nearly continuous. A few steps of these are given in Fig. 7.8.

Both multiple shooting and collocation methods rely on more complex optimization methods than we can discuss in detail here, and which will require exactly solving ODEs rather than getting close. The scheme derived above is a simplification for illustrative purposes only. See Baake et al. (1992), Hooker and Biegler (2007) for details of implementation and tutorials on running them.

## 7.7 Fitting Features

A final class of methods for fitting exact solutions of differential equations involves matching derived features from observed data. Rather than trying to fit all the data in detail, we derive some statistics from the observed data set and simply look for parameters that reproduce these well. This approach is largely applicable to systems with complex dynamical behavior, particularly when you do not attempt to recreate the process in detail. This approach was taken in Tien and Guckenheimer (2008) who fit a model of a "bursting" neuron, in which sharp spikes in voltage come in

pulses of several spikes. They looked for parameters to match the spike amplitude, the interval between the spikes and the number of spikes in a burst.

Similar approaches have been taken with stochastic dynamic models. Wood (2010) matches statistical features—taken as parameters from fitting time series models—to repeated simulation from a stochastic model. A related approach is given in Approximate Bayesian Computation, coined by Beaumont et al. (2002) though with many precursors, in which the likelihood is replaced by an indicator that you are "close" to derived features. See Csilléry et al. (2010) for a review.

These methods are efficient if the dynamical features selected are sufficient statistics for the parameters. Indeed, some of the development of ABC has focussed on finding features that yield high precision estimation. We feel that there is also value in choosing features that ignore details of the data that are not relevant to discovering the particular dynamical qualities of the data or fitting a simplified model.

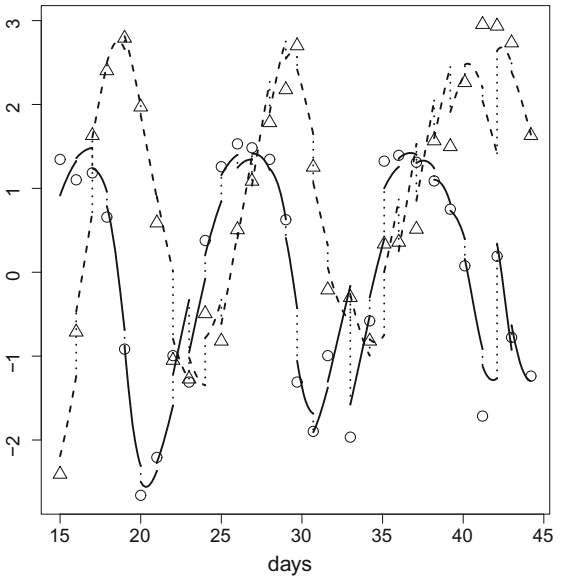


**Fig. 7.8** Application of multiple shooting to the chemostat data. New initial conditions are allowed at each observation time; the difference between the solution at the next observation time and the new initial conditions is gradually constrained to zero as we minimize squared error for $\boldsymbol{\theta}$ at the same time

Here we will illustrate this approach briefly using the chemostat data already described. Here we ignore initial conditions and only examine long term behavior. Our goal here was to find steady-state behavior that matches the cycles we observe in the chemostat. To do this, we run a chemostat model for 30 days and then look at the following statistics from the following 30 days. These statistics were chosen to describe the cycles and ought to be insensitive to the initial conditions $\mathbf{x}_0$:

1. Maximum and minimum of both rotifer and algae populations.
2. Average time between cycles, defined as the peak of the rotifer cycle.

3. Lag time from Algal to Rotifer peaks.
4. Maximum positive and negative slopes of both rotifer and algae through the cycle.

This yields 10 features with which to fit five parameters. We extracted these features from the smoothed estimates of the trajectories described in Chap. 8. We calculated these statistics by visually choosing points from the data to find maxima, minima and slopes for each cycle. The chosen points chosen and the statistics calculated are indicated in Fig. 7.9. Note that in one case we have averaged two measurements corresponding to the bottom of the algal cycle.

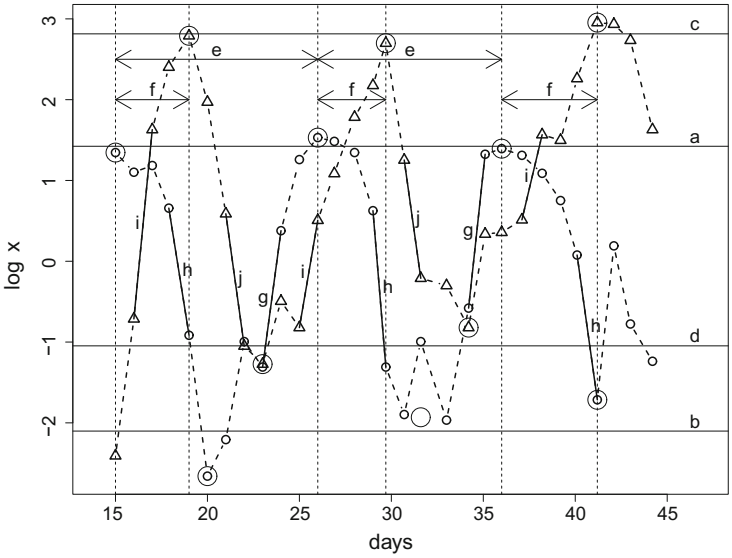


**Fig. 7.9** Dynamic features derived from algae (C) and rotifer (B) measurements from a chemostat experiment. *Circles* and *solid lines* indicate the data points used to calculate *(a)* maximum algae *(b)* minimum algae *(c)* maximum rotifers, *(d)* minimum rotifers, *(e)* time between algal peaks, *(f)* delay between algal and rotifer peaks, *(g)* maximum rate of algal growth, *(h)* maximum rate of algal decline, *(i)* maximum rate of rotifer growth and *(j)* maximum rate of rotifer decline. All parameters were averaged over the cycles where they could be calculated

We then fit parameters by minimizing the sum of squares between these statistics and equivalently calculated statistics from solutions to the Rosenzweig–MacArthur equations as indicated above. The result of this fit, where we have plotted both the original and fitted statistics is given in Fig. 7.10. Here we see a fairly good agreement in most of the data, although the difference in lag between algal and rotifer peaks is missed by a fairly wide margin. This may give some indication of ways to improve the model, although we find the gradient matching diagnostics discussed in Chap. 8 somewhat more direct. The reader will also notice that our trajectories have not, in

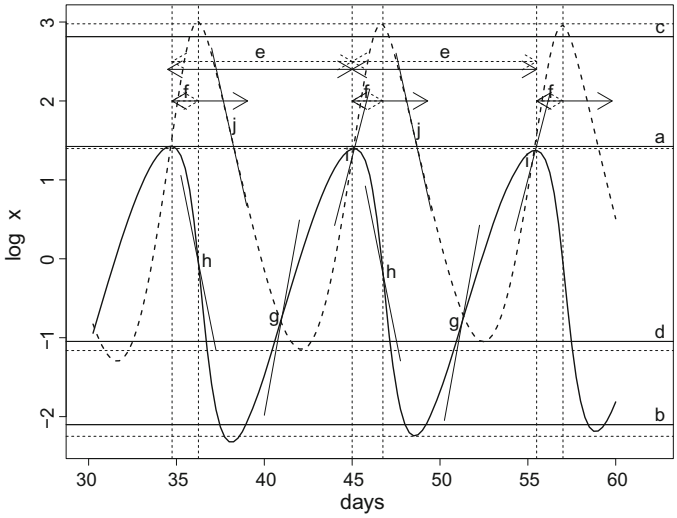


**Fig. 7.10** A fit of features found in Fig. 7.9 to the Rosenzweigh–MacArthur model. Algal trajectories are given in *solid lines*, rotifers *dashed*. *Straight lines* and *arrows* illustrate the statistics calculated: *solid lines* indicate statistics derived from the data (shown in Fig. 7.9), *dashed lines* are those obtained from the fitted trajectory. Fitted slopes are not visible

fact, achieved a limit cycle. While we could ensure that they exhibit less variance by running the model for longer, we cannot guarantee to measure that cycle exactly.

We could, of course, have weighted each feature (this would make sense if they had very different scales) or used a criterion other than squared error. We do not deal with inference here. Wood (2010) derives confidence intervals by assuming the statistics calculated statistics have a multivariate normal distribution and allowing the parameter estimates to inherit variability from this; the idea is extended beyond normality by Fasiolo et al. (2016). However, the reliability (and bias) of this procedure will depend very much on the statistics employed—choosing peaks from real data, for example, can be difficult—and we hesitate to provide a general prescription.

## 7.8 Applications: Head Impacts

This section uses the head impact data in Chap. 1 to illustrate some aspects of working with ODEs. Figure 7.11 plots head impact data described in Chap. 1.

Here we model the observed displacement as a second order system driven by a short (3 ms) impetus starting at 14 ms. This gives us the following equation in general:

$$\mathrm{D}^2 x = \beta_0 + \beta_1 x + \beta_2 \mathrm{D}x + \beta_3 u(t)$$

where $u(t)$ is the indicator function for $14 < t < 17$. We can simplify this somewhat from the discussion above, however, by observing that we clearly start at a steady-state of 0. This means that we can assume initial conditions $(x(0), \mathrm{D}x(0)) = (0, 0)$ and that $\beta_0 = 0$.

The resulting equation can be solved explicitly, but we will continue with a numeric solver here. Expanding the equation to also describe velocity: $v = \mathrm{D}x$ results in

$$\mathrm{D}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \beta_1 & \beta_2 \end{pmatrix}\begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} 0 \\ u(t) \end{pmatrix}$$

To this we need to append the sensitivity equations where we can write out the derivatives with respect to $\beta$ (we fix initial conditions at zero) as

$$\begin{aligned} \mathrm{D}\begin{pmatrix} \partial_{\beta_1} v \\ \partial_{\beta_2} v \\ \partial_{\beta_3} v \end{pmatrix} &= \beta_1 \begin{pmatrix} \partial_{\beta_1} x \\ \partial_{\beta_2} x \\ \partial_{\beta_3} x \end{pmatrix} + \beta_2 \begin{pmatrix} \partial_{\beta_1} v \\ \partial_{\beta_2} v \\ \partial_{\beta_3} v \end{pmatrix} + \begin{pmatrix} x \\ v \\ u \end{pmatrix} \mathrm{D}\begin{pmatrix} \partial_{\beta_1} x \\ \partial_{\beta_2} x \\ \partial_{\beta_3} x \end{pmatrix} \\ &= \begin{pmatrix} \partial_{\beta_1} v \\ \partial_{\beta_2} v \\ \partial_{\beta_3} v \end{pmatrix} \end{aligned} \tag{7.21}$$

in which these six auxiliary equations also have initial conditions zero.

We selected an initial guess $(\beta_1, \beta_2, \beta_3) = (-0.05, -0.075, -3)$ via methods in Chap. 8 and then minimized squared error via 20 Gauss–Newton steps. Recall that only $x$ is measured, so setting $\mathbf{x}(\boldsymbol{\beta}) = (x(t_1, \boldsymbol{\beta}), \ldots, x(t_n, \boldsymbol{\beta}))$ to be the vector of evaluations at the observation time points and defining $\partial_{\boldsymbol{\beta}} \mathbf{x}(\boldsymbol{\beta})$ to be the matrix given by their derivatives with respect to $\boldsymbol{\beta} = c(\beta_1, \beta_2, \beta_3)$, we can write down the iteration as

$$\boldsymbol{\beta}_{j+1} = \boldsymbol{\beta}_j + \left[\partial_{\boldsymbol{\beta}} \mathbf{x}(\boldsymbol{\beta}_j)^T \partial_{\boldsymbol{\beta}} \mathbf{x}(\boldsymbol{\beta}_j)\right]^{-1} \partial_{\boldsymbol{\beta}} \mathbf{x}^T (\mathbf{y} - \mathbf{x}(\boldsymbol{\beta}_j)).$$

Running this algorithm reduces squared error from 252,265 to 69,912 resulting in parameters $(-0.072, -0.213, -19.1)$ and an estimated measurement standard deviation of 22.9. The starting and final solutions of the ODE are plotted in Fig. 7.11. We can now calculate confidence intervals using the estimated variance

$$\mathrm{var}(\hat{\boldsymbol{\beta}}) = \frac{1}{n} \|\mathbf{y} - \mathbf{x}(\boldsymbol{\beta})\|^2 \left[\partial_{\boldsymbol{\beta}} \mathbf{x}(\boldsymbol{\beta})^T \partial_{\boldsymbol{\beta}} \mathbf{x}(\boldsymbol{\beta})\right]^{-1},$$

resulting in confidence intervals $[-0.078, -0.066]$, $[-0.257, -0.169]$ and $[-21.78 - 16.45]$ for $\beta_1$, $\beta_2$ and $\beta_3$ respectively.

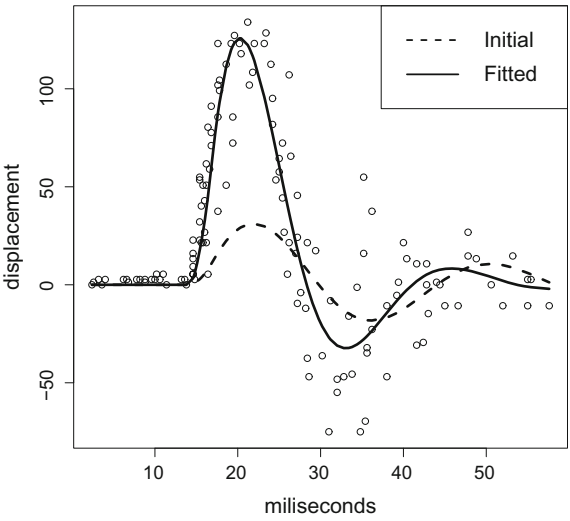


**Fig. 7.11** Least-squares fit to the head impact data using a linear second-order ODE. *Dashed lines* gives solutions at an initial guess, *solid* the final estimate

# Chapter 8
# Two-Stage Least Squares or Gradient Matching

## 8.1 Introduction

Chapter 7 examined estimating parameters for differential equations by making solutions to the differential equation match the data as well as possible. This is possibly the most direct means of estimating parameters. However, these may be unappealing for a number of reasons:

- Repeatedly solving ODEs at different parameter values and initial conditions can be computationally costly, although this is mostly a problem only for high dimensional systems or those with very nonlinear dynamics.
- Finding the best-fitting parameters may require optimizing over a complex surface with multiple local minima.
- The differential equations may be approximately, but not exactly correct.

Instead, the structure of ODEs allows a number of *indirect* methods that do not require solving the differential equations explicitly. Here we will examine the most common of these.

The idea is rather simple: if we could observe $\mathbf{x}(t)$ directly, we could simply choose parameters to make the right hand side of the ODE fit $\mathrm{D}\mathbf{x}$ as well as possible, say by minimizing

$$\texttt{ISSE}(\boldsymbol{\theta}) = \int ||\mathrm{D}\mathbf{x}(t) - \mathbf{f}(\mathbf{x}, \boldsymbol{\theta})||^2 \, dt \; . \tag{8.1}$$

We describe this approach as *gradient matching* to contrast it with the use of the term *trajectory matching* in Chap. 7. Gradient matching avoids some of the difficulties that trajectory matching encounters:

1. It does not require us to repeatedly solve the differential equation at each candidate value of $\boldsymbol{\theta}$.
2. Frequently $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ is more nearly linear in $\boldsymbol{\theta}$ – and hence the landscape is closer to quadratic—than are solutions $\mathbf{x}(t, \boldsymbol{\theta}, \mathbf{x}_0)$. In many cases, $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ is exactly linear



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_8

in $\boldsymbol{\theta}$ and in this case the minimum of $\texttt{ISSE}(\boldsymbol{\theta})$ can be given explicitly in terms of $\mathbf{x}(t)$ and $\mathrm{D}\mathbf{x}(t)$.
3. When $\mathrm{D}\mathbf{x}$ is only approximately equal to $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ we might hope that the minimizer of (8.1) will exhibit less bias – because it allows deviations of $\mathrm{D}\mathbf{x}$ from $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ – than the estimates obtained from the trajectory matching approach.

The least-square objective function (8.1) is amenable to the same Gauss-Newton methods described in Chap. 7. Moreover, this approach does not require us to either know or estimate initial conditions $\mathbf{x}(0)$.

However, a drawback is that we must have enough data to estimate both $\mathbf{x}$ and $\mathrm{D}\mathbf{x}$ accurately without reference to the model. This can be done a number of ways, either by differencing the data directly, or by various smoothing techniques. Another limitation is that we assume that each component of $\mathbf{x}$ is measured directly; a limitation that we will relax when we consider the profiling methods in Chap. 9.

Gradient matching has been re-invented several times and has gone by different names. These include Varah (1982), Ramsay (1996), Pascual and Ellner (2000) and Timmer and Voss (2000); our earliest reference is Bellman and Roth (1971) but we would not be surprised to learn that Newton used it, too. The statistical properties of variations on this idea have been studied in, for example, Brunel (2008), Gugushvili and Klaassen (2012) and Liang and Wu (2008) where it is often referred to as "two-stage least squares". We feel that gradient matching is the most descriptive of the names given to this approach.

We will review some non-parametric methods used to obtain estimates of $\mathbf{x}$ and $\mathrm{D}\mathbf{x}$ before detailing the estimation of $\boldsymbol{\theta}$ and obtaining confidence intervals for our estimate. The presentation of nonparametric smoothing will be brief but will help to illustrate parameter estimation methods. A more comprehensive treatment of smoothing can be found in Eubank and Spiegelman (1990), Ramsay and Silverman (2005) and Ruppert et al. (2005).

## 8.2 Smoothing Methods and Basis Expansions

This section is intended to provide a brief overview of nonparametric smoothing in order to produce an intuition for gradient matching methods. We will make use of the ecological data from Chap. 7, as well as returning to them later. We focus on basis expansion methods because of their relationship to the profiling methods described in Chaps. 9 and 10.

For notational simplicity and because we apply smoothing to each variable in turn, we will treat the case of a single variable $x$.

The technology we employ here and in the next chapter is to represent $x(t)$ as a linear combination of basis functions, $(\phi_1(t), \ldots, \phi_K(t))$:

$$x(t) \approx \sum_{k=1}^{K} c_k \phi_k(t) = \boldsymbol{\phi}(t)^T \mathbf{c}$$

where we combine the basis functions into the vector $\boldsymbol{\phi}(t)$.

There are many potential choices of basis system. The first panel in Fig. 8.1 illustrates a *B-spline* basis system that we employ through the examples in this chapter and those in Chaps. 9 and 10. These show a system of 10 basis functions over the time interval [0, 1]. In this case we have defined the basis by partitioning the time interval into 8 equally-sized blocks on which each of the $\phi_k(t)$ is a cubic polynomial. We have chosen these *knots* for display purposes: typically many more knots (and therefore basis functions) would be used and they need not be equally spaced.

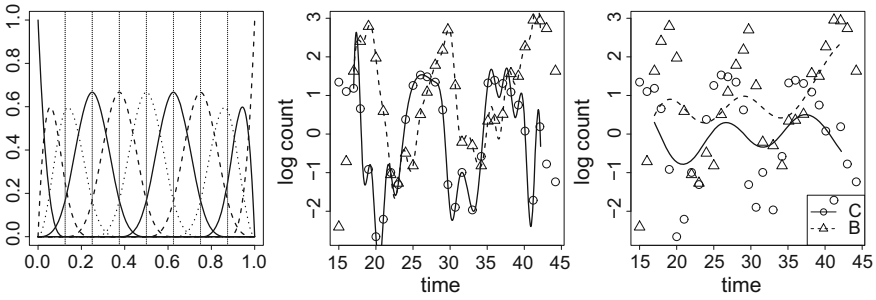


**Fig. 8.1** Basis expansion smoothing. *Left panel* 10 cubic B-spline basis functions. *Middle* and *right channels* smooth's resulting from small and large values of λ, *solid lines* give estimates of the trajectory of rotifers (*B*) and dashed for algae (*C*)

Once we have a set of basis functions, we need to obtain values for the vector of coefficients $\mathbf{c}$. The most direct means of doing this is to minimize the sum of squared errors. Let $\boldsymbol{\Phi}$ be the $n$ by $K$ matrix contain the values of $\phi_k(t_j)$. We compute the coefficients as

$$\hat{\mathbf{c}} = \text{argmin} \sum_{j=1}^{n} \left(y_j - \boldsymbol{\phi}(t_j)^T \mathbf{c}\right)^2 = \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \mathbf{y} \ .$$

Given the coefficient estimates, we can then estimate the value of $x(t)$ at any time $t$ from

$$\hat{x}(t) = \boldsymbol{\phi}(t)^T \hat{\mathbf{c}} = \boldsymbol{\phi}(t)^T \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi}\right)^{-1} \boldsymbol{\Phi}^T \mathbf{y} \ .$$

This approach can yield unsatisfying results in the sense that, if many basis functions are used, we can end up with a very rough curve which nearly interpolates what can be noisy data. Too few, of course, will fail to capture the trajectory well. We therefore need to somehow decide how complex our curve should be.

We can control the smoothness of the fit to the data by our choice of the number of basis functions, but this provides only coarse control over the resulting estimates. As an alternative, we use fairly large number of basis functions, but modify the way we estimate the coefficients in $\mathbf{c}$. Rather than simply minimizing squared error, we add a term that measures the extent the $\hat{x}$'s are smooth. This makes our objective

function

$$\texttt{PENSSE}(\mathbf{c}) = (1-\rho)\sum_{j=1}^{n}\left(y_j - \boldsymbol{\phi}(t_j)^T\mathbf{c}\right)^2 + \rho\int\left[\mathrm{D}^2\boldsymbol{\phi}(t)^T\mathbf{c}.\right]^2 dt\ . \qquad (8.2)$$

We have represented this as a trade-off between two terms, the first of which is squared error, our data fitting criterion. The second term is referred to as a *smoothing penalty* and it trades off making $x(t)$ close to the observed data against requiring that its curvature, $\mathrm{D}^2x(t)$, is small. This means that as $\rho$ increases from zero, the "wiggles" that can be seen in middle panel of Fig. 8.1 are increasingly ironed out. The right panel of Fig. 8.1 shows the result of changing $\rho$ from being very small, hence obtaining rough estimates, to very large in which the resulting smooth is nearly linear.

We note that $\texttt{PENSSE}(\mathbf{c})$ is often written in terms of unweighted squared error plus a weighted penalty, usually designated by $\lambda$. This is equivalent to our formulation, but we feel that the trade-off is more apparent here, and it will be a particularly useful perspective when we present the profiling methods in Chaps. 9 and 10.

The use of the second derivative $\mathrm{D}^2x(t)$ in the roughness penalty term implies that straight lines are smooth since they receive zero penalty, and that curvature implies roughness. This is by no means the only way of representing smoothness; any derivative or combination of derivatives could be employed. If, for example, we replace $\mathrm{D}^2x(t)$ with $\mathrm{D}^3x(t)$ we state that quadratic functions are also smooth. Alternatively, using $\mathrm{L}x = \mathrm{D}^3x(t) - \omega\mathrm{D}x(t)$ in its place would define $\sin(\omega t)$ and $\cos(\omega t)$ as smooth, along with the constant function. Here, we advocate using a higher derivative as a penalty – at least $\mathrm{D}^3x(t)$. This is because we will want to use the estimate $\mathrm{D}\hat{\mathbf{x}}(t)$ below and we want this to be smooth in the sense of having low curvature. In order to be general, we will use $\mathrm{L}x(t)$ to indicate some combination of derivatives of $x(t)$ that is used to measure roughness; Chaps. 9 and 10 use $\mathrm{D}^m x(t) - f(t, x, \boldsymbol{\theta})$ as a smoothing penalty, and add the feature that the parameter vector $\boldsymbol{\theta}$ is simultaneously estimated.

We should also note another effect of this penalization; it deliberately introduces bias into the estimate $x(t)$. In particular, if $\mathrm{D}^2\mathbf{x}(t)$ is large at some point (we might expect this to be the case when $\mathbf{x}(t)$ solves an ordinary differential equation, for example) then we can expect that our estimate will not be particularly accurate nearby. This can be alleviated by choosing a penalty that corresponds to the true $x(t)$ as closely as possible. This is the central idea behind Chaps. 9 and 10.

We can now reformulate the expression of the coefficient vector $\mathbf{c}$ by observing that

$$\int\left[\mathrm{L}\mathbf{c}'\boldsymbol{\phi}(t)\right]^2 \mathrm{d}t = \mathbf{c}^T\mathbf{R}\mathbf{c}$$

where $\mathbf{R}$ is the matrix with entries

$$R_{kl} = \int \mathrm{L}\phi_k(t)\mathrm{L}\phi_l(t)\mathrm{d}t\ .$$

From this we obtain

$$\hat{\mathbf{c}} = \left((1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \rho\mathbf{R}\right)^{-1}\boldsymbol{\Phi}^T\mathbf{y},$$

and that

$$\hat{x}(t) = \boldsymbol{\phi}(t)^T\hat{\mathbf{c}}, \quad \text{and} \quad \widehat{\mathrm{D}x}(t) = \mathrm{D}\boldsymbol{\phi}(t)^T\hat{\mathbf{c}}\ .$$

So far, we have not specified a value of $\rho$ or how to choose it. By its nature, if we wish to minimize squared error on the observations, our best choice of $\rho$ is 0. Instead, it is common to minimize *cross validation error*; this is the error that you get when you leave one data point out of the model and try to predict it. Explicitly we define the cross validation error to be

$$\mathrm{CV}(\rho) = \sum_{j=1}^{n}\left(y_j - \hat{x}_\rho^{-j}(t_j)\right)^2$$

where $\hat{x}_\rho^{-j}(t)$ is the estimate obtained with all the data except $(y_j, t_j)$. The value of $\mathrm{CV}(\rho)$ thus represents the error obtained when making predictions for new data. $\mathrm{CV}(\rho)$ may be calculated efficiently by defining the *smoothing matrix*

$$\mathbf{S}(\rho) = \boldsymbol{\Phi}\left((1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi} + \rho\mathbf{R}\right)^{-1}\boldsymbol{\Phi}^T$$

and the predictions at the data points (without cross validation)

$$\hat{\mathbf{y}} = \mathbf{S}(\rho)\mathbf{y}\ .$$

We can now express the cross-validation error to be

$$\mathrm{CV}(\rho) = \sum_{j=1}^{n}\left(\frac{y_j - \hat{y}_j}{1 - S(\rho)_{jj}}\right)^2\ . \tag{8.3}$$

For various reasons, it is also common to use the related criterion of *Generalized Cross Validation*

$$\mathrm{GCV}(\rho) = \frac{\sum_{j=1}^{n}(y_j - \hat{y}_j)^2}{\sum_{j=1}^{n}(1 - S(\rho)_{jj})^2}\ . \tag{8.4}$$

Alternative methods of choosing $\rho$ are based on representing the smooth in terms of random effects; see Ruppert et al. (2005). Whichever criterion is used, it is often easier to search for a minima by looking at a grid of values of in the *logit* scale $\eta = \log\rho/(1-\rho)$, and a grid search often suffices to find a reasonable value.

This procedure must be carried out for each component $x(t)$ of the differential equation separately. It is often convenient to use the same basis expansion for each component, in which case state *vector* is written as $\mathbf{x}(t) = \boldsymbol{\phi}(t)^T C$ with $C$ containing the coefficients for each state vector stacked in columns. However, the choice of

roughness measure, basis expansion and penalty parameter can all be tailored to each component of the state vector if needed. We suggest that experimentation with various values of $\rho$ should be the norm, as opposed to blind reliance on any automatic method such as those discussed here.

Figure 8.2 presents the results of carrying this out on the ecological data from Chap. 7. Here we minimised the sum of GCV for both $C$ and $B$. The resulting curve is somewhat rougher than it is visually pleasing, however this results in better predictive performance; it also produces better parameter estimates from the methods below.

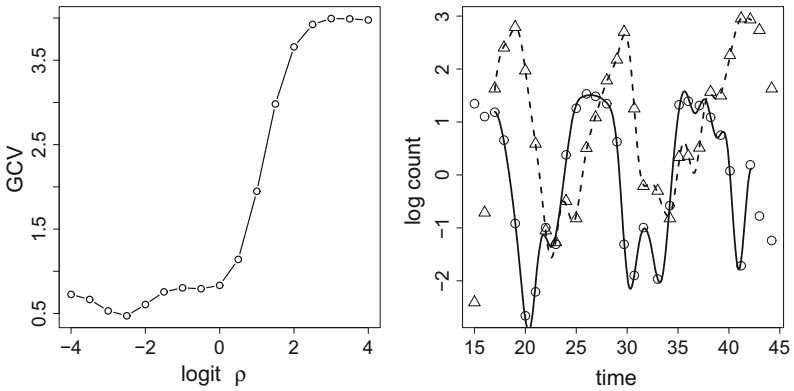


**Fig. 8.2** Optimal smoothness of the chemostat data. *Left panel* GCV as a function of $log(\rho/(1-\rho))$. It is usually useful to look over values on the logit scale for a minimum. *Right panel* the resulting smooths

## 8.3 Fitting the Derivative

### *8.3.1 Optimizing Integrated Squared Error (ISSE)*

We now return to the the multivariable context. Suppose we use some smoothing method to obtain estimates $\hat{\mathbf{x}}(t)$ and $\widehat{\mathrm{D}\mathbf{x}}(t)$ of $\mathbf{x}(t)$ and $\mathrm{D}\mathbf{x}(t)$ respectively. We can now estimate $\boldsymbol{\theta}$ by making $\mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta})$ look as much like $\widehat{\mathrm{D}\mathbf{x}}(t)$ as possible. The easiest way to do this is via non-linear regression and we define $\boldsymbol{\theta}$ to be the minimizer of the *integrated squared error*:

$$\texttt{ISSE}(\boldsymbol{\theta}) = \sum_{i=1}^{d} \int \omega_i(t) \left(\widehat{\mathrm{D}x}_i(t) - f_i(\hat{\mathbf{x}}(t), \boldsymbol{\theta})\right)^2 \mathrm{d}t \ . \tag{8.5}$$

This objective can be fairly readily extended to provide different weights to different components $x_i$ of the model and even to provide more weight to some particular parts

of the time domain. Squared error is also not the only possible objective although, as we see below, it is one of the easiest to work with.

Variables will often need weighting using the weight function $\omega_i(t)$, which is generally chosen to be constant through most of the region, but for theoretical reasons should be zero at the ends of the interval. This is because the non-parametric estimates $\hat{x}_i$ and $\widehat{\mathrm{D}x_i}$ are poorly behaved at the edge of the data and this can impact parameter estimates – see, Wu et al. (2012) and Gugushvili and Klaassen (2012). In practice, we can often visually assess poor behaviour of $\hat{x}_i$ and $\widehat{\mathrm{D}x_i}$ at either end of the time interval and remove these from our estimates. The weights are designed to equilibrate the impact of variables and of the error levels in the observations across variables. We again remind ourselves that transformations of variables is often helpful. Comparing standard errors of observations around their fits is also advisable.

Since we expect that $\mathbf{f}$ will be nonlinear, the integral above is usually not tractable. Consequently, we need to approximate it by a quadrature rule in which we evaluate the integrand at time points $t_q$ for $q = 1, \ldots, Q$ and approximate the integral by a weighted sum

$$\widehat{\mathtt{ISSE}}(\boldsymbol{\theta}) = \sum_{i=1}^{d} \sum_{q=1}^{Q} w_{iq} \left( \widehat{\mathrm{D}x}_i(t_q) - f_i(\hat{\mathbf{x}}(t_q), \boldsymbol{\theta}) \right)^2 .$$

Here in addition to accounting for the weighting values of $\omega_i(t_q)$ above, the weights $w_{iq}$ also incorporate the quadrature rule that you use, and there are a number available, although we will not detail them here. Simpson's rule (Kincaid and Cheney 2002) can be readily applied on small intervals, for example between the knots in a basis expansion, but the simpler trapezoidal rule is often sufficiently accurate.

We now observe that $\widehat{\mathtt{ISSE}}(\boldsymbol{\theta})$ is a weighted least squares criterion and we can apply a Gauss-Newton scheme to minimize it just as in Chap. 7. To do this we only need to deal with the derivative of $f_i(\mathbf{x}, \boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ for each $i$ and at specific values of $\mathbf{x}$. In particular, if we define the vectors $F_i(\boldsymbol{\theta})$ and matrices $\partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta})$ for $i = 1, \ldots, d$ by

$$[F_i(\boldsymbol{\theta})]_q = f_i(\hat{\mathbf{x}}(t_q), \boldsymbol{\theta}), \quad \text{and} \quad [\partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta})]_{q\ell i} = \partial_{\theta_\ell} f_i(\hat{\mathbf{x}}(t_q), \boldsymbol{\theta}) .$$

These contain the value of $f_i$ and gradient of $f_i$ with respect to $\boldsymbol{\theta}$ evaluated at each time point $t_q$. We will also need to define $\mathbf{W}_i$ to be the $Q \times Q$ matrix with the $w_{iq}$ on the diagonal and to let $\hat{X}_i$ and $\widehat{\mathrm{D}X_i}$ be the vectors giving the evaluation of $\hat{x}_i(t_q)$ and $\widehat{\mathrm{D}x_i}(t_q)$ respectively for $q = 1, \ldots, Q$. We now start with some sensible guess $\hat{\boldsymbol{\theta}}_0$ for $\boldsymbol{\theta}$ and on iteration $\ell$ make the update

$$\hat{\boldsymbol{\theta}}^{\ell+1} = \hat{\boldsymbol{\theta}}^{\ell} - \left[ \sum_{i=1}^{d} \sum_{q=1}^{Q} \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}^\ell)^T \mathbf{W}_i \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}^\ell) \right]^{-1} \sum_{i=1}^{d} \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}^\ell)^T \mathbf{W}_i \left( \widehat{\mathrm{D}X}_i - F_i(\boldsymbol{\theta}^\ell) \right) .$$

If we start $\hat{\boldsymbol{\theta}}_0$ close to the optimal value, this iteration will converge quickly, but experimentation with different optimization strategies may be required.

### 8.3.2 Gradient Matching for the Refinery Data

We illustrate gradient matching with the refinery data from Chaps. 1 and 7 where we fitted a first-order forced linear ODE:

$$\mathrm{D}x = \beta_0 + \beta_1 x + \alpha u(t) \ .$$

In this case, the minimizing values of the parameters can be obtained explicitly:

$$\begin{pmatrix} \beta_0 \\ \beta_1 \\ \alpha \end{pmatrix} = \begin{pmatrix} \int w & \int w\hat{x} & \int wu \\ \int w\hat{x} & \int w\hat{x}^2 & \int w\hat{x}u \\ \int wu & \int w\hat{x}u & \int wu^2 \end{pmatrix}^{-1} \begin{pmatrix} \int w\widehat{\mathrm{D}x} \\ \int w\hat{x}\widehat{\mathrm{D}x} \\ \int wu\widehat{\mathrm{D}x} \end{pmatrix},$$

where we have suppressed "$(t)\,\mathrm{d}t$" for clarity.

Figure 8.3 presents the results of this estimate. We have shown both $\widehat{\mathrm{D}x}$ and our prediction of it from $\hat{x}$ as well as the error between the two. This example shows why gradient matching is appealing. Besides avoiding solving the differential equation, gradient matching also often involves a better conditioned optimization problem. When the ODE is linear in the parameters (even if it is not linear in the state variables) we can write down parameter estimates explicitly.

As a case in point, in Fig. 7.3 from Chap.7 we showed the optimization problem for the FitzHugh-Nagumo model where we observe that there are many local minima, and that the shape of the surface is influenced by the choice of ODE solver. In gradient matching, this ODE is linear in $a$ and $b$ meaning that we can give explicit expressions for their estimates and the objective function is exactly quadratic. Even when the ODE is not linear in parameters, the gradient matching objective is often more nearly convex than the trajectory matching objective; this makes for a much easier optimization problem with less need to be careful about using an initial guess.

### 8.3.3 Gradient Matching and the Chemostat Data

We have employed this approach with the chemostat data and the smooth shown in Fig. 8.2 where we have removed the first two and last two days of the series to down-weigh the end points and fit them with the Rosenzweig–MacArthur model used in Chap. 7. Here, our parameters differ from those found by trajectory matching. Figure 8.4 presents a comparison of $\widehat{\mathrm{D}\mathbf{x}}$ and $\mathbf{f}(\hat{\mathbf{x}}, \hat{\boldsymbol{\theta}})$ plotted over time and against each other (left and middle plots) where we observe that we have mimicked the derivative

pretty well. The right plot compares the data to the solution of the Rosenzweig–MacArthur ODE at the parameters found by gradient matching. Here we can see that gradient matching has resulted in trajectories with a longer period and smaller amplitude than the data exhibit. This is partly the result of statistical inaccuracies in estimating $\widehat{\mathrm{D}\mathbf{x}}$ and $\hat{\mathbf{x}}$, but can also be due to error in the ODE as well as in our measurements, something we discuss in the next section.

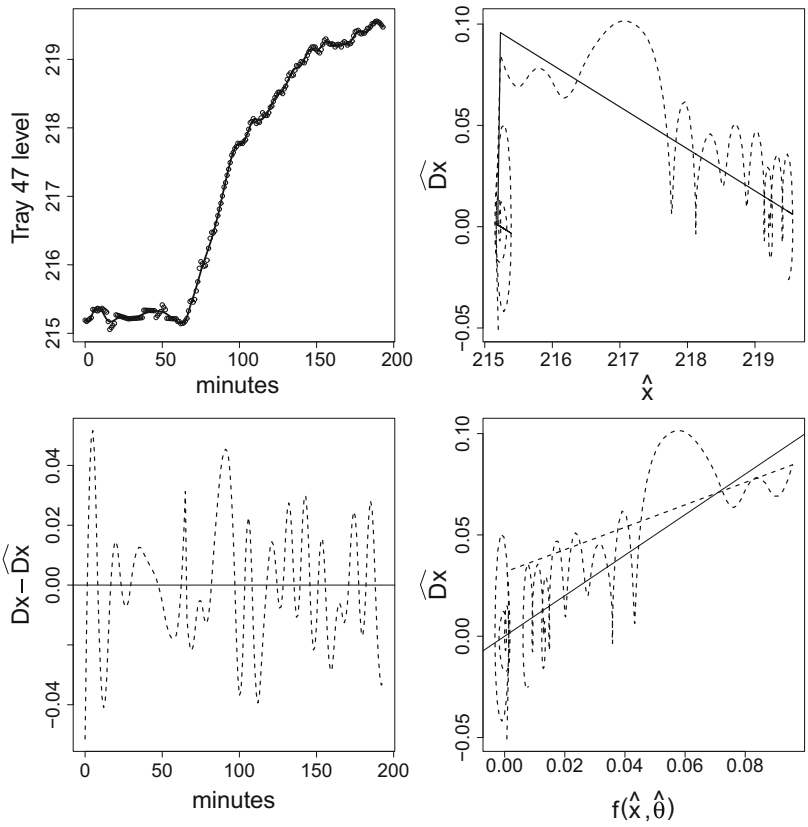


**Fig. 8.3** Gradient matching on the refinery data. *Top left* level in Tray 47 and RS move of these data. *Top right* $\widehat{\mathrm{D}x}$ plotted against $\hat{x}$ (*dashed*) and the prediction of it (*solid*) using estimated parameters. *Bottom left* difference between $\widehat{\mathrm{D}x}$ and its prediction plotted over time. *Bottom right* predicted versus fitted $\widehat{\mathrm{D}x}$

## 8.4 System Mis-specification and Diagnostics

### *8.4.1 Diagnostic Plots*

An advantage of gradient matching is that it provides some robustness to system disturbances that have not been modeled. This can be important: most ODE models are written down for simplified systems that assume all external influences are accounted for in the model. Importantly, we often expect that any discrepancies between the model and the process generating the data are best modeled in terms of the derivative of the system rather than the state vector itself. Gradient matching explicitly allows us to visually examine how $\widehat{\mathrm{D}\mathbf{x}}$ differs from $\mathbf{f}(\hat{\mathbf{x}}, \hat{\boldsymbol{\theta}})$.

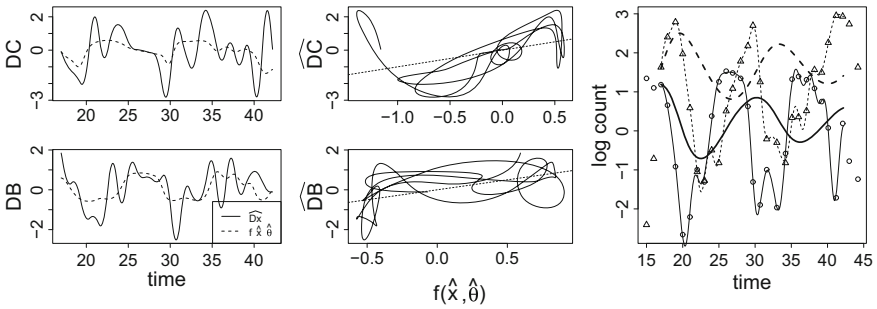


**Fig. 8.4** Gradient matching on the chemostat data. The *left panel* plots $\widehat{\mathrm{D}\mathbf{x}}$ (*dashed*) and the fitted $\mathbf{f}(\hat{\mathbf{x}}, \hat{\boldsymbol{\theta}})$ (*solid*) as a function of time for $C$ (*top*) and $B$ (*bottom*). The *middle panel* presents $\widehat{\mathrm{D}\mathbf{x}}$ plotted against $\mathbf{f}(\hat{\mathbf{x}}, \hat{\boldsymbol{\theta}})$. The *right panel* plots log data and smooth (*thin lines*) along with the trajectories obtained by solving the ODE at parameters obtained from gradient matching (*thick lines*)

The final panel in Fig. 8.3 presents the basic approach. We have plotted process residuals

$$\mathbf{r}(t) = \widehat{\mathrm{D}\mathbf{x}}(t) - \mathbf{f}(\hat{\mathbf{x}}(t), \hat{\boldsymbol{\theta}}) \; .$$

We think of this as a lack-of-fit function which indicates how far $\hat{\mathbf{x}}$ is from satisfying the ODE, even at the best possible parameter estimates. Figure 8.3 lets us examine $\mathbf{r}$ where we see what appears to be random oscillations – we will deal with accounting for these in Sect. 8.5. However, this also gives us the opportunity to examine whether $\mathbf{f}$ has been poorly chosen; we can now visually conduct diagnostics exactly as one might in more classical regression by plotting $\mathbf{r}$ versus $\mathbf{x}$ and looking for reliable patterns. This is a much more immediate means of improving models than is available for the trajectory matching methods in Chap. 7 where changing the algebraic form of $\mathbf{f}(\mathbf{x}, \boldsymbol{\theta})$ can have counter–intuitive effects on the characteristics of solutions to an ODE.

An example where we might suspect some lack of fit comes from the middle panel of Fig. 8.4 where we see some evidence of curvature in the relationship between $\widehat{DC}$ and $f_C(\hat{\mathbf{x}}, \hat{\boldsymbol{\theta}})$. In Fig. 8.5, we investigate this further by plotting $\mathbf{r}(t)$ against $\hat{\mathbf{x}}(t)$ to look for relationships that might be used to modify $\mathbf{f}$. Just as in linear regression, we can use this to add terms to the ODE; it is also often helpful to think of mechanistic explanations for the relationships you see. It is important to recall that the relationships in Fig. 8.5 are what is left over after fitting a nonlinear model, and simply adding terms to the equations (as we might for linear regression) may produce a much more complicated model than necessary.

To illustrate this, Fig. 8.5 produces a sequence of plots. On the left, we have plotted $\mathbf{r}(t)$ as a two-dimensional function of $\hat{C}(t)$ and $\hat{B}(t)$. Here there may be some patterns but they are difficult to discern; it can also help to use plotting software that will allow you interactively rotate these plots when looking for patterns. The middle plot compares this to what we get when fitting a Lotka–Volterra model

$$\begin{aligned} \mathrm{D}C &= \alpha C - \beta C B \\ \mathrm{D}B &= \gamma C B - \delta B \end{aligned}$$

(note that this is linear in the parameters $(\alpha, \beta, \gamma, \delta)$ and we can therefore obtain estimates using linear regression). Here some more curvature is evident in the $C$ equation (black lines) although this can still be difficult to make out. The right panel provides an example template – we have taken the solutions to the ODE displayed in Fig. 8.4 and then fit the Lotka–Volterra model to them using gradient matching. Here we have a much "cleaner" picture where curvature associated with the half-saturation rate $\chi_C$ is more evident as is the logistic growth model for $C$, although it would require both some careful thought about appropriate ways to modify the ODE, and some experimentation, to get back to a Rosenzweig–MacArthur model.

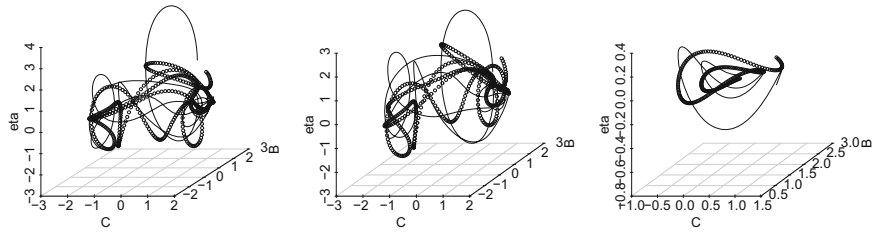


**Fig. 8.5** Diagnostics for fitting the chemostat data. The *left panel* provides the lack of fit $\mathbf{r}(t)$ plotted against the estimated state variables using the Rosenzweig–MacArthur ODE. The lack of fit in the $C$ equation is given by *lines*, in the $B$ equation by *circles*. The middle panel uses a simpler Lotka–Volterra ODE. The right panel provides a prototype diagnostic resulting from fitting solutions to a Rosenzweig–MacArthur model with a Lotka–Volterra ODE

## 8.5 Conducting Inference

In this section, we will discuss methods to produce inference for parameters after performing derivative fitting. How to do this is less obvious than it is for trajectory fitting in Chap. 7 where standard nonlinear regression methods are available. In this case, variance is derived from both the estimate $\hat{\mathbf{x}}$ and its derivative $\widehat{\mathrm{D}\mathbf{x}}$; these are strongly correlated.

In order to derive a reasonable means of estimating variance for this model, we will think of our estimate as a functional of both $\hat{\mathbf{x}}$ and $\widehat{\mathrm{D}\mathbf{x}}$. In fact, since we plug in the values of these estimates at discrete times, we can really write it as $\hat{\boldsymbol{\theta}}(\hat{X}, \widehat{\mathrm{D}X})$, (recall from Sect. 8.3 that $\hat{X}$ and $\widehat{\mathrm{D}X}$ are matrices containing the evaluation of $\hat{\mathbf{x}}$ and $\widehat{\mathrm{D}\mathbf{x}}$ at times $t_q$ for $q = 1, \ldots, Q$).

In order to understand variability, we need to have a notion of what $\hat{\boldsymbol{\theta}}$ is varying around. To do this, we'll define $\boldsymbol{\theta}_0 = \boldsymbol{\theta}(E\hat{X}, E\widehat{\mathrm{D}X})$ to be the estimator based on the expected values of $\hat{X}$ and $\widehat{\mathrm{D}X}$. These will be biassed and we expect that $\boldsymbol{\theta}_0$ may not, in fact, lie exactly on the truth – we will examine bootstrap-based means of reducing this bias below. To try and understand the variability in $\hat{\boldsymbol{\theta}}$, we will consider taking one step of the Gauss-Newton algorithm starting at $\boldsymbol{\theta}_0$:

$$\hat{\boldsymbol{\theta}}_1 = \boldsymbol{\theta}_0 - \left[\sum_{i=1}^{d}\sum_{q=1}^{Q} \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}_0)^T \mathbf{W}_i \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}_0)\right]^{-1} \sum_{i=1}^{d} \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}_0)^T \mathbf{W}_i \left(\widehat{\mathrm{D}X}_i - F_i(\boldsymbol{\theta}_0)\right) ,$$

and we will write out the difference in the last term as

$$\begin{aligned}\widehat{\mathrm{D}X}_{qj} - F_{qj}(\boldsymbol{\theta}_0) = & \\ & \widehat{\mathrm{D}X}_{qj} - E\widehat{\mathrm{D}X}_{qj} + E\widehat{\mathrm{D}X}_{q_i} - f_i(E\hat{X}_q, \boldsymbol{\theta}_0) + f_i(E\hat{X}_q, \boldsymbol{\theta}_0) - f_i(\hat{X}_q, \boldsymbol{\theta}_0) \\ & = \\ & \widehat{\mathrm{D}X}_{qj} - E\widehat{\mathrm{D}X}_{qj} + \partial_{\mathbf{x}} f_i(\hat{X}_q, \boldsymbol{\theta}_0)(\hat{X}_q - E\hat{X}_q) + E\widehat{\mathrm{D}X}_{q_i} - f_i(E\hat{X}_q, \boldsymbol{\theta}_0).\end{aligned} \tag{8.6}$$

These are the terms on which we will base our estimate of variance.

The last of these terms only involves expectations, so has zero variance (it does contribute bias but we will worry about that later) so we only have to worry about the variance in $\hat{X}_i$ and $\widehat{\mathrm{D}X}_i$. To express this, we'll need to define the matrices

$$[\partial_{\mathbf{x}} F_i]_{qi} = \partial_{x_i} f_i(\hat{\mathbf{x}}(t_q), \boldsymbol{\theta}) .$$

These contain derivatives with respect to $\mathbf{x}$ on each row and use this to construct

$$H(\boldsymbol{\theta}) = \sum_{i=1}^{d} \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta})^T W_i \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta})$$

$$J_i(\boldsymbol{\theta}) = H(\boldsymbol{\theta})^{-1} \left( \sum_{k=1}^{d} \partial_{\boldsymbol{\theta}} F_k(\boldsymbol{\theta})^T W_k \left[ \operatorname{diag}(\partial_{\mathbf{x}} F_k(\boldsymbol{\theta})_i) \operatorname{diag}(1_{i=k}) \right] \right)$$

where $1_{i=k}$ is an identity matrix if $i = k$ and a matrix of zeros, otherwise. Then, putting things together we have

$$\hat{\boldsymbol{\theta}} - \boldsymbol{\theta}_0 = \sum_{i=1}^{d} J_i(\boldsymbol{\theta}) \begin{bmatrix} \hat{X}_i - E\hat{X}_i \\ \widehat{\mathrm{D}X}_i - E\widehat{\mathrm{D}X}_i \end{bmatrix}$$

(in fact, this takes some re-arranging of terms to collect all the columns of $\hat{X} - E\hat{X}$ in the same place). With this expression we can now calculate a variance as being

$$\operatorname{var}(\hat{\boldsymbol{\theta}}) = V = \sum_{i=1}^{d} \sum_{k=1}^{d} J_i(\boldsymbol{\theta}) \begin{bmatrix} \operatorname{var}(\hat{X}_i, \hat{X}_k) & \operatorname{cov}(\hat{X}_i, \widehat{\mathrm{D}X}_k) \\ \operatorname{cov}(\widehat{\mathrm{D}X}_i, \hat{X}_k) & \operatorname{var}(\widehat{\mathrm{D}X}_i, \widehat{\mathrm{D}X}_k) \end{bmatrix} J_k(\boldsymbol{\theta})^T \ . \tag{8.7}$$

Of course, these expressions all depend on $\boldsymbol{\theta}_0$, but we can substitute $\hat{\boldsymbol{\theta}}$ in $J_i(\boldsymbol{\theta})$ and still produce an (asymptotically) correct answer. To manage this, we need to express the variance and covariances of our initial smooths.

### 8.5.1 Nonparametric Smoothing Variances

Our estimates $(\hat{X}, \widehat{\mathrm{D}X})$ inherit variability from data modeled as $Y_{ji} = X_i(t_j) + \varepsilon_{ji}$ where $\varepsilon_{ji}$ is standardly assumed to be distributed as $N(0, \sigma_j^2)$ and represents any random disturbances in the data. Because we might expect that measurements made at the same time may be subject to the same random errors (for example, the volume of a sample from a chemical reaction will increase the volume of all the reactants) we model these and dependent between different measurements at the same time. That is, we say that the vector of measurements $\boldsymbol{\varepsilon}_j$ has covariance matrix $\Sigma$ and we will write the elements of the matrix as $\sigma_{ik} = \operatorname{cov}(\varepsilon_{ij}, \varepsilon_{jk})$. We can obtain these estimates simply by obtaining the empirical *observation residuals* $\hat{\varepsilon}_{ij}$ (because they estimate variability in our observations) and estimating

$$\hat{\sigma}_{ik} = \frac{1}{n} \sum_{j=1}^{n} \hat{\varepsilon}_{ji} \hat{\varepsilon}_{jk} \ .$$

Often, we will divide by $n - \mathrm{df}$ where df is an estimate of the smoothing degrees of freedom, frequently given as $\mathrm{df} = \operatorname{tr}(S(\rho))$ (see Ramsay and Silverman 2005;

Eubank and Spiegelman 1990), although $\mathrm{df} = 2\mathrm{tr}(S(\rho)) - \mathrm{tr}(S(\rho)^T S(\rho))$ has also been suggested; see Ruppert et al. (2005).

In the case of basis-expansion smoothing, the variance of $\mathbf{x}$ and $\widehat{\mathrm{D}\mathbf{x}}$ can now be be expressed fairly readily. Since

$$\hat{x}_i(t) = \boldsymbol{\phi}(t)^T(\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_i + \lambda_i\mathbf{P})^{-1}\boldsymbol{\Phi}_i^T Y_i \ ,$$

$$\mathrm{D}\hat{x}_i(t) = \mathrm{D}\boldsymbol{\phi}(t)^T(\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_i + \lambda_i\mathbf{P})^{-1}\boldsymbol{\Phi}_j^T Y_i$$

and the $Y_i$ have the same variance as the residuals $\varepsilon$, our estimates just inherit this variance. In this case, we can write

$$\begin{aligned}
\mathrm{cov}(\hat{x}_i(t), \hat{x}_k(s)) &= \sigma_{ik}\boldsymbol{\phi}(t)^T(\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_i + \lambda_i\mathbf{P})^{-1}\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^T\boldsymbol{\Phi}_k + \lambda_k\mathbf{P})^{-1}\boldsymbol{\phi}(t)^T \\
\mathrm{cov}(\hat{x}_i(t), \widehat{\mathrm{D}x}_k(s)) &= \sigma_{ik}\boldsymbol{\phi}(t)^T(\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_i + \lambda_i\mathbf{P})^{-1}\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^T\boldsymbol{\Phi}_k + \lambda_k\mathbf{P})^{-1}\mathrm{D}\boldsymbol{\phi}(t)^T \\
\mathrm{cov}(\widehat{\mathrm{D}x}_i(t), \widehat{\mathrm{D}x}_k(s)) &= \sigma_{ik}\mathrm{D}\boldsymbol{\phi}(t)^T(\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_i + \lambda_i\mathbf{P})^{-1}\boldsymbol{\Phi}_i^T\boldsymbol{\Phi}_k(\boldsymbol{\Phi}_k^T\boldsymbol{\Phi}_k + \lambda_k\mathbf{P})^{-1}\mathrm{D}\boldsymbol{\phi}(t)^T \ ,
\end{aligned}$$

which allows us to construct the covariances above.

### 8.5.2 Example: Refinery Data

We will illustrate this process with the simplest example: the refinery data examined above. Here we first use our estimated $\eta(t)$ to obtain an estimate for $R(u)$ plotted on the bottom right panel in Fig. 8.3. We also need to calculate the variability of our observations $Y$ using

$$\hat{\sigma}^2 = \frac{1}{n - \mathrm{tr}S(\lambda)}\sum_{j=1}^{n}(y_j - \hat{y}_j)^2$$

which comes out to 0.002 with $\lambda$ being chosen to be essentially 0. Since our equation is linear, we can readily calculate that

$$\partial_{\boldsymbol{\beta}}\mathbf{f}(t, x, u, \boldsymbol{\beta}) = (1, x(t), u(t))$$

and we define the matrix $X$ to have rows $(1, x(t_q), u(q))$. Without using weights, it is fairly easy to calculate that

$$H = X^T X, \ J = X^T\begin{bmatrix}\beta_1 I \\ I\end{bmatrix}$$

applying our formulae above, Table 8.1 reports the variance due to the $\varepsilon$ observation errors, $\eta(t)$ process errors and their combination as well as confidence intervals.

**Table 8.1** Variances and confidence intervals after gradient matching on the refinery data

| | Estimated variance | Confidence intervals | |
|---|---|---|---|
| | | Lower | Upper |
| $\beta_0$ | 0.137091 | 7.6932 | 9.1743 |
| $\beta_1$ | 0.000001 | −0.0228 | −0.0187 |
| $\alpha$ | 0.000059 | −0.2101 | −0.1795 |

## 8.6 Related Methods and Extensions

A number of related methods have been proposed which we will briefly outline here

### *8.6.1 Alternative Smoothing Methods*

We have presented gradient matching using basis expansion methods as a means to estimate $\mathbf{x}(t)$ and its derivative, partly because of their connection to profiling methods (see Chap. 9). However, as we have indicated, they are not the only tools available.

Local linear regression has often been used, particularly since it is easier to analyze mathematically. Briefly, this method fits a linear regression *locally* to the point $t$ of interest. That is we fit

$$(\beta_0(t), \beta_1(t)) = \text{argmin} \sum_{j=1}^{n} K(|t - t_j|/h) \left(y_j - \beta_0 - \beta_1(t - t_j)\right)^2$$

where $K(u)$ is a function that decreases as $u$ gets large, down-weighting observations far away from the point $t$. Constant $h$, referred to as the *bandwidth* controls how quickly this down-weighting takes in; just like $\lambda$ in the basis expansions above, as $h$ gets small we fit points only very near to $t$, when $h$ is very large we tend to linear regression. Here, $\beta_0(t)$ estimates $\mathbf{x}(t)$, but $\beta_1(t)$ provides an estimate of it's slope, i.e. $\mathrm{D}\mathbf{x}(t)$ that does not necessarily equate to the derivative of $\beta_0(t)$. Since these do yield estimates that are linear in $y$, they can be plugged into our methods above.

A variety of other smoothing methods can also be used. Pascual and Ellner (2000) used neural networks; earlier literature such as Varah (1982) or Bellman and Roth (1971) used splines without a penalty, but in which the knots were chosen adaptively, often by hand.

### 8.6.2 Numerical Discretization Methods

Wu et al. (2012) proposed that, rather than fit $\mathbf{f}(t, \mathbf{x}, \theta)$ to $\mathrm{D}\mathbf{x}(t)$, we could instead consider finding parameters to make a numerical solution fit the smooth as well as possible. To illustrate, they consider evaluating $\mathbf{x}(t)$ at points $t_1, \ldots, t_m$ which are $h$ units apart; they then try to make an Euler update fit this smooth as well as possible:

$$\hat{\boldsymbol{\theta}} = \operatorname{argmin} \sum_{i=1}^{d} \sum_{j=1}^{n} w_{ij} \left[ x_i(t_j + h) - x_i(t_j) - h f_i(t_j, \mathbf{x}, \theta) \right]^2 . \tag{8.8}$$

Beyond taking just a simple Euler update, they can also examine updates based on Runge Kutta schemes. Using Runge Kutta methods, $m$ can be taken to smaller, relative to $n$, than for the gradient matching we have examined above. When $m$ is quite large (hence $h$ is small), there is little difference in these objective functions after re-scaling by $1/h^2$ so that the methods of inference described above can still be employed.

A significant advantage of this approach is that it avoids having to estimate $\widehat{\mathrm{D}\mathbf{x}}$, something that adds variability to parameter estimates. However, by dividing the term inside the square by $h$ in (8.8), we can see that we are essentially producing a finite difference approximation to $\mathrm{D}\hat{\mathbf{x}}$ which may not perform much better. We also do not so directly access the diagnostics discussed in Sect. 8.4.

### 8.6.3 Unobserved Covariates

One of the largest disadvantages of gradient matching is that we must have observations of every state variable, and we must observe these with enough frequency and accuracy to be able to estimate derivatives. This is very often not the case. When this happens there are two possible ways to deal with this (at least within gradient matching):

1. Re-express the equation in terms of a higher-order equation. In principle, this is always possible, according to the literature on *attractor reconstruction* (see Arbabanel 1996; Kantz and Schreiber 2003). For example the FitzHugh-Nagumo equations which we discussed in Chap. 7 can be written either as

$$\begin{aligned} \mathrm{D}V &= c(V - V^3/3 + R) \\ \mathrm{D}R &= -(V - a - bR)/c \end{aligned}$$

   in which $R$ is unobserved, or as

$$\mathrm{D}^2 V = c(1 - V^2)\mathrm{D}V - V + a + bR$$

which we achieve by differentiating the first equation and substituting the second for D$R$. However, this is not always algebraically so simple, and can require a large number of derivatives.

2. We can also represent missing components via a basis expansion whose coefficients we optimize along with $\boldsymbol{\theta}$. That is, if we divide $\mathbf{x}(t)$ into $(\mathbf{x}_o(t), \mathbf{x}_m(t))$ to represent observed and missing components, we can represent $\hat{\mathbf{x}}_m(t) = \boldsymbol{\phi}(t)^T C_m$ and minimize $\texttt{ISSE}(\boldsymbol{\theta}, C_m)$ over both. This will bring $\hat{\mathbf{x}}_m(t)$ as much into line with $\mathbf{x}_o(t)$ and the ODE as possible.
   Naturally, this involves a large number of additional parameters and so potentially adds considerable uncertainty. Similar ideas have been explored in the integral matching methods in Sect. 8.7 below. These ideas have connections to the profiling methods explored in Chap. 9, which we think provides a better solution to these problems.

There has been very little exploration of the relative benefits of these approaches. We do note that they produce quite different representations of lack of fit in the ODE, and therefore different degrees of uncertainty in parameter estimates if the trajectories are treated as random.

### *8.6.4 Nonparametric Models*

Besides missing state variables, basis expansions can also be used for non-parametric components that vary either with time or with state variables. Often, the structure of a model may be known, but we may be less sure of our ability to estimate a specific component. For example, in the Rosenzweig–MacArthur model above, we might be agnostic about the saturation effect of the prey: $C/(\chi_B + C)$, we could instead replace this with a non-parametric estimate $g(C) = \boldsymbol{\phi}(C)\mathbf{d}$, say, where $\mathbf{d}$ represents a vector of coefficients. The model is then written as

$$\begin{aligned} \mathrm{D}C &= \rho C(\chi_C - C) - Bg(C) \\ \mathrm{D}B &= \gamma Bg(C) - \delta B \ . \end{aligned}$$

This type of approach was taken in Ellner et al. (2002) and Cao et al. (2008) among others.

We can also allow some parameters to vary over time. This can be parametric (see Hooker et al. 2011, for an example) or it can be allowed to vary nonparametrically. A particular example of this arises out of the Functional Data Analysis literature (Ramsay and Silverman 2005). If we have multiple replicates of a process $\mathbf{x}_1(t), \ldots, \mathbf{x}_n(t)$, Ramsay (1996) described *principal differential analysis* as estimating a model of the form

$$\mathrm{D}\mathbf{x}_i(t) = \mathbf{B}(t)\mathbf{x}_i(t) + \boldsymbol{\eta}_i(t) \ .$$

That is $\mathrm{D}\mathbf{x}_i(t)$ is linear in $\mathbf{x}_i(t)$ but the coefficients change over time. When we have replicate observations, we can estimate $\mathbf{B}(t)$ through gradient matching just by performing linear regression at each time point $t$[1]. This can be particularly useful as a way of exploring dynamics as a means of developing a more mechanistic model.

One doesn't have to perform gradient matching in order to estimate non-parametric components. Paul et al. (2011) included non-parametric estimates of root-growth responses while performing trajectory matching; see also Hooker (2009), Hooker and Biegler (2007).

### *8.6.5 Sparsity and High Dimensional ODEs*

Exploratory analyses of ODE models can also be conducted. Particularly in the field of systems biology, hundreds of proteins, or even more, can be involved in a complex dynamical network – some inhibiting or accelerating the production of others. Because we don't know a lot about these relationships, it makes sense to start with linear model:

$$\mathrm{D}\mathbf{x}(t) = B\mathbf{x}(t) \ .$$

However, we also don't expect that any given protein is affected, or affects, very many others – ie, we expect most of the entries in $B$ to be zero. The LASSO (Tibshirani 1996) is a statistical method designed to automatically set many parameters to zero; in the context of gradient matching it involves minimizing the penalized squared error

$$\mathtt{ISSE}(B) + \lambda \sum_i \sum_k |b_{ik}|$$

where the non-differentiability of the absolute value at zero tends to make many of the minimizing values of $B$ exactly zero. These ideas were explored in Lu et al. (2011); one cannot simply import LASSO methods to gradient matching directly but they were able to identify relationships between groups of similar proteins.

There are many variants on the LASSO, few of which have been examined in the context of dynamical systems, but many of which may potentially be applied where high-dimensional measurements are made over time.

[1] Although Ramsay (1996) added some smoothing penalties to regularize $\mathbf{B}(t)$ and this can be helpful

## 8.7 Integral Matching

The idea of matching derivatives has a complement in that of matching integrals. We can re-write the ODE specification by obtaining the indefinite integral on each side of the equation:

$$\mathbf{x}(t) = \mathbf{x}_0 + \int_0^t \mathbf{f}(\mathbf{x}(t), \boldsymbol{\theta})\mathrm{d}t \ .$$

Here we can approximate the integral in terms of either the smooth or the observed data, yielding an objective one of the following forms, where we have also indicated our numerical approximate to them

$$\sum_{i=1}^{d} \int_0^T w_i(t) \left( \hat{x}_i(t) - x_{i0} - \int_0^t f_i(\hat{\mathbf{x}}(s), \boldsymbol{\theta}) ds \right)^2 \mathrm{d}t \tag{8.9}$$

$$\approx \sum_{i=1}^{d} \left(X_i - 1x_{i0} - U F_i(\boldsymbol{\theta})\right)^T \mathbf{W}_i \left(X_i - 1x_{0i} - U F_i(\boldsymbol{\theta})\right)$$

or

$$\sum_{j=1}^{n} \sum_{i=1}^{d} w_{ji}^* \left( y_{ji} - x_{0i} - \int_0^{t_i} f_i(\hat{\mathbf{x}}(s), \boldsymbol{\theta}) ds \right)^2 \tag{8.10}$$

$$\approx \sum_{i=1}^{d} \left(Y_i - 1x_{0i} - U^* F_i(\boldsymbol{\theta})\right)^T W_i^* \left(Y_i - 1x_{0i} - U^* F_i(\boldsymbol{\theta})\right)$$

where $\hat{\mathbf{x}}(t)$ is any approximation to $\mathbf{x}(t)$ from smoothing the data as above. In our approximations, $U$ and $U^*$ are matrices whose rows calculate approximations to the integral $\int_0^t f_j(\hat{\mathbf{x}}(s), \boldsymbol{\theta}) ds$ where $t$ is either a quadrature point in (8.9) or an observation time in (8.10). As above, $\mathbf{W}_i$ is a diagonal weight matrix that both incorporates the weight function $w_i(t)$ and the quadrature rule.

There are numerous variations on this method, dating back to Himmelblau et al. (1967) – see Yermakova et al. (1982), Vajda et al. (1986), Font and Fabregat (1997), Dattner and Klaassen (2015), Dattner and Gugushvili (2015), involving different smoothing methods, different matching criteria and different means of performing the integration. The method is often referred to as the *direct integral* approach or the *HJB* method for it's first originators; however, we like the term *integral matching* to unify it with trajectory and gradient matching. The reader will note that it shares many of the same advantages of gradient matching: we do not need to solve an ODE

and the minimization task is often easier. Like gradient matching, if $\mathbf{f}$ is linear in the parameters, we can obtain their estimates explicitly (up to evaluating integrals). As a distinct advantage, it also avoids evaluating D$\mathbf{x}$, improving the stability of the estimate. As a disadvantage, $\mathbf{x}_0$ must be estimated again – either as an intercept in a nonlinear model, or from the smooth (we take the former option here) – and it is difficult to perform visual diagnostics and model improvement without returning to an estimation of D$\mathbf{x}$.

It is possible to conduct the same variance calculations as we performed above. In this case, we need to calculate values for the second derivative with respect to both $\boldsymbol{\theta}$ and $\mathbf{x}_0$. We will do this by first writing

$$\mathbf{G}_i(\boldsymbol{\theta}) = \left[ U \partial_{\boldsymbol{\theta}} F_i(\boldsymbol{\theta}) \; E_i \right]$$

where $E_i$ is a zero matrix apart from 1's on the $i$th column do indicate the $i$th component of $\mathbf{x}_i$. The second derivative is then

$$H(\boldsymbol{\theta}) = \sum_{i=1}^{d} \mathbf{G}_i(\boldsymbol{\theta})^T \mathbf{W}_i \mathbf{G}_i(\boldsymbol{\theta})$$

where an equivalent formulation for the second derivative of (8.10) can be obtained by using the appropriate starred quantities.

Setting $\boldsymbol{\vartheta} = (\boldsymbol{\theta}, \mathbf{x}_0)$ as in Chap. 7, we can approximate

$$\hat{\boldsymbol{\vartheta}} \approx \boldsymbol{\vartheta} - H(\boldsymbol{\theta})^{-1} \sum_{i=1}^{d} \mathbf{G}_i(\boldsymbol{\theta})^T \mathbf{W}_i (\hat{X}_i - 1 x_{0j} - U F_i(\boldsymbol{\theta})) \; ,$$

from which we can derive a variance as in Sect. 8.5 from the approximation

$$\hat{X}_i - 1 x_{0i} - U F_i(\boldsymbol{\theta}) \approx (\hat{X}_i - E\hat{X}_i) - \sum_{k=1}^{d} \mathrm{diag}([U \partial_{\mathbf{x}} F_j]_k)(\hat{X}_k - E\hat{X}_k) \; .$$

Thus writing

$$J_i(\boldsymbol{\theta}) = H(\boldsymbol{\theta})^{-1} \left( \mathbf{G}_i(\boldsymbol{\theta}) \mathbf{W}_i - \sum_{k=1}^{d} G_k(\boldsymbol{\theta}) W_k \mathrm{diag}([U \partial_{\mathbf{x}} F_k(\boldsymbol{\theta})]_j) \right)$$

we have

$$\text{var}(\hat{\boldsymbol{\theta}}|\boldsymbol{\eta}) = V = \sum_{i=1}^{d}\sum_{k=1}^{d} J_i(\boldsymbol{\theta})^T \text{cov}(\hat{X}_i, \hat{X}_k) J_k(\boldsymbol{\theta}) \ .$$

Our variance estimate is slightly different in the case of (8.10), where we write

$$Y_i - 1x_{0i} - U F_i(\boldsymbol{\theta}) \approx Y_i - \sum_{k=1}^{d} \text{diag}([U \partial_{\mathbf{x}} F_i(\boldsymbol{\theta})]_k)(\hat{X}_k - E\hat{X}_k)$$

so that in this case we must account for

$$\text{cov}(Y_i, \hat{X}_k) = \sigma_{ik} \boldsymbol{\Phi} \left(\boldsymbol{\Phi}^T \boldsymbol{\Phi} + \lambda P\right)^{-1} \boldsymbol{\Phi}^T$$

if we employ a penalized smoothing method to estimate $\hat{X}_k$. We thus need to modify our variance estimate $\text{var}(\hat{\boldsymbol{\theta}})$ (with analogous starred quantities) to specify

$$J_i^*(\boldsymbol{\theta}) = H^*(\boldsymbol{\theta})^{-1} \left( - \sum_{k=1}^{d} G_k^*(\boldsymbol{\theta}) W_k^* \text{diag}([U^* \partial_{\boldsymbol{x}} F_i(\boldsymbol{\theta})]_k) \right)$$

$$\sum_{i=1}^{d}\sum_{k=1}^{d} \left[ G_i^*(\boldsymbol{\theta})^T \mathbf{W}_i^* \ J_i^*(\boldsymbol{\theta})^T \right] \begin{bmatrix} \text{cov}(Y_i, Y_k) & \text{cov}(Y_i, X_k) \\ \text{cov}(X_i, Y_k) & \text{cov}(X_i, X_k) \end{bmatrix} \begin{bmatrix} W_k^* G_k^*(\boldsymbol{\theta}) \\ J_k^*(\boldsymbol{\theta}) \end{bmatrix} \ .$$

We illustrate this on our simple refinery data first. In this case, the system is sampled very frequently and at unit intervals, meaning that we can use the observation times as quadrature points and approximate the integral via a cumulative sum:

$$\int_0^{t_j} \mathbf{f}(\hat{\mathbf{x}}(t), \boldsymbol{\theta}) \text{d}t \approx \sum_{i=1}^{i} \mathbf{f}(\mathbf{x}(t_j), \boldsymbol{\theta})$$

thus writing $Z$ to be the $n \times 4$ matrix obtained by adding a column of 1's to the (column-wise) cumulative sum of $X$ from Sect. 8.5, we have $G(\boldsymbol{\beta}) = Z$ and

$$J(\boldsymbol{\beta}) = (Z^T Z)^{-1} Z^T (I - \text{diag}(\mathbf{t}))$$

where $\mathbf{t}$, is the vector of observation times. Note that since we have just projected the data onto a basis system, there is algebraically no difference between using (8.9) and (8.10) (this is left as an exercise for the curious reader).

In Fig. 8.6 we have compared plots of predicted versus fitted values for both gradient and integral matching as well as their process residuals. Here we see much better agreement on the integral than the derivative scale. Table 8.2 replicates Table 8.1 but for integral matching where we see similar results (with $x_0$ an additional estimated parameter) but with somewhat smaller variances.

As with gradient matching, missing components can be modeled via a basis expansion using additional parameters; it is possible to convert an ODE to higher-order, but the equivalent methods for higher-order ODEs are somewhat more complicated.

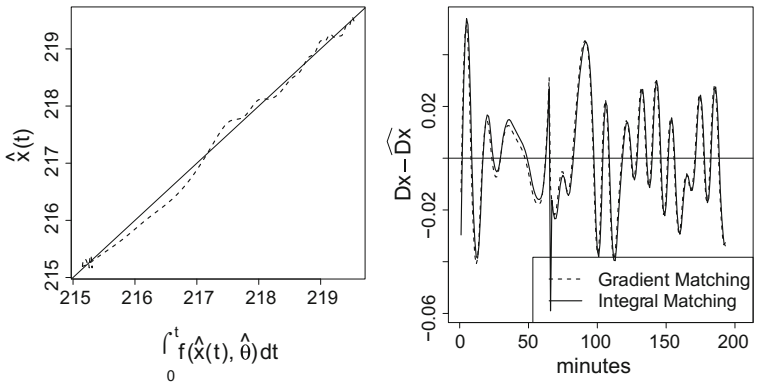


**Fig. 8.6** Integral matching on the refinery data. The *left plot* shows $\hat{x}(t)$ versus $\int f(\hat{x}, \boldsymbol{\theta})\mathrm{d}t$. Here we observe a much closer relationship than we had for gradient matching. The *right plot* shows $\widehat{\mathrm{D}x} - f(\hat{x}, \boldsymbol{\theta})$ for both gradient and integral matching. These give virtually identical results

**Table 8.2** Variances and confidence intervals after integral matching on the refinery data

| | Estimated variance | Confidence intervals | |
|---|---|---|---|
| | | Lower | Upper |
| $x_0$ | $2.592 \times 10^{-4}$ | 215.2779 | 215.3423 |
| $\beta_0$ | $4.204 \times 10^{-2}$ | 8.3517 | 9.1718 |
| $\beta_1$ | $3.422 \times 10^{-7}$ | −0.0225 | −0.0201 |
| $\alpha$ | $1.629 \times 10^{-5}$ | −0.2128 | −0.1967 |

## 8.8 Applications: Head Impacts

Here we re-examine the head impact data from Chap. 7. Recall that these data are described by a second-order system

$$\mathrm{D}^2 x = \beta_1 x + \beta_2 \mathrm{D}x + \beta_3 u(t) \tag{8.11}$$

where $u(t)$ is the indicator function for $14 < t < 17$. W can also write this model as a first order system, but with two state variables

$$\mathrm{D}\begin{pmatrix} x \\ v \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ \beta_1 & \beta_2 \end{pmatrix}\begin{pmatrix} x \\ v \end{pmatrix} + \begin{pmatrix} 0 \\ u(t) \end{pmatrix} .$$

However, in this representation, we only have measurements of $x$ and not $v$.

Treating a second order system is relatively straightforward in gradient matching; we simply estimate $(\widehat{\mathrm{D}^2 x}, \widehat{\mathrm{D}x}, \hat{x})$ and treat (8.11) as a linear regression to estimate $(\beta_1, \beta_2, \beta_3) = c(-0.062, -0.135, -7.757)$. Figure 8.7 shows the result of this. If we wished to assess the variability of the resulting parameter estimates, the approaches above can be readily extended for this case.

An alternative approach to this problem is to use the extended representation of the system in terms of $x$ and $v$. Here we face the problem that $v$ is unobserved. However, we can instead represent its value via a basis expansion: $\hat{v}(t, \mathbf{c}_v) = \boldsymbol{\phi}(t)^T \mathbf{c}_v$ in which $\mathbf{c}_v$ are taken to be further parameters to estimate within the gradient matching criterion:

$$\begin{aligned} \texttt{ISSE}(\beta_1, \beta_2, \beta_3, \mathbf{c}_v) = &\int \left(\widehat{\mathrm{D}x}(t) - \boldsymbol{\phi}(t)^T \mathbf{c}_v\right)^2 \mathrm{d}t + \\ &\int \left(\mathrm{D}\mathbf{c}_v' \boldsymbol{\phi}(T) - \beta_1 \hat{x}(t) - \beta_2 \boldsymbol{\phi}(t)^T \mathbf{c}_v - \beta_3 u(t)\right)^2 \mathrm{d}t \ . \end{aligned}$$

Here, the terms in the integrals could be weighted differently, but we have simply used them together. The minimizers of $\texttt{ISSE}(\beta_1, \beta_2, \beta_3, \mathbf{c}_v)$ result in parameter estimates $(\beta_1, \beta_2, \beta_3) = c(-0.062, , -0.121, -7.749)$. and have been used to create the plots in the bottom row of Fig. 8.7 where, our results are not too different to fitting the second order model.

While these results are not very different in this case, it is important to note that there are real differences between these choices. Most importantly, using a *system* of equations relaxes two relationships: the equations for $x$ and for $v$. Thus, model errors can occur in both terms. In the second-order system, we only see model errors in one equation. If these are treated as random forcing terms, as above, it could have considerable impact on our estimates. Of course, if we give much more relative weight to the first integral in $\texttt{ISSE}(\beta_1, \beta_2, \beta_3, \mathbf{c}_v)$ we could expect to return to fitting $\widehat{\mathrm{D}^2 x}$.

We have not included variance estimates here. These could be obtained by simulation. In the case of fitting (8.11) directly, the approaches discussed for first-order models can be readily extended. When estimating entire state vectors in our second approach, the variance properties of this estimate are less clear.

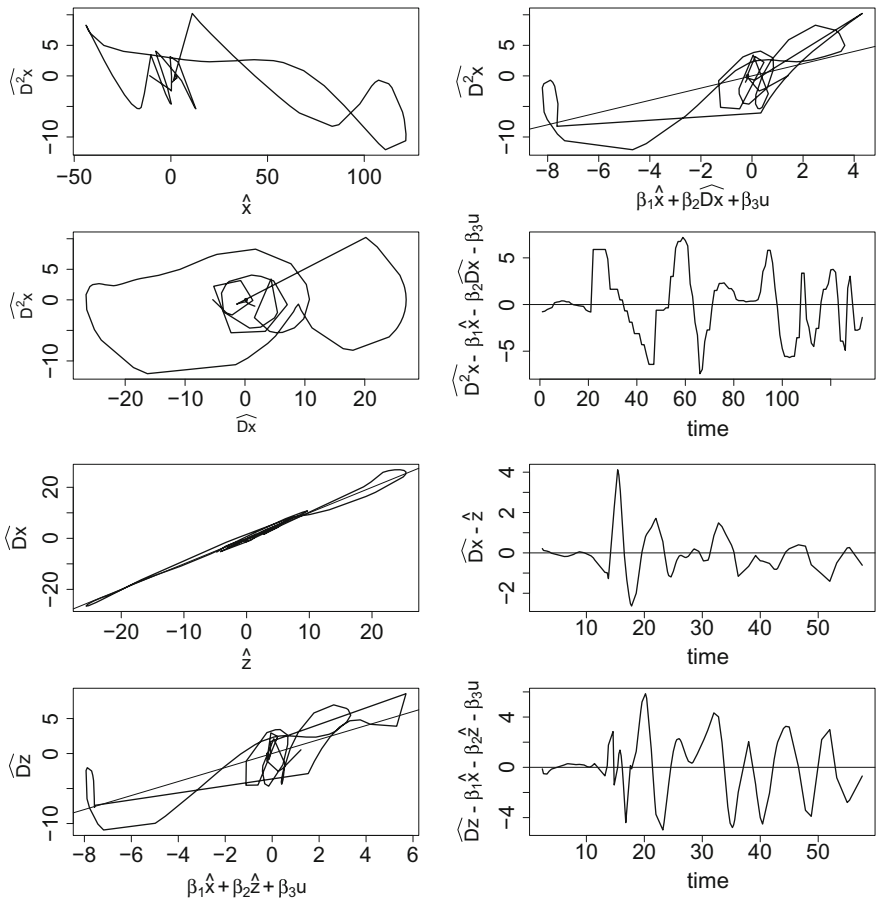


**Fig. 8.7** Fitting the head impact data. *Top panel* fitting a second-order ODE. *Left panel* plots $\widehat{\mathrm{D}^2 x}$ versus $\widehat{\mathrm{D}x}$ and $\hat{x}$; *right panel* plots $\widehat{\mathrm{D}x}$ versus fitted values (*top*) and the process residuals (*bottom*). *Bottom panel* fitting a two-dimensional first order ODE system with $v$ unmeasured. *Left panel* plots observed versus fitted values for $\widehat{\mathrm{D}x}$ and $\widehat{\mathrm{D}z}$, *right panel* gives process residuals for both $\hat{x}$ and $\hat{z}$. Here the two methods mostly differ in whether there is one source of system error (on $\widehat{\mathrm{D}^2 x}$), or two

Finally note that in our original development of the model, we knew the initial conditions of the state $x(0)$ and $\mathrm{D}x(0)$ from the experimental setup. This information is not available to gradient matching, but could be enforced in the smoothing step. In the direct integral approach, we would have to evaluate nested indefinite integrals – something that would be difficult – but could similarly hard-code initial conditions if they are known.

# Chapter 9
# Profiled Estimation for Linear Systems Estimated by Least Squares Fitting

## 9.1 Introduction and Chapter Overview

We now describe an approach to parameter estimation and statistical inference called *parameter cascading* or *generalized profiling* that combines the virtues of both the nonlinear least squares method or trajectory matching described in Chap. 7 and the gradient matching approach in Chap. 8, but in a way that avoids the disadvantages of both. We use $\boldsymbol{\theta}$ and the term "parameters" for the vector of all the constants that define the differential equation system and that must be estimated from the data. That is, there may be other constants defining the equation(s) that are regarded as known and held fixed in any analysis, and we ignore them as parameters.

Parameter cascading defines the estimation problem as a function of only the unknown parameters in one or more differential equations, and yet simultaneously estimates the solutions $\mathbf{x}$ as a byproduct of the computations. In comparison with the nonlinear least squares method discussed in Chap. 7, the computational overhead is greatly reduced; and unlike the derivative matching approach, differential equations with partially observed or indirectly observed variables are accommodated. The method shares with gradient matching in Chap. 8 the use of basis function expansions to represent the solution functions $x_i$. A number of other advantages come with parameter cascading as well, including having a continuum of choices between fitting the data and solving the differential equation.

This chapter begins with a conceptual introduction to parameter cascading or generalized profiling estimation. In this and the next next three sections only the single equation case is considered in order to keep the notation as simple as possible. Sections 9.3 and 9.4 look at the selecting the roughness penalty $\rho$ and the computation of confidence intervals for parameters. Up to this point only a single equation has been involved, but the next Sect. 9.5 generalizes the parameter cascading estimation strategy to the multiple equation context. Four case study sections follow illustrating aspects of data fitting by linear dynamical systems. The final section shows how an alternative compact shape-based basis system can be constructed once the dynamical system is estimated.



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_9

As a further aid to understanding how parameter cascading works, we will weave into the narrative the account of an analysis of a single set of data generated from the simplest unforced first order equation $\mathrm{D}x = -\beta x$, the solution to which is $x(t) = C\exp(-\beta t)$. We remind ourselves here that the solution has *two* parameters while the differential form has only one. That is, the differential equation defines a *linear space* of solutions corresponding to the possible values of $C$. Scale parameter $C$ would be defined if we knew the initial value $x(0) = C$ or, as is more usual in statistical investigations we had a reasonable amount of noisy data distributed over the observation interval. Our "data" here will be $N = 21$ values generated by adding independent normally distributed errors with mean 0 and standard deviation 0.1 to the function $\exp(-\beta t)$ at 21 equally spaced locations over [0,4], so that $\beta = C = 1$. One such simulated sample, the analysis of which we will consider in detail, is displayed in Fig. 9.1. We, in fact, generated and analyzed 500 such samples, and will also report results averaged over these simulated data samples.

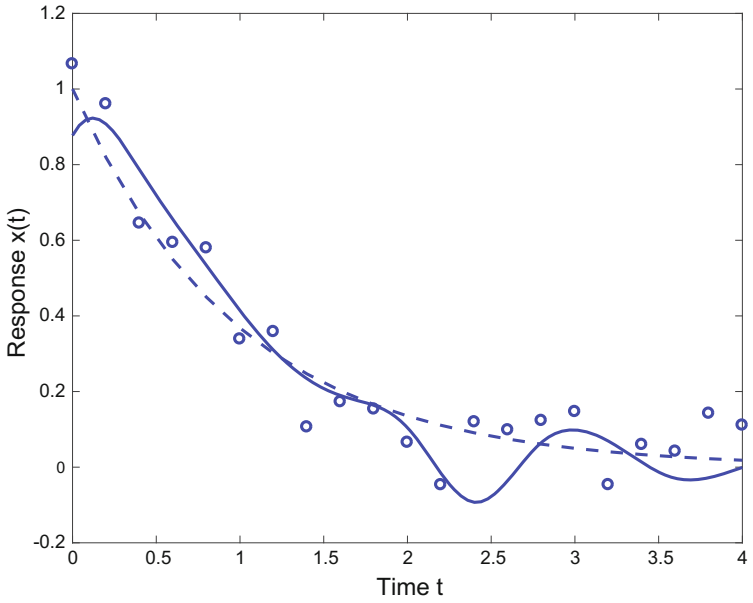


**Fig. 9.1** An exponential decay process $x(t) = C\exp(-\beta t)$ for which $C = \beta = 1$. The *circles* are observations at 21 equally spaced time points simulated by adding independent Gaussian errors with mean 0 and standard deviation 0.1. The *solid curve* shows a least squares fit constructed using 13 order 4 B-spline basis functions. The *dashed curve* is the errorless version

## 9.2 The Parameter Cascading Strategy for Estimating Parameters

The elements of parameter cascading in the context of a single differential equation are:

- Approximate the solution $x(t)$ to the differential equation by the basis function expansion

$$\hat{x}(t) = \sum_{k}^{K} c_k \phi_k(t) = \mathbf{c}^T \boldsymbol{\phi}(t) \; . \tag{9.1}$$

- Constrain the coefficient vector $\mathbf{c}$ to be a smooth function $\mathbf{c}(\boldsymbol{\theta})$ of the parameters in vector $\boldsymbol{\theta}$ that define the differential equation.
- Optimize the fit to observations $y_1, \ldots, y_n$ in vector $\mathbf{y}$ by $\hat{x}(\mathbf{t}) = \mathbf{c}(\boldsymbol{\theta})^T \boldsymbol{\phi}(\mathbf{t})$ with respect to $\boldsymbol{\theta}$ by optimizing a fitting criterion $H(\boldsymbol{\theta}|\mathbf{y})$.

Figure 9.1 shows the least squares fit to the data using $K = 13$ order 4 B-spline basis functions with equally spaced knots at $0, 0.4, 0.8, \ldots, 4.0$. The recovery of the true function in this way is poor, but this number of spline bases can fit the true exponential curve with a maximum error of 0.00002.

### *9.2.1 Two Classes of Parameters*

Parameter cascading pivots around the idea of replacing the coefficients $c_k$ in the basis expansion by values of a smooth function of the parameters in $\boldsymbol{\theta}$. In effect, the status of the $K$-dimensional coefficient vector $\mathbf{c}$ as a set of parameters is eliminated. Instead, the coefficient estimates are actually the values of the function $\mathbf{c}(\hat{\boldsymbol{\theta}})$ at the optimal value of the $\boldsymbol{\theta}$. But why would we want to do this?

The parameters in $\boldsymbol{\theta}$ typically impact the trajectory over either the entire interval $[0, T]$, or at least over large portions of it. As a rule we regard the number of such parameters as fixed, and in our toy example we have only one parameter $\beta$. By contrast, we have in our analysis 13 coefficients $c_k$, and there is no particular reason why this number might not be either larger or smaller.

The parameters also often stand for something substantively interesting in the processes generating the data, as is the speed of decline in our simple example, and their estimated values are therefore important to our clients and to understanding the model. Statisticians refer to these as *structural* parameters, although the designation *global* might also be appropriate.

By contrast, any of the $K$ coefficients $c_k$ defining the basis function expansions of the variable $x$ usually defines only a small piece of the trajectory. For spline bases, for example, each $c_k$ is approximately proportional to the height of $x_i$ at the center of the small interval over which the $k$th spline basis function is positive. In our example, the value of $c_6$ contributes to the definition of the fitted curve only over the interval

[0.4, 2.4]. Typically a client researcher would prefer to just examine a plot of the estimated trajectory $x$ rather than the sizes of each of the coefficients.

The number $K$ of coefficients $c_k$ may be orders of magnitude larger than the number of structural parameters, as it is in our demonstration example. Moreover, the numbers of these coefficients tend to be tied to the amount of data collected for each observed variable. It is common to place a spline function knot at the location of each observation, so that if a client arrives with more data, the number of coefficients will increase proportionately. In our demo example, this would increase the number of coefficients from 13 to 23.

The coefficients $c_k$ are required in a model to accommodate *local* variation so as to leave more global types of variation to be explained in terms of structural parameters. Parameters like these are often referred to as *incidental* or *nuisance* parameters, but the less pejorative term *local* seems appropriate, too.

The central problem for a model requiring large numbers of local parameters is how to prevent over-fitting the data, so that enough variation is left to guarantee useful and efficient estimation of the structural parameters that are of primary interest. For example, we can always make the basis size $K$ large enough to exactly fit each observation; and indeed, we are here counselling making $K_i$ much larger on occasion to capture sharp variation in the functions in the equation solution space. Moreover, interpolating the data would also interfere with the inspection of the shapes of the $x_i$'s, and certainly with the usefulness of their derivatives. That is, a data–interpolation or a high dimensional data smoothing of $x$ when the data are noisy, cannot be close to a solution of its corresponding differential equation because the equations imply a much stronger type of smoothness. This is certainly so for the solid curve in Fig. 9.1, which is a least squares fit to the data using only the 13 B-spline basis functions. The data smooth does not look much like the true curve, and the fit of its first derivative to the true derivative $-\beta \exp \beta t$ is much worse.

A secondary but important reason for using parameter cascading is that the computation required to optimize the fit of the model to the data is greatly reduced since the final fit depends only on the smaller number of parameters, rather than on sum of the number of parameters and the number of coefficients.

We will see over and over again that forcing $\mathbf{c}(\boldsymbol{\theta})$ to be a smooth function of the parameters will results in better parameter estimates with smaller confidence intervals without seriously harming the fit to the data that $\mathbf{c}(\boldsymbol{\theta})$ defines. For examples in this Chapter, have a look at Figs. 9.5, 9.11 and 9.12 as well as our personal favorite Sect. 1.6.

### 9.2.2 *Defining Coefficients as Functions of Parameters*

A natural approach to defining $\mathbf{c}(\boldsymbol{\theta})$ is to define an *inner fitting criterion* $J(\mathbf{c}|\boldsymbol{\theta})$ that optimizes $\mathbf{c}$ given any candidate value for the parameters in $\boldsymbol{\theta}$, and that is re-optimized *each time* $\boldsymbol{\theta}$ is changed in order to optimize an *outer fitting criterion* $H(\boldsymbol{\theta})$. That is, the functional relation is *implicitly* defined by specifying the $\mathbf{c}$-optimality for $J$ as

*conditional* on a prior choice for $\boldsymbol{\theta}$. We are, in effect, placing a higher priority on the estimation $\boldsymbol{\theta}$ and a lower priority on the estimation of $\mathbf{c}$ and the $x(t)$ that it defines.

But there are common situations where this strategy can be short-circuited by working out the optimizing coefficients in the function $J$ explicitly, and one of these is the use of penalized least squares for $J(\mathbf{c}|\boldsymbol{\theta})$. Before we do that, however, we need to look for a way of blending together the information in the data and the information in the equation.

### 9.2.3 The Symmetric Relation Between the Data and the Differential Equation

Most dynamic models for observed systems are only partially founded upon basic scientific principles, usually because a complete model would be too complex to identify given the amount of data that are available. The simple second order equation used for the head impact data, for example, ignores the wide variation in the mechanical properties of the brain tissue being deformed, as well as the bone material transmitting the shock to the soft tissue inside the skull.

Or it may be that many alternative processes, such as chemical reactions, can lead to the same end result. Typically there are no data available to tease apart what fraction of processes is due to any specific case. Or, finally, the goal of the modeller may be fundamentally humble: Just capturing the prominent shape features can be an important achievement. We see this often in physiological studies, such as the Nobel Prize winning Hodgkins–Huxley neural spike potential model.

In these circumstances, we don't wish to put all of our money on the equation. In order to control this relative emphasis, we fix a value $\rho \in [0, 1)$, called a *smoothing parameter* or a *bandwidth parameter*, that modulates the relative emphasis on fitting the data as opposed to satisfying the differential equation. By choosing $\rho$ judiciously, we can see how much of the variation in the data can be accommodated by what is perhaps a too-simple dynamic model. If this is possible, then the residual variation is apt to give important cues about either how the model is to be elaborated, or about what kind and quantity of data would be more revealing.

### 9.2.4 Inner Optimization Criterion $J$

Our first task is to define the inner optimization function $J(\mathbf{c}|\boldsymbol{\theta}, \rho)$. If we choose to use the familiar squared residual fit measures, then

$$J(\mathbf{c}|\boldsymbol{\theta}, \rho) = (1-\rho)\sum_{j}^{n}[y_j - x(t_j)]^2/n + \rho\int_0^T \{\mathrm{D}^m x(t) - f[x(t)]\}^2\,\mathrm{d}t/T. \quad (9.2)$$

The *smoothing* or *bandwidth* parameter $\rho$ is in the half-closed interval $[0, 1)$. If $\rho = 0$, we are ignoring the differential equation entirely and concentrating solely on data-fitting; which, depending on the number $K$ of basis functions, can amount to either least squares curve fitting or to data interpolation. But as $\rho \to 1$, we place more and more emphasis on $x$ being close to a solution to the differential equation. Whether the solution is exact or not will depend on whether the basis system is powerful enough to capture all the shape variation that an exact solution will require. We define the right side of the $\rho$-interval to be open since will surely not want to ignore the data entirely, primarily because a solution to most differential equations is not uniquely defined without some additional information, and the data provides this information so long as $\rho < 1$.

### 9.2.5 The Least Squares Cascade Coefficient Function

As is usual in statistics, working with least squares criteria brings a lot of mathematical convenience. Let us assume for a couple of paragraphs that there are no forcing functions, so that the differential equation residual function $\mathrm{D}^m x(t) - f[x(t)|\boldsymbol{\theta}]$ in (9.2) can be re-expressed as $\mathrm{L}x = 0$ where L is the differential operator form of the differential equation $\mathrm{D}^m x = f(x|\boldsymbol{\theta})$.

Using matrix notation, the fitting function is $\hat{x}(t) = \mathbf{c}^T\boldsymbol{\phi}(t)$, where the coefficient vector $\mathbf{c}$ and basis function vector $\boldsymbol{\phi}$ are of length $K$. Let $n$ by $K$ matrix $\boldsymbol{\Phi}$ contain the values of the basis functions evaluated at the observation points. Then inner criterion (9.2) becomes

$$\begin{aligned} J(\mathbf{c}|\boldsymbol{\theta}) &= (1-\rho)(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})^T(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})/n + \rho\int_0^T [\mathbf{c}^T\mathrm{L}\boldsymbol{\phi}(t)][\mathbf{c}^T\mathrm{L}\boldsymbol{\phi}(t)]^T\,\mathrm{d}t/T \\ &= (1-\rho)(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})^T(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})/n + \rho\mathbf{c}^T\left(\int_0^T [\mathrm{L}\boldsymbol{\phi}(t)][\mathrm{L}\boldsymbol{\phi}(t)^T]\,\mathrm{d}t\right)\mathbf{c}/T \\ &= (1-\rho)(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})^T(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})/n + \rho\mathbf{c}^T\mathbf{R}(\boldsymbol{\theta})\mathbf{c}/T \end{aligned} \tag{9.3}$$

where

$$\mathbf{R}(\theta) = \int_0^T [\mathrm{L}\boldsymbol{\phi}(t)][\mathrm{L}\boldsymbol{\phi}(t)^T]\,\mathrm{d}t\ . \tag{9.4}$$

Note that order $K$ matrix $\mathbf{R}(\boldsymbol{\theta})$ is a function of the parameter vector $\boldsymbol{\theta}$ because the linear differential operator L depends on $\boldsymbol{\theta}$ through the right side function $f(x|\boldsymbol{\theta})$.

With $\mathbf{R}(\boldsymbol{\theta})$ in hand as well as the last equation in (9.3) it is a simple exercise in matrix calculus to work out that the minimizing value of $\mathbf{c}$ that satisfies the equation

$$\mathbf{c}(\boldsymbol{\theta}) = [(1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi}/n + \rho\mathbf{R}(\boldsymbol{\theta})/T]^{-1}(1-\rho)\boldsymbol{\Phi}^T\mathbf{y}/n\ . \tag{9.5}$$

Only a minor extension of (9.5) is required for forced linear systems where the differential equation has the operator form

$$\mathrm{L}x(t) = \sum_{\ell}^{\mathrm{L}^*} \alpha_{\ell|\boldsymbol{\theta}}(t) u_\ell(t) \ .$$

The parameter vector $\boldsymbol{\theta}$ now also contains whatever coefficients define the forcing function rates $\alpha_\ell$. Along with the order $K$ matrix $\mathbf{R}(\boldsymbol{\theta})$, we now define $K$ by $\mathrm{L}^*$ matrix $\mathbf{S}(\theta)$ to be

$$\mathbf{S}(\theta) = \int_0^T [\mathrm{L}\boldsymbol{\phi}(t)]\mathbf{u}^T(t)\,\mathrm{d}t \ . \tag{9.6}$$

The inner criterion is then

$$J(\mathbf{c}|\boldsymbol{\theta}) = (1-\rho)(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})^T(\mathbf{y}-\boldsymbol{\Phi}\mathbf{c})/n + \rho\mathbf{c}^T\mathbf{R}(\boldsymbol{\theta})\mathbf{c}^T/T + \rho\mathbf{c}^T\mathbf{S}(\boldsymbol{\theta})/T \ , \tag{9.7}$$

and we have that

$$\mathbf{c}(\boldsymbol{\theta}) = [(1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi}/n + \rho\mathbf{R}(\boldsymbol{\theta})/T]^{-1}[(1-\rho)\boldsymbol{\Phi}^T\mathbf{y}/n + \rho\mathbf{S}(\boldsymbol{\theta})/T] \ . \tag{9.8}$$

Equation (9.8) is a simple strategy for blending together two varieties of multiple regression. Both $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$ and $\mathbf{R}(\boldsymbol{\theta})$ are matrices of inner products, taken for the first term with respect to values of $\boldsymbol{\phi}$ at discrete sampling points and in the second term taken with integration over all values of $t$. Each is an analogue of $\mathbf{Z}^T\mathbf{Z}$ in the conventional regression problem $\mathrm{E}[\mathbf{y}] = \mathbf{Z}\boldsymbol{\beta}$. Moreover, $\boldsymbol{\Phi}^T\mathbf{Y}$ and $\mathbf{S}(\theta)$ are also cross-product structures representing the inner products of covariates with data. The main difference between the two problems is the source of the data. If the differential equation is homogeneous, the analogue of $\mathbf{Y}$ in the second term of $J$ is the zero function, since in effect there are no external variables to approximate. If, on the other hand there are one or more additive forcing functions, then the matrix $\mathbf{S}$ is the corresponding data factor that corresponds to $\mathbf{Z}^T\mathbf{y}$ in conventional regression.

Figure 9.2 shows how the coefficients $c_3(\rho)$, $c_6(\rho)$, and $c_9(\rho)$ vary as $\beta$ ranges from 0.5 to 1.5 for the relatively low smoothing parameters $\rho = 0.27$, and for the high value $\rho = 0.98$. As we move along these trajectories to find the minimizing values of $\beta$, we automatically specify a varying set of coefficients that identify $x$. That is, for any value of $\rho$, $\mathbf{c}(\beta)$ defines a space curve in 13-dimensional coordinate space; and the larger $\rho$ is, the smoother this curve will be. In the Figure we see a sharper curvature for $\rho = 0.27$ than we do for $\rho = 0.98$, reflecting the goal of defining $\mathbf{c}$ as a function of $\boldsymbol{\theta}$ that becomes more and more smooth as we increase $\rho$.

From (9.8) we can now see more clearly the linear relation between the fitting vector $\hat{x}(\mathbf{t})$ and the data in $\mathbf{y}$, namely that

$$\hat{x}(\mathbf{t}) = \boldsymbol{\Phi}\mathbf{c}(\boldsymbol{\theta}) = \mathbf{M}(\boldsymbol{\theta})\mathbf{y} + \mathbf{N}(\boldsymbol{\theta}) \tag{9.9}$$

where

$$\mathbf{M}(\boldsymbol{\theta}) = (1-\rho)\boldsymbol{\Phi}[(1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi}/n + \rho\mathbf{R}(\boldsymbol{\theta})/T]^{-1}\boldsymbol{\Phi}^T/n$$

and

$$\mathbf{N}(\boldsymbol{\theta}) = \rho\boldsymbol{\Phi}[(1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi}/n + \rho\mathbf{R}(\boldsymbol{\theta})/T]^{-1}\mathbf{S}(\boldsymbol{\theta})^T/T\ .$$

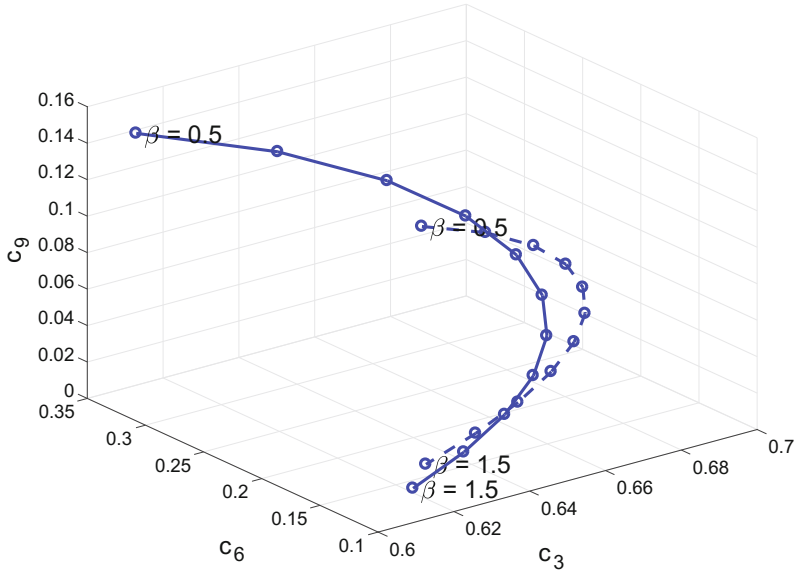


**Fig. 9.2** The *curves* display three of the 13 coefficient functions $c_3(\beta)$, $c_6(\beta)$, and $c_9(\beta)$ defining $x(t)$ varying over values of $\beta$ between 0.5 and 1.5 for the model and data in Fig. 9.1, and for values of $\rho$ of 0.27 (*dashed*) and 0.98 (*solid*)

The symmetric matrix function $\mathbf{M}(\boldsymbol{\theta})$, called the *smoothing map*, plays an important role in defining an equivalent degrees of freedom measure, the GCV criterion, and in computing confidence intervals for estimates of parameters in $\boldsymbol{\theta}$.

At first sight, the need for evaluating $\mathbf{R}(\boldsymbol{\theta})$ and $\mathbf{S}(\boldsymbol{\theta})$ each time $\boldsymbol{\theta}$ is changed by numerical approximations of the integrals defining them implies a major computational overhead. We will return to this problem later and show that almost all of the work in computing these matrices can be completed once and for all prior to optimization by a computational strategy called *memoization*.

The computation of $\mathbf{c}(\boldsymbol{\theta})$ depends critically on the non-singularity of the matrix $(1-\rho)\boldsymbol{\Phi}^T\boldsymbol{\Phi}/n + \rho\mathbf{R}(\boldsymbol{\theta})/T$. There is usually little concern with the cross-product matrix $\boldsymbol{\Phi}^T\boldsymbol{\Phi}$, and especially if B-spline basis functions are used, since in this case the matrix is band-structured and well-conditioned. For example, in our demo example this matrix is of order 13 and its eigenvalues range from 1.929 down to 0.89, and therefore is well conditioned.

But this is not so for $\mathbf{R}(\boldsymbol{\theta})$. For an $m$-order linear differential operator L, the equation $\mathrm{L}x = 0$ has a solution space of dimension $m$, and if our basis function expansion is sufficiently powerful, it will come close to being able to approximate functions in this solution set, called the *kernel* of operator L. For our demo example with $\beta = 1$, the smallest eigenvalue is about $10^{-8.5}$, the next largest is about $10^{-1}$, and the largest is about 1, which is exceedingly dangerous when $\rho \to 1$. Fortunately, as we shall see, values such as $\rho = 0.999$ are well within what is tolerable and can also give excellent approximations to exact solutions of the linear differential equation and at the same time of the parameters in $\boldsymbol{\theta}$ that define them.

### 9.2.6 The Outer Fitting Criterion $H$

We remain in the simpler single-equation context, where $\mathbf{y}$ is a vector of length $n$ containing the data and $x(\mathbf{t})$ is a vector of length $n$ containing the fitted values at observations times $t_j$.

To complete the parameter cascading strategy, we specify an *outer* fitting criterion,

$$H(\boldsymbol{\theta}|\rho) = G_0[\mathbf{y}, x(\mathbf{t})|\boldsymbol{\theta}, \rho] \; . \tag{9.10}$$

If we stick with least squares as our criterion, then

$$H(\boldsymbol{\theta}|\rho) = \sum_j [y_j - \hat{x}(t_j)]^2 = \sum_j [y_j - \mathbf{c}^T(\boldsymbol{\theta}|\rho)\boldsymbol{\phi}(t_j)]^2 \; . \tag{9.11}$$

The least squares criterion will no longer be a simple quadratic function of the unknown parameter values, but the optimization problem is nevertheless easier if we use the Gauss-Newton algorithm. The shape of the criterion function also varies with the level of $\rho$.

Figure 9.3 shows how $H(\boldsymbol{\theta}|\rho)$ varies over $\beta$ for our demo example using $\rho = 0.5, 0.73, 0.88, 0.95, 0.98, 0.99$. We see that the curvature is less and also more quadratic for the bottom curve ($\rho = 0.5$). As we increase $\rho$ the height of $H$ increases, its asymmetry increases, as does its curvature, but we also see that the minimizing values moves closer and closer to the true value $\beta = 1$.

Ideally, the choice of fit measure $G_0$ will depend on knowing something about the distribution of the data in vector $\mathbf{y}$, including how that distribution would depend on the fit vector $x(\mathbf{t})$. The choice of least squares is justified if:

1. The residual values $e(t_j) = y_j - x(t_j)$ have a Gaussian distribution given that $x(t_j) = \mathrm{E}[y_j]$ and that these values are independently distributed but with the same variance.
2. The differential equation correctly identifies the processes that have generated the data.

3. The basis system $\boldsymbol{\phi}$ has sufficient fitting power to represent the function $x$ and its derivatives up to order $m$ to a level of accuracy that allows us to ignore discrepancies between the values of the equation's right and left sides.
4. The smoothing parameter $\rho$ has been set to a value that is sufficiently close to 1.

Then the single-equation least squares criterion is an optimal choice from the standpoint of statistical theory.

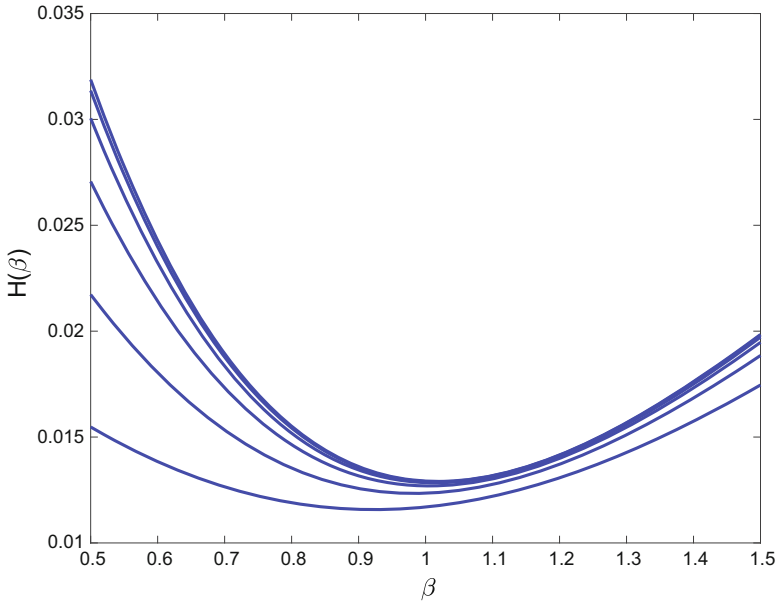


**Fig. 9.3** Each *curve* shows the variation of values of $H(\beta)$ over $\beta$ for the demo data analysis of the data in Fig. 9.1. The values of the $\rho$ used in the same order as the heights of the *curves* are (0.5, 0.73, 0.88, 0.95, 0.98, 0.99)

These assumptions implying least squares are, needless to say, strong. The easiest of these to fix in practice is the distributional assumption (1). We may prefer that $G_0$ is the negative of the log of the likelihood function associated with associated with a probability density function $p(y|x)$ that is more appropriate for the data at hand. For example, if the data are not independent, but are Gaussian, then it may be possible to estimate a covariance matrix $\hat{\Sigma}$ and use a weighted least squares criterion where the weighting matrix is $\hat{\Sigma}^{-1}$.

Some common non-normal distributional situations are:

Positive data: The values $x_j$ are known to be positive, we may opt for a lognormal density and its associated negative log likelihood (minus terms not depending on $x$)

$$H(\boldsymbol{\theta}|\rho) = \sum_j [\log(y_j) - \log(\hat{x}(t_j))]^2 \ . \tag{9.12}$$

If the odd $y_j$ is 0, such as is the case for counts of rare events over a large population, we are apt to substitute a small positive value such as $y = 1/(2N)$ where $N$ is the population size. Alternatively, as we saw in Chap. 2, the differential equation can itself be transformed to a nonlinear form so that $x(t) > 0$.

Non-negative data: If the number of zero $y$ values is not negligible, we may opt for more exotic distributions drawn from what is often called the *zero-inflated* class.

Binary data: If the data are binary with values zero or one, then a Bernoulli distribution is a logical choice with $x$ constrained to satisfy the condition $0 \leq x \leq 1$.

Nevertheless, when the differential equation is linear, as we assume in this chapter, we can gain a great deal computational efficiency if inner criterion involves least squared error measures for both terms. Non-normal distributions in the *generalized linear model (GLM)* family can use iterated least squares solutions as a technique for capitalizing on this efficiency.

On the other hand, if the type of data demand something other than a least squares criterion for outer criterion $H$, it is only logical that the same applies to inner criterion $J$. We have assumed in this section that the coefficient function $\mathbf{c}(\boldsymbol{\theta})$ is explicitly available, but it is not apt to be the case if $J$ is other than a few simple cases, including least squares. If an explicit expression for $\mathbf{c}(\boldsymbol{\theta})$ is not available, the techniques in the next Chap. 10 will have to be employed.

## 9.3 Choosing the Smoothing Parameter $\rho$

The choice of $\rho$ value can depend on many factors. Is our goal to estimate as efficiently as possible the parameters in $\boldsymbol{\theta}$, or is it only to find a useful curve $x$ that does a nice job of fitting the data but has at least some relationship to the differential equation defined by $\boldsymbol{\theta}$? Or is our focus a blend of the two where we are exploring a family of differential equations, perhaps forced by data-defined exogenous terms $u$, hoping that the data will help us to find the magic dynamic model that is the analogue of Maxwell's equations or Planck's black-body radiation function upon which future scientific advances will be based? A simple and effective answer to these questions is to estimate both $\boldsymbol{\theta}$ and $x$ for a wide range of $\rho$ values so as to calculate the *functional flow* $\mathbf{x}(\rho)$ or *parametric flow* $\boldsymbol{\theta}(\rho)$.

An important reason to take this approach is computational. As $\rho$ approaches one, the topology or curvature variation of the response surface defined by $H(\boldsymbol{\theta})$ tends to be become more and more complex, with sharper and sharper curved ridges and more and more possibilities for local optima. Consequently, it can be crucial to have an initial value of $\boldsymbol{\theta}$ that is close to the global optimum. Fortunately, we have found, by contrast, that when $\rho$ is smaller, the opposite holds: the topology of the response surface becomes milder and milder, and more and more like a quadratic surface for which the optimum can be located with only a few iterations. Therefore a working strategy that we use routinely is to begin with a small value of $\rho$, and then increase $\rho$ relatively gently, optimizing $\boldsymbol{\theta}$ for each increment, using that optimum as

an initial value for the next increment. These increments can be conveniently defined by constant increments of the *log-odds* or *logit* transformation $\gamma = \log[\rho/(1-\rho)]$. Base-10 common logarithms are nice to work with in this regard; and, for example, $\rho = 0.9999$ corresponds to the easily remembered value $\gamma = 4$ and $\gamma = 0$ means that $\rho = 0.5$.

If the goal is more focussed on obtaining a nice fitting function $\hat{x}$, there are various data-driven techniques for choosing an optimal $\rho$. In the interests of limiting the discussion to one option among the many choices available, we develop the *GCV* criterion. We define the equivalent degrees of freedom measure

$$df(\boldsymbol{\theta}) = \text{trace}[2\mathbf{M}(\boldsymbol{\theta}) - \mathbf{M}(\boldsymbol{\theta})\mathbf{M}(\boldsymbol{\theta})^T] \,. \tag{9.13}$$

Here *degrees of freedom* refers to the effective dimensionality of the fitting function as a function of $\rho$. For $\rho \to 0$, it turns out that this measure converges to $K$, the number of basis functions. But as $\rho \to 1$, the measure converges to $K - \text{rank}(\mathbf{M}) = m$, which corresponds to the dimensionality of the solution space for the differential equation. For more information, see Ramsay and Silverman (2005).

The GCV criterion is

$$GCV(\rho) = \frac{n}{[n - df(\rho)]^2} \texttt{SSE}(\rho) \tag{9.14}$$

where $\texttt{SSE}(\rho)$ is the residual sum of squares. Notice that GCV is same as the unbiassed estimate of error variance used in linear and nonlinear squares analyses $\texttt{SSE}/(n-p)$ except for the multiplication by the factor $n/(n - df(\rho))$. This factor represents an additional discounting of SSE over and beyond the division by $n - df(\rho)$ to allow for the fact that we are optimizing with respect to both $\boldsymbol{\theta}$ and $\rho$. For theoretical reasons, and backed up by our experience, the value of $\rho$ that minimizes this GCV measure will provide a nearly optimal estimate of $x$ most of the time if the residuals are reasonably close to being uncorrelated.

Many other methods for selecting bandwidth or smoothing parameters have been proposed and may be tried. Useful discussions of their relative merits and some cautions against relying too heavily on automatic methods can be found in Ruppert et al. (2005) and Ramsay and Silverman (2005).

It is important to emphasize, however, that a minimum-GCV estimate of $x$ is not necessarily associated with a best estimate of $\boldsymbol{\theta}$ itself. In fact, it can often be better to use a value of $\rho$ close to one, and especially when the differential equation is obviously doing a fine job of explaining the data.

When we analyzed the demonstration data, we found that the GCV criterion was minimized for the largest value of $\rho$, which was 0.99, and where the estimated value of $\hat{\beta} = 1.019$. For $\rho = 0.5$, on the other hand, $\hat{\beta} = 0.923$. The fit to the data is shown in Fig. 9.4, and we see a nice approximation of the true curve value in the upper panel. But of course the differential equation also defines a model for the first derivative of the fitted function, so that we should see a close agreement between

the left and right sides of the equation. The lower panel shows that this indeed is the case.

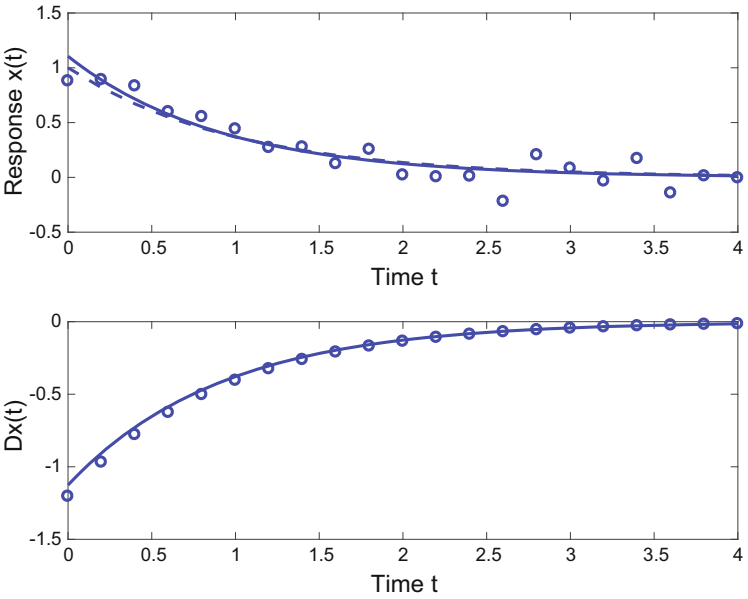


**Fig. 9.4** The *circles* in the *upper panel* are observations at 21 equally spaced time points simulated by adding independent Gaussian errors with mean 0 and standard deviation 0.1. The *solid curve* shows the parameter cascaded least squares fit constructed using 13 order 4 B-spline basis functions and for $\rho = 0.99$. The *dashed curve* is the errorless *curve*. The *lower panel* displays the estimates of the derivative $Dx(t)$ of the estimated $x(t)$'s at the observation points as *circles* and the fit $-\beta x(t)$ of the right side of the differential equation to these derivative estimates

## 9.4 Confidence Intervals for Parameters

The construction of confidence intervals for parameter estimates in $\hat{\boldsymbol{\theta}}$ is an essential sequel to the fitting of a model to data. We want to know how well the data define these estimates, or how much latitude there is for alternative parameter values that could also be seen to be adequate for characterizing the data. From a frequentist statistician's perspective, the question is, "If a new set of data were collected from the same population that these were drawn from, how different could the results be?" A Bayesian approach would frame the question differently, "Given the prior information that we had about the parameters, expressed as a prior density function, and now given the data that we observe, what posterior density function defined on parameter space would characterize our revised knowledge?" Both reasonable questions, of course,

but both fraught with issues. Is there such a thing as a "population" from which the data were randomly drawn? Can a Bayesian really express prior knowledge in terms of a probably density function, potentially over a high dimensional parameter space? And, of course, both schools of thought assume that the right model has been fit to the data.

But confidence intervals do give us vital information, and somewhat biassed confidence intervals are far more useful than none at all, provided that we regard these intervals with a healthy skepticism. One way to put it is, "I think that these intervals may be a somewhat optimistic, but I can be pretty sure that the right limits are at least this bad." Statisticians who propose new methodology are now routinely required to back up other checks on confidence interval quality such as asymptotic analyses by simulated data studies, hopefully emulating a situation that reflects a real-life estimation problem. Although these studies not infrequently provide discouraging news suggesting that further research is needed, the larger picture is a fairly simple one: confidence intervals of the kind we are now proposing are fairly reasonable, if a touch optimistic; and both the frequentist and Bayesian approaches produce rather similar intervals.

In the simplest case of a single equation and least squares data fitting, confidence interval estimation closely follows linear regression analysis in how it works. That is, it proceeds by approximating the relationship between the parameters and the data by a linear relation, and then uses this approximating relation to construct intervals. We now follow the frequentist path, not because we have any disdain for the Bayesian perspective, but merely because this line of argument is well represented in the textbook literature on nonlinear regression, such as Bates and Watts (1988).

In linear regression analysis, we fit data vector $\mathbf{y}$ by the linear map of the regression coefficient vector $\boldsymbol{\beta}$ into data space $\hat{\mathbf{y}} = \mathbf{X}\boldsymbol{\beta}$. Estimating $\boldsymbol{\beta}$ in the least squares sense requires that we construct the inverse to this map $\mathbf{X}$, which is $(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$, and the estimate $\hat{\boldsymbol{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$ is defined directly in terms of this "data—to—parameter" inverse map.

The estimate $\hat{\boldsymbol{\beta}}$ defined in this least squares approach inherits the variation in the data about the so-called "true" fit vector $\mathbf{y}_{pop} = \mathbf{X}\boldsymbol{\theta}$. Let's denote the variance-covariance matrix of the residuals as $\boldsymbol{\Sigma}$. Then, because the variance–covariance matrix of a linear transformation of a random variable is produced by the pre–multiplication of the original variance–covariance matrix by the transpose of the transformation followed by the post–multiplication of the transformation itself, we have in the linear regression situation the result

$$\mathrm{Cov}(\hat{\boldsymbol{\beta}}) = (\mathbf{X}^T\mathbf{X})^{-1}(\mathbf{X}^T\boldsymbol{\Sigma}\mathbf{X})(\mathbf{X}^T\mathbf{X})^{-1}. \tag{9.15}$$

Now we frequently assume that the residuals are independently and identically distributed with mean 0 and variance $\sigma^2$, so that $\boldsymbol{\Sigma} = \sigma^2\mathbf{I}$, and then the right side simplifies to

$$\mathrm{Cov}(\hat{\boldsymbol{\beta}}) = \sigma^2(\mathbf{X}^T\mathbf{X})^{-1}\ .$$

Scalar variance $\sigma^2$ is usually estimated from the data as

$$\hat{\sigma}^2 = \frac{1}{n-p}\sum_i^n (y_j - \hat{y}_j)^2$$

where the number $p$ of regression coefficients is subtracted from the number $n$ of data points to compensate for the fitted values $\hat{y}_j$ being optimizers of the fit of a $p$-dimensional model.

With all this in place, and assuming that confidence intervals are symmetric about the estimated parameter vector, which in turn is assumed to have a roughly normal distribution (or the $t$-distribution if $n = p$ is small), we can offer intervals $\hat{\boldsymbol{\beta}} \pm 1.96\hat{\boldsymbol{\sigma}}_\beta$ where the *parameter standard error* vector

$$\hat{\boldsymbol{\sigma}}_\beta = \hat{\sigma}\sqrt{\text{diag}[(\mathbf{X}^T\mathbf{X})^{-1}]}\,.$$

Well, we confess, we often change 1.96 to 2.

Now let's review all this, but after replacing the regression parameter $\boldsymbol{\beta}$ by our general differential equation parameter $\boldsymbol{\theta}$. Our data–fitting function $\hat{\mathbf{y}}(\boldsymbol{\theta}) = \boldsymbol{\Phi}\mathbf{c}(\boldsymbol{\theta})$ is no longer linear since now $\mathbf{c}$ is not a linear function of $\boldsymbol{\theta}$, but we can use the Taylor Series linear approximation

$$\hat{\mathbf{y}} \approx \boldsymbol{\Phi}\frac{d\mathbf{c}}{d\boldsymbol{\theta}}. \tag{9.16}$$

where $\boldsymbol{\Phi}$ is the $n$ by $K$ matrix of $K$ basis functions evaluated at $n$ observation times, and $d\mathbf{c}/d\boldsymbol{\theta}$ is a $K$ by $p$ matrix. The $n$ by $p$ composite matrix

$$\mathbf{J} = \boldsymbol{\Phi}\frac{d\mathbf{c}}{d\boldsymbol{\theta}}\,,$$

often called the *Jacobian matrix*, is the nonlinear regression analogue of the matrix $\mathbf{X}$ in linear least squares regression. The standard error vector for $\boldsymbol{\theta}$ is therefore

$$\hat{\boldsymbol{\sigma}}_\theta = \hat{\sigma}\sqrt{\text{diag}[(\mathbf{J}^T\mathbf{J})^{-1}]}. \tag{9.17}$$

The estimated standard error (9.17) for the analysis of the demonstration data was 0.1143, providing 95% confidence limits of $\hat{\beta} = 1.019 \pm 0.23$, so that the true value of 1.0 is well within the confidence limits.

### *9.4.1 Simulation Sample Results*

The analysis of our demonstration data for the six values of $\rho$ in Fig. 9.3 required about 1/5 of a second on a typical desktop computer, and for all 3000 analyses for

the 500 samples a minute or so. Some averages across the 500 samples are found in Table 9.1. The standard deviations of the parameter estimates and the averages of the estimates (9.17) improve steadily as $\rho$ increases, and for larger values of $\rho$ are in close agreement. We pay a small amount in terms of bias, but the improvement in RMSE more than offsets this. The average degrees of freedom move toward one as $\rho \to 1$, as they should since a first order system has a one-dimensional solution space. Finally, the GCV values are minimized for the largest $\rho$'s as expected because the data vary randomly about the true model defined by the differential equation.

**Table 9.1** Averages across 500 simulated samples for various values of smoothing parameter $\rho$. The RMSE values are the square roots of the average variances of the parameter estimates $\hat{\beta}$. The values labelled df and GCV are for the average degrees of freedom and the generalized cross-validation index

| $\rho$ | RMSE | | Bias | df | GCV |
|---|---|---|---|---|---|
| | Actual | Estimated | | | |
| 0.50 | 0.1189 | 0.1605 | −0.0017 | 2.860 | 0.0109 |
| 0.73 | 0.1140 | 0.1321 | −0.0053 | 1.853 | 0.0102 |
| 0.88 | 0.1132 | 0.1208 | −0.0066 | 1.349 | 0.0099 |
| 0.95 | 0.1131 | 0.1165 | −0.0070 | 1.134 | 0.0098 |
| 0.98 | 0.1131 | 0.1149 | −0.0072 | 1.050 | 0.0098 |
| 0.99 | 0.1131 | 0.1143 | −0.0072 | 1.019 | 0.0097 |

## 9.5 Multi-variable Systems

The exposition in the last four sections assumed that the model involved only a single equation in order to minimize the mathematical detail. However, it is more frequently the case that a linear system will involve multiple variables $x_i, i = 1, \ldots, d > 1$, so that the general formulation is

$$\mathrm{D}^{M_i} x_i(t) = \sum_{k=1}^{d} \sum_{j=0}^{M_k - 1} b_{ijk} \beta_{ijk}(t|\boldsymbol{\theta}) \mathrm{D}^j x_k(t) + \sum_{\ell}^{\mathrm{L}_i^*} a_{i\ell} \alpha_{i\ell}(t|\boldsymbol{\theta}) u_{i\ell}(t), \quad M_i \geq 0, \quad i = 1, \ldots, d \,. \tag{9.18}$$

That is, each equation can have contributions from any variable in the system, including its own, as well as from any allowable derivative of a variable and, in addition, from any number of forcing functions $u_{i\ell}$.

Recall that the $b_{ijk}$'s and $a_{i\ell}$'s are fixed known constants which are most often, if not 0, either $-1$ or 1. Recall, also, that the term "rate parameter" or "rate function" refers to the possibly time-varying functions $\beta$ and $\alpha$ that multiply variables and their derivatives as well as forcing terms in the linear differential equations described in Chap. 3. A rate function may be expressed in terms of a linear combination of a set of basis functions, in which case the coefficient vector for this linear expansion is a component of the overall parameter vector $\boldsymbol{\theta}$. However, the user may also supply code for evaluating the rate function and its first derivative, and this code will have as an argument the subset of parameter values in $\boldsymbol{\theta}$ that define the rate function.

Only some of these equations may be observed, and we will use the notation $\mathscr{D}$ to indicate the subset of observed variables. Some or all of the observations may be replicated $N$ times.

The number $K_i$ and the nature of the basis functions may vary from one variable to another. Since the typical sizes of variables can vary substantially we need to allow for a differential weighting of variables by weights $\omega_i > 0$. A natural consequence of using differential weighting will be that residual variances $\sigma_i$ must be assumed to also vary over observed variables.

Nevertheless, the general form of the expressions involving $\mathbf{R}(\boldsymbol{\theta})$ and $\mathbf{S}(\boldsymbol{\theta})$ remain valid, except that these matrices will have a block structure. Using $\mathbf{R}_{i_1,i_2}(\boldsymbol{\theta})$, $i_1, i_2 = 1, \ldots, d$ to indicate the matrix in (9.4) for variables $i_1$ and $i_2$, super–matrix $\mathbf{R}(\boldsymbol{\theta})$ will have $\sum_i K_i$ rows and columns, with the matrices $\mathbf{R}_{i_1,i_2}(\boldsymbol{\theta})$ in block $(i_1, i_2)$. Super-matrix $\mathbf{S}(\boldsymbol{\theta})$ will have the analogous block structure with $\sum_i K_i$ rows and number columns determined by total number of forcing functions in the system.

The vector $\mathbf{c}(\boldsymbol{\theta})$ will be composed of a stack of $d$ coefficient vectors, one for for each variable; and therefore also be of length $\sum_i K_i$. The sum of squared errors for the data will be summed across all the observed variables as well as summed over observations within a variable, as will the penalty terms. And of course there may be $d$ smoothing parameters $\rho_i$ rather than only one.

The inner criterion $J$ will now be

$$J(\mathbf{c}|\boldsymbol{\theta}) = (\mathbf{y} - \boldsymbol{\Phi}\mathbf{c})^T \mathbf{W}_1 (\mathbf{y} - \boldsymbol{\Phi}\mathbf{c}) + \mathbf{c}^T \mathbf{W}_2 \mathbf{R}(\boldsymbol{\theta})\mathbf{c} + \rho \mathbf{c}^T \mathbf{W}_2 \mathbf{S}(\boldsymbol{\theta}) \ , \tag{9.19}$$

where this time $\mathbf{y}$ will be the stacked data vectors for only the observed variables and $\boldsymbol{\Phi}$ will be a block-diagonal super-matrix containing evaluations of basis functions in the diagonal blocks. The diagonal weighting matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ will containing the variable weighting coefficients $(1 - \rho_i)\omega_i / n_i$ and $\rho_i \omega_i / T$ along their diagonals, respectively. Then we have that

$$\mathbf{c}(\boldsymbol{\theta}) = [\mathbf{W}_1 \boldsymbol{\Phi}^T \boldsymbol{\Phi} + \mathbf{W}_2 \mathbf{R}(\boldsymbol{\theta})]^{-1} [\mathbf{W}_1 \boldsymbol{\Phi}^T \mathbf{y} + \mathbf{W}_2 \mathbf{S}(\boldsymbol{\theta})] \ . \tag{9.20}$$

Finally, the parameter error variance vector becomes

$$\hat{\boldsymbol{\sigma}}_\theta^2 = \text{diag}[(\mathbf{J}^T \boldsymbol{\Omega} \mathbf{J})^{-1} (\mathbf{J}^T \boldsymbol{\Omega} \mathbf{W}_3 \mathbf{J}) (\mathbf{J}^T \boldsymbol{\Omega} \mathbf{J})^{-1}] \tag{9.21}$$

where $\mathbf{J}$ contains the stacked Jacobian matrices, diagonal matrix $\boldsymbol{\Omega}$ contains the variable weights $\omega_i$, and diagonal matrix $\mathbf{W}_3$ contains $\hat{\sigma}_i^2$ in the diagonal of the $i$th block, $i \in \mathscr{D}$.

## 9.6 Analysis of the Head Impact Data

The data in Fig. 9.5 were collected in study to measure the effects of motorcycle accident on the driver's brain tissue. They are measurements of acceleration of brain tissue within the cranium of a corpse which was struck by a blunt object with a force chosen to be representative of that received by a motorcycle driver's head colliding with a hard surface. Analyses of the data have been reported by many authors, including Härdle (1990) and Silverman (1985).

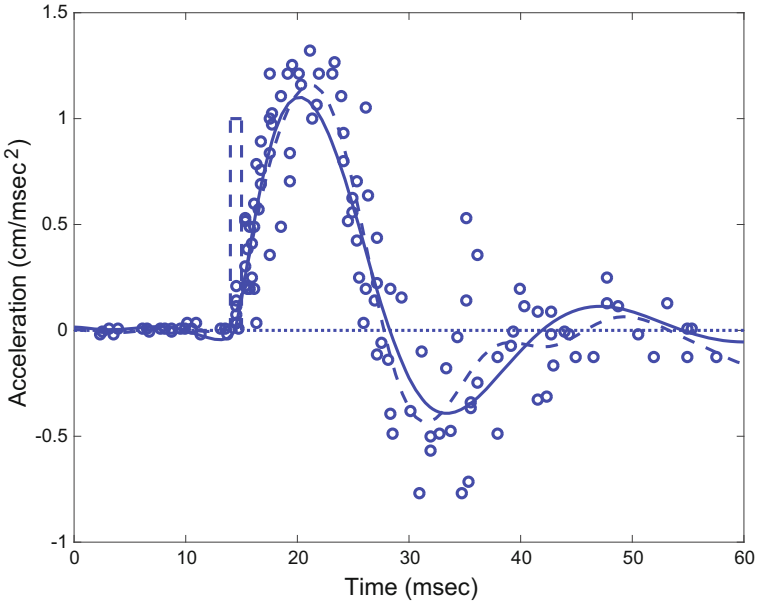


**Fig. 9.5** The *circles* indicate observations of acceleration of brain tissue from five replications of an experiment involving striking the cranium of a corpse with a blunt object. The *box* was constructed to represent the impact itself spread over one time unit. The *dashed* and *solid lines* indicate the fits from the parameter cascading analysis of the data in Fig. 9.5 using the Eq. (9.22) and corresponding to $\rho = 0.5$ and $\rho = 0.998$, respectively

The shape of the data closely resembles what we noted in Chap. 3 as the response to a second order constant rate system to a point impulse. We therefore analyzed the data using the three-parameter forced damped harmonic equation

$$D^2x(t) = -\beta_0 x(t) - \beta_1 Dx(t) + \alpha u(t) \tag{9.22}$$

where the standardized forcing function $u$ was a box function located at the impact time and with unit height and width, which is displayed in the Figure.

Setting up the knot structure for this problem required some care. The box function is discontinuous, and the additive nature of the forcing implies a discontinuity in $D^2x$. This can be achieved in a B-spline expansion by using multiple knots. In the usual case when there is only one knot at each location, and order $m$ B-spline has a continuous derivative of order $m - 2$, so that a cubic B-spline (order four) has a continuous second derivative, which is composed of piece-wise linear segments that join at the knot locations. But each additional knot at a location decreases the order of the continuous spline by one, so that two knots at a cubic spline location would result in the second derivative being discontinuous at that location but with a continuous first derivative. This would be useful, for example, in approximating a first derivative system forced by a discontinuous function.

We used order six spline basis functions defined over [0,60], which have discontinuous fourth derivatives if all knots are singletons. To achieve curvature discontinuity at the impact point $t = 14$ and at $t = 15$, we placed three knots at these locations, so that the second derivative (acceleration) at these points was continuous. In addition, we put no knots between the first observation and the impact point, where the data

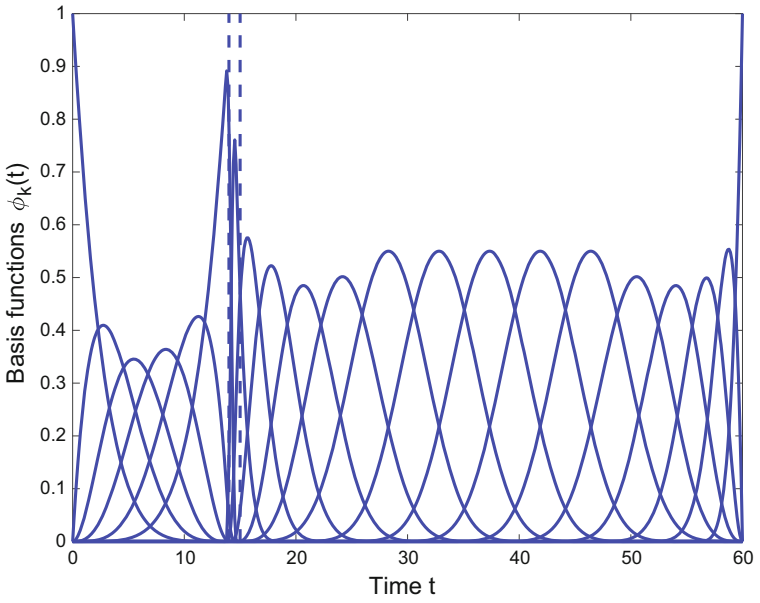


**Fig. 9.6** The *solid lines* are the 21 order 6 B-spline basis functions active in a region around the box forcing function for the head impact data. The *vertical dashed lines* indicate the two locations of three coincident knots

indicate a flat trajectory, and finally nine equally spaced knots between $t = 15$ and $t = 60$. Figure 9.6 displays the B-spline basis functions, and shows that both the basis functions and their derivatives are continuous at the edges of the box, even though their second derivatives are discontinuous across locations.

Figure 9.7 shows how the parameter estimates vary over $\rho$ values 0.50, 0.73, 0.88, 0.95, 0.98, 0.99 and 0.998. The values of parameters $\beta_0$, $\beta_1$, and $\alpha$, along with two standard errors at $\rho = 0.998$ are $0.056 \pm 0.011$, $0.128 \pm 065$ and $0.383 \pm 0.101$, respectively. The GCV criterion was minimized at 0.556 for $\rho = 0.98$, corresponding to 10.3 degrees of freedom, and for $\rho = 0.998$, GCV = 0.574 corresponding to 6.4 degrees of freedom.

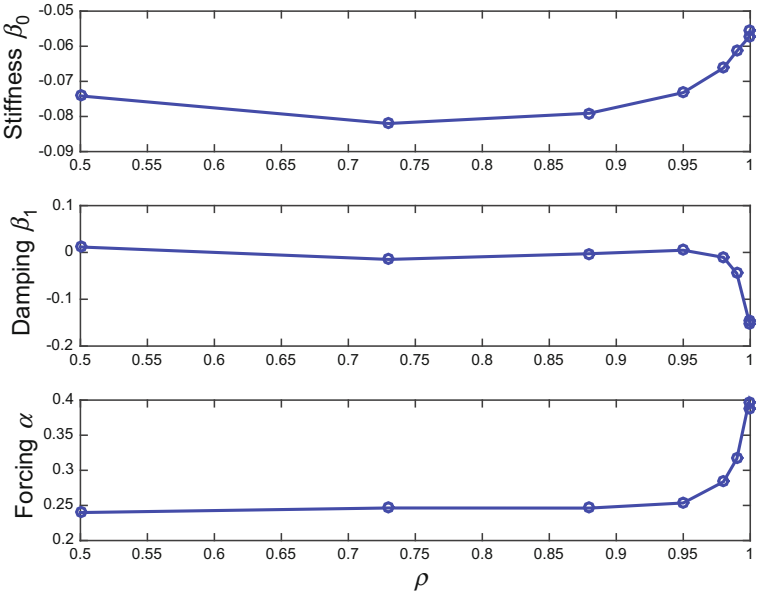


**Fig. 9.7** The three parameter values as a function $\rho$, the largest value of which is 0.998

Figure 9.5 displays the fits corresponding to $\rho = 0.5$ and $\rho = 0.998$. We see a slightly better fit to the data by $x$ for $\rho = 0.5$, but it is striking how well the $\rho = 0.998$ solution fits these data. Figure 9.8 shows how well the second derivative of the fitting function at the observation points is approximated by the right side of the differential equation, and the recovery is remarkable save for some small deviations near the force pulse locations. The model seems quite satisfactory at both the function and acceleration levels.

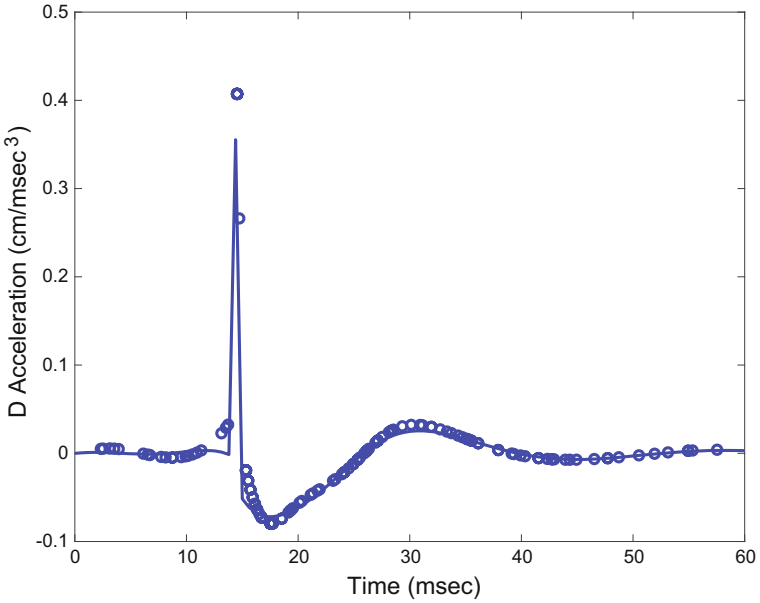


**Fig. 9.8** The *circles* indicate the second derivative of the fitting function for $\rho = 0.998$ evaluated at the observation points. The *solid curve* is the right side of the equation, $-\beta_0 x(t) - \beta_1 \mathrm{D}x(t) + \alpha u(t)$

## 9.7 A Feedback Model for Driving Speed

We now consider a two-variable stationary first order differential equation system that models a simple feedback process such as what might roughly hold when a driver regulates speed so as to respect posted speed limits. We first express the model in terms of two variables, speed and a control level. Then we insert the solution for the control level into the speed equation so as to produce a single second order equation. In both cases, we will see how well the process can be estimated from noisy data.

### *9.7.1 Two-Variable First Order Cruise Control Model*

The concept of *process control* in chemical engineering refers to constructing feedback loops in order to keep aspects of the output of a chemical reactor close to target values, called *set-points*. A great deal of the modelling in this field involves using first order linear stationary equations, $\mathrm{D}x = -\beta x$, to represent components of the process, such has how the output responds to a step change in input. Feedback uses a measurement of the output to modify some aspect of the input, and this feedback process is itself often modelled using a first order equation.

The law requires us to keep the speed of our car at or below a sign-posted limits, plus or minus a small tolerance that we usually abuse to our advantage. We are the control process, the input is the flow of the gas–air mixture to the cylinders; and, when we press the accelerator, the output is speed.

Here is a simple pair of first-order forced differential equations that could be used to represent the process, where $S$ is speed, $S_0$ is the target speed, $C$ is the control variable, and $t$ is time in seconds:

$$\begin{aligned} \mathrm{D}S(t) &= -\beta_S S(t) + \alpha_S C(t) \\ \mathrm{D}C(t) &= \alpha_C [S_0(t) - S(t)] \, . \end{aligned} \tag{9.23}$$

There is only one external forcing function, the set–point $S_0$, it affects only the control variable $C$ and its rate is $\alpha_C$. In the control equation $S$ has the special constraint that its rate is $-\alpha_C$, so that we have to effectively use this rate twice in the equations.

Parameter $\beta_S$ is $4/\tau_S$ where $\tau_S$ is the time it takes a certain car to change the speed to the new level. This is somewhat unrealistic for cars, since the time to reach 100 km/h is much longer than the time to reach 10 km/h; but for speeds in the normal driving range, we tend to compensate for this by pressing the pedal to the floor when entering a highway, and we apply only small pressures to reach low speeds. For example, operating a small car on a snowy day in winter, one of the authors took about 8 s to reach 60 km/h.

When $S$ exceeds the set–point $S_0$, the control level $C$ drops, and positive $\alpha_C$ in the control equation implies a corresponding drop in speed modulated by $\alpha_S > 0$. We assume a black box that measures speed at frequent intervals with negligible noise. The control level $C$ may also be measured, although typically the response time $4/\beta_C$ and forcing rate $\alpha_C$ have already been determined in the controller–design phase.

Figure 9.9 displays as a dashed line the solutions of the differential equations (9.23) corresponding to $\beta_S = 1$, $\alpha_S = 1/4$ and $\alpha_C = 1$; these values produce speed changes roughly consistent with what the author–driver observed. The set–point $S_0$ is shown in Fig. 9.9 as the dotted line in the upper panel, and corresponds to the situation where a driver is driving in a 60 km/h speed zone for 16 s, a 40 km/h school zone for 16 s, an 80 km/h controlled-access street for 16 s, and finally returns to 60 km/h. The solution is computed for the initial conditions $S(0) = C(0) = 0$. With these parameters, we see that the speed reaches the target speed, after each of three set–point changes, in about 8 s. The control level change is proportional to the speed–discrepancy since $\beta_C = 0$. Figure 9.9 also shows a set of 41 simulated observations produced by adding a mean–zero Gaussian random deviate with standard deviation 5.0 to the speed solution in Fig. 9.9 and with standard deviation 20 to the control level.

The speed variable in the control equation has second order curvature because, although it is has order 1 it is forced by a first order system. We used 51 order 6 splines defined by locating a knot at every other observation time, with the exception of the times of the step changes, where added two additional knots to allow for a discontinuity of the third derivative at these points. We used order 5 splines with

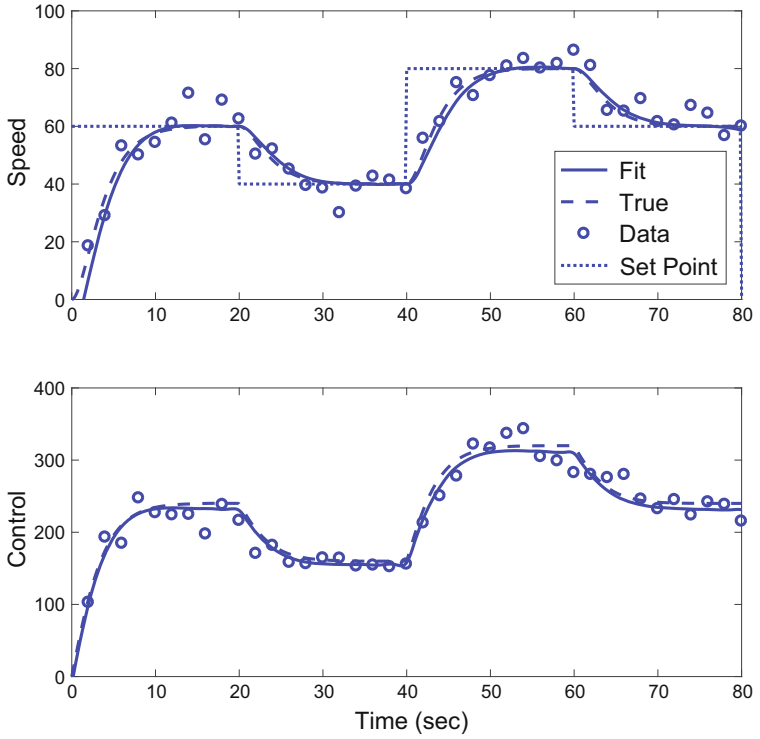


**Fig. 9.9** The *upper panel* shows as a *dashed line* the true speed solution to (9.23) for initial conditions $S(t) = \mathrm{D}S(t) = 0$, the data used to estimate the solution as *circles*, the estimated solution as a *solid line*, and the set–point function $S_0$ as a *dotted line*. The control level true solution, data and the estimated solution for the control variable $C$ are shown in the *lower panel*

the same knot sequence for the control variable, implying 50 basis functions. We stepped $\rho$ from 0.500 to 0.998 using the transform of the integer sequence 0:1:6.

The GCV index bottomed out at the highest level 0.998 of $\rho$, associated with 4.4 degrees of freedom. Figure 9.9 displays the fit to the data for this $\rho$, which produced parameter estimates of $0.670 \pm 0.029$, $0.172 \pm 0.054$ and $0.858 \pm 0.0140$ for the three estimated rates having true values 1, 1/4 and 1, respectively. The confidence intervals given here are at the 95% level, and contain the true values for the two $\alpha$ coefficients, but for the estimate of $\beta_S$ is biassed significantly downward.

We analyzed the same data using only the data for speed, and the result was a disaster. But, when we fixed $\alpha_C$ at its true value, as might be appropriate if the controller were previously calibrated, we obtain results comparable to those with observed control values.

### 9.7.2 One-Variable Second Order Cruise Control Model

We can, without losing much modelling power, simplify the equations by assuming that the gain $K = \alpha/\beta$ is fixed at 1.0 and the controller reaction speed is so large that the controller responds virtually immediately to a change in speed/set–point discrepancy. That is, $\mathrm{D}C = S_0 - S$, or

$$C(t) = \int_0^t [S_0(u) - S(u)]\, du + C_0 \ ,$$

where $C_0$ is an arbitrary constant of integration. Inserting the right side of this equation into the $\mathrm{D}S$ equation in (9.23), and differentiating both sides of the result, we have the single second order equation

$$\mathrm{D}^2 S(t) = -\alpha_S S(t) - \beta_S \mathrm{D}S(t) + \alpha_S S_0(t) \ . \tag{9.24}$$

That the conversion of the two-variable equation implies a single second order equation implies that a first order process with a single first order feedback loop exhibits second order behaviour. We can just see this reflected in the smooth take–off of speed from 0 on the left of the upper panel of Fig. 9.9, even though the control level changes its slope instantaneously, corresponding to $\beta_C = \infty$.

The simplified equation (9.24) resembles the forced harmonic equation, but with some interesting differences. Parameter $\alpha_1 > 0$ plays the role of the stiffness parameter $\beta_0$ in the harmonic equation defining the natural sinusoidal response frequency $P = \sqrt{2\pi/\alpha_1}$, and $\beta_1$ is the damping rate in both equations. The constraint $-\beta_0 - \alpha_1 = 0$ reduces the number of parameters by one. Figure 9.10 shows what happens when $\beta_1$ is set to the values 1/2, 1 and 2: the under–damped value of 1/2 results in an overshoot of the target, 1 is the critically damped solution displayed in Fig. 9.9, and 2 produces an over–damped solution.

This single variable formulation of the cruise control model implicitly assumes that we will only measure speed itself. We now analyze simulated data with the same noise level that we used for Fig. 9.9, but we now assume that the data have been collected for 10 independent occasions, with some variation as well in the set–points, reflecting how different drivers interpret what will be tolerated rather than recommended. Figure 9.11 displays the data and the estimated solutions to the equations. Now we see an excellent recovery of the true solutions shown in Fig. 9.9 along with nice fits to the data. The two parameter estimates are 0.19 and 0.83, respectively, which is rather closer to the true values of 1/4 and 1, respectively. The minimum-gcv value of $\rho$ was 0.99 and the equivalent degrees of freedom was 5.7, which is consistent with what should happen when the right model is fit to the data.

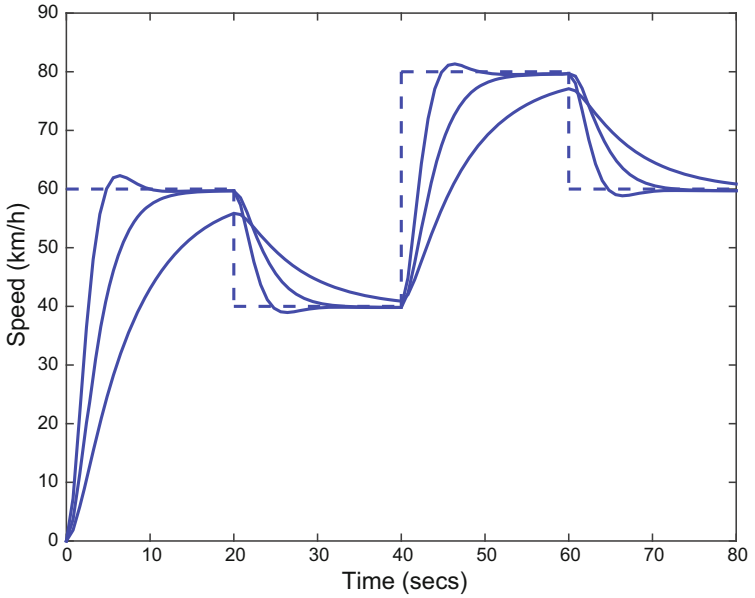


**Fig. 9.10** The speed solution to (9.24) for initial conditions $S(t) = \mathrm{D}S(t) = 0$ for various values of damping rate $\beta_1$. Moving from *top* to *bottom* for *curve* values at time 2, the values are 1/2, 1 and 2

## 9.8 The Dynamics of the Canadian Temperature Data

Solar radiation has both direct and indirect effects on temperature. The absorption of heat by the atmosphere at ground level is a relatively immediate impact, and the seasonal variation of this source can be crudely modelled by a constant $\alpha$ times $u(t) = cos[2\pi(t + 10)/365]$, where the translation by 10 days places the minimum at the winter solstice.

But the sun also heats both land and sea surfaces, which continue to release heat for a few weeks after the winter solstice, and the circulation of atmosphere tends to bring heat from elsewhere, such as the Gulf of Mexico via the Mississippi valley for locations in the St. Lawrence valley. That is, seasonal climate variation is a buffered process which, to a first order, can be modelled by the differential equation $\mathrm{D}T = -\beta T$. We expect that this buffer is itself non-stationary, and therefore we want to allow $\beta$ to vary slowly over the year. We add to the homogeneous differential equations two forcing functions: $U_1(t) = 1$ to capture the annual mean temperature variation, and $U_2(t) = \cos[2\pi(t + 192)/365]$ because we want to highlight the model's performance in winter by starting the year at July 1. The equation is

$$\mathrm{D}T(t) = -\beta(t)T(t) + \alpha_1(t)U_1(t) + \alpha_2 U_2(t) \; . \tag{9.25}$$

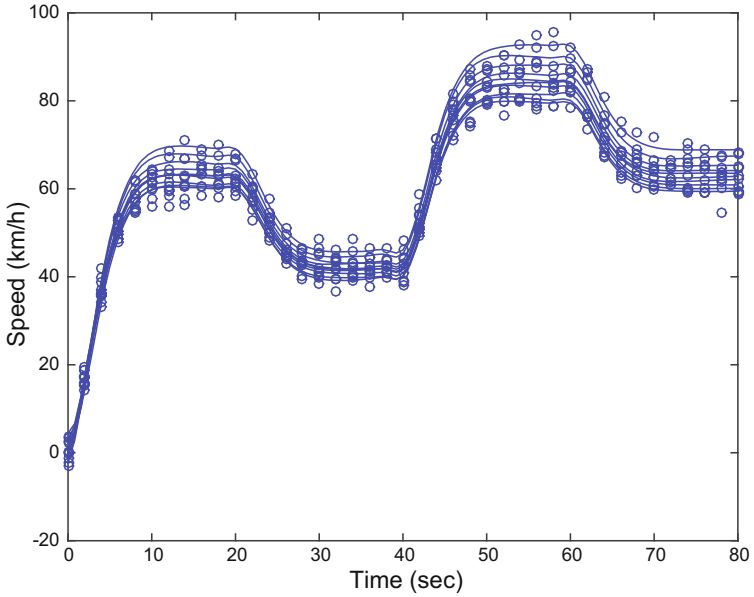


**Fig. 9.11** The *upper panel* contains the estimated speed solution to (9.23) provided by the parameter cascading algorithm using the single second order equation (9.24) with ten replications of the noisy observations

We used as data the 73 5-day averages that span a year's observations. We represented $T$ with an order five B-spline basis function expansion with 77 basis functions resulting from placing a knot at the center of each block. For each station in turn we stepped $\rho$ through $\log[\rho/(1-\rho)]$ values from 0 ($\rho = 0.5$) to 9 ($\rho = 0.9998$) and selecting for display the $\rho$ value corresponding to the minimum GCV measure.

Our preliminary investigations revealed, with no surprise, that the modulation functions $\beta$, $\alpha_1$ and $\alpha_2$ varied substantially from one weather station to another. The 22 stations in the Arctic, the eastern maritimes, central Canada and the prairies all displayed similar shapes, represented by seven Fourier basis functions for $\beta$ and constant functions for the $\alpha$'s.

The results for the city of Montreal, Quebec, were typical, and are shown in Figs. 9.12 and 9.13. As with most of these stations, one of the highest values $\rho = 0.9991$, minimized GCV and corresponded to a degrees of freedom value for the fitting curve $T(t)$ of about 7. We see in Fig. 9.12 that both the fitting function and its derivative are smooth, that the fit tracks the data well, and that the difference between the right and left sides of the differential equation is small everywhere. The nine degrees of freedom for the parameters used to achieve the fit seems like a remarkably compact model, and all nine parameters are defined well by the data.

As expected, we see in Fig. 9.13 that the value of $\beta(t)$ is positive. The value $\tau(t) = 1/\beta(t)$ is the length of time for a buffered change in temperature to attain about 2/3 of its value, and we see that this corresponds to about 30 days in the summer months, but more like 100 days in late autumn and early winter when the speed of reaction is more sluggish. The constant forcing rate is $\alpha_1 = 0.23$ with 0.01 representing two standard errors of estimate, reflecting the fact that Montreal's annual average is greater than 0 °C. The cosine forcing modulation function is $\alpha_1 = 0.56$ with the same standard error indicating that radiative forcing is important throughout the year.

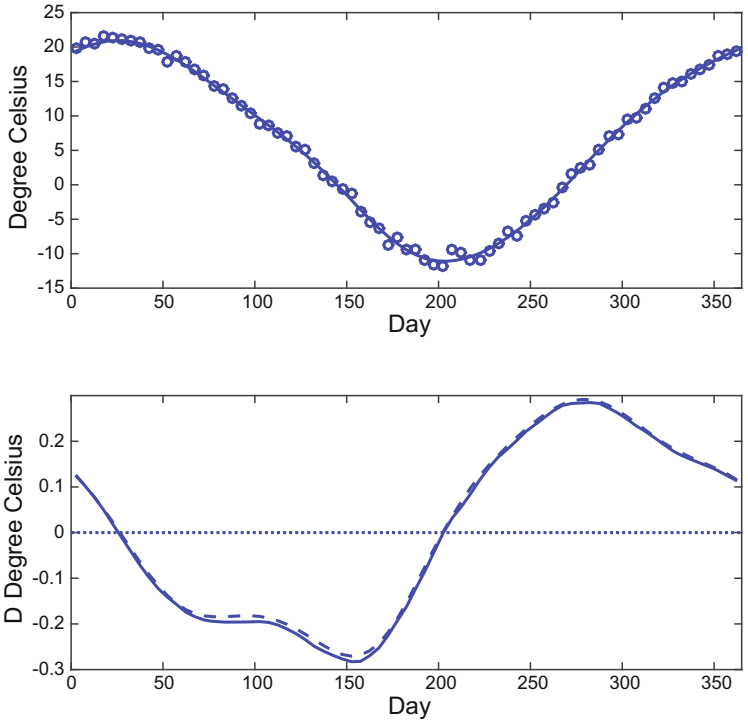


**Fig. 9.12** The *top panel* displays the 5-day averages for Montreal as *circles* and the fitting function $T(t)$ corresponding to $\rho = 0.9991$ as a *solid line*. The *solid line* in the *bottom panel* is the derivative function $\mathrm{D}T(t)$ and the *dashed line* is the value of the right side of the differential equation (9.25). Day 0 is July 1st

The other 13 weather stations are located at or near coastal areas, including Vancouver, Victoria and Prince Rupert on the west coast and Iqaluit on Baffin Island. Each of these has lovely prairie-like summers with plenty of sun shine, but in the fall descend into nearly non-stop drizzle as low-pressure ridges dominate the climate.

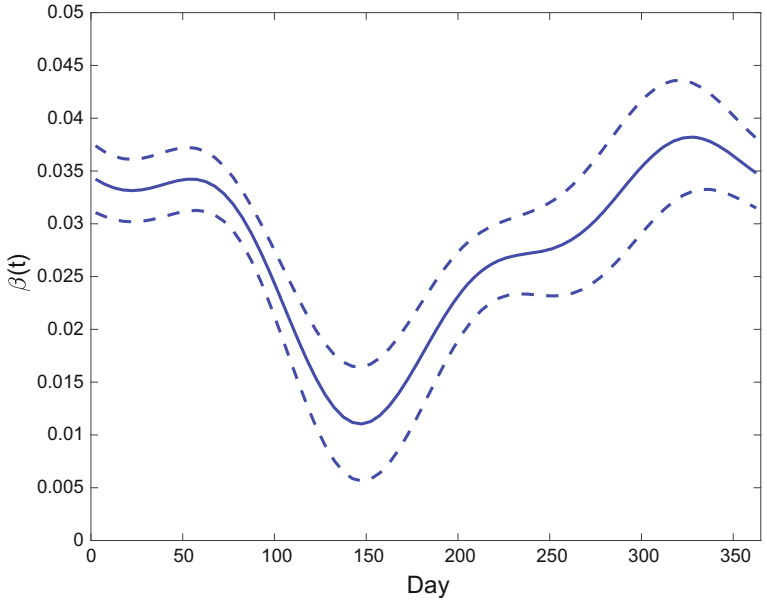


**Fig. 9.13** The *solid line* is the estimated reaction speed $\beta(t)$ in equation (9.25), and the *dashed lines* indicate two pointwise standard errors of estimate above and below the estimate itself. Day 0 is July 1st

We also noted that these stations had winter minimum temperatures rather earlier, casting doubt on the proposition of the simple buffering hypothesis. The GCV criterion selected a substantially lower value of $\rho$, implying more like 40 degrees of freedom for $T(t)$. We conjectured that the modulation function for radiative forcing needed to reflect seasonal effects, so we chose Victoria, British Columbia, to develop a model that did a better job. Figure 9.14 shows the result of expanding $\alpha_2$ using five Fourier basis functions. We see that the shape of $\beta$ is quite unlike that of Montreal, and involves episodes of positive values. The radiative forcing modulation function takes on negative values in the summer months and is positive over the winter. These results are not easy to interpret, but at $\rho = 0.9991$ and the fit was associated with only three degrees of freedom, the differential equation seems to do a fine job of capturing the city's temperature profile. Nearby inland weather stations like Kamloops and Prince George responded similarly to making $\alpha_2$ seasonal, but required only three Fourier basis functions to do so.

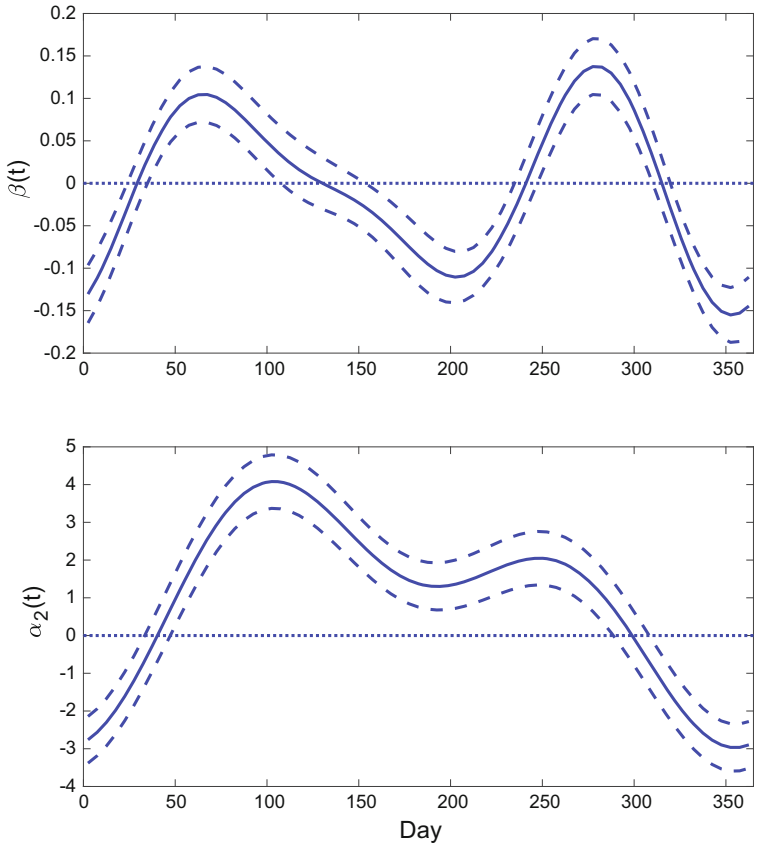


**Fig. 9.14** The *upper panel* contains the estimated value of $\beta$ in Eq. (9.25) (*solid line*) and its 95% pointwise confidence limits (*dashed lines*). The *lower panel* displays the corresponding rate of the cosine radiative forcing function

## 9.9 Chinese Handwriting

We return now to the remarkably successful model in Sect. 1.1.6 in Chap. 1 for the first of 50 replications of Chinese handwriting. Recall that the model was, for each coordinate $X$ and $Y$, one of passing a step function input through a harmonic buffer,

$$\begin{aligned} \mathrm{D}^2 X(t) &= -\beta_X X(t) + \alpha_X(t) \\ \mathrm{D}^2 Y(t) &= -\beta_Y Y(t) + \alpha_Y(t). \end{aligned} \tag{9.26}$$

An event in this handwriting is a stroke, a cusp or a lift off the page. We counted 46 of these, and noted that they seemed to be at equally spaced time intervals. Section 1.1.6

is reproduced here in Fig. 9.15, but with circles at the boundaries of these subintervals rather than numbers at their centers

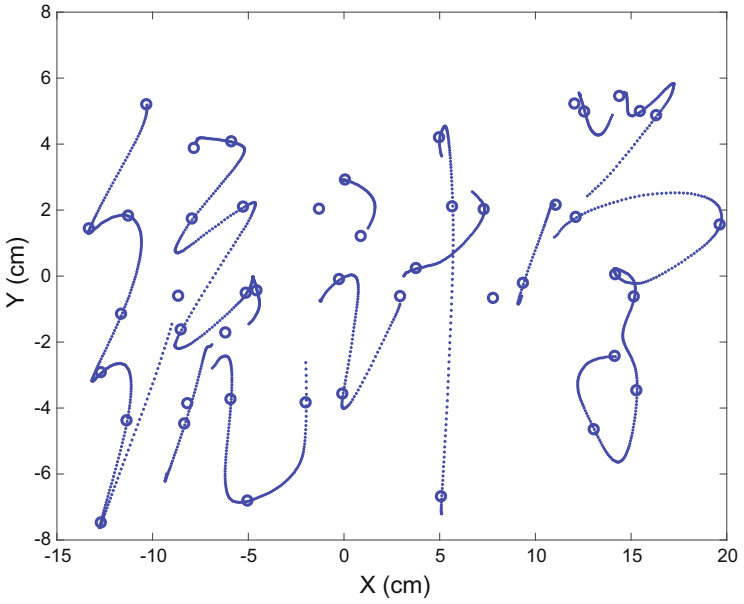


**Fig. 9.15** The *solid line* segments show the position of a pen while on a writing surface during a single writing of "statistics" in Chinese. The writing took 6 s, and the *circles* are displayed at the boundaries between 46 equally spaced time intervals

The homogeneous portions of these equations define two sinusoidal variations; and, viewed as a trajectory on the plane, we would see an ellipse. A physical realization of this object would a rotating disk spinning at a certain angular velocity and with a mass that we have to determine from the data. Moreover, the spinning disk is free to move in a frictionless way over the writing surface. That is, as a hand describing circular movements and suspended over the surface by an arm.

Each step in the second terms of the right sides of the equations represents a force vector applied to the center of the rotating disk, lasting the duration of the step, which we calculated be 130.4 ms. According to the model, the hand continues its eliptical motion, but the short-lived force fields launch it in one new direction after another. Because this disk (hand) has a substantial mass, the response to each new pulse of force is smooth, like we saw in the head impact model, but without any damping.

The order 6 B-spline basis system for representing the two coordinate trajectories assured a smooth second derivative. We used 197 basis functions with equally spaced knots, which assigned 32 basis functions per second, or four for each 7.7 Herz cycle. Preliminary smoothing of the data confirmed that this basis could track the data with

sufficient potential accuracy. We used the sequence of $\rho$ values $e^m/(1+e^m)$, $m = 0, \ldots, 7$ so that $\rho$ ranged from 0.5 to 0.999. We were specifically interested in obtaining a solution to the equation, and therefore we displayed the results for the highest value. Figure 9.16 shows the fits of the right sides of the equation to the left sides, and these fits to the accelerations $\mathrm{D}^2X(t)$ and $\mathrm{D}^2Y(t)$ look quite satisfactory.

The two dynamic rate parameter estimates $\beta_X$ and $\beta_Y$ along with two standard errors were $0.0386 \pm 0.0018$ and $0.0284 \pm 0.0027$. Their periods of rotation, $P = 2\pi/\sqrt{\beta}$, were 319.6 and 373.1 ms, respectively. These values reflect that the fact that the rotation of the wrist adds additional agility for the production of lateral movements that is not available for vertical movements. The 95% confidence limits quoted with the values of $\beta$ suggest a small level of estimation error, that is, roughly about 5%.

If we represent $X(t)$ as $a_{XS}\sin(\omega_X t) + a_{XC}\sin(\omega_Y t)$ and then solve the equations for the values of $\mathrm{D}X(0)$ and $\mathrm{D}^2X(0)$ for constants $a_{XS}$ and $a_{XC}$, respectively, and then do the same for $Y(t)$, we obtain a close approximation to the first stroke. This is not exactly correct since we haven't allowed for the force field applying over the first stroke, but it is a useful check on results.

Figure 9.17 shows the two estimated forcing step functions $\alpha_X(t)$ and $\alpha_Y(t)$ along with their 95% confidence limits. These indicate that each of these steps is estimated with considerable accuracy, as we would expect from the fact that each interval contains at least 51 accurate observations of pen position. Figure 9.18 plots the $Y$ forcing function against its $X$ counterpart, and see that this discrete trajectory roughly follows that of the handwriting itself. This information is all that is required, according to our model, for reproducing this script over and over again. It represents a compression ratio of about 50–1 relative to recording rate of the cameras that provided the data, and a 4–1 compression relative to the 197 basis functions that we used to represent the curves themselves.

This analysis can be seen as a new form of data smoothing. Instead of applying a smoothing operation directly to the data, we estimate a discrete low-dimensional representation for the final smooth curve, and pass this low-dimensional representation through a pair of harmonic buffers to arrive at the final data smooth. We like to think of this as *dynamic smoothing*. Its principal advantage, the compression ratio with respect to the dimension of the spline basis, is achieved by knowing a considerable amount about the neurology and physiology of the motor control system for the human hand.

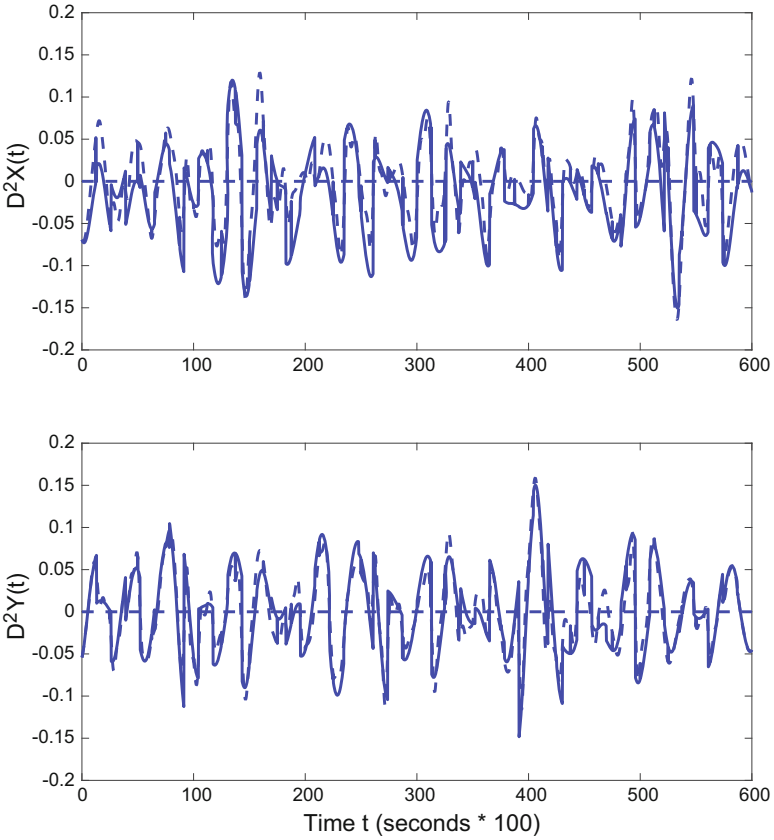


**Fig. 9.16** The *top panel* shows the right side of the equation for the $X$ coordinate as a *solid line*, and the actual acceleration $D^2X$ as a *dashed line*. The *bottom panel* displays these values for coordinate $Y$

## 9.10 Exploring Complexity Through A Transformation of Basis Functions

We emphasized the connection between the parameter cascading algorithm and conventional regression in the context of a linear dynamic model. In particular we noted that order $K$ matrix $\mathbf{R}(\boldsymbol{\theta})$ plays the same role as the covariate cross-product matrix $\mathbf{Z}^T\mathbf{Z}$ in linear regression. We now ask, "Would an eigenanalysis be as revealing and useful here as it has proven to be in regression analysis?" That is, given the decomposition $\mathbf{R} = \mathbf{V}\mathbf{D}\mathbf{V}^T$ what can we learn from the eigenvectors in the columns of $\mathbf{V}$ and the eigenvalues in the diagonal matrix $\mathbf{D}$?

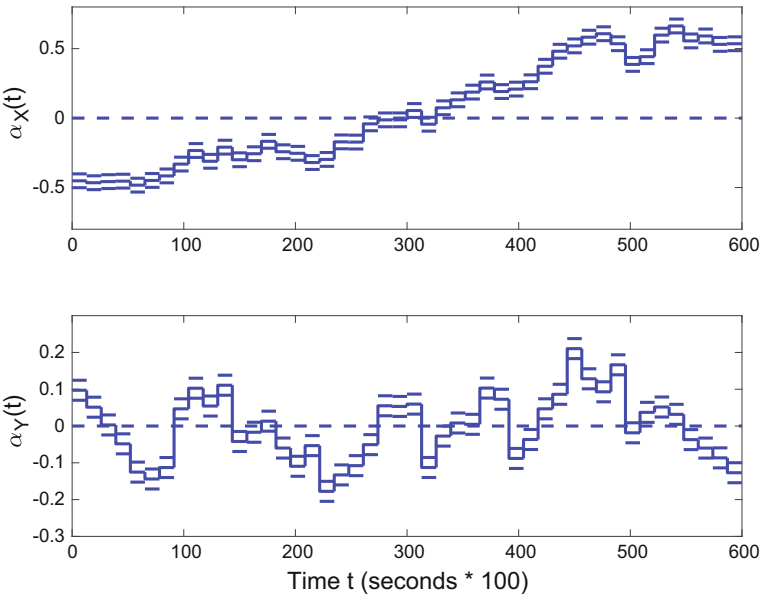


**Fig. 9.17** The *top panel* shows the estimated forcing function values $\alpha_X(t)$, along with two standard errors up and down, which provide about 95% confidence limits. The *bottom panel* displays these values for coordinate *Y*

We call the basis system $\mathbf{r} = \mathbf{V}^T\boldsymbol{\phi}$, where columns of $\mathbf{V}$ are indexed in increasing order of the eigenvalues in the diagonal of $\mathbf{D}$, *R-functions*. These new basis functions are a rotation of the original $\phi$-basis, and define a shape basis system that can have many useful applications as an alternative way of decomposing the shape-variation in the observed curves. These functions are defined by the homogeneous linear differential operator L defining $\mathbf{R}(\rho)$ and, of course, by the basis system $\boldsymbol{\phi}$.

Matrix $\mathbf{R}$ is symmetric and positive semidefinite because it is a cross-product matrix. That is, all of its eigenvalues will be either zero or positive. Also, for a linear differential operator L of order $m$, we know that there is a linear space of functions $x_0$ of dimension $m$ that satisfy $\mathrm{L}x_0 = 0.$, called the *kernel* of L. Consequently, if the basis $\boldsymbol{\phi}$ is sufficiently rich to capture the necessary detail in this kernel space, the $m$ smallest eigenvalues will be either near or at zero since, because any $x_0$ in this subspace can be represented by a set of coefficients $\mathbf{c}$ satisfying $\mathbf{c}^T\boldsymbol{\phi} = 0$, the $m$ eigenvectors associated with the smallest eigenvalues will span the same space.

If we define the *complexity* of a function $x$ as the size of $\int[\mathrm{L}x(t)]^2\,dt$, then the approximately zero eigenvalues corresponding to eigenfunctions satisfying $\mathrm{L}\mathbf{V}_j^T\boldsymbol{\phi} \approx 0$ have near zero complexity. These eigenfunctions are a set of orthogonal functions spanning the kernel of the homogeneous linear differential operator L. Therefore, we get this space of solutions to the equation $\mathrm{L}x = 0$ for virtually free by simply doing an eigenanalysis of $\mathbf{R}$ and using its eigevectors as coefficients in a set of $K$ basis function expansions, $\sum_k v_{k,j}\phi_k = \mathbf{v}_j^T\boldsymbol{\phi}$, $j = 1, \ldots, K$.

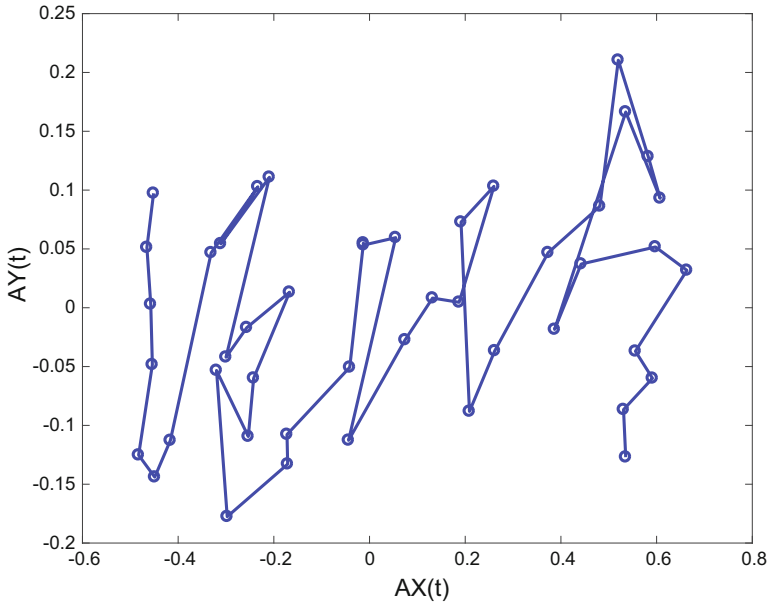


**Fig. 9.18** The $Y$ forcing values plotted against the $X$ values. This is a discrete and rough caricature of the script

Figure 9.19 displays the common logarithms of the eigenvalues of $\mathbf{R}(\boldsymbol{\theta})$ associated with $\rho = 0.9999$ for the head impact data. The two smallest eigenvalues are 4 to 5 orders of magnitude smaller that the remainder, and as such are small enough to be associated with zero complexity. Figure 9.20 displays the two zero-complexity basis functions. We see that they are damped sinusoids.

How well do these R-basis functions manage to approximate the original data? Fig. 9.21 plots the squared multiple correlation achieved by using the first $k$ complexity basis functions to approximate the data in $\mathbf{y}$ as a function of $k = 1, \ldots, 21$. We see that $k = 13$ achieves about all of the fit possible with this basis system, correspond to $R^2 = 0.84$.

Figure 9.22 shows how the complexity increases beyond these two kernel basis functions, each with complexity values of about $10^{-4}$. Their shapes capture additional low-frequency wobbles in the data, and permit us to capture about 40% of the data variation. Adding a sixth basis would bring us to about 73% of a perfect fit. Figure 9.23 displays the three most complex basis functions. We see that the complexity functions 19 an 21 represent sharp variation in the neighbourhood of the onset and offset of the box function, while complexity function 20 captures variation near the end of the observation interval.

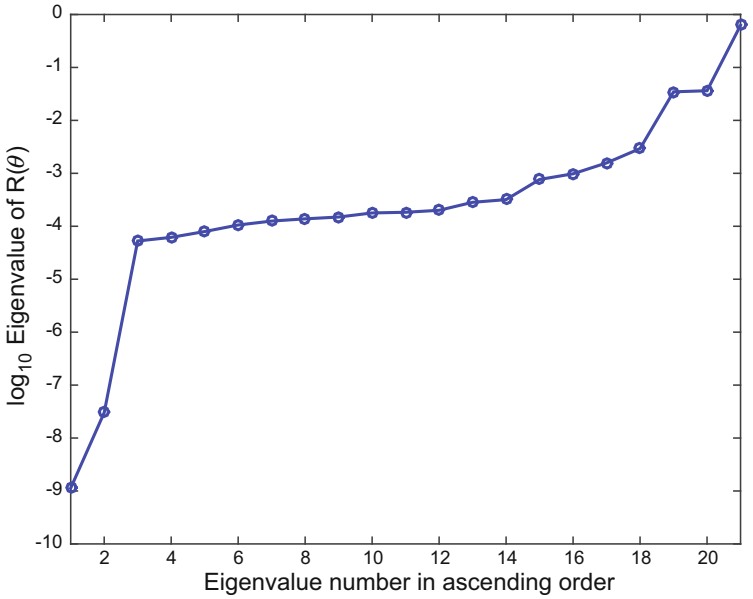


**Fig. 9.19** The common logarithms of the 21 eigenvalues of the matrix $\mathbf{R}(\boldsymbol{\theta})$ for the head-impact data

It is a common practice in the analysis of functional data to use as a basis system a small number of the dominant functional principal components computed from a sample of $N$ sets of observations (Ramsay and Sliverman 2005). A common complaint from substantive area researchers is that these components are difficult to interpret in terms of real-world processes, and at the same time statisticians can be concerned that the number of principal components used (3 is the most common choice) can give an overly optimistic impression of the dimensionality of the data. These R-functions can offer a nice compromise, since they have only a light dependency on the data via the estimation of operator L, while still offering a fairly compact basis representation, and each R-function tends to have an easy description of the shape that it captures. Finally, this basis system has written in its core the shape of the dynamics of the process being analyzed.

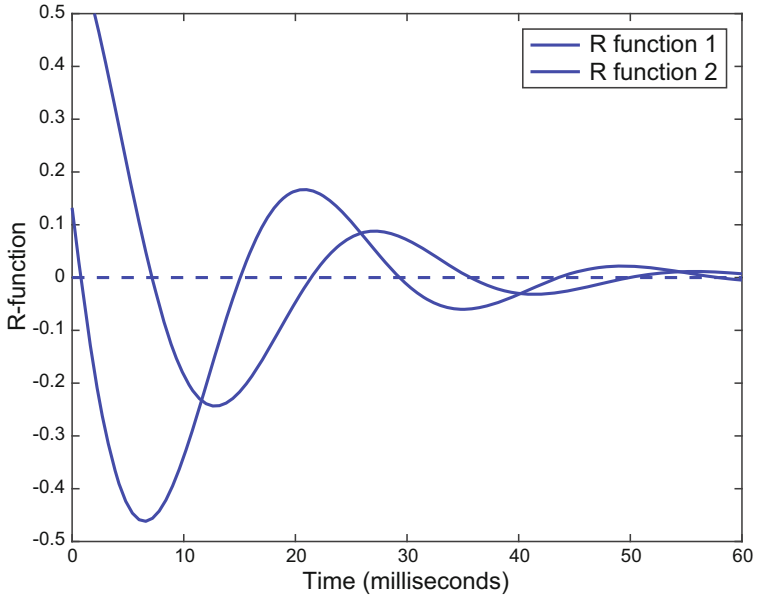


**Fig. 9.20** The first two complexity basis functions, having complexity near zero and therefore spanning the space of solutions to the homogeneous linear differential equation estimated for the head impact basis

## 9.11 Notes on Software and Computation

The structure of the computation for linear systems is quite different than that for the R package code `CollocInfer` and its Matlab version. This primarily due to the elimination of the inner optimization loop in the least squares version, but is also due to the fact that the parameters in $\boldsymbol{\theta}$ play the much more specific role of defining the rate functions $\beta$ and $\alpha$, rather the many possible ways that they could define a nonlinear right side function $f(x)$ in the next chapter. As a consequence, a separate set of code `Data2LD` is used for linear systems.

### *9.11.1 Rate Function Specifications*

A typical `Data2LD` analysis requires the user to supply a set of rate function specifications, each by a list (R) or struct (Matlab) object. There need not be a one-to-one correspondence between these specifications and the terms in the equations, since some rate functions may appear in more than one term. The field names and contents are;

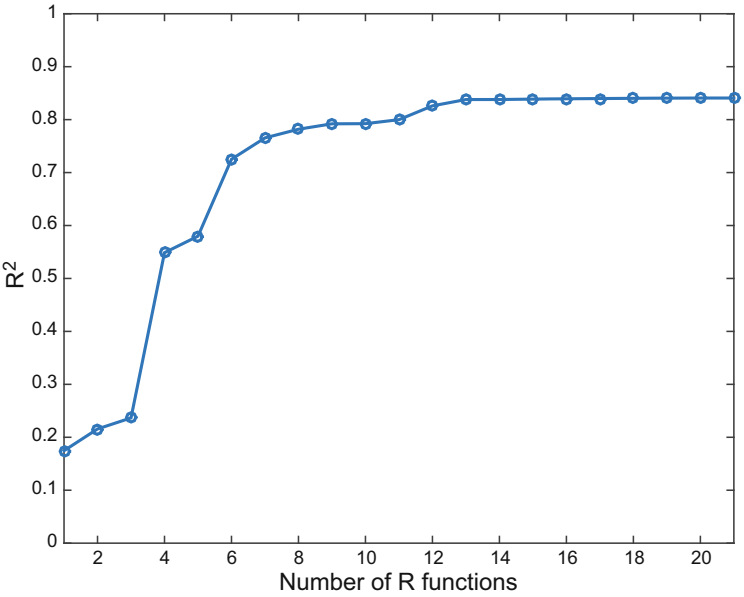


**Fig. 9.21** The squared multiple correlation for approximations of the data based on an increasing number of R-functions

fun: This field can be one of two types:

- a functional parameter object, defined in the functional data analysis code in the R package `fda` or its Matlab equivalent. In this case, this object defines a basis function expansion of the coefficient along with a possible smoothing or roughness penalty.
- a list (R) or struct (Matlab) object, in which case, it contains three fields:

fn: a function object (R) or a function handle (Matlab) giving code for evaluating the rate coefficient

Dfun: a function object (R) or a function handle (Matlab) giving code for evaluating the derivative of the rate coefficient with respect to its parameters

more: a function object (R) or a function handle (Matlab) providing whatever additional information is required for evaluating the rate function or its derivatives

parvec: A vector of values for the parameters defining the rate function. These are used either as initial values for optimization or as definitions of a fixed non-estimated rate function

estimate: A logical value. TRUE (R) or a nonzero real number (Matlab) indicates that the parameters defining the rate function are to be estimated from the data. FALSE or zero indicates that this is a fixed rate function whose parameter values will not be changed during computation.

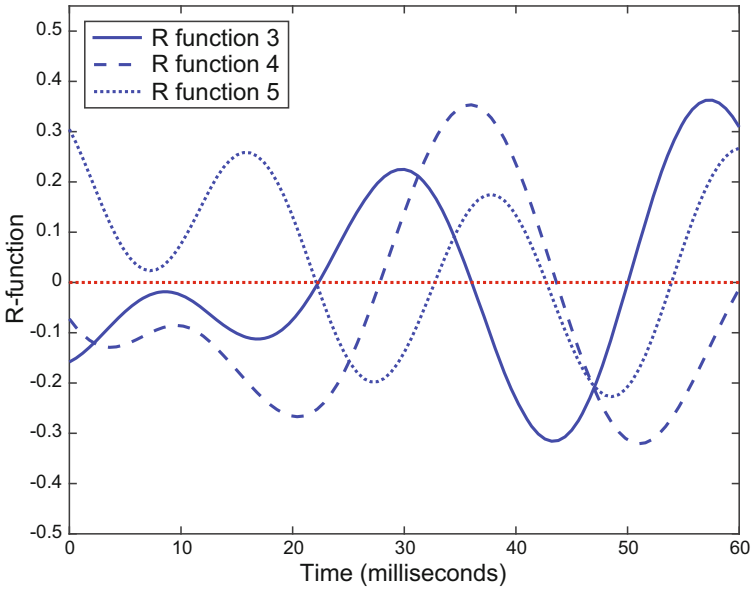


**Fig. 9.22** The next three complexity basis functions after the two kernel functions for the head impact basis

coeftype: A string, either `'beta'` or `'alpha'`, indicating whether the rate function is for a rate function in the homogeneous part of the equation or for a forcing function, respectively.

These rate functions are loaded into a list (R) or cell array (Matlab) with numbered members, one for each rate function.

### *9.11.2 Model Term Specifications*

The terms for each variable in the model must also be specified. These are supplied in a list (R) or cell array (Matlab) with numbered members, one for each variable.

Each member or cell contains a specifications for the terms that appear in the right side of that variable's equation. These specifications are contained in a named list (R) or a struct (Matlab) object. The field names and their contents are as follows, using R terminology:

XList: A named list that in turn has these fields:

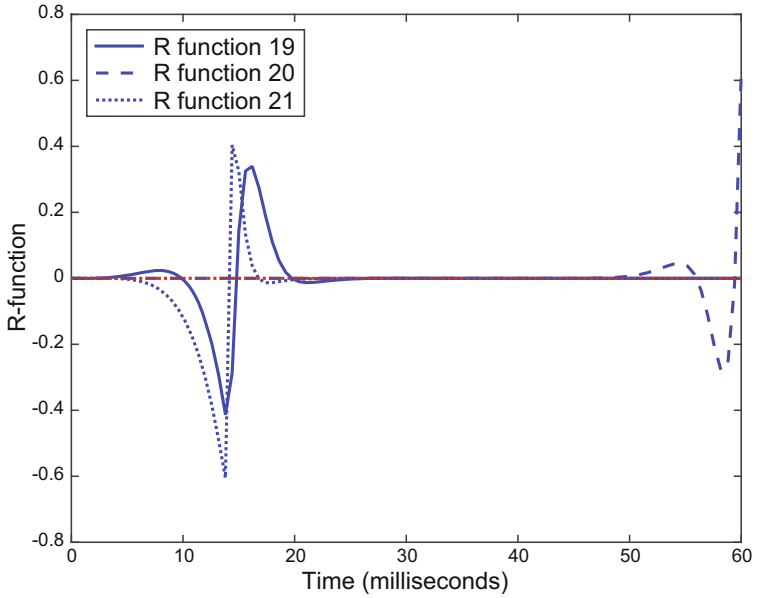


**Fig. 9.23** The three complexity basis functions having the highest complexity values for the head impact basis

variable: An integer $i, i = 1, \ldots, d$ specifying the variable in the system for that term.

derivative: A non-negative integer $0 \leq j < M_i$ indicating the order of derivative used for that variable and term

ncoef: An integer indicating which of the rate functions is to used.

factor: A real number that is used as a multiplier for that term. This is often -1.

FList: A named list that in turn has these fields:

Ufd: A functional data object, defined in the `fda` package, for the forcing function..

ncoef: An integer indicating which of the rate functions is to used.

factor: A real number that is used as a multiplier for that term.

order: The order $M_i$ of the derivative on the left side of the differential equation for variable $i$.

name: An optional string specifying a name for the variable

weight: An optional positive weight to be applied to that variable in computing the inner and outer criteria.

### *9.11.3 Memoization*

Rate functions are often defined as basis function expansions, and if so, this can be used to considerably speed up computation. In this case the matrices $\mathbf{R}(\boldsymbol{\theta})$ and $\mathbf{S}(\boldsymbol{\theta})$ are defined in terms of a set of four-way arrays or tensors involving the products of two rate function basis functions and two variable basis functions. For large-basis problems such as the handwriting example, computing these arrays needs only to happen once. The first time the function `Data2LD` is invoked, this is done, and may require considerable computing time. The arrays are then stored, and on subsequent invocations of `Data2LD` they are read in from memory. This process is referred to as *memoization*, and is often used to improve computation speed.

# Chapter 10
# Profiled Estimation for Nonlinear Systems

## 10.1 Introduction and Chapter Overview

We have seen in the last chapter that linear differential equations can provide nice answers for a wide variety of problems, in the same way that multiple linear regression is the workhorse of applied statistics. The versatility of linear equations becomes especially obvious when we work with time-varying rate functions with values $\beta(t)$ and $\alpha(t)$, which themselves can have any internal structure that we desire, except that they can not depend on the values of any of the variables in the model. An especially useful example is where a rate function is required to be positive, and therefore is represented as the exponential of the basis expansion of its logarithm. Linear equations also can deal with the situation where a rate function is forced by an known external variable, so that the notation $\beta[t, u(t)]$ is appropriate.

We saw, too, that the special structure of a linear equation leads to important computational strategies that make the calculations more reliable, faster, and more accurate, and especially when least squares type criteria are involved. This is especially when we can provide an explicit algebraic expression for the optimizer of the inner criterion $J(\mathbf{c}|\boldsymbol{\theta})$.

All very nice, but what are we losing in exchange? The answer is in the concept of *interactions* between variables. Our prototypical example has been the SIR infection model, or its close cousin the Lotka–Volterra predation model, where a product of two variables is involved which can produce a rapid increase in the sizes of some variables coupled with rapid exhaustion of others. We saw, too, that powers of variables, as in the Fitz–Hugh Nagumo model, can generate periodic spiky variation because a low power term dominates change near zero but a higher power with an opposing sign takes over for larger variable values. The class of equations involving products and powers is probably the most common nonlinear equation category. Interactions like these can be the primary focus of the investigation.

Linear equations cover the situation where one variable affects another by an additive contribution, but often one variable also affects the rate function of another variable directly, and in a way not representable by a product. An example is rate



J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9_10

functions with a Michaelis-Menton structure such as $\beta[T/(K+T)]$ where $T > 0$ is another variable in the system, such as temperature in degrees Kelvin. If $T$ is large, $\beta$ is nearly constant, but as $T$ falls below $K$, $\beta$ goes to zero and the reaction speed becomes slower and slower.

In this chapter we extend our discussion to systems of differential equations in general, that is, of the form

$$\mathrm{D}^{M_i} x_i(t) = f_i(\mathbf{x}, \mathbf{u}_i|\boldsymbol{\theta}), \quad M_i \geq 0, \quad i = 1, \ldots, d \,. \tag{10.1}$$

The right-side function $f_i$ can now be a function of all variables in the system, as well as of one or more forcing functions in vector $\mathbf{u}_i$. How $\mathbf{x}$ and $\mathbf{u}_i$ are involved in the equation is left open, so that, for example, a forcing function can directly affect a rate function, rather than having only additive effects. If there are replications of the observation process, for example, the forcing functions may vary from one replicate to another.

We will also generalize the analysis by allowing the data to have complicated relationships to the variables. In addition to only a subset of them being observed, the data can be observations of functions of the $x_i$, including sums, products, ratios and more complex functional relationships. And the data may be such that a least squares approximation does not make sense, such as when the data are counts or binary. Outer fitting criterion $H$ may, for example, be defined as a negative log likelihood, computed either for a single variable, or for groups of variables, and may involve only some of the parameters in $\boldsymbol{\theta}$.

This chapter therefore considers a potentially more challenging environment, and some of the hazards in working with nonlinear systems have been reviewed in Chaps. 4 and 6. Solutions may exist in only a small interval around the initial point, local and global instabilities can occur, and parameters may not be defined by the data at hand, or else be acutely sensitive to data values.

In the linear case, the user had to supply only a dictionary of rate functions $\beta$ and $\alpha$ along with a blueprint for how these entered into the linear equations, but in the general case the user must supply actual computer code for computing the values of each of the right-side functions $f_i$ and, optionally, the values of the partial derivatives $\partial f_i/\partial \mathbf{x}$, $\partial f_i/\partial \boldsymbol{\theta}$, $\partial^2 f_i/\partial \boldsymbol{\theta}^2$, and $\partial^2 f_i/\partial \mathbf{x} \partial \boldsymbol{\theta}$.

This additional setup effort is necessary because almost surely the inner criterion $J(\mathbf{c}|\boldsymbol{\theta})$ will require numerical optimization methods to calculate the new optimal $\mathbf{c}$ after a change in $\boldsymbol{\theta}$. In the next section we first consider how the concept of parameter cascading must be adapted to the nonlinear situation. At first sight this may seem complicated, but we will try to show that, in fact, the modification is comparatively minor. That is, if the reader has a grasp of how the parameter cascading algorithm works in the linear case, this chapter will be an easy read. As we did in the previous chapter, we will begin by using least squares data fitting in the context of a single nonlinear differential equation before working with multivariable systems.

## 10.2 Parameter Cascading for Nonlinear Systems

### *10.2.1 The Setup for Parameter Cascading*

We saw in Chap. 9 that parameter cascading involves inner and outer fitting criteria, $J(\mathbf{c}|\boldsymbol{\theta})$ and $H(\boldsymbol{\theta})$, respectively.

For $J(\mathbf{c}|\boldsymbol{\theta})$ we will revert to the older and better known non-equilibrated formulation of the penalized least square criterion as involving a data fitting term plus a penalty term multiplied by roughness penalty or bandwidth constant $\lambda$. That is, in the least squares data-fitting case,

$$J(\mathbf{c}|\boldsymbol{\theta}, \lambda) = \sum_{j}^{n} [y_j - x(t_j)]^2 + \lambda \int_0^T \{\mathrm{D}^m x(t) - f[x(t), \mathbf{u}(t)|\boldsymbol{\theta}]\}^2 \, \mathrm{D}t \tag{10.2}$$

where the bandwidth or smoothing parameter $\lambda > 0$ controls the emphasis on fitting the equation but the data-fitting first term remains fixed. We do this because the existing package `CollocInfer` in the R language is set up this way. However, future releases of our software may well switch to the $(1 - \rho)$, $\rho$ weighting structure.

The outer criterion $H(\boldsymbol{\theta}|\lambda)$ defines fit in terms of only the parameter vector $\theta$. The *total* derivative of $H$ with respect to $\boldsymbol{\theta}$ must allow for the fact that $H$ may depend both directly on $\boldsymbol{\theta}$ and indirectly via its dependency on $\mathbf{c}(\boldsymbol{\theta})$:

$$\frac{dH}{d\boldsymbol{\theta}} = \frac{\partial H}{\partial \boldsymbol{\theta}} + \left(\frac{\partial H}{\partial \mathbf{c}}\right)\left(\frac{d\mathbf{c}}{d\boldsymbol{\theta}}\right), \tag{10.3}$$

although for for most fitting criteria, including least squares, $H$ has no direct dependency at all on $\boldsymbol{\theta}$, so that $\partial H/\partial \boldsymbol{\theta} = 0$ and can be dropped from the total derivative equation. It is the derivative $d\mathbf{c}/d\boldsymbol{\theta}$ that is the missing link, since we no longer have an explicit expression for $\mathbf{c}$.

### *10.2.2 Parameter Cascading Computations*

In the least squares fitting of linear systems in Chap. 9, we worked out explicitly the optimizing value of $\mathbf{c}$, but in general, this will not be possible, and numerical methods must be used to achieve the optimum of $J$. But this does not imply that we know nothing about $\mathbf{c}(\boldsymbol{\theta})$ or its gradient $\partial \mathbf{c}/\partial \boldsymbol{\theta}$ with respect to $\boldsymbol{\theta}$.

We use *implicit differentiation* or the *Implicit Function Theorem* to calculate this gradient. Let us suppose that we have optimized $J$ using a high quality numerical optimization strategy to obtain an optimum value $\mathbf{c}$, so that we may now say that $\partial J/\partial \mathbf{c} \approx 0$ to a high level of accuracy. We now compute the total derivative $d/d\boldsymbol{\theta}$ of the gradient of $J$ with respect to $\boldsymbol{\theta}$, which, since $\partial J/\partial \mathbf{c} = 0$, will also be 0:

$$\frac{d}{d\boldsymbol{\theta}}\left(\frac{\partial J}{\partial \mathbf{c}}\right) = \frac{\partial^2 J}{\partial \mathbf{c}\partial \boldsymbol{\theta}} + \left(\frac{\partial^2 J}{\partial \mathbf{c}^2}\right)\left(\frac{d\mathbf{c}}{d\boldsymbol{\theta}}\right) = 0$$

so that

$$\frac{d\mathbf{c}}{d\boldsymbol{\theta}} = -\left(\frac{\partial^2 J}{\partial \mathbf{c}^2}\right)^{-1}\left(\frac{\partial^2 J}{\partial \mathbf{c}\partial \boldsymbol{\theta}}\right)$$

and, substituting this into (10.3) we see that

$$\frac{dH}{d\boldsymbol{\theta}} = \frac{\partial H}{\partial \boldsymbol{\theta}} - \left(\frac{\partial H}{\partial \mathbf{c}}\right)\left(\frac{\partial^2 J}{\partial \mathbf{c}^2}\right)^{-1}\left(\frac{\partial^2 J}{\partial \mathbf{c}\partial \boldsymbol{\theta}}\right). \tag{10.4}$$

However, assuming that the inner optimization does proceed smoothly, then the amount of computation involved in each re-iteration is apt to be small since the outer optimization typically involves small changes in $\boldsymbol{\theta}$, and consequently a small perturbation in $J$ as the function to be optimized, so that only a few iterations will be required within each outer optimization step. It is typical, too, that $J$ is optimized with respect to $\mathbf{c}$ conditional on whatever starting values are available for the parameters in $\boldsymbol{\theta}$ prior to commencing the $H$-optimization process.

Although we lose the explicit expression for $\mathbf{c}(\boldsymbol{\theta})$ in the nonlinear system context, we do not lose the other advantages of least squares estimation. If appropriate, least squares objective functions for $J$ and $H$ can be optimized by the usually stable and efficient Gauss–Newton process. Moreover, since the criterion $H(\boldsymbol{\theta})$ essentially defines a nonlinear least squares problem in $\boldsymbol{\theta}$, we can use the same techniques for computing confidence intervals that we used in the last chapter.

### *10.2.3 Some Helpful Tips*

When $\lambda$ is small, the data-fitting term dominates the derivative-fitting term in (10.2). Since the data-fitting criterion is quadratic in $\mathbf{c}$ for each variable for a least squares criterion and nearly so for other loss functions, the inner loop converges quickly and reliably, and the outer fitting criterion is not especially sensitive to nonlinearities in the differential equation, so that the optimization of $H$ also goes well. But as we increase $\lambda$, the greater and greater domination of the second term means that both optimizations become more and more difficult, and therefore dependent on good starting values. For this reason, we routinely step $\lambda$ up by a uniform sequence in the transformation $\log_{10}\lambda$, and for each $\lambda$ we start the optimization with the values of $\boldsymbol{\theta}$ and $\mathbf{C}$ achieved at the end of the previous optimization.

It can happen that the vital *Hessian* matrix $\partial^2 J/\partial \mathbf{c}^2$ will have negative or zero eigenvalues, and especially for larger values of $\lambda$. This problem is often located in the second penalty term, where, in the single equation case,

$$\frac{\partial^2[\mathrm{D}x - f(x|\boldsymbol{\theta})]^2}{\partial \mathbf{c}^2} = 2\left[\mathrm{D}\boldsymbol{\phi} - \left(\frac{\partial f}{\partial x}\right)\boldsymbol{\phi}\right][\mathrm{D}\boldsymbol{\phi}^T - \left(\frac{\partial f}{\partial x}\right)\boldsymbol{\phi}^T\Big] - 2[\mathrm{D}x - f(x)]\left(\frac{\partial^2 f}{\partial x^2}\right)\boldsymbol{\phi}\boldsymbol{\phi}^T \,. \tag{10.5}$$

The first term is automatically at least non-negative definite, but the second term may not be. However, the total contribution of the second term is usually small and dominated by that of the first cross-product term, and a frequent fix is to just drop it. The impact on the optimization of $J$ is often only a few extra iterations when the Hessian matrix is positive definite. The statistical community refer to this as optimization by *scoring*. The coefficient gradient $d\mathbf{c}(\boldsymbol{\theta})/d\boldsymbol{\theta}$ is now only an approximation, of course, but we have found that this has little impact on the outer optimization of $H$ with respect to the parameter values beyond adding a few extra iterations. An example of this scoring strategy will be illustrated in Sect. 10.7.

The R package `CollocInfer`, described in Hooker et al. (2014), and its Matlab version use the parameter cascading approach described in this paper. The `CollocInfer` program has a simple setup for least squares fitting, but requires the user at a minimum to provide a function that evaluates the right side of the differential equation $\mathbf{f}$. The computation does require a reasonably accurate estimate of the four partial derivatives $\partial\mathbf{f}/\partial\mathbf{x}$, $\partial\mathbf{f}/\partial\boldsymbol{\theta}$, $\partial^2\mathbf{f}/\partial\mathbf{x}^2$, and $\partial^2\mathbf{f}/\partial\mathbf{x}\partial\boldsymbol{\theta}$. If these are not provided, `CollocInfer` estimates them by differencing, a process that works reasonably well most of the time, and certainly might serve in the preliminary stages of investigating a model.

Providing explicit functions, however, assures a smoother and more rapid progress to final solutions, tends to avoid some computational issues on the way such as being trapped in local minima, and yields better derivative estimates to be used in computing confidence intervals.

We strongly suggest, as a consequence of much wastage of time in our early work, that symbolic computation tools such as Maple, Mathematica, and the Symbolic Toolbox in Matlab be used for this purpose no matter how simple the equation system appears to be. To this end, a Matlab function called `make_file_build` is supplied with the Matlab version of `CollocInfer` that only requires a few arguments, including definitions in Matlab symbolic toolbox code of the equations, and produces a `.m` file that can be invoked to compute the values of the right sides of the equations and all the required derivatives.

### *10.2.4 Nonlinear Systems and Other Fitting Criteria*

We use the following expression of the inner fitting criterion involving a general data-fitting function $G_J$, in the special case where all $x_i$ trajectories are expanded using the same set $\boldsymbol{\phi}$ of $K$ basis functions:

$$J(\mathbf{c}|\boldsymbol{\theta}, \lambda) = G_J[\mathbf{Y}, \mathbf{C}(\boldsymbol{\theta}|\boldsymbol{\lambda})\boldsymbol{\phi}] + \sum_i^d \lambda_i \int_0^T \{\mathrm{D}x_i(t) - f[\mathbf{x}(t), \mathbf{u}_i(t)|\boldsymbol{\theta}]\}^2 \, \mathrm{D}t/T \ . \tag{10.6}$$

The coefficient matrix $\mathbf{C}$ will, in this simple case, have $K$ rows and $d$ columns. In this formulation, also for simplicity, we retain the integrated squared residuals in the penalty term, but we certainly envisage other penalty structures that might be more appropriate. The corresponding expression for the outer criterion $H$ is

$$H(\boldsymbol{\theta}|\lambda) = G_H[\mathbf{Y}, \mathbf{C}(\boldsymbol{\theta}|\boldsymbol{\lambda})\boldsymbol{\phi}] \ . \tag{10.7}$$

However, the two data fitting functions $G_J$ and $G_H$ may in many applications be the same.

## 10.3 Fitting the Lotka–Volterra Model to the Chemostat Data

We now fit the simple Lotka–Volterra model to the chemostat data analyzed in Chaps. 7 and 8:

$$\begin{aligned} \mathrm{D}C &= \alpha C - \beta_C BC \\ \mathrm{D}B &= \beta_B BC - \delta B \ . \end{aligned} \tag{10.8}$$

In this application, variable $C$ is the abundance of variety of single-cell algae called Chorella, and $B$ the abundance of a multi-cell rotifer called Brachionis that feeds on algae cells. Parameter $\alpha$ is the birth rate of algae cells, $\beta_C$ is the rate of mortality of these cells per unit of algae abundance, $\beta_B$ is the rate of mortality of these cells per unit of rotifer abundance, and $\delta$ is the death rate of rotifer cells. Although this model resembles the SIR model closely, notice that here we have two different populations interacting rather than one, and this consequently requires two rate constants for the effects of predation.

The nonlinearity in this model is due to the product $BC$ appearing in both equations, and $-\beta BC$ can be viewed as a mortality term for algae with a rate $\beta B$, that is, as rate-forcing by rotifer abundance. Each of these parameters is nonnegative in principle, so that we actually estimate the logarithms of the parameters, and convert these estimates to their exponential values for display purposes. Even though the amount of data available is modest and the model is elementary, we can still explore a number of aspects of the profiling approach to data modelling.

Although a reader may be insulted if it is suggested that coding up the right sides and their derivatives for a model as simple as this should be avoided, here are six lines of code that will do the entire job in Matlab:

```
sysname  = 'Lotka_Volterra';
filename = 'make_Lotka_Volterra';
xeqn = cell(2,1);
xeqn{1} = 'exp(p{1})*x{1}
          - exp(p{2})*x{1}*x{2}';
xeqn{2} = 'exp(p{3})*x{1}*x{2}
          - exp(p{4})*x{2}';
make_file_build(sysname, filename, xeqn, 3);
```

The result is a Matlab file with 62 lines of code, which we consider to be a useful saving in effort.

A first design issue is data and model transformation. It is often desirable, when there is large amplitude variation among variables, as there often is for spread of disease and population dynamics models, to model the logarithm of the data. This is achieved by dividing each equation's right side by its variable value, so that

$$\begin{aligned} \mathrm{D}\ln C &= \alpha - \beta_C B \\ \mathrm{D}\ln B &= \beta_B C - \delta\ . \end{aligned} \tag{10.9}$$

Although this seems to render the equations linear, $C$ and $B$ are now the exponentials of the corresponding log-variables, so that the equations remain nonlinear. The data for variables $\ln(C)$ and $\ln(B)$ are in the panels of Fig. 10.1. We see that both populations oscillate in the manner predicted by the Lotka–Volterra model, which includes the prediction that the predator population will lag the prey population by about five time units. A warning, however, is that the domain over which the equations are defined should not contain zero if either the data or the values of the variables are zero at that time. This caution applies to any other time where a close approach to zero by one of the populations happens.

We first had to set up a basis system for representing the two trajectories. We used for each variable 34 order five B-splines with knots positioned at equal steps of size one between times 15 and 45, so that each knot except the last was close to an observation time. We chose splines of order five because we are using first order differential equations and we want to ensure that the first derivatives are smooth. We also needed quadrature points and weights for approximating the integrals, and we used 150 equally spaced quadrature points, so that a point is positioned at each knot and there are four points within each inter-knot interval. The quadrature weights were those of Simpson's Rule.

We noted above that fitting the model for an increasing sequence of values of $\lambda$ allows us to both examine the trade-off between fitting the data and satisfying the equation, and generates better starting values for the more difficult situation where we place heavy emphasis on fitting the equation. Consequently, we defined $\lambda$ by setting the sequence $\log_{10}\lambda$ to the 11 values from $-2$ to 3 with step size 0.5. Figure 10.1 displays the data along with estimated trajectories for values $\lambda = 0.1$, 3.2 and 1000. The sum of the square roots of the mean squared residuals (RMSE) for the fit to the

log abundance data for these values were 0.29, 1.06, and 1.48, where their increases reflect increasing emphasis on fitting the equation. By contrast, the RMSE's for the differences between the right and left sides of the equation were 1.94, 0.46 and 0.05, so that the trajectories for $\lambda = 1000$ were close to being solutions to the differential equation (10.9).

The model that we are inclined to favour is indicated by the solid lines in Fig. 10.1, where $\log_{10} \lambda = 1/2$. It seems to provide a reasonable balance between fitting the data and fitting the differential equation. But we were impressed by how the best equation fit also gave a rather satisfactory account of the data as well. Lotka–

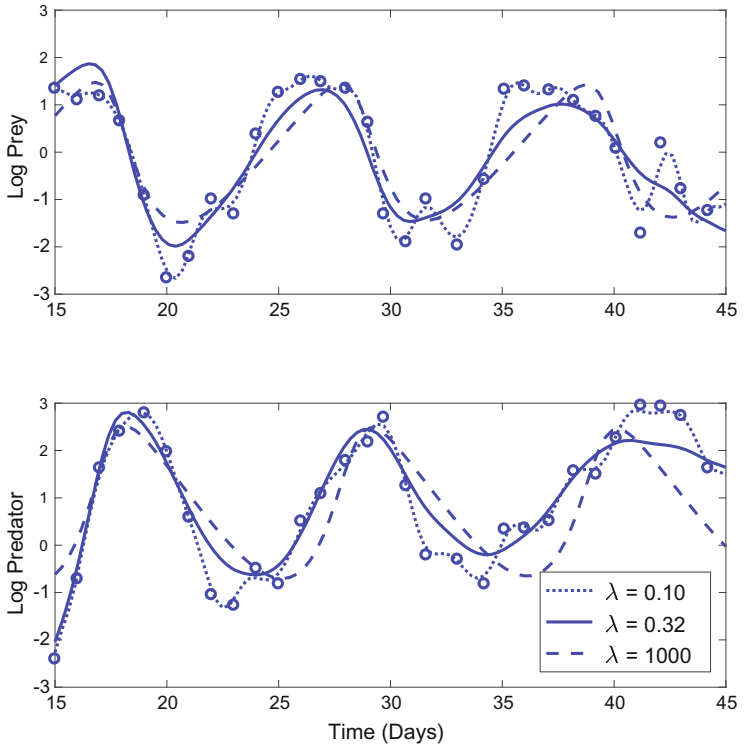


**Fig. 10.1** The *upper panel* and *lower panels* contain the observations on the prey ($C$) and predator ($B$) abundances, respectively, indicated by *circles*. The *lines* in each panel indicate the fits to the data from the model for a *low*, *medium* and *high* values of smoothing parameter $\lambda$

Volterra model fits to actual data are relatively rare in the ecology literature; and we believe that the `CollocInfer` process, by sneaking up on a tight equation fit, has managed be quite successful for these data.

Figure 10.2 shows how the parameter estimates vary across the values of $\log_{10} \lambda$. Beyond $\lambda = 10$, all four parameters are relatively stationary, except that the predator

rate $\beta_B$ for predation declines and this is compensated for by a decrease in predator morality.

Figure 10.3 shows how the logarithm of the root-mean-squared error for the data fit increases for both variables as $\lambda$ goes from $-0.01$ to 1000, which the corresponding values for the left and right sides of the equation decline to the point where the equation is fit precisely and the data are left to pick up what fit they can.

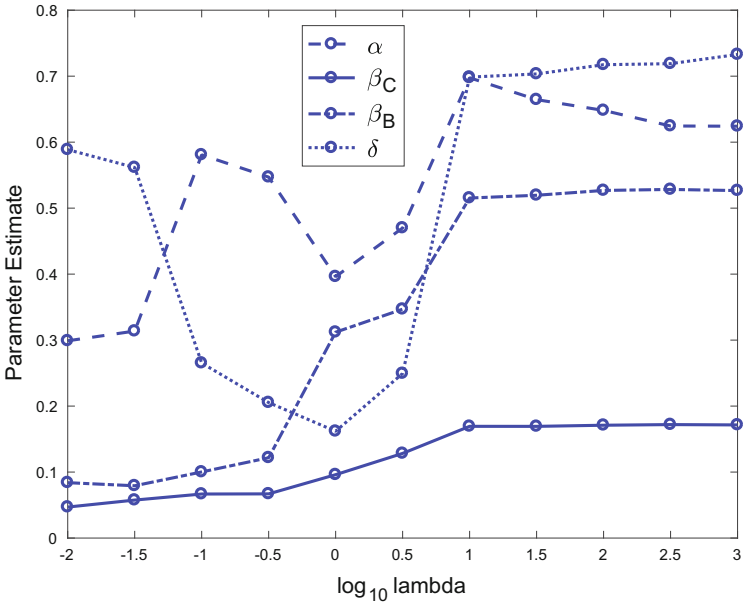


**Fig. 10.2** The parameter estimates for the Lotka–Volterra model for the chemostat data over a range of values of $\log_{10}(\lambda)$

## 10.4 A `CollocInfer` Analysis of the Head Impact Data

We set up the analysis of these data in order to illustrate the use of two first order linear equations to solve a second order linear equation. The equation $\mathrm{D}^2 x(t) = -\beta_0 x(t) - \beta_1 \mathrm{D}x(t) + \alpha u(t)$ can be solved by the equivalent system

$$\begin{aligned} \mathrm{D}x_1(t) &= x_2(t) \\ \mathrm{D}x_2(t) &= -\beta_0 x_1(t) - \beta_1 x_1(t) + \alpha u(t) \end{aligned} \tag{10.10}$$

where the forcing function $u$ is the pulse function displayed in Fig. 9.5. In this setup only variable $x_1$ is observed.

Figure 10.4 contains the data and the fit to them offered by variable $x_1$ for $\lambda = 100$. We see that the fit is almost identical to that in Chap. 9 where $\rho = 0.95$. The estimated parameters values are 0.060, 0.024 and 0.139 for $\beta_0$, $\beta_1$ and $\alpha$, respectively, which are close to those in the previous chapter. The bottom panel of the figures shows the values of $\mathrm{D}x_2 = \mathrm{D}^2 x_1$ at the observation times as points, and the right side of the equation as a smooth curve. Clearly the differential equation is simultaneously well satisfied by the estimated trajectory $x_2$ and the data nicely fit by the estimated trajectory $x_1$.

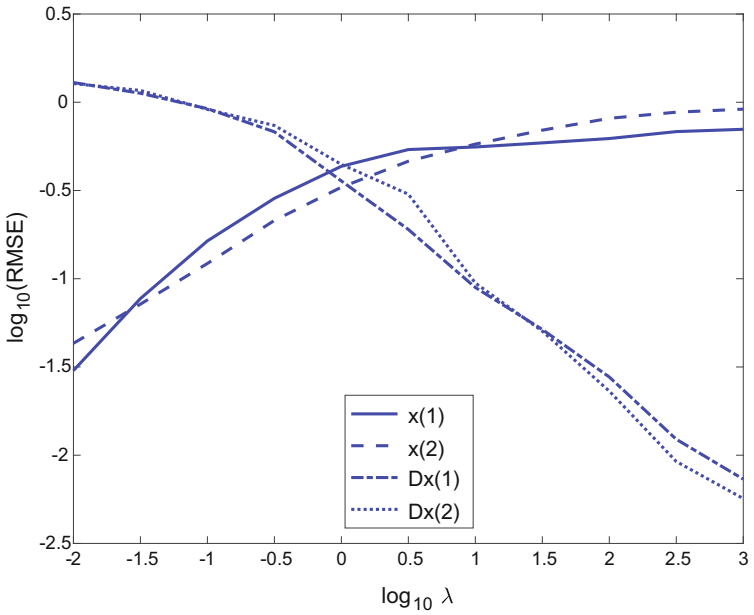


**Fig. 10.3** The common logarithms of the root-mean-square errors for the data and for the equation, and for the two chemostat variables $C$ and $B$, over a range of values of $\log_{10}(\lambda)$

A small disclaimer is required, however. In principle the two variables should have the same basis expansions, and hence the same coefficients. In fact, in this setup the two variables are expanded independently, so that there is more fitting power in this model than in that of Chap. 9. In fact, though, the two sets of coefficients are quite similar, as they should be. One way to deal with this discrepancy is by requiring the two coefficient vectors to be equal by using a linear constraint matrix.

## 10.5 Fitting a Compound Model to Blood Alcohol Data

This and the next two examples are taken from the invaluable collection of differential equation models in Schittkowski (2002). The systems in this collection includes many types other initial value value problems, but these alone contain 232 examples where data from actual experiments are provided, along with, in most cases, literature citations. Along with these data, the author also distributes a computer program called `EASYFIT` that uses the trajectory matching methods in Chap. 7 to compute parameter estimates along with confidence limits. The text, in addition, offers a comprehensive review of numerical methods for parameter estimation.

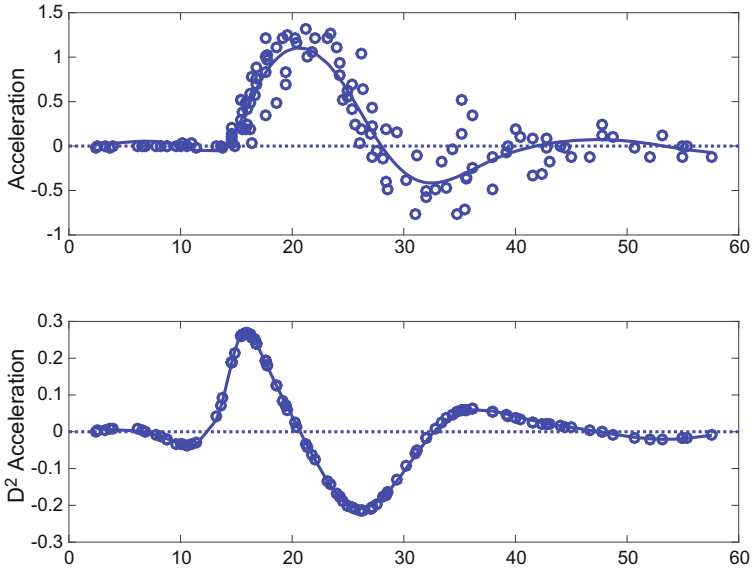


**Fig. 10.4** The *upper panel* displays the head impact data analyzed in Chap. 9 along with the fit to the data defined by the `CollocInfer` analysis for $\lambda = 100$. The *lower panel* shows the values of the second derivative at the observation times as points along with the right hand function value as a *curve*

Varah (1982) contains several interesting data sets fit by differential equations. Among these is a set of block ethanol concentrations over a two hour period of alchohol consumption followed by a 6.5 hour period of abstinence. Pharmacodynamic data like these can be modelled in various ways, but Varah selected a compound Michaelis–Menton model that switched as follows at the two-hour point:

$$\begin{aligned} \mathrm{D}x &= \alpha + \beta x/(\gamma + x), \quad t < 2 \\ \mathrm{D}x &= -\beta x/(\gamma + x), \quad t \geq 2. \end{aligned} \tag{10.11}$$

We can also set this up as a single equation with both additive- and rate-forcing:

$$\mathrm{D}x = \alpha_1 u_1 + u_2 \beta x/(\gamma + x)$$

where the additive forcing function $u_1$ switches from 1 to 0 and the rate forcing function switches from 1 to $-1$ at 2 h.

As above we set up the analysis with order 5 B-splines with knots at observation points and ranged $\log_{10}\lambda$ from $-2$ to 1 in unit steps. The four fits to the data and the equation are shown in Fig. 10.5 in the top and bottom panels, respectively. Again we see that the data and equation fits converge to a tight adherence to the differential equation. But a fair amount of fit is sacrificed along the way, and the fit at $\lambda = 0.1$ seems to strike a good compromise. Moreover, there is little change in the three coefficients over the $\lambda$ range, and the values $\alpha = 0.34$, $\beta = 181$, $\gamma = 1981$ can be considered as quite satisfactory estimates when compared with $\alpha = 0.22$, $\beta = 183$, $\gamma = 1957$ for the tight fit to the equation.

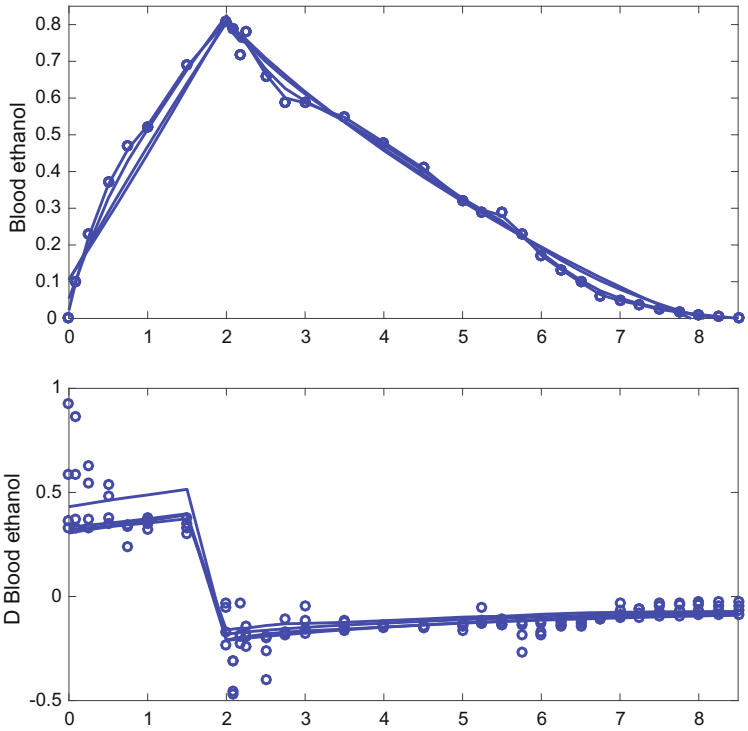


**Fig. 10.5** The *upper panel* displays blood alcohol concentration as as *circles*, and the fit to the data defined by the catalytic equation and $\lambda = 0.01, 0.1, 0, 10$ as *solid lines*. The *lower panel* displays the corresponding derivatives of the fit as *circles* and the *right sides* of the equation as *solid lines*

## 10.6 Fitting the Catalytic Growth Model to the Root Length Data

It is instructive to look at a bad idea now and then, as a warning that it is important to consider whether the data at hand can define the model in mind.

Differential equations are natural for modelling growth, and we have noted that most growth processes have upper limits defined by both internal structural factors and external constraints. The catalytic equation 4.4, which we express here as $\mathrm{D}x = \beta x(\gamma - x)$, is usually seen as the first model to consider for limited growth processes. We illustrate how we would use `CollocInfer` to estimate the rate parameter $\beta$ and the upper limit of root length parameter $\gamma$ by using 11 observations of root length of a fungus published by Buwalda et al. (1982).

The data, displayed in Fig. 10.6, involve root lengths ranging from 3.8 to 11591 centimeters, a *dynamic range* = 11591/3.8 of about 3000, and, as in the population dynamics data above, it is usually wise to analyze the logarithm of the data with a model altered to define this scale. Thus, we switched to log version $\mathrm{D}\log x = \beta(\gamma - x)$ as our differential equation for log root length. We also defined the parameters to be estimated as the logarithms of $\beta$ and $\gamma$ so that their values would be positive. We see that the two parameters have a trade-off relationship via their product, and preliminary analyses revealed that an estimate $\beta$ would be of the order of $10^{-5}$ while the limit parameter $\gamma$ be in the range of $10^4$. We switched to estimating the logarithms of the parameters after rescaling them, resulting the equivalent equation for log root length $\mathrm{D}\log x = \beta(10^{-4}\gamma - 10x)$.

We used order five B-spline basis functions with knots positioned at the times of observation and allowed $\log_{10}\lambda$ to step by 0.5 from 0 to 4. The optimizations converged rapidly at each step with a maximum number of iterations being four. Figure 10.7 displays these data in its upper panel and the values of derivative of the fit (circles) and the right side of the equation (line) for $\lambda = 10^4$. We see that the data are tightly fit, but that in the bottom panel the right and left sides of the differential equation differ enormously on the right. Moreover, these fits prevailed no matter how high we set $\lambda$, indicating that the optimum parameters value fit the data tightly and ignore the conflict between right and left sides of the equation.

We see what the problem is in both Figs. 10.6 and 10.7. The largest root lengths are still a very long way from whatever maximum root length might hold for this plant. There is simply no data to define $\gamma$, but the trade-off between $\beta$ and $\gamma$ in the catalytic model means that $\beta$ is therefore undefined by the data as well. We would be better off to work with a unexciting first order unrestricted growth model $\mathrm{D}x = \beta x$ or $\mathrm{D}\log x = -\beta$, possibly with a little time variation in the rate parameter. You can only estimate a model for the data that you have, and the catalytic model is an over–parameterization in this case.

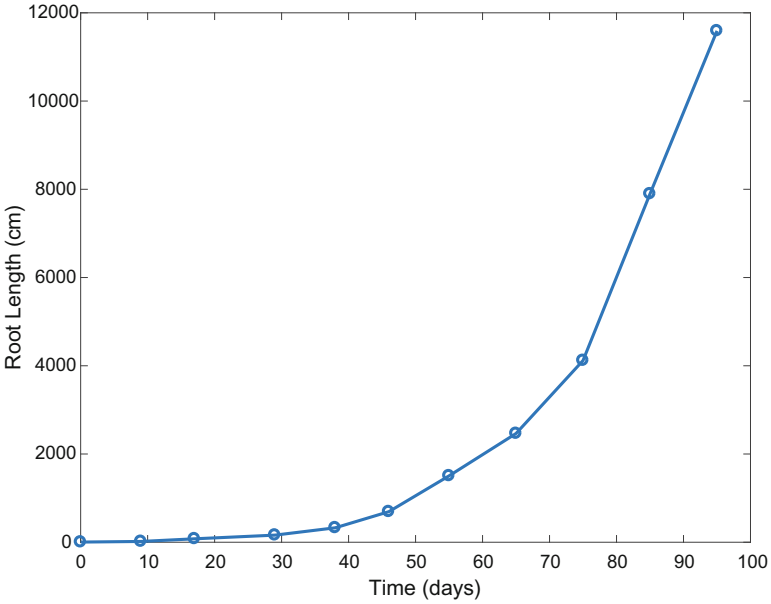


**Fig. 10.6** The length of the root of a fungus as a function of days of growth

## 10.7 Chemical Reactions Among Aromatic Compounds

We turn now to a model involving rather more equations than we have seen so far and with only partially observed variables. The system was proposed to H-G. Bock by a chemical engineer working in the Shell laboratory in Amsterdam as an interesting test case for his *multiple shooting* algorithm PARFIT (Bock 1983) described in Chap. 7. The system, some new data and an analysis by the EASY-FIT algorithm were later reported by Schittkowski (2005) and are available from that author. We report here results for these data.

The nine equations are

$$\begin{aligned}
\mathrm{D}x_1 &= -p_1x_1x_2 + p_2x_3 + 10^5x_3x_7 - 10^3x_1x_8 + 7\times 10^4x_6x_7 - 1.4\times 10^3x_1x_9\\
\mathrm{D}x_2 &= -p_1x_1x_2 + p_2x_3 - p_5x_2x_5 + p_6x_6\\
\mathrm{D}x_3 &= p_1x_1x_2 - p_2x_3 - p_3x_3 + p_4x_4x_5 - 10^5x_3x_7 + 10^3x_1x_8 + 7\times 10^3x_6x_8\\
&\quad -1.4\times 10^4x_3x_9
\end{aligned}$$

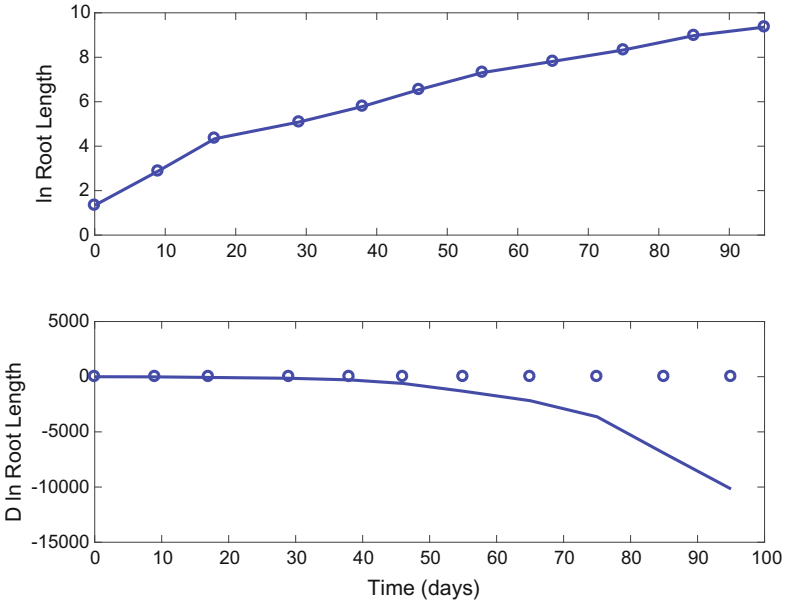


**Fig. 10.7** The *upper panel* displays the natural logarithm of root length as a function of days of growth as *circles*, and the fit to the data defined by the catalytic equation and $\lambda = 10^4$ as a *solid line*. The *lower panel* displays the derivative of the fit as *circles* and the *right side* of the equation as a *solid line*

$$
\begin{aligned}
\mathrm{D}x_4 &= p_3x_3 - p_4x_4x_5 \\
\mathrm{D}x_5 &= p_3x_3 - p_4x_4x_5 - p_5x_2x_5 + p_6x_6 \\
\mathrm{D}x_6 &= p_5x_2x_5 - p_6x_6 - 7\times 10^4 x_6x_7 + 1.4\times 10^3 x_1x_9 - 7\times 10^3 x_6x_8 \\
&\quad +1.4\times 10^4 x_3x_9 \\
\mathrm{D}x_7 &= -10^5 x_3x_7 + 10^3 x_1x_8 - 7\times 10^4 x_6x_7 + 1.4\times 10^3 x_1x_9 \\
\mathrm{D}x_8 &= 10^5 x_3x_7 - 10^3 x_1x_8 - 7\times 10^3 x_6x_8 + 1.4\times 10^4 x_3x_9 \\
\mathrm{D}x_9 &= 7\times 10^4 x_6x_7 - 1.4\times 10^3 x_1x_9 + 7\times 10^3 x_6x_8 - 1.4\times 10^4 x_3x_9\,. \qquad (10.12)
\end{aligned}
$$

These equations are typical of those used to describe chemical reactions among multiple reagents, where all variables are non-negative over all values of $t$. Each variable represents the concentration of a chemical species called an aromate, a molecule with a cyclic planar structure that is both highly stable and reactive. The terms are either single variables or products of variables, so that the system is effectively quadratic rather than linear. Thus, the equations are essentially extensions of the Lotka–Volterra and SIR models, and can be viewed as an essentially linear network but with rate-forcing of each variable by one or more other variables. The number of linkages of each variable to the others in the system ranges from two for $x_4$ to eight

or all others for the lynch-pin variable $x_3$. But, since $x_4$ is additively forced by $x_3$, it, too, has a central role in the system. We know that rate-forcing is apt to make a system twitchy in the sense that a variable may be highly sensitive to small changes in other variables.

There are six parameters, $p_1, \ldots, p_6$, that must be estimated from these data; and they are rate constants for, respectively,:

1. $x_1 x_2$ in equations 1, 2 and 3,
2. $x_3$ in equations 1, 2 and 3,
3. $x_3$ in equations 3, 4 and 5,
4. $x_4 x_5$ in equations 3, 4 and 5,
5. $x_2 x_5$ in equations 2, 5 and 6,
6. $x_6$ in equations 2, 5 and 6.

The right sides of equations 7, 8 and 9 are entirely determined by known constants. The EASY-FIT estimated parameter values were 0.037, 4.79, 47.4, 0.371, 0.217, and 49.3, respectively.

Only variables 4, 5, 7 and 8 are observed, and each at at 11 common time points, (0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.9, 1.2, 1.8, 2.4, 3.0).

We tweaked the initial values $x_i(0)$ used in the original EASY-FIT NLS analysis to (0.5, 3.0, 0, 0, 0.02, 0.01, 0.52, 0, 0.5), respectively. The approximation to the solution of the initial value problem using and unweighted least squares criterion is shown in Fig. 10.8, along with the data. The fit to the measurements seems quite credible, although with some small bias in the fits to variables 5, 7 and 8. At the scale of the plot, the variables seem to have a nice smooth exponential–like shape that would not seem to require a lot of basis functions to approximate. Variables $x_3$, $x_4$, $x_5$, and $x_8$ increase monotonically over the observation interval and have the profile of outputs from the reaction. Variables $x_1$, $x_6$, and $x_9$ look like chemical species used up by the reaction, and variables $x_2$ and $x_7$ look like by-products that will eventually return to zero values.

But there are indications that there might be problems. The approximation to the solution of the equation for our initial values by Matlab function `ode45` required 245,385 time steps over a range of error tolerance settings. Figure 10.9 shows a close-up of the approximation of $x_3$ over a sequence of 100 successive time points near the end of the interval. Although we can't be sure about how much of this high-frequency variation is due to the Runge–Kutta algorithm used for the approximation, this remains an impressive illustration of how a dynamic system that seems to have a straightforward structure can in fact determine substantial variation on a very fine time scale.

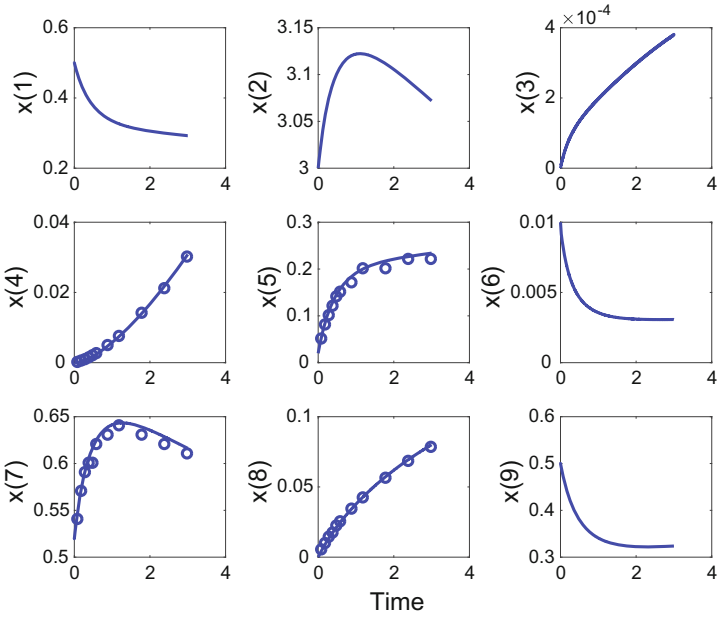


**Fig. 10.8** The *panels* show as *solid lines* the approximations to the solutions of nine equations representing reactions of aromatic compounds. The initial values were chosen to provide a good fit to the observations of variables $x_4$, $x_5$, $x_7$ and $x_8$ shown as *circles*

We also see a large variation in the sizes of the variables, amounting to four orders of magnitude, which would suggest that either we should switch to a logarithmic scale for our `CollocInfer` analysis, or use different weightings for a least squares analysis of the actual data. We opted for the second approach since one of the observations was zero, and therefore we multiplied the variable values by 10 to the powers (1, 0, 4, 2, 1, 2, 1, 1, 1), respectively. We positioned knots at 0 and at each of the 11 observation points and used an order 5 B-spline basis, so that there 15 basis functions used to approximate each variable.

We used as starting parameter values for our `CollocInfer` analysis those used in the EASY-FIT analysis. After some experimentation, we found that a sequence of smoothing parameter $\lambda$ values on a common log scale beginning at -1 and with step size 0.5 produced data and equation fits ranging from close for the data at $\log_{10}\lambda = -1$ to close for the fit to the first derivative by the right-side of each equation at about $\log_{10}\lambda = 2$. Along the way, we encountered serious problems with the negativity of some eigenvalues of the coefficient Hessian matrix for higher $\lambda$ values until we switched to the simpler scoring-type Hessian matrix produced by dropping the second partial derivative term. But, as in the numerical approximation for the initial value problem, we noted that as much as 1000 iterations of the inner coefficient loop were required for convergence.

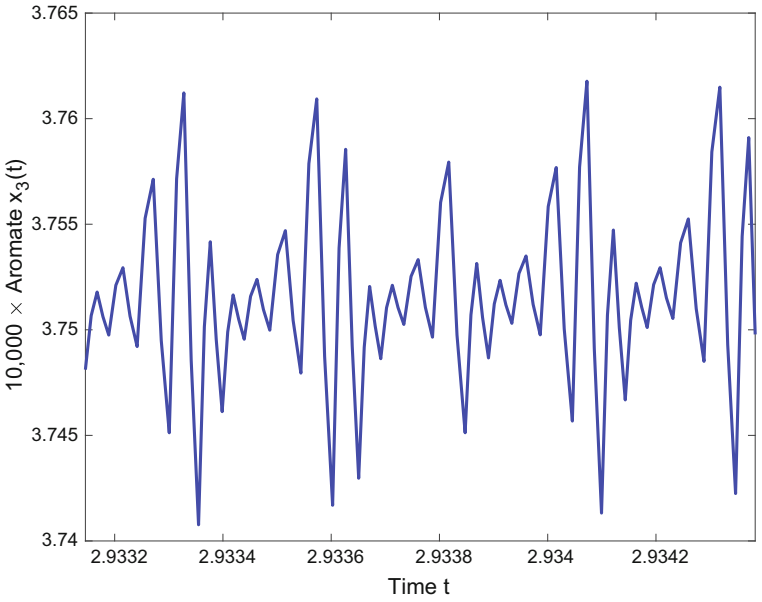


**Fig. 10.9** The variation of variable $x_3$ over 100 steps late in the entire sequence of length 245,385 of the Runge–Kutta initial value approximation

Figure 10.10 shows the solution at $\log_{10}\lambda = 2$, where the fit to the data just begins to deteriorate but the fit of the right side of the equation to the first derivative is close. Also shown in each panel as a dashed line is the EASY-FIT approximation in Fig. 10.8. The fits to the data are excellent, but the EASY-FIT and `CollocInfer` solutions for unmeasured variables differ. Variables $x_1$ and $x_2$ differ primarily in terms of a vertical shift, while the other unmeasured variables appear to differ primarily by a multiplying factor. Otherwise the shapes of all variables are essentially the same. We also started a `CollocInfer` analysis off by the optimal EASY-FIT values, which produce only a small improvement in fit over that displayed in Fig. 10.8. The converged values of outer function $H$ were 0.00035 and 0.00039 for the EASY-FIT and `CollocInfer` analyses, respectively. That is, the two solutions are nearly identical in terms of how they fit the data.

Figure 10.11 displays the values at the observation times of the first derivative of each variable along with the right side of the equation shown as a solid line. For all but small-amplitude variable $x_3$ these agree closely, so that we may say that the differential equation model provides an excellent account of both the data and the equations for either set of parameter values.

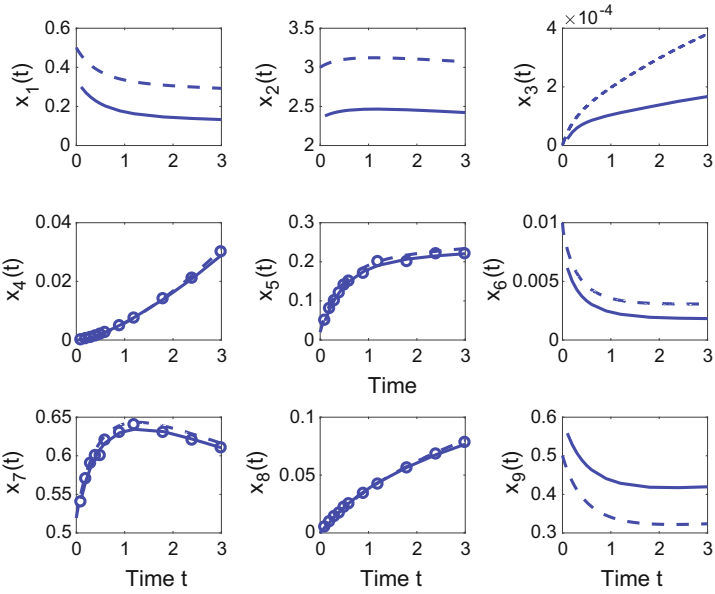


**Fig. 10.10** The *panels* show as *solid lines* the approximations to the solutions of nine equations representing reactions of aromatic compounds computed a `CollocInfer` analysis. The observed values are shown as *circles*. The *dashed lines* display the corresponding EASY-FIT solution in Fig. 10.8

How did the parameters change as we increased $\lambda$ from 0.1 to 100? Figure 10.12 displays this variation in common log terms, and we see that the amount of variation is rather small, so that we see here as well as in our other sample analyses that `CollocInfer` parameter estimates may be quite adequate without running the value of $\lambda$ to very high values. It is striking that the changes in initial values for parameters $p(2)$ and $p(4)$ were essentially negligible. This is consistent with 95% confidence limits reported by (Bock87) for these parameters that included zeros. Moreover, when we replaced the values of log(0.01) that we used as initial values by the corresponding EASY-FIT optimal values, we observed essentially the same estimated solutions and fits to the data. When we re-analyzed the data by optimizing only the other four parameters, we found no deterioration in fit and no change in the values of the parameters. It would appear that values of these parameters have no bearing on the fits to the four sets of observations. We may say, therefore, that our `CollocInfer` estimates for $p_1$, $p_3$, $p_5$ and $p_6$ are 0.085, 82.3, 0.218 and 61.4, respectively, but that we consider $p_2$ and $p_4$ as inestimable given the data at hand.

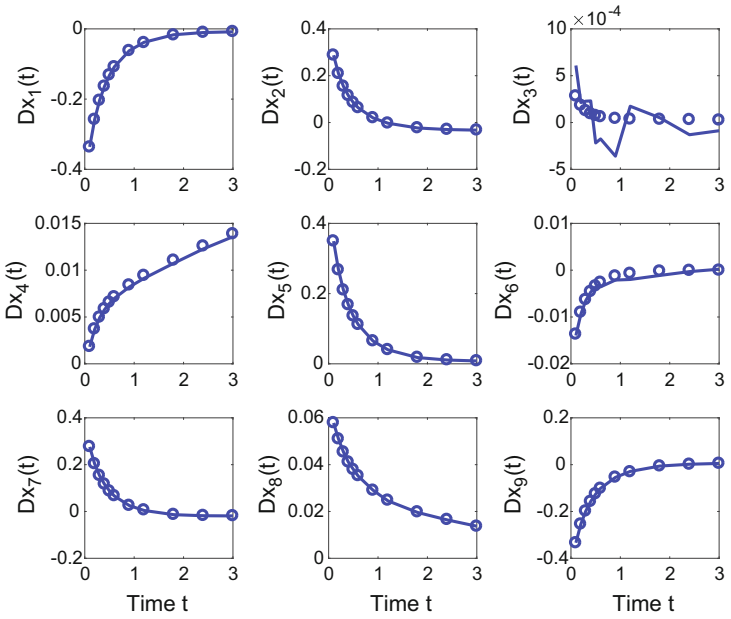


**Fig. 10.11** The *panels* show as *solid lines* the *right sides* of nine equations representing reactions of aromatic compounds computed a `CollocInfer` analysis. The observed values of the first derivatives at the observation points are shown as *circles*

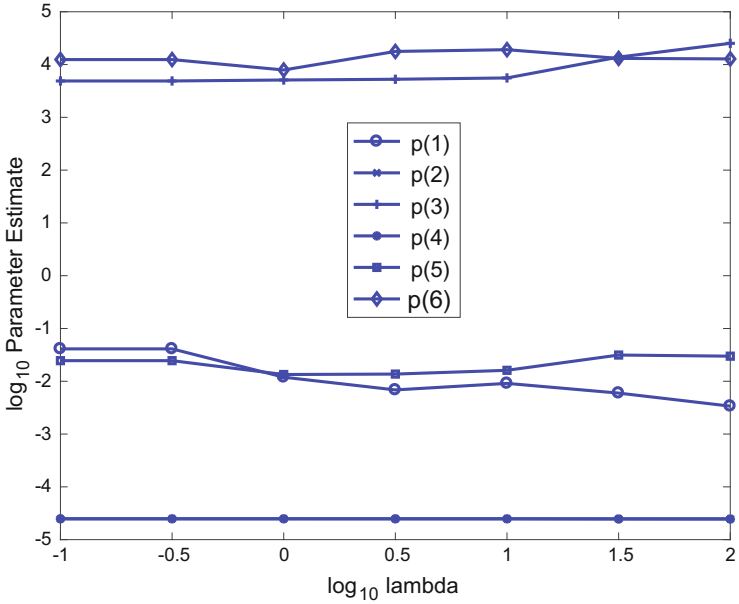


**Fig. 10.12** The variation in estimated parameter values over the range of smoothing parameter $\lambda$ used in the `CollocInfer` analysis of the aromate data

# References


Albert, J. (2009). *Bayesian computation with R*. New York: Springer Science & Business Media.

Alligood, K., Sauer, T., & Yorke, J. (1997). *Chaos: An introduction to dynamical systems*. New York: Springer.

Andrieu, C., Doucet, A., & Holenstein, R. (2010). Particle markov chain monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *72*(3), 269–342.

Arbabanel, H. D. I. (1996). *Analysis of observed chaotic data*. New York: Springer.

Azzimonti, L., Nobile, F., Sangalli, L. M., & Secchi, P. (2014). Mixed finite elements for spatial regression with pde penalization. *SIAM/ASA Journal on Uncertainty Quantification*, *2*(1), 305–335.

Baake, E., Baake, M., Bock, H. G., & Briggs, K. M. (1992). Fitting ordinary differential equations to chaotic data. *Physical Review A*, *45*, 5524–5529.

Barnes, B., & Fulford, G. R. (2009). *Mathematical modelling with case studies: A differential equations approach using Maple and MATLAB*. Boca Raton, Florida: Chapman and Hall/CRC.

Bates, D. M., & Watts, D. B. (1988). *Nonlinear regression analysis and its applications*. New York: Wiley.

Beaumont, M. A., Zhang, W., & Balding, D. J. (2002). Approximate bayesian computation in population genetics. *Genetics*, *162*(4), 2025–2035.

Bellman, R., & Roth, R. S. (1971). The use of splines with unknown end points in the identification of systems. *Journal of Mathematical Analysis and Applications*, *34*, 26–33.

Beskos, A., Papaspiliopoulos, O., Roberts, G. O., & Fearnhead, P. (2006). Exact and computationally efficient likelihood-based estimation for discretely observed diffusion processes (with discussion). *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *68*(3), 333–382.

Bhowmik, P., Das, S., & Konar, A. (2010). Emotion clustering from stimulated electroencephalographic signals using a duffing oscillator. *International Journal for Computers in Healthcare*, *1*, 66–84.

Blanchard, P., Devaney, R. L., & Hall, G. R. (2006). *Differential equations* (3rd ed.). Belmont, CA: Brooks/Cole.

Bock, H. G. (1983). Recent advances in parameter identification techniques for ODE. In P. Deuflhard & E. Harrier (Eds.), *Numerical treatment of inverse problems in differential and integral equations* (pp. 95–121). Basel: Birkhäuser.

Borrelli, R., & Coleman, C. (2004). *Differential equations: A modeling perspective*. New York: Wiley.

Brooks, S., Gelman, A., Jones, G., & Meng, X.-L. (2011). *Handbook of Markov chain Monte Carlo*. Boca Raton: CRC Press.




J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9

Brown, K. S., Hill, C. C., Calero, G. A., Myers, C. R., Lee, K. H., Sethna, J. P., et al. (2004). The statistical mechanics of complex signaling networks: Nerve growth factor signaling. *Physical Biology*, *1*(3), 184.

Brunel, N. (2008). Parameter estimation of ODEs via nonparametric estimators. *Electronic Journal of Statistics*, *2*, 1242–1267.

Buwalda, J. G., Ross, G. J. S., Stribley, D. B., & Tinker, P. B. (1982). The development of endomycorrhizal root systems. *New Phytologist*, *92*, 319–399.

Campbell, D., & Steele, R. (2012). Smooth functional tempering with application to nonlinear differential equation models. *Statistical Computing*, *22*, 429–443.

Cao, J., Fussmann, G. F., & Ramsay, J. O. (2008). Estimating a predator-prey dynamical model with the parameter cascades method. *Biometrics*, *64*, 959–967.

Coddington, E. A. (1961). *An introduction to ordinary differential equations*. New York: Dover.

Coddington, E. A., & Carlson, R. (1997). *Linear ordinary differential equations*. Philadelphia: SIAM.

Cortez, M. H., & Ellner, S. P. (2010). Understanding rapid evolution in predator-prey interactions using the theory of fast-slow dynamical systems. *The American Naturalist*, *176*(5), E109–E127.

Csilléry, K., Blum, M. G., Gaggiotti, O. E., & François, O. (2010). Approximate bayesian computation (abc) in practice. *Trends in Ecology & Evolution*, *25*(7), 410–418.

Dattner, I., & Gugushvili, S. (2015). Accelerated least squares estimation for systems of ordinary differential equations. arXiv:1503.07973.

Dattner, I., & Klaassen, C. A. (2015). Optimal rate of direct estimators in systems of ordinary differential equations linear in functions of the parameters. *Electronic Journal of Statistics*, *9*(2), 1939–1973.

Deuflhard, P., & Bornemann, F. (2000). *Scientific computing with ordinary differential equations*. New York: Springer.

Earl, D. J., & Deem, M. W. (2005). Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, *7*, 3910–3916.

Edwards, C. H. (1979). *The historical development of the calculus*. New York: Springer.

Ellner, S. P., & Guckenheimer, J. (2006). *Dynamic models in biology*. Princeton: Princeton University Press.

Ellner, S. P., Seifu, Y., & Smith, R. H. (2002). Fitting population dynamic models to time-series data by gradient matching. *Ecology*, *83*(8), 2256–2270.

Eubank, R. L., & Spiegelman, C. H. (1990). Testing the goodness of fit of a linear model via nonparametric regression techniques. *Journal of the American Statistical Association*, *85*(41), 387–392.

Fan, J., & Yao, Q. (2005). *Nonlinear time series: Nonparametric and parametric methods*. New York: Springer.

Fasiolo, M., Wood, S. N., Hartig, F., & Bravington, M. V. (2016). An extended empirical saddlepoint approximation for intractable likelihoods. arXiv:1601.01849.

Fearnhead, P., Giagos, V., & Sherlock, C. (2014). Inference for reaction networks using the linear noise approximation. *Biometrics*, *70*(2), 457–466.

FitzHugh, R. (1961). Impulses and physiological states in models of nerve membrane. *Biophysical Journal*, *1*, 445–466.

Font, J., & Fabregat, A. (1997). Testing a predictor-corrector integral method for estimating parameters in complex kinetic systems described by ordinary differential equations. *Computers and Chemical Engineering*, *21*(7), 719–731.

Girolami, M., & Calderhead, B. (2011). Riemannian manifold Landgevin and Hamiltonian Monte Carlo. *Journal of the Royal Statistical Society*, *73*(2), 123–214.

Golightly, A., & Wilkinson, D. J. (2011). Bayesian parameter inference for stochastic biochemical network models using particle Markov chain Monte Carlo. *Interface Focus*, *1*(6), 1–14.

Gugushvili, S., & Klaassen, C. A. (2012). n-consistent parameter estimation for systems of ordinary differential equations: Bypassing numerical integration via smoothing. *Bernoulli*, *18*(3), 1061–1098.

Gutenkunst, R. N., Waterfall, J. J., Casey, F. P., Brown, K. S., Myers, C. R., & Sethna, J. P. (2007). Universally sloppy parameter sensitivities in systems biology models. *PLoS Computational Biology*, *3*(10), e189.

Härdle, W. (1990). *Applied nonparametric regression*. Cambridge: Cambridge University Press.

He, D., Ionides, E. L., & King, A. A. (2010). Plug-and-play inference for disease dynamics: Measles in large and small populations as a case study. *Journal of the Royal Society Interface*, *7*(43), 271–283.

Himmelblau, D., Jones, C., & Bischoff, K. (1967). Determination of rate constants for complex kinetics models. *IEC Fundamentals*, *6*(4), 539–543.

Hirsch, M. W., Smale, S., & Devaney, R. L. (2013). *Differential equations, dynamical systems and an introduction to chaos*. Amsterdam: Academic Press.

Hodgkin, A. L., & Huxley, A. F. (1952). A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, *133*, 444–479.

Hooker, G. (2009). Forcing function diagnostics for nonlinear dynamics. *Biometrics*, *65*, 613–620.

Hooker, G., & Biegler, L. (2007). IPOPT and neural dynamics: Tips, tricks and diagnostics. Technical Report BU-1676-M, Dept. Bio. Stat. and Comp. Bio., Cornell University.

Hooker, G., Ellner, S. P., Roditi, L. D. V., & Earn, D. (2011). Parameterizing state-space models for infectious disease dynamics by generalized profiling: Measles in ontario. *Journal of the Royal Society Interface*, *8*, 961–975.

Hooker, G., Xiao, L., & Ramsay, J. (2014). CollocInfer: Collocation inference for dynamic systems. *R package version*, *1*,

Ionides, E. L., Bretó, C., & King, A. A. (2006). Inference for nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, *103*, 18438–18443.

Jagacinski, R. J., & Flach, J. M. (2003). *Control theory for humans: Quantitative approaches to modeling performance*. Mahwah, New Jersey: Lawrence Erlbaum Associates.

Kandel, E. R., Schwartz, J. H., & Jessell, T. M. (Eds.). (2000). *Principles of neuroscience*. New York: McGraw-Hill.

Kantz, H., & Schreiber, T. (2003). *Nonlinear time series analysis*. Cambridge: Cambridge University Press.

Kaufman, C. G., Bingham, D., Habib, S., Heitmann, K., & Frieman, J. A. (2011). Efficient emulators of computer experiments using compactly supported correlation functions, with an application to cosmology. *The Annals of Applied Statistics*, *5*, 2470–2492.

Keeling, M. J., & Rohani, P. (2008). *Modeling infectious diseases in humans and animals*. New York: Princeton University Press.

Kennedy, M., & O'Hagan, A. (2001). Bayesian calibration of computer models. *Journal of the Royal Statistical Society, Series B*, *63*, 425–464.

Kermack, W. O., & McKendrick, A. G. (1927). A contribution to the mathematical theory of epidemics. *Proceedings of the Royal Society of London, A*, *115*, 700–721.

Kincaid, D., & Cheney, E. (2002). *Pure and applied undergraduate texts. Numerical analysis: Mathematics of scientific computing*. Providence: American Mathematical Society.

Kutznetsov, Y. A. (2004). *Elements of applied bifurcation theory*. New York: Springer.

Lele, S. R., Nadeem, K., & Schmuland, B. (2010). Estimability and likelihood inference for generalized linear mixed models using data cloning. *Journal of the American Statistical Association*, *105*, 1617–1625.

Liang, H., & Wu, H. (2008). Parameter estimation for differential equation models using a framework of measurement error in regression models. *Journal of the American Statistical Association*, *103*, 1570–1583.

Lu, T., Liang, H., Li, H., & Wu, H. (2011). High dimensional ODEs coupled with mixed-effects modeling techniques for dynamic gene regulatory network identification. *Journal of the American Statistical Association*, *106*(496), 1242–1258.

Marlin, T. E. (2000). *Process control*. New York: McGraw-Hill.

Miao, H., Xia, X., Perelson, A., & Wu, H. (2011). On identifiability of nonlinear ODE models and applications in viral dynamics. *SIAM Review*, *53*, 3–39.

Morris, C., & Lecar, H. (1981). Voltage oscillations in the barnacle giant muscle fiber. *Biophysics Journal*, *35*, 193–213.

Nagle, R., Saff, E., & Snider, A. (2008). *Fundamentals of differential equations and boundary value problems*. Boston: Pearson Addison-Wesley.

Neidinger, R. D. (2010). Introduction to automatic differentiation and matlab object-oriented programming. *SIAM Review*, *52*, 545–563.

Nocedal, J., & Wright, S. J. (2006). *Numerical optimization* (2nd ed.). New York: Springer.

Pascual, M., & Ellner, S. P. (2000). Linking ecological patterns to environmental forcing via nonlinear time series models. *Ecology*, *81*(10), 2767–2780.

Paul, D., Peng, J., & Burman, P. (2011). Semiparametric modeling of autonomous nonlinear dynamical systems with applications to Plant Growth. *The Annals of Applied Statistics*, *5*(3), 2078–2108.

Pearson, K. (1895). Contributions to the mathematical theory of evolution, ii: Skew variation in homogeneous material. *Philosophical Transactions of the Royal Society*, *186*, 343–414.

Ramsay, J. (1996). Principal differential analysis: Data reduction by differential operators. *Journal of the Royal Statistical Society: Series B (Methodological)*, *58*(3), 495–508.

Ramsay, J. O. (2000). Functional components of variation in handwriting. *Journal of the American Statistical Association*, *95*, 9–15.

Ramsay, J. O., & Heckman, N. (2000). Some theory for l-spline smoothing. *CRM Proceedings and Lecture Notes*, *18*, 371–380.

Ramsay, J. O., & Silverman, B. W. (2005). *Functional data analysis*. New York: Springer.

Ramsay, J. O., Hooker, G., Campbell, D., & Cao, J. (2007). Parameter estimation in differential equations: A generalized smoothing approach. *Journal of the Royal Statistical Society, Series B*, *16*, 741–796.

Ramsay, T. (2002). Spline smoothing over difficult regions. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *64*(2), 307–319.

Rossi, N., Wang, X., & Ramsay, J. (2002). Nonparametric Item Response Function Estimates with the EM Algorithm. *Journal of Educational and Behavioral Statistics*, *27*(3), 291–317.

Rosenzweig, M., & MacArthur, R. (1963). Graphical representation and stability conditions of predator-prey interaction. *American Naturalist*, *97*, 209–223.

Ruppert, D., Wand, M. P., & Carroll, R. J. (2005). *Semiparametric regression*. Cambridge: Cambridge University Press.

Schittkowski, K. (2002). *Numerical data fitting in dynamical systems*. Dordrecht: Kluwer Academic Publishers.

Schittkowski, K. (2005). *Numerical data fitting in dynamical systems*. Dordrecht: Kluwer Academic Publishers.

Schmidt, M., & Lipson, H. (2009). Distilling free-form natural laws from experimental data. *Science*, *324*(5923), 81–85.

Schulte, O., & Drew, M . S. (2010). Discovery of conservation laws via matrix search. *Discovery science* (pp. 236–250). Berlin: Springer.

Seber, G. A. F., & Wild, C. J. (2003). *Nonlinear regression*. New York: Wiley.

Silverman, B. W. (1985). Some aspects of the spline smoothing approach to non-parametric regression curve fitting. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *47*, 1–52.

Tennenbaum, M., & Pollard, H. (1963). *Ordinary differential equations*. New York: Harper and Row.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 267–288.

Tien, J. H., & Guckenheimer, J. (2008). Parameter estimation for bursting neural models. *Journal of Computational Neuroscience*, *24*(3), 358–373.

Timmer, J., Rust, H., Horbelt, W., & Voss, H. U. (2000). Parametric, nonparametric and parametric modelling of a chaotic circuit time series. *Physical Letters A*, *274*, 123–134.

Tjoa, I.-B., & Biegler, L. (1991). Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *Industrial Engineering and Chemical Research*, *30*, 376–385.

Tong, H. (1990). *Non-linear time series: A dynamical system approach*. Oxford: Oxford Science Publications.

Transtrum, M. K., Machta, B. B., & Sethna, J. P. (2011). Geometry of nonlinear least squares with applications to sloppy models and optimization. *Physical Review E*, *83*(3), 036701.

Vajda, S., Valko, P., & Yermakova, A. (1986). A direct-indirect procedure for estimation of kinetic parameters. *Computers and Chemical Engineering*, *10*(1), 49–58.

Varah, J. M. (1982). A spline least squares method for numerical parameter estimation in differential equations. *SIAM Journal on Scientific Computing*, *3*, 28–46.

Wilson, H. R. (1999). *Spikes, decisions and actions: The dynamical foundations of neuroscience*. Oxford: Oxford University Press.

Wood, S. N. (2010). Statistical inference for noisy nonlinear ecological dynamic systems. *Nature*, *466*, 1102–1104.

Wu, H., Xue, H., & Kumar, A. (2012). Numerical discretization-based estimation methods for ordinary differential equation models via penalized spline smoothing with applications in biomedical research. *Biometrics*, *68*(2), 344–52.

Yermakova, A., Vajda, S., & Valko, P. (1982). Direct integral method via spline-approximation for estimating rate constants. *Applied Catalysis*, *2*, 139–154.

Yoshida, T., Ellner, S. P., Jones, L. E., Bohannan, B. J. M., Lenski, R. E., & Hairston, N. G. (2007). Cryptic population dynamics: Rapid evolution masks trophic interactions. *PLOS Biology*, *5*, 1868–1879.

Zheng, W., McAuley, K., Marchildon, K., & Yao, K. Z. (2005). Effects of end-group balance on melt-phase nylon 612 polycondensation: Experimental study and mathematical model. *Industrial and Engineering Chemistry Research*, *44*, 2675–2686.

# Index





J. Ramsay and G. Hooker, *Dynamic Data Analysis*, Springer Series in Statistics,
DOI 10.1007/978-1-4939-7190-9