MATH5824 Generalized Linear and Additive Models

Robert G Aykroyd

5/25/23

Table of contents

Weekly schedule

Revision and Examination Weeks (15 - 31 May)

- Please do not forget that the examination this year is **closed-book** and all questions and you will be expected to attempt all questions.

Week 11 (8 - 12 May)

Revision and examination preparation.

Week 10 (1 - 5 May)

Questions 10 & 11 from the MATH3823 *Exercises from last year* which are available on Minerva. These will consider situations which we have not really met in Lectures and will fill gaps compared to the syllabus.
On Friday we can look at some of: Questions 2, 3, 4 and 7 on the MATH5824 *Exercises from last year* sheet on Minerva.
If there is time, then other questions from the same Exercise sheets will also be discussed.

Week 9 (27 - 31 March)

- **Before next Lecture:** Re-read MATH3823 *Chapter 6: Log-linear Models* and MATH5824 *Chapter 5: Choosing the Smoothing Parameter.*
- **Lecture on Tuesday:** Start MATH3823 *Chapter 7: Extensions to Loglinear models.*
- **Lecture on Thursday:** Continue MATH3823 *Chapter 7: Extensions to Log-linear models.*
- **Lecture on Friday:** MATH5824 *Chapter 6: General Additive Models.*
- **Weekly feedback:** Relevant exercise questions from last year (solutions online).

Coursework Practical Sessions (20 - 24 March)

- Coursework for this module involves a single written report worth 20% of the module grade. This will mainly involve investigating different models using **R** and interpreting the results. Tasks are expected to be handed out on 14 March with **extended** hand-in deadline expect to be **5 April**. See *Learning Resources / Practical Assessment* for details of the tasks and for submission links.

Week 8 (20 - 24 March)

- **Before next Lecture:** Re-read *Chapter 5: Sections 5.1-5.3.*
- **Lecture on Tuesday:** Cancelled due to UCU strike.
- **Computer Practical on Tuesday:** Cancelled due to UCU strike.
- **Computer Practical on Wednesday:** Cancelled due to UCU strike.
- **Lecture on Thursday:** Continue *Chapter 6: Loglinear Modelling.*
- **Lecture on Friday:** Complete MATH5824 *Chapter 5: Choosing the Smoothing Parameter* with *Sections 5.4-5.6.*
- **Weekly feedback:** Relevant exercise questions from last year (solutions online).

Week 7 (13 - 17 March)

- **Before next Lecture:** Re-read *Chapter 5: Sections 5.1-5.3.*
- **Lecture on Tuesday:** Complete *Chapter 5:* by looking at *Section 5.4 - Application to dose-response experiments.*
- **Lecture on Thursday:** Cancelled due to UCU strike, please self-study *Chapter 6: Loglinear Modelling* by reading *Section 6.1 - Motivating Examples.*
- **Lecture on Friday:** Cancelled due to UCU strike, please self-study MATH5824 *Chapter 5: Choosing the Smoothing Parameter* by reading *Sections 5.1-5.3.*
- **Weekly feedback:** Relevant exercise questions from last year (solutions online).

Week 6 (6 - 10 March)

- **Before next Lecture:** Re-read *Chapter 4: Sections 4.1-4.4.*
- **Lecture on Tuesday:** Complete **Chapter 4:** with Section 4.5 Fitting GLMs in R
- **Lecture on Thursday:** Start *Chapter 5: Logistic Regression.*
- **Lecture on Friday:** MATH5824 *Chapter 4 Sections 4.5 Smoothing splines in R.*
- **Weekly feedback:** Relevant exercise questions from last year (solutions online).

Week 5 (27 February - 3 March)

- **Before next Lecture:** Re-read *Chapter 4: Section 4.1.*
- **Lecture on Tuesday:** Cancelled due to illness. Please read *Chapter 4: Section 4.2.*
- **Lecture on Thursday:** *Chapter 4: Sections 4.3, 4.4 & 4.5.*
- **Weekly feedback:** Start Exercises in *Chapter 4.*

Week 4 (20 - 24 February)

- **Before next Lecture:** Be confident with all material up to, and including, *Section 3.4 Moments of exponential-family distributions.*
- **Lecture on Tuesday:** *Chapter 3: Sections 3.5 & 3.6*
- **Lecture on Thursday:** Start Chapter 4 by covering *Section 4.1.*
- **Weekly feedback:** Complete Exercises in *Chapter 3.*

Week 3 (13 - 17 February)

- **Before next Lecture:** Be confident with material in *Chapter 2: Essentials of Normal Linear Models.*
- **Lecture on Tuesday:** Cancelled due to UCU strike. Instead, self-study *Chapter 3: Sections 3.1 & 3.2.*
- **Lecture on Thursday:** Cancelled due to UCU strike. Instead, self-study *Chapter 3: Sections 3.3 & 3.4.*
- **Before next Lecture:** Complete questions and check solutions, including video(s), for all Exercises in *Chapters 1 and 2.* Start Exercises in *Chapter 3.*

Week 2 (6 - 10 February)

- **Before next Lecture:** Please re-read *Section 2.1: Overview* and read *Section 2.2: Types of normal linear model.*
- **Lecture on Tuesday:** We will briefly cover all remaining material in *Chapter 2: Essentials of Normal Linear Models.*
- **Before next Lecture:** Please re-read *Chapter 2* carefully.
- **Lecture on Thursday:** Cancelled due to UCU strike.
- **Weekly feedback:** Self-study the Exercises in *Section 2.6* – solutions to be posted during Week 3.

Week 1 (30 January - 3 February)

- **Before next Lecture:** Please read the *Overview.*
- **Lecture on Tuesday:** We will briefly cover all material in *Chapter 1: Introduction.*
- **Before next Lecture:** Please re-read *Chapter 1* carefully.
- **Lecture on Thursday:** Start *Chapter 2: Essentials of Normal Linear Models* with *Section 2.1: Overview.*
- **Weekly feedback:** Self-study the Exercises in *Section 1.5* – solutions to be posted during Week 1.

Overview

Preface

These lecture notes are produced for the University of Leeds module "MATH5824 - Generalized Linear and Additive Models" for the academic year 2022-23. They are based on those used previously for this module and I am grateful to previous module lecturers for their considerable effort: Lanpeng Ji, Amanda Minter, John Kent, Wally Gilks, and Stuart Barber. This is the first year, however, that they have been produced in accessible format and hence some errors might occur during this conversion process. For information, I am using Quarto (a successor to RMarkdown) from RStudio to produce both the html and PDF, and then GitHub to create the website which can be accessed at rgaykroyd.github.io/MATH3823/. Please note that the PDF versions will only be made available on the University of Leeds Minerva system. Although I am a long-term user of RStudio, I have not previously used Quarto/RMarkdown nor Github and hence please be patient if there are hitches along the way.

In the Level 3 component of this module, we extend the simple linear regression model to the generalized linear model which can cope with non-normally distributed response variables, in particular data following binomial and Poisson distributions. However, we still just use linear functions of the predictor variables. A further extension of the linear model is the generalized additive model. Here, we no longer insist on the predictor variables affecting the response via a linear function of the predictors, but allow the response to depend on a more general smooth function of the predictor. In the Level 5 component of this module, we study splines and their use in interpolating and smoothing the effects of explanatory variables in the generalized linear models of the Level 3 component of this module (see separate Lecture Notes accompanying MATH3823).

RG Aykroyd, Leeds, November 22, 2022

⚠ Warning

Statistical ethics and sensitive data
Please note that from time to time we will be using data sets from situations which some might perceive as sensitive. All such data sets will, however, be derived from real-world studies which appear in textbooks or in scientific journals. The daily work of many statisticians involves applying their professional skills in a wide variety of

situations and as such it is important to include a range of commonly encountered examples in this module. Whenever possible, sensitive topics will be signposted in advance. If you feel that any examples may be personally upsetting then, if possible, please contact the module lecturer in advance. If you are significantly effected by any of these situations, then you can seek support from the Student Counselling and Wellbeing service.

Official Module Description

Module summary

Linear regression is a tremendously useful statistical technique but is limited to normally distributed responses. Generalised linear models extend linear regression in many ways - allowing us to analyse more complex data sets. In this module we will see how to combine continuous and categorical predictors, analyse binomial response data and model count data.A further extension is the generalised additive model. Here, we no longer insist on the predictor variables affecting the response via a linear function of the predictors, but allow the response to depend on a more general smooth function of the predictor.

Objectives

On completion of this module, students should be able to:

- carry out regression analysis with generalised linear models including the use of link functions, deviance and overdispersion;
- fit and interpret the special cases of log linear models and logistic regression;
- compare a number of methods for scatterpot smoothing suitable for use in a generalised additive model;
- use a backfitting algorithm to estimate the parameters of a generalised additive model;
- interpret a fitted generalised additive model;
- use a statistical package with real data to fit these models to data and to write a report giving and interpreting the results.

Syllabus

Generalised linear model; probit model; logistic regression; log linear models; scatterplot smoothers; generalised additive model.

University Module Catalogue

For any further details, please see MATH5824 Module Catalogue page

1 Non-parametric Modelling

1.1 Motivation

Table 1.1 reports on the depth of a coal seam determined by drilling bore holes at regular intervals along a line. The depth y at location $x = 6$ is missing: could we estimate it?

Table 1.1: Coal-seam depths (in metres) below the land surface at intervals of 1 km along a linear transect.

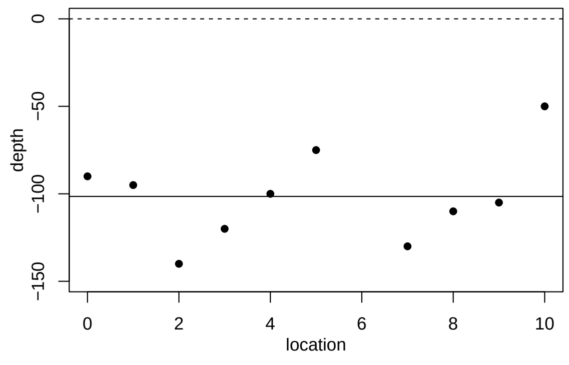
Location, x	0	1	2	3	4	5	6	7	8	9	10
Depth, y	-90	-95	-140	-120	-100	-75	NA	-130	-110	-105	-50

Figure 1.1 plots these data, superimposed with predictions from several polynomial regression models.

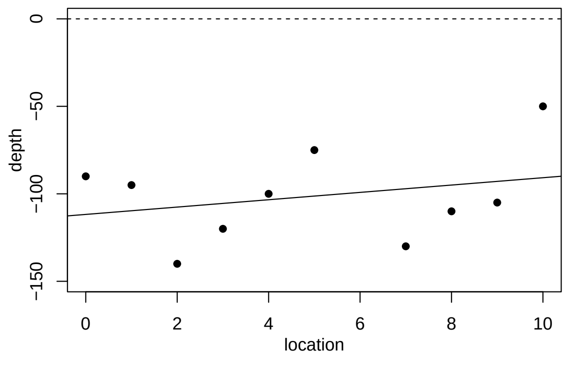
Each of these models would predict a different value for the missing observation y_6. We do not know the accuracy of the depth measurements, so in principle any of these curves could be correct. Clearly, the residual variance is largest for the constant-depth model in Figure 1.1a, and smallest for the cubic polynomial in Figure 1.1c. However, none of these models produces a convincingly good fit. Moreover, these models are not particularly believable, since we know that geological pressures exerted over very long periods of time cause the landscape and its underlying layers of rock to undulate and fracture. This suggests we need a different strategy.

Next, consider the simulated example in Figure 1.2. At first look we might be happy with the fitted curves in Figure 1.2a or Figure 1.2b. The data, however, are created with a *change-point* at $x = 0.67$ where the relationship changes from linear with slope 0.6 to a constant value of 0.75. This description is completely lost with these two models.

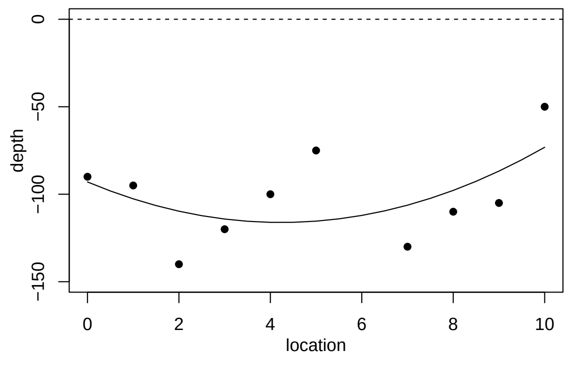
Figure 1.2c shows the result of fitting one linear function to the data below 0.67 and a second linear function above. Clearly, this fits well but it has assumed that the change-point location is known – which is unrealistic. Finally, Figure 1.2d shows a fitted *cubic smoothing spline* to the data – we will studies these models later. This shows an excellent fit and leads to appropriate conclusions. That is, the relationship is approximately linear for small values, then there is a rapid increase, and finally a near constant value for high values. Of course, this is not exactly as the true relationship with a discontinuity at $x = 0.67$ but



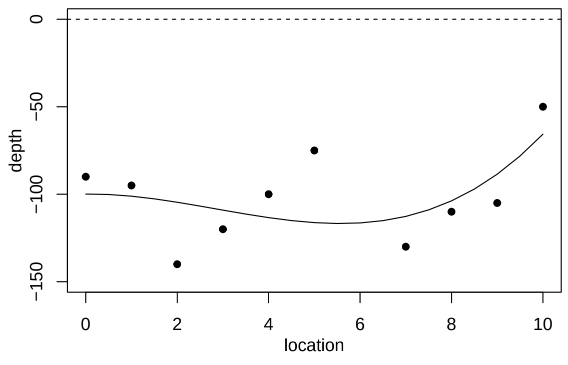
(a) Constant model



(b) Linear model

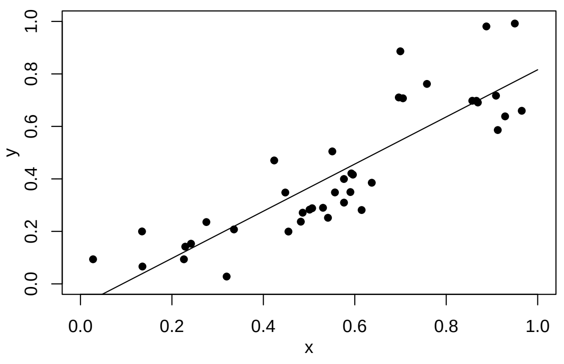


(c) Quadratic model

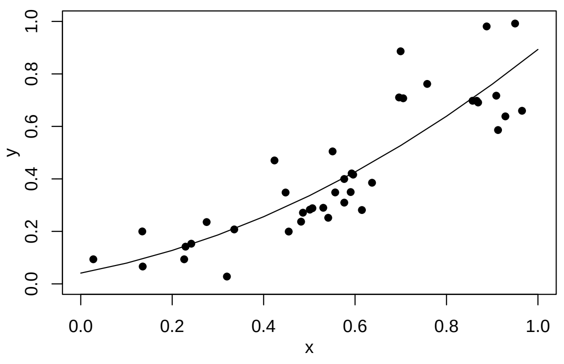


(d) Cubic model

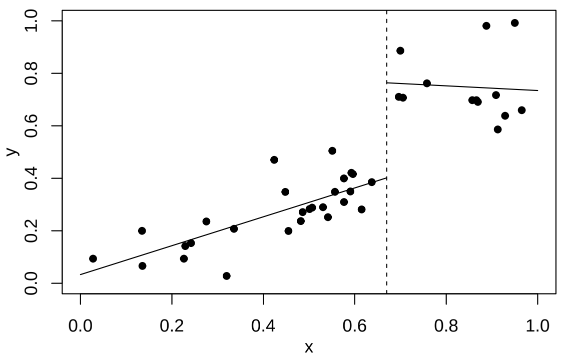
Figure 1.1: The coal-seam data superimposed with predictions from polynomial regression models.



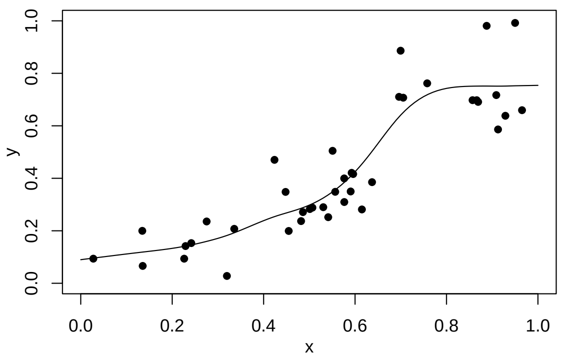
(a) Linear model



(b) Quadratic model



(c) Piecewise linear model

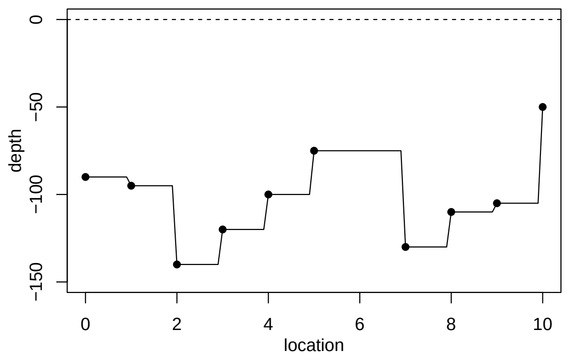


(d) Cubic smoothing spline

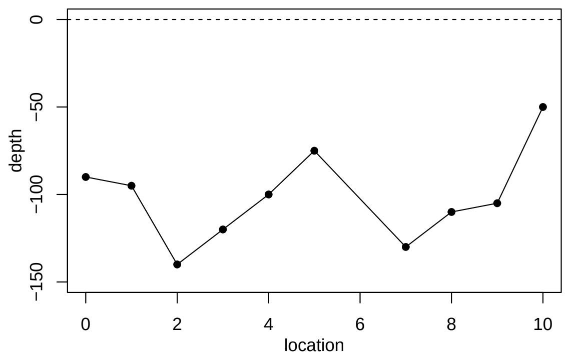
Figure 1.2: Simulated data superimposed with predictions from various models.

it would definitely suggest something extreme occurs between about 0.6 to 0.7. Full details will follow later, but the cubic spline fits local cubic polynomials which are constrained to create a continuous curve.

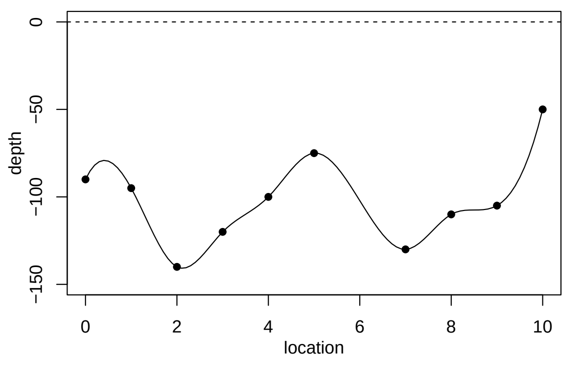
Now returning to the coal seam data. Figure 1.3 shows the data again, superimposed with predictions from methods which are not constrained to produce such smooth curves.



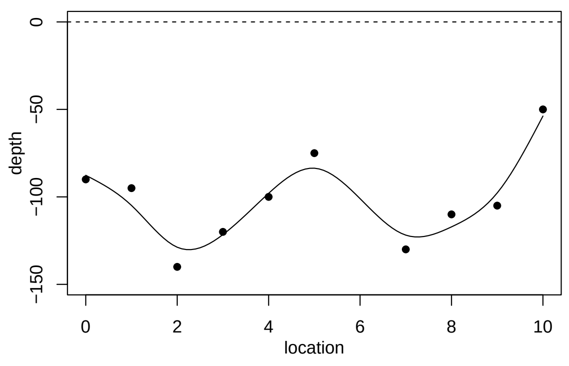
(a) Constant interpolating spline



(b) Linear interpolating spline



(c) Cubic interpolating spline



(d) Cubic smoothing spline

Figure 1.3: The coal-seam data superimposed with predictions from various spline models.

The simplest method, *constant-spline interpolation*, assumes that the dependent variable remains constant between successive observations, with the result shown in Figure 1.3a. However, the discontinuities in this model make it quite unreliable. A better method, whose results are shown in Figure 1.3b, is *linear-spline interpolation*, which fits a straight line between successive observations. Even so, this method produces discontinuities in the *gradient* at each data point. A better method still, shown in Figure 1.3c, is *cubic spline interpolation*, which fits a cubic polynomial between successive data points such that both the gradient and the curvature at each data point is continuous.

A feature of all these interpolation methods is that they fit the data exactly. Is this a good thing? The final method assumes that there may be some measurement error in the observations, which justifies fitting a smoother cubic spline than the cubic interpolating spline, but as we see in Figure 1.3d which does not reproduce the data points exactly. Is this a bad thing? We will see during this module how to construct and evaluate these curves. Here, the results are presented only for motivation.

1.2 General modelling approaches

We wish to model the dependence of a response variable y on an explanatory variable x, where y and x are both continuous. We observe y_i at each time x_i, for $i = 1, \dots, n$, where the observation locations are ordered: $x_1 < x_2 < \dots < x_n$. We imagine that the y's are noisy versions of a smooth function of x, say $f(x)$. That is,

$$y_i = f(x_i) + \epsilon_i, \tag{1.1}$$

where the $\{\epsilon_i\}$ are i.i.d:

$$\epsilon_i \sim \mathrm{N}(0, \sigma^2). \tag{1.2}$$

We suppose we do not know the correct form of function f: how can we estimate it?

It is useful to divide modelling approaches into two broad types: parametric and non-parametric.

Parametric models

By far the most common parametric model is simple linear regression, for example, $f(x) = \alpha + \beta x$, where parameters α and β are to be estimated. This is, of course, the simplest example of the polynomial model family, $f(x) = \alpha + \beta x + \gamma x^2 + \cdots + \omega\ x^p$, where p is the *order* of the polynomial and where all of $\alpha, \beta, \gamma, \dots, \omega$ are to be estimated. This has as special cases: quadratic, cubic, quartic, and quintic polynomials models. Also common are exponential models, for example $f(x) = \alpha e^{-\beta x}$, where α, β are to be estimated – do not confuse this with the exponential probability density function.

Note that the polynomial models are all linear functions *of the parameters*. They are standard forms in regression modelling, as studied in MATH3714 (Linear regression and Robustness) and MATH3823 (Generalized linear models). The exponential model, however, is an example of a model which is non-linearly in the parameters – it is an example of a *non-linear regression model*.

Although very many parametric models exist, they are all somewhat inflexible in their description of f. They cannot accommodate arbitrary fluctuations in $f(x)$ over x because they contain only a small number of parameters (degrees-of-freedom).

Non-parametric models

In such models, f is assumed to be a smooth function of x, but otherwise we do not know what f looks like. A *smooth function* f is such that $f(x_i)$ is close to $f(x_j)$ whenever x_i is close to x_j. To characterize and fit f we will use an approach based on *splines*. In practice, different approaches to characterizing and fitting smooth f lead to similar fits to the data. The spline approach fits neatly with normal and generalized linear models (NLMs and GLMs), but so do other approaches (for example, kernel smoothing and wavelets). Methods of fitting f based on kernel smoothing and the Nadaraya–Watson estimator are studied in the Level 5 component of MATH5714 (Linear regression, robustness and smoothing) where the choice of bandwidth in kernel methods is analogous to the choice of smoothing parameter value in spline smoothing.

Piecewise polynomial models

A common problem with low-order polynomials is that they can often fit well for part of the data but have unappealing features elsewhere. For example, although none of the models in Figure 1.1 fit the data at all well, we might imagine that three short linear segments might be a good fit to the coal-seam data. Also, the piecewise linear model was a good description of the data in Figure 1.2c. This suggests that local polynomial models might be useful. In some situation, for example when we know that the function f is continuous, jumps in the fitted model, as in Figure 1.2c, are unacceptable. Alternatively, we may require differentiability of f. Such technical issues lead to the use of *splines,* which is introduced in the next chapter.

2 Introducing Splines

2.1 Basic definitions

Let $t_1 < t_2 < \ldots < t_m$ be a fixed set of *sites* or *knots* which need not correspond to observation locations, as in Figure 2.1.

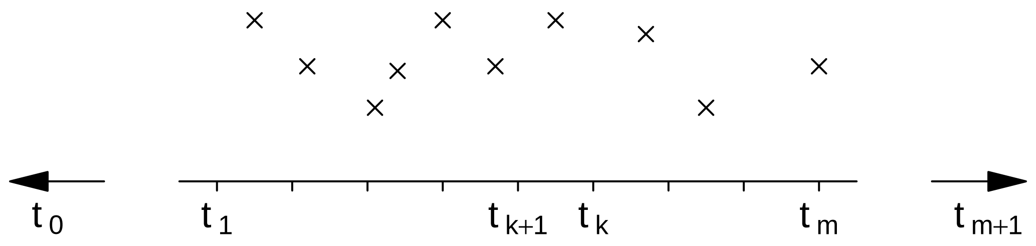


Figure 2.1: Diagram of knots and data points.

Note that we use the symbol t, rather than x, so that we do not confuse knots and observation locations.

A spline of order $p \geq 1$ is a piecewise-polynomial of order p which is $(p-1)$ times differentiable at the knots. Thus there are coefficients $\{a_{k\ell},\ k = 0, \ldots, m,\ \ell = 0, \ldots, p\}$ such that

$$f(t) = \sum_{\ell=0}^{p} a_{k\ell}\, t^{\ell}, \qquad \text{for } t_k \leq t < t_{k+1}, \tag{2.1}$$

where we take $t_0 = -\infty$ and $t_{m+1} = +\infty$.

If we are using cubic polynomials, $(p = 3)$, then f is given by the following equations:

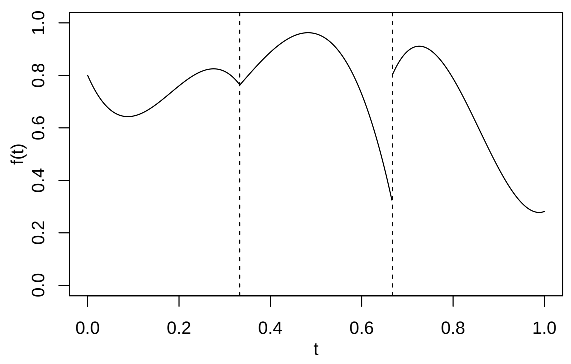
$$f(t) = a_{00} + a_{01}t + a_{02}t^2 + a_{03}t^3, \quad t_0 \leq t < t_1$$

to the left of the first knot,

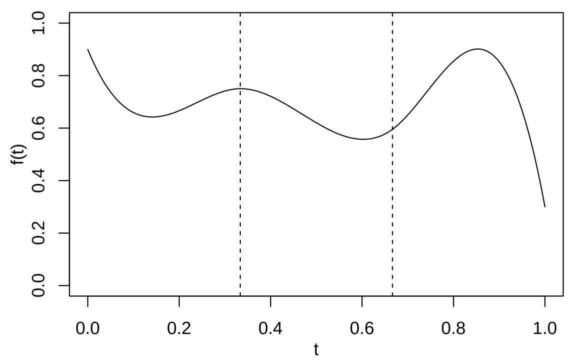
$$f(t) = a_{10} + a_{11}t + a_{12}t^2 + a_{13}t^3, \quad t_1 \leq t < t_2$$

between the first and second knots, and so on until

$$f(t) = a_{m0} + a_{m1}t + a_{m2}t^2 + a_{m3}t^3, \quad t_m \leq t < t_{m+1}$$



(a) No smoothness constraints



(b) With smoothness constraints

Figure 2.2: Piecewise-cubic functions in three intervals with knot positions indicated with vertical lines.

to the right of the final knot. This is illustrated in Figure 2.2a with $m = 2$.

Because of the use of polynomials, f is smooth *between* each successive pair of knots. At the knots, however, f might not be continuous and it might not be differentiable – in such cases we would say that the function is not smooth.

To ensure that f is also smooth at each of the knots, we impose smoothness constraints which control continuity of the function and its derivatives at the knots.

Let $f^{(\ell)}$ be the ℓ-th order derivative, with $f^{(0)} = f$ being the function itself, $f^{(1)} = f'$ is the first derivative and $f^{(2)} = f''$ the second derivative. Further, let $f^{(\ell)}(t - \epsilon)$ and $f^{(\ell)}(t + \epsilon)$, for $\epsilon \geq 0$, denote evaluation of the function or its derivative at points just below and just above t – we will be interested in their relative values as $\epsilon \to 0$.

To impose smoothness, we require that

$$\lim_{\epsilon \to 0} f^{(\ell)}(t_k - \epsilon) = \lim_{\epsilon \to 0} f^{(\ell)}(t_k + \epsilon), \tag{2.2}$$

for all $k = 1, \ldots, m$ and for $\ell = 0, \ldots, (p - 1)$.

In other words we say that f is smooth if the limits, from below and from above, of the function and its $(p - 1)$ derivatives exist and are equal.

The meaning of these smoothness constraints is illustrated in Figure 2.2. In Figure 2.2a, a piecewise cubic function with two knots has been plotted. The first derivative, f', is discontinuous at the first knot and the function itself, f, is discontinuous at the second knot. Figure 2.2b shows a similar shaped cubic spline with two knots. This time, the function f and its first two derivatives are continuous at both knots.

The smoothness conditions in Equation 2.2 induce constraints on the coefficients $\{a_{k\ell}\}$. A polynomial of order p has $p + 1$ coefficients, and there are $m + 1$ intervals when we have m

knots. This leads to $(p+1)\times(m+1)$ coefficients but there are p constraints at each of the m knots. Thus the total *degrees of freedom* of the system is

$$\mathrm{df}_{\mathrm{spline}} = (p+1)(m+1) - pm = m + p + 1. \tag{2.3}$$

These degrees of freedom provide the necessary flexibility in the spline.

Note that f is infinitely differentiable everywhere, except at the knots where it is $p-1$ times differentiable. In particular, for $p=1$, f is a linear spline comprising linear pieces constrained to be continuous at the knots, although the slope of f is discontinuous at the knots. Also, for $p=3$, f is a cubic spline comprising cubic polynomial pieces continuous at the knots; where the first and second derivatives of f are also continuous, but the third derivative is discontinuous at the knots.

2.2 Exercises

2.1 Why is it not sensible to define a smooth function made-up of constant components? Similarly, why is not sensible to create a differentiable function from linear splines?

2.2 In the situation illustrated in Figure 2.2b, where $p=3$ and $m=2$, clearly identify the $(p+1)\times(m+1)=12$ model parameters and the $pm=6$ smoothness constraints in terms of the cubic polynomials and their derivatives.

2.3 Further consider the situation illustrated in Figure 2.2b. Suppose now that we require the splines to pass through specified coordinates $(t_1, f(t_1))$ and $(t_2, f(t_2))$. What is the degrees of freedom for this model? How many such cubic splines would satisfy these constraints? Discuss potential additional constraints which would lead to a unique fitted model. Do you think having a unique solution is a positive or negative property?

2.4 For a general problem, what would be the effect of requiring additional constraints of the form of Equation 2.2 but with $\ell = p$? Would this lead to an acceptable fitted cubic spline model? Justify your answer.

Note

Exercise 2.2 Solutions can be found here.

3 Interpolating Splines

3.1 Overview

Chapter 1 considered general limitations of parametric models, and polynomial regression in particular (see Figure 1.1), which motivated the use of the more flexible spline models (see Figure 1.3) – though at that stage no mathematical details were presented. In Chapter 2, basic spline definitions were given, including the notation of smoothness constraints, and these ideas were further explored in the Exercises in Section 3.6. This chapter will now give mathematical details of the interpolating spline problem and consider application to data. A feature of all these interpolation methods is that they fit the data exactly and that the fitted functions are smooth. Figure 3.1, is *cubic spline interpolation*, which fits a cubic polynomial between successive data points such that the function, gradient and the curvature are all continuous at each data point. The solid line shows the fitted values within the range of the data, whereas the dashed line shows the fitted values outside the range of the data – *extrapolation*.

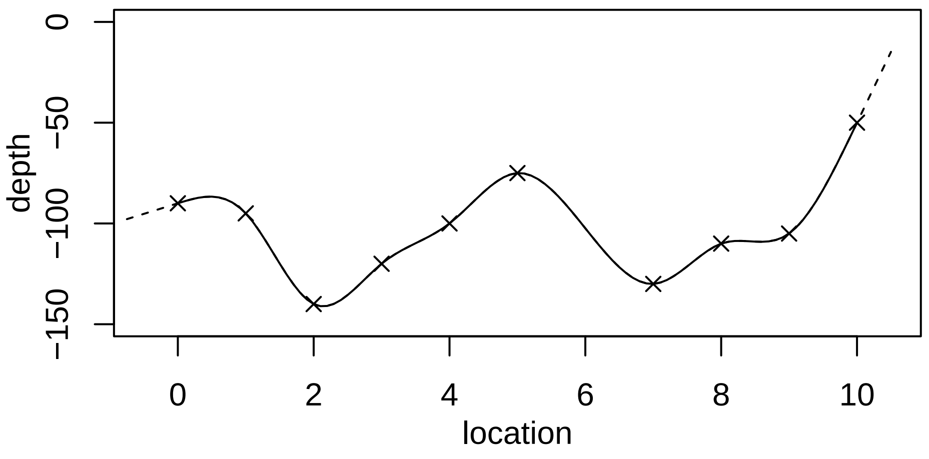


Figure 3.1: A cubic interpolating spline fitted to the coal-seam data, with the dashed line showing extrapolation.

3.2 Natural splines

Suppose we have n observations $\{y_1, \dots, y_m\}$ at locations $\{t_1, \dots, x_m\}$. We can construct a cubic spline (that is with $p = 3$) to pass through (interpolate) all the points (t_i, y_i), $i = 1, \dots, m$. In fact, for any given set of points, there is an infinite number of cubic splines which interpolate them, see Figure 3.2 for examples. Exactly one of these splines has the property that, in the leftmost and rightmost intervals, it is a straight line. Such a spline is called a *natural* cubic spline.

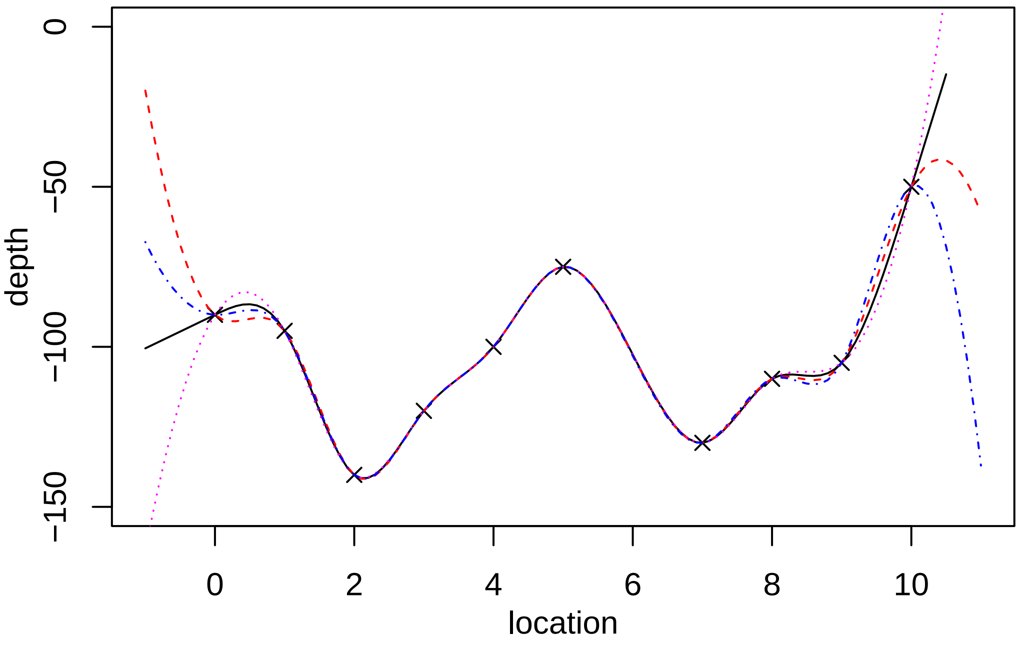


Figure 3.2: Cubic interpolating splines fitted to the coal-seam data, with the dashed lines showing extrapolation – the natural spline is shown in solid black.

3.3 Properties of natural splines

Natural splines are a special case of polynomial splines of *odd* order p. Thus we have natural linear splines ($p = 1$), natural cubic splines ($p = 3$), etc. A spline is said to be *natural* if, beyond the boundary knots t_1 and t_m, its $(p + 1)/2$ higher-order derivatives are zero:

$$f^{(j)}(t) = 0, \tag{3.1}$$

for $j = (p + 1)/2, \dots, p$ and either $t \leq t_1$ or $t \geq t_m$.

Thus a natural spline of order p has the following $p + 1$ constraints, in addition to those of Equation 2.2 :

$$\lim_{\epsilon \to 0} f^{(\ell)}(t_1 - \epsilon) = \lim_{\epsilon \to 0} f^{(\ell)}(t_m + \epsilon) = 0, \tag{3.2}$$

for $\ell = (p+1)/2, \dots, p$.

In particular,

- a natural *linear* spline has $p+1=2$ additional constraints:

$$\lim_{\epsilon\to 0} f^{(1)}(t_1-\epsilon) = \lim_{\epsilon\to 0} f^{(1)}(t_m+\epsilon) = 0, \tag{3.3}$$

  implying that $f(t)$ is constant in the outer intervals of a natural linear spline,
- a natural *cubic* spline has $p+1=4$ additional constraints:

$$\begin{aligned} \lim_{\epsilon\to 0} f^{(2)}(t_1-\epsilon) &= \lim_{\epsilon\to 0} f^{(2)}(t_m+\epsilon) = 0, \\ \lim_{\epsilon\to 0} f^{(3)}(t_1-\epsilon) &= \lim_{\epsilon\to 0} f^{(3)}(t_m+\epsilon) = 0, \end{aligned} \tag{3.4}$$

  implying that $f(t)$ is linear in the outer intervals of a natural cubic spline.

The total degrees of freedom of a natural spline is, starting from Equation 2.3, but taking into account the additional $p+1$ additional constraints is

$$\mathrm{df}_{\text{nat.spline}} = m+p+1-(p+1) = m. \tag{3.5}$$

That is the degrees of freedom for natural splines equals m whatever the value of p.

Proposition 3.1. *Linear and cubic natural splines have the following representations:*

- *Linear natural splines:*

$$f(t) = a_0 + \sum_{i=1}^{m} b_i\,|t-t_i|\,; \quad \sum_{i=1}^{m} b_i = 0 \tag{3.6}$$

- *Cubic natural splines:*

$$f(t) = a_0 + a_1 t + \sum_{i=1}^{m} b_i\,|t-t_i|^3\,; \quad \sum_{i=1}^{m} b_i = \sum_{i=1}^{m} b_i t_i = 0. \tag{3.7}$$

Proof: Not covered here (but may be included later in the module if time allows).

3.4 Roughness penalties

An aim of spline models is to describe an unknown function using piecewise-polynomials which are smooth. In the previous section, smoothness was imposed by explicitly constraining specified high-order derivatives. An alternative approach is to measure and control the degree of smoothness of the splines. In practice the *roughness* of the spline is usually measured and one definition of roughness is:

$$J_\nu(f) = \int_{-\infty}^{\infty} \left[f^{(\nu)}(t)\right]^2 \mathrm{d}t \tag{3.8}$$

where $\nu \geq 1$ is an integer and $f^{(\nu)}$ denotes the νth derivative of f. Thus $f^{(1)}(t)$ denotes the first derivative and $f^{(2)}(t)$ denotes the second derivative of f.

Intuitively, roughness measures the "wiggliness" of a function.

Aim might be to find the smoothest function which interpolates the data points. Hence, an alternative approach to that in previous sections is to find the function f which minimizes Equation 3.8 and satisfies $f(t_i) = y_i$ for $i = 1, \dots, m$. We refer to the solutions of this problem as the *optimal interpolating function.*

It turns out that there is a very close link between $J_\nu(\cdot)$ and pth-order natural splines, where $p = 2\nu - 1$ (so p is odd). Important special cases are: $\nu = 1$ and $p = 1$, and $\nu = 2$ and $p = 3$. This relationship is defined in the following proposition.

Proposition 3.2. *The optimal interpolating function is a pth-order natural spline, where* $p = 2\nu - 1$*. That is, the natural spline* f *is the unique minimizer of* $J_\nu(f)$*.*

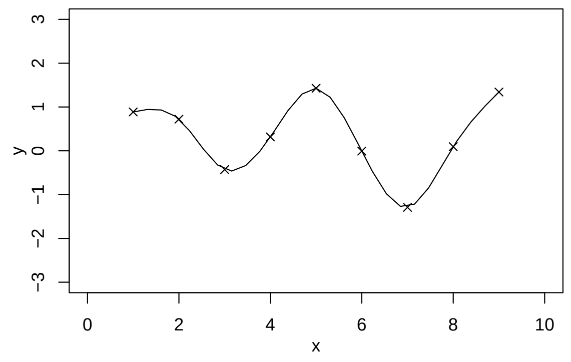
Proof: Not covered here (but may be included later in the module if time allows).

Comments

- Linear and cubic interpolating splines are also of interest in numerical analysis, for example to interpolate tables of numbers.
- The linear interpolating spline is simply the piecewise-linear path connecting the data points.
- Of course, in the linear spline case, knot points are clearly visible as kinks in the interpolating function.
- But, in the cubic spline case, knots points are invisible to the naked eye. Hence, in general, there is little motivation to use higher-order splines.
- Numerical considerations: the interpolating spline solutions involve matrix inversion. The inversion of an $n \times n$ matrix involves $O(n^3)$ operations – hence it is time consuming if n is large (for example, $n = 1000$ or 10000). Fortunately there are tricks to reduce the computation to $O(n)$.

3.5 Fitting interpolating splines in R

There are two main function within **R** for fitting interpolating splines to data, `spline` which outputs fitted values for specified points or `splinefun` which returns an **R** *function* which can be used directly by other commands, such as `curve`. The following illustrates the two approaches.


(a) Using the spline command

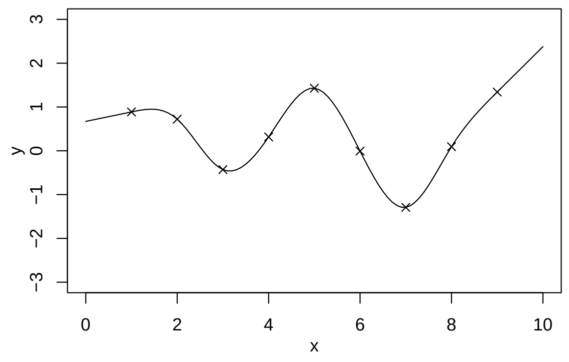

(b) Using the splinefun command

Figure 3.3: R code for cubic interpolating splines.

The following, illustrates the different ways to draw the spline and to calculated fitted values.

```
$x
[1] 2.5 7.5

$y
[1]  0.08785792 -0.78655273

[1]  0.08785792 -0.78655273
```

3.6 Exercises

3.1 For the situation shown in Figure 2.2, but taking $p = 1$, write-down the linear functions for the three intervals and clearly identify all the 6 model parameters. Next, write down the constraints required to make the functions pass through the $m = 2$ data points, and the two constraints which impose continuity of function. What additional constraints are needed to fix the first derivative at zero for the outer two intervals?

3.2 Continuing the problem described in Exercise 3.1, write the constraints as a system of 6 linear equations in the 6 unknown model parameters. How might you solve this system to give the parameter values which solve the interpolation problem?

3.3 Continuing the linear system described in Exercise 3.2, create a synthetic problem by choosing two data response values. Then solve the system in **R**, or otherwise, and plot the fitted spline interpolating function.

3.4 Again, considering the situation shown in Figure 2.2, but taking $p = 1$. Using the alternative representation in Equation 3.6, write down two constraints involving the data points and the additional constraint on the b_i parameters. Write this linear system of 3 equations in three unknowns in matrix form.

3.5 Continuing the linear system described in Exercise 3.4, using the same points created in Exercise 3.4, calculate the parameter values in this new parameterization. Check that your two fitted interpolating spline give the same answers. Which approach do you prefer? Justify you answer.

3.6 Create you own version of the R code used to produce Figure 3.3 and experiment with the two alternative spline fitting commands. Remove the `set.seed(15342)` command so that you produce different data each time and comment on the similarities and differences when using different data sets.

3.7 Let

$$f(t) = 3 + 2t + 4|t|^3 + |t-1|^3.$$

Write f as a cubic polynomial in each of the intervals $(-\infty, 0)$, $(0, 1)$ and $(1, \infty)$. Verify that f and its first two derivatives are continuous at the knots.

Is f a spline? Is f a natural spline?

3.8 Let

$$f(t) = 3 + |t| - |t-2|.$$

Show by direct calculation that

$$\int_{-\infty}^{\infty} \{f'(t)\}^2 \, dt = 8.$$

Show that this integral can also be written in the form $-2\mathbf{b}^T K \mathbf{b}$, where you should define $\mathbf{b}$ and K.

4 Smoothing Splines

4.1 Overview

In Section 1.2 we described a general statistical model with a response variable y and an explanatory variable x. We observe y_i at each location t_i, for $i = 1, \dots, n$. We imagined that the y's are noisy versions of a smooth function of t, say $f(\cdot)$ where the errors follow a normal distribution with constant variance. That is

$$y_i = f(t_i) + \epsilon_i, \quad \epsilon_i \sim \mathrm{N}(0, \sigma^2),$$

for $i = 1, \dots, n$, where f is smooth, the ϵ_i are i.i.d., and f and σ^2 are unknown.
The log-likelihood for this situation is:

$$l(f; \mathbf{y}) = -\frac{1}{2\sigma^2} \sum_{i=1}^{n} \left(y_i - f(t_i)\right)^2 - n \log \sigma \tag{4.1}$$

and we wish to estimate f for a given data set $\mathbf{y} = \{y_1, \dots, y_n\}$. With no constraints on f, the log-likelihood would be maximized by setting $f(t_i) = y_i$ for all i, and we would estimate the noise variance as $\hat{\sigma}^2 = 0$. This takes no account of randomness in the data and f would in general need to be quite wiggly to achieve this fit.

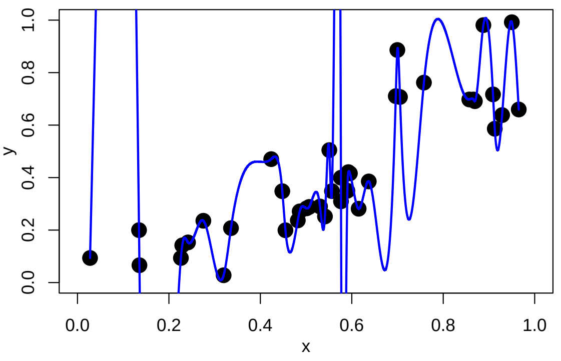
Figure 4.1a shows such an interpolation of noisy data. This would be of little use for explanation, interpolation or prediction.

We do not expect, or even want, the fitted function f to pass exactly through the data points $\{(t_i, y_i)\}$, but merely to lie close to them. We would rather trade-off goodness-of-fit against smoothness of f. Figure 4.1b shows a smoothing spline fit to the same data. This is much better as there is a clear explanation of the relationship, it could be used reasonably well for interpolation and prediction.

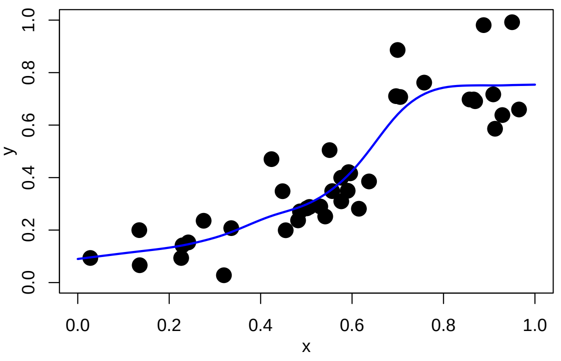
4.2 The penalized least-squares criterion

Noting that maximizing Equation 4.1 is equivalent to minimizing the residual sum of squares: $\sum_{i=1}^{n} \left(y_i - f(t_i)\right)^2$, we can achieve this trade-off by minimizing a *penalized* sum of squared residuals:

$$R_\nu(f, \lambda) = \sum_{i=1}^{n} \left(y_i - f(t_i)\right)^2 + \lambda J_\nu(f), \tag{4.2}$$



(a) Interpolating spline



(b) Smoothing spline

Figure 4.1: Comparison of interpolating and smoothing methods applied to a noisy dat set.

where $J_\nu(f)$, as first defined in Equation 3.8, penalizes the roughness of f. The *smoothing parameter* $\lambda \geq 0$ controls the severity of this penalty. For now we will assume λ, which absorbs the σ^2 in Equation 4.1, is known.

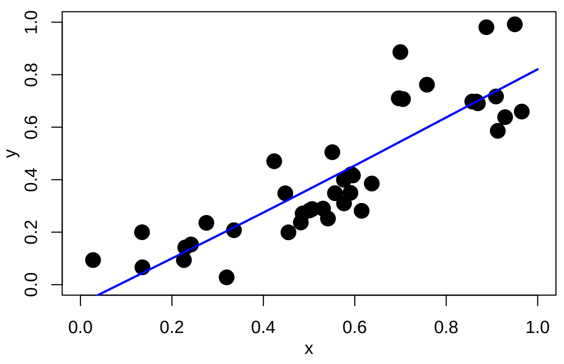
Figure 4.2 shows example smoothing spline fits using a range of smoothing parameters, λ. In Figure 4.2a the fit is essentially a straight line, perhaps Figure 4.2b and Figure 4.2c show acceptable fits. Figure 4.2d, with a very small λ value is close to an interpolating spline fit and is clearly unacceptable.

As the smoothing parameter λ increases, the optimal f becomes smoother. In particular, it can be shown that as $\lambda \to \infty$, the vector of coefficients $\mathbf{b} \to \mathbf{0}$, and $\mathbf{a}$ tends to the OLS estimate of the regression parameters. Thus, the smoothing spline converges to the sample mean $f(t) = \bar{y}$ when $\nu = 1$, and to the ordinary least squares fitted line, $f(t) = \hat{\alpha} + \hat{\beta} t$, when $\nu = 2$. In the other direction, as $\lambda \to 0$ the smoothing solution converges to the interpolating spline.

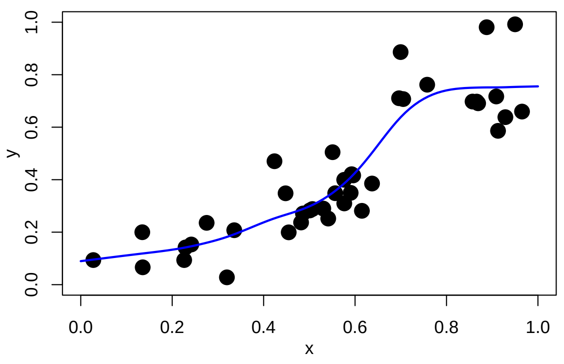
4.3 Relation to interpolating splines

We show in the following proposition that the function f which minimizes Equation 4.2 is the interpolating spline of its fitted values.

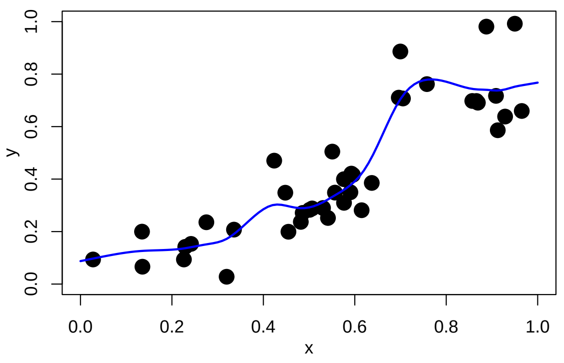
Proposition 4.1. *Suppose $\hat{f}$ minimizes $R_\nu(f, \lambda)$ and let $\hat{y}_i = \hat{f}(t_i), i = 1, \ldots, n$ denote the corresponding fitted values. Then, $\hat{f}$ solves the interpolation problem for the* artificial *data set $(t_i, \hat{y}_i), i = 1, \ldots, n$. That is, $\hat{f}$ minimizes $J_\nu(f)$ over functions f satisfying $\hat{f}(t_i) = \hat{y}_i$, $i = 1, \ldots, n$. Consequently, $\hat{f}$ is a p^{th}-order natural spline, where $p = 2\nu - 1$. This means that, when solving the smoothing problem, we only need consider spline functions given by the representations in Proposition 3.1.*



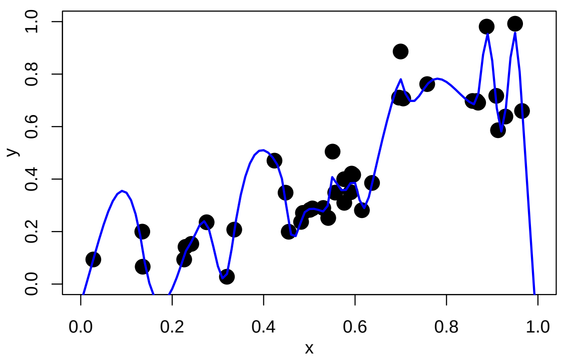
(a) Very strong smoothing



(b) Moderate smoothing

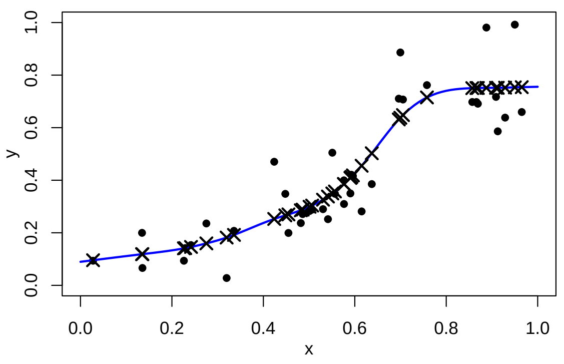


(c) Weak smoothing

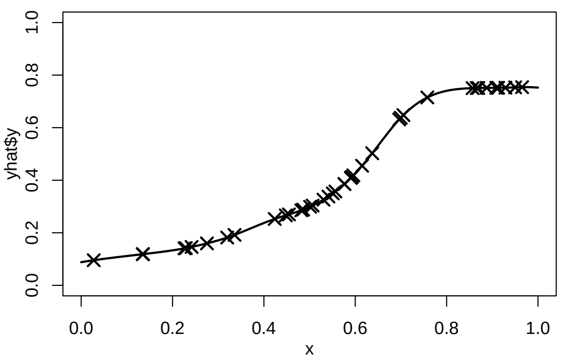


(d) Very weak smoothing

Figure 4.2: Comparison of interpolating and smoothing methods applied to a noisy dat set.



(a) Smoothing spline of data shown as dots and with fitted values shown as crosses



(b) Interpolating spline of fitted values from smoothing problem

Figure 4.3: Illustration of proposition showing that solution of the smothing problem is a natural interpolating spline.

Proof: Suppose the assertion is not true. In this case, we must be able to find a function $\hat{f}^\star$, say, which also interpolates the artificial data $(t_i, \hat{y}_i), i = 1, \dots, n$, but which has a smaller roughness penalty. That is

$$J_\nu(\hat{f}^\star) < J_\nu(\hat{f}) \text{ with } \hat{f}^\star(t_i) = \hat{y}_i,\ i = 1, \dots, n.$$

Note that the fitted values from the function $\hat{f}^\star$ are also equal to $\hat{y}_i, i = 1, \dots, n$ as it interpolates the same artificial data as $\hat{f}$.

Now, from Equation 4.2,

$$\begin{aligned} R_\nu(\hat{f}^\star, \lambda) &= \sum_i (y_i - \hat{y}_i)^2 + \lambda J_\nu(\hat{f}^\star) \\ &< \sum_i (y_i - \hat{y}_i)^2 + \lambda J_\nu(\hat{f}) = R_\nu(\hat{f}, \lambda). \end{aligned}$$

Hence $R_\nu(\hat{f}^\star, \lambda) < R_\nu(\hat{f}, \lambda)$. But, by construction $\hat{f}$ minimizes $R_\nu(f, \lambda)$, which is a contradiction. Hence, it must not be possible to find a function $\hat{f}^\star$ which also interpolates the artificial data but which has a smaller roughness penalty.

We have shown that $\hat{f}$ is the optimal interpolant of the fitted values $\hat{y}_i,\ i = 1, \dots, n$, so it follows from Proposition 4.1 that $\hat{f}$ is a natural spline of order $p = 2\nu - 1$.

4.4 The smoothing problem in matrix notation

We have just proved that the function $\hat{f}$ that minimises $R_\nu(f, \lambda)$ must be a natural spline (linear if $\nu = 1$, cubic if $\nu = 2$) with knots at $\{t_i,\ i = 1, \dots, n\}$. That is, as in Proposition 3.1, we can write:

$$\hat{f}(t) = \sum_{i=1}^{n} b_i \,|t - t_i|^p + \begin{cases} a_0, & \nu = 1 \\ a_o + a_1 t, & \nu = 2, \end{cases} \tag{4.3}$$

where $p = 2\nu - 1$ and constraints

$$\sum_i b_i = 0 \ \text{ for } \ \nu = 1 \qquad \sum_i b_i = \sum_i b_i\, t_i = 0 \ \text{ for } \ \nu = 2. \tag{4.4}$$

However, we have not yet figured out how to calculate the parameter values $\hat{a}_0, \dots, \hat{a}_{\nu-1},\ \hat{b}_1, \dots, \hat{b}_n$ in Equation 4.3 which optimally fit the data $y_1, \dots, y_n$. For this, it is convenient to re-express the penalized sum of squared residuals Equation 4.2 in matrix notation.

Proposition 4.2. *The roughness of a natural linear spline ($\nu = 1$, i.e. $p = 1$) is*

$$J_1(f) = -2 \sum_{i=1}^{n} \sum_{k=1}^{n} b_i b_k \,|t_i - t_k| = c_1\, \mathbf{b}^T K_1\, \mathbf{b}, \tag{4.5}$$

and the roughness of a natural cubic spline ($\nu = 2$, i.e. $p = 3$) is

$$J_2(f) = 12 \sum_{i=1}^{n} \sum_{k=1}^{n} b_i b_k \left|t_i - t_k\right|^3 = c_2\, \mathbf{b}^T K_2\, \mathbf{b}, \tag{4.6}$$

where the constants are given by

$$c_1 = -2, \quad c_2 = 12. \tag{4.7}$$

Here $\mathbf{b} = (b_1, \dots, b_n)^T$ is the vector of spline coefficients and K_ν is the $n \times n$ matrix whose (i,k)th element is $\left|t_i - t_k\right|^p$, where $p = 2\nu - 1$ that is

$$K_\nu = \begin{bmatrix} |t_1 - t_1|^p & |t_1 - t_2|^p & \dots & |t_1 - t_n|^p \\ |t_2 - t_1|^p & |t_2 - t_2|^p & \dots & |t_2 - t_n|^p \\ \vdots & \vdots & \ddots & \vdots \\ |t_n - t_1|^p & |t_n - t_2|^p & \dots & |t_n - t_n|^p. \end{bmatrix}$$

Proof: To be covered later if there is sufficient time.

From Proposition 4.2 the roughness penalty satisfies:

$$J_\nu(\hat{f}) = c_\nu\, \mathbf{b}^T K_\nu \mathbf{b} \tag{4.8}$$

where $c_1 = -2$; $c_2 = 12$; $\mathbf{b} = (b_1, \dots, b_n)^T$; and K_ν is the $n \times n$ matrix whose (i,k)th element is $\left|t_i - t_k\right|^p$.

From Equation 4.3, the value of $\hat{f}$ at knot t_k is

$$\hat{f}(t_k) = \sum_{i=1}^{n} b_i \left|t_k - t_i\right|^p + \begin{cases} a_0, & \nu = 1 \\ a_0 + a_1 t_k, & \nu = 2 \end{cases}. \tag{4.9}$$

In matrix form, this is

$$\hat{\mathbf{f}} = \begin{bmatrix} \hat{f}(t_1) \\ \vdots \\ \hat{f}(t_n) \end{bmatrix} = K_\nu \mathbf{b} + L_\nu \mathbf{a}_\nu \tag{4.10}$$

where

$$L_1 = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \quad \mathbf{a}_1 = a_0, \qquad \text{and} \qquad L_2 = \begin{bmatrix} 1 & t_1 \\ \vdots & \vdots \\ 1 & t_n \end{bmatrix}, \quad \mathbf{a}_2 = \begin{bmatrix} a_0 \\ a_1 \end{bmatrix}.$$

Thus, from Equation 4.4 – Equation 4.10, the penalized least-squares criterion Equation 4.2 reduces to a quadratic function of the parameters **a** and **b**:

$$R_\nu(f, \lambda) = (\mathbf{y} - K_\nu\, \mathbf{b} - L_\nu\, \mathbf{a})^T (\mathbf{y} - K_\nu\, \mathbf{b} - L_\nu\, \mathbf{a}) + \lambda\, c_\nu\, \mathbf{b}^T K_\nu\, \mathbf{b} \tag{4.11}$$

subject to

$$L_\nu^T\, \mathbf{b} = 0, \tag{4.12}$$

where $\mathbf{y} = (y_1, \dots, y_n)^T$.

To find the explicit values for **b** and **a**, we must minimize the quadratic function Equation 4.11 subject to the linear constraints Equation 4.12.

Proposition 4.3. *The solution to the smoothing spline problem is given by*

$$\begin{bmatrix} \hat{\mathbf{a}} \\ \hat{\mathbf{b}} \end{bmatrix} = \begin{bmatrix} 0 & L_\nu \\ L_\nu^T & K_\nu + \lambda^* I_n \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{y} \end{bmatrix} \tag{4.13}$$

and $\lambda^* = c_\nu \lambda$.

Proof: Omitted.

4.5 Smoothing splines in R

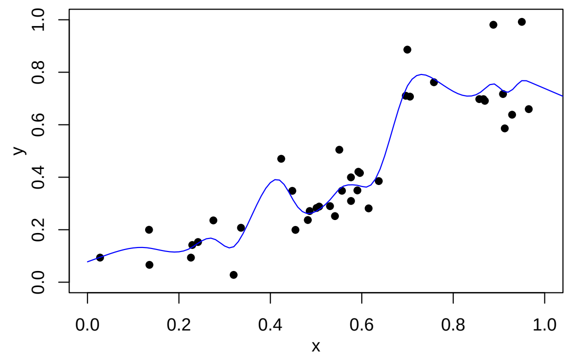
Fitting a smoothing spline to data involved estimating values of **a** and **b** to minimize the penalized roughness measure in Equation 4.11 subject to the constraints in Equation 4.12 which yields the explicit equation in Equation 4.13.

Of course, it is possible to code the solution of this matrix system, but it is usual instead to use in-built commands in software such as **R**.

In **R** the basic command is:

where `x` and `y` are vectors of data coordinates, and `lambda` specifies the value of the smoothing parameter to use.

For example, consider the synthetic data set which contains a break-point at $x = 2/3$ first met in Figure 1.2. Fitted spline curves with $\lambda = 0.0001$ and $\lambda = 1$, shown in Figure 4.4, are based on the following commands:


(a) Weak smoothing

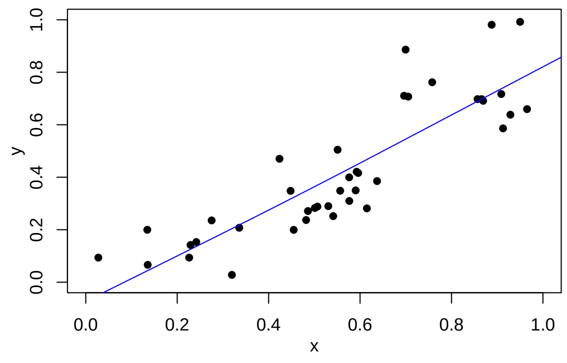

(b) Strong smoothing

Figure 4.4: Simulated data superimposed with fitted smoothing splines.

If no value of the smoothing parameter is specified, the the optimal value is calculated using generalized cross-validation as discussed in the next chapter.

4.6 Exercies

4.1 Consider the *Old Faithful* data set on geyser eruptions available in R. Use the following commands to learn more about and visualize the data:

```
data(faithful)
help(faithful)
plot(faithful)
```

Fit a cubic smoothing spline to these data, using a range of values for the smoothing parameter λ. By eye, suggest a suitable value for λ.

5 Choosing the Smoothing Parameter

5.1 Overview

Suppose we are given data $D = \{(t_i, y_i),\ i = 1, \dots, n\}$ and that our model is:

$$y_i = f(t_i) + \epsilon_i, \qquad \epsilon_i \sim \mathrm{N}(0, \sigma^2) \tag{5.1}$$

where the ϵ_i are i.i.d. $\sim \mathrm{N}(0, \sigma^2)$ and $f(t)$ is assumed to be smooth. Given knot positions $\{t_i,\ i = 1, \dots, n\}$, we can estimate $f(t)$ with a smoothing spline $\hat{f}_\lambda(t)$.

How then should we choose the value of the smoothing parameter λ? By setting $\lambda \to 0$, we obtain exactly the interpolating spline $\hat{f}_0(t)$ and a perfect fit to the data. However, this tends to *overfit* the data: applying it to to a new sample of data where model Equation 5.1 still applies would produce a poor fit. Conversely, by setting $\lambda \to \infty$, we get:

$$f_\infty(t) = \begin{cases} \hat{a}_0, & \nu = 1,\ p = 1 \\ \hat{a}_0 + \hat{a}_1 t, & \nu = 2,\ p = 3. \end{cases}$$

Here, $\hat{a}_0 = \bar{y}$, for the $\nu = 1, p = 1$ case, and $\{\hat{a}_0,\ \hat{a}_1\}$, for the $v = 2, p = 3$ case, are the OLS linear regression parameters.
If the true $f(t)$ was constant or linear, this solution would be reasonable, but often we are interested in less regular functions.

5.2 Training/test approach

One way to approach estimation of λ is to partition the set of indices $I = \{1, \dots, n\}$ into two subsets I_1 and I_2, where $I_1 \cup I_2 = I$ and $I_1 \cap I_2 = \phi$. Thus we obtain two datasets:

- Training dataset: $D_1 = \{(t_i, y_i),\ i \in I_1\}$,
- Test dataset: $D_2 = \{(t_i, y_i),\ i \in I_2\}$.

We fit a smoothing spline $\hat{f}_{\lambda, I_1}(t)$ to the training dataset, and judge the quality of the fit using the test dataset:

$$Q_{I_1:I_2}(\lambda) = \sum_{i \in I_2} \left(y_i - \hat{f}_{\lambda, I_1}(t_i) \right)^2 . \tag{5.2}$$

We choose λ to minimize $Q_{I_1:I_2}(\lambda)$. Many algorithms exist for such minimization, for example through evaluation on a fine grid of λ values, although many more computationally efficient algorithms exist.

5.3 Cross-validation or *leave-one-out*

This is an extreme form of the above principle. The test dataset D_2 comprises a single observation, (t_j, y_j), for a given value of j. The training set D_1 is then $D_{-j} = \{(t_i, y_i),\ i \in I_{-j}\}$, where I_{-j} denotes the full set I excluding j. Then in a slightly amended notation we can write

$$Q_{-j:j}(\lambda) = \left(y_j - \hat{f}_{\lambda,-j}(t_j)\right)^2$$

to assess the quality of fit. Of course, j is arbitrary, so we repeat this process for each $j \in \{1, \dots, n\}$ then average the assessments to form the *ordinary cross-validation criterion*:

$$Q_{OCV}(\lambda) = \frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{f}_{\lambda,-j}(t_j)\right)^2. \tag{5.3}$$

We then choose the value $\hat{\lambda}$ which minimizes $Q_{OCV}(\lambda)$. Hopefully, a plot of $Q_{OCV}(\lambda)$ will appear as in Figure 5.1, but there is no theoretical guarantee that this curve will have a unique turning point, making it difficult to locate the minimum.

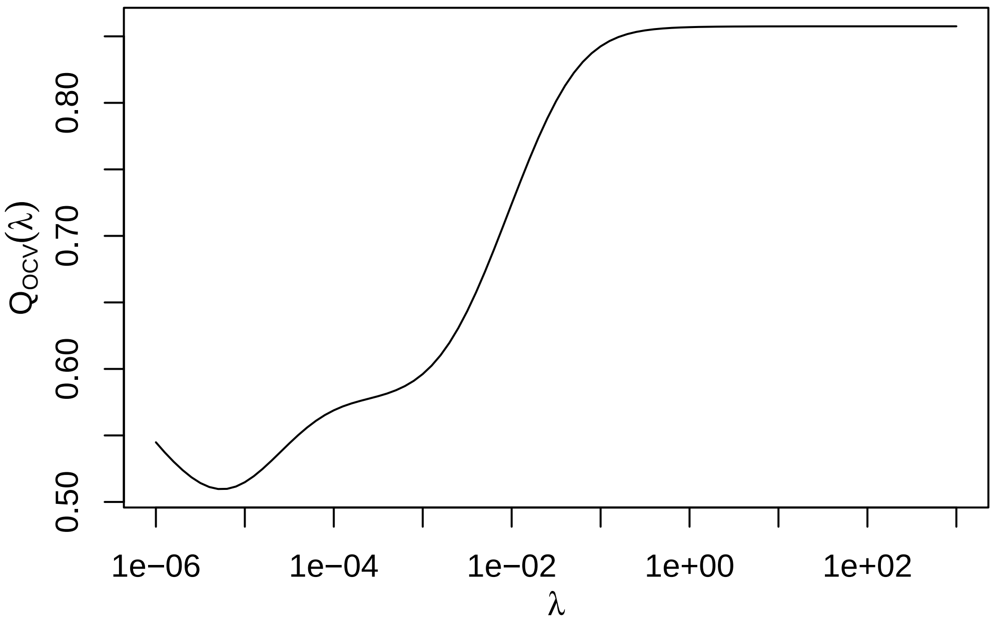


Figure 5.1: Ordinary cross validation plot

At first sight, evaluation of $Q_{OCV}(\lambda)$ for a given λ appears computationally intensive: we must compute n different smoothing solutions, each corresponding to one of the *left-out* data points. Fortunately, there is a computational trick which enables us to compute $Q_{OCV}(\lambda)$ directly from the smoothing spline solution constructed from the whole dataset

5.4 The smoothing matrix

Here we show, for a given value of the smoothing parameter λ and the index $\nu \geq 1$, that the fitted value $\hat{f}_\lambda(t_k)$ at each knot t_k may be written as a linear combination of the observations, $y_1, \cdots, y_n$.

Recall from Proposition 4.3 that the smoothing spline $\hat{f}_\lambda(t)$, which minimizes the penalized sum of squares criterion Equation 4.2 for a given value of the smoothing parameter λ, has coefficients $\hat{\mathbf{a}}, \hat{\mathbf{b}}$ where

$$\begin{bmatrix} \hat{\mathbf{a}} \\ \hat{\mathbf{b}} \end{bmatrix} = M_\lambda^{-1} \begin{bmatrix} \mathbf{0} \\ \mathbf{y} \end{bmatrix} \quad \text{where} \quad M_\lambda = \begin{bmatrix} 0 & L_\nu^T \\ L_\nu & K_\nu + \lambda^* I_n \end{bmatrix}$$

The fitted values of the smoothing spline at the knots can be represented in matrix form as

$$\hat{\mathbf{f}} = \begin{bmatrix} \hat{f}(t_1) \\ \vdots \\ \hat{f}(t_n) \end{bmatrix} = K\,\hat{\mathbf{b}} + L\,\hat{\mathbf{a}} = \begin{bmatrix} K & L \end{bmatrix} \begin{bmatrix} M_\lambda^{12} \\ M_\lambda^{22} \end{bmatrix} \mathbf{y},$$

where M_λ^{-1} has been partitioned in the form

$$M_\lambda^{-1} = \begin{bmatrix} M_\lambda^{11} & M_\lambda^{12} \\ M_\lambda^{21} & M_\lambda^{22} \end{bmatrix},$$

where M_λ^{12} is $\nu \times \nu$ and M_λ^{22} is $n \times n$.

It can be shown that the matrix

$$S_\lambda = \begin{bmatrix} K & L \end{bmatrix} \begin{bmatrix} M_\lambda^{12} \\ M_\lambda^{22} \end{bmatrix}$$

is a symmetric positive definite matrix for $\lambda > 0$ called the *smoothing* matrix. Then, S_λ connects the data $\mathbf{y}$ to the fitted values through

$$\hat{\mathbf{f}} = S_\lambda\,\mathbf{y}. \tag{5.4}$$

That is, the fitted values are simple linear function of the data.

5.5 Effective degrees of freedom

How many degrees of freedom are there in the smoothing spline? There are altogether $n+\nu$ parameters, ν in $\mathbf{a}$ and n in $\mathbf{b}$, but these are not completely free.

We showed earlier that as the smoothing parameter $\lambda \to \infty$, the smoothing spline $\hat{f}(t)$ becomes the least-squares regression solution for model formula $y \sim 1$ when $\nu = 1$, or model formula $y \sim 1 + t$ when $\nu = 2$. Thus, when $\lambda = \infty$, the degrees of freedom in the spline is ν. We also showed that when $\lambda = 0$, the smoothing spline $\hat{f}(t)$ becomes the interpolating spline, for which the degrees of freedom is the number of observations, n.

Thus, intuitively, for values of λ between the extremes of 0 and ∞, the spline degrees of freedom should lie somewhere between ν and n; the greater the smoothing, the fewer the *effective* degrees of freedom. How can we capture this notion precisely?

A clue comes from Ordinary Least Squares (OLS) regression, in which the fitted values are given by

$$\hat{\mathbf{y}} = X(X^TX)^{-1}X^T\mathbf{y} = H\mathbf{y},$$

where X is the $n \times p$ design matrix, where p is the number of model parameters. Here,

$$H = X(X^TX)^{-1}X^T$$

is called the **hat** matrix, which linearly maps the data $\mathbf{y}$ onto the fitted values $\hat{\mathbf{y}}$. Using the property that $\text{trace}(QR) = \text{trace}(RQ)$ for matrices Q, R of conformable dimensions, the trace of the hat matrix is:

$$\begin{aligned}\text{trace}(X(X^TX)^{-1}X^T) &= \text{trace}((X^TX)^{-1}X^TX)\\ &= \text{trace}(I_p)\\ &= p,\end{aligned}$$

where I_p is the $p \times p$ identity matrix. Thus, for OLS regression, we see that the trace of the hat matrix equals the number of model parameters.

Now in Equation 5.4 we see that the smoothing matrix S_λ takes the role of a hat matrix, since it linearly maps the data onto the fitted values. This suggests that, for the smoothing spline, we can calculate an effective number of degrees of freedom as:

$$\text{edf}_\lambda = \text{trace}\, S_\lambda. \tag{5.5}$$

It can be shown from the limiting behaviour of the smoothing splines as $\lambda \to \infty$ and $\lambda \to 0$ that $\text{edf}_\infty = \nu$ (the number of parameters in the OLS solution) and $\text{edf}_0 = n$ (the number of parameters in the interpolating spline).

5.6 Generalized Cross Validation

The cross-validation criterion $Q_{OCV}(\lambda)$, defined in Equation 5.2, is used to set the smoothing parameter, λ. However, as noted in Section 5.3, using Equation 5.2 to compute $Q_{OCV}(\lambda)$ would be impractical for large n, since it would require fitting a new smoothing spline $\hat{f}_{\lambda,-j}$ for each *leave-one-out* dataset I_{-j}.

Fortunately, it is possible to compute $Q_{OCV}(\lambda)$ directly from the spline $\hat{f}_\lambda$ fitted to the full dataset. It can be shown that Equation 5.2 can be rewritten:

$$Q_{OCV}(\lambda) = \frac{1}{n}\sum_{j=1}^{n}\left(\frac{y_j - \hat{f}_\lambda(t_j)}{1 - s_{jj}}\right)^2, \tag{5.6}$$

where $\hat{f}_\lambda(t_j)$ is the full-data fitted spline value at t_j given by Equation 5.4, and s_{jj} is the jth diagonal element of the smoothing (hat) matrix S_λ.

Prior to the discovery of algorithm to compute Equation 5.6 quickly, a computationally efficient approximation to the cross-validation criterion Equation 5.2 was proposed, called the **Generalized Cross-Validation** criterion (GCV):

$$Q_{GCV}(\lambda) = \frac{\frac{1}{n}\sum_{j=1}^{n}\left(y_j - \hat{f}_\lambda(t_j)\right)^2}{\left(1 - \frac{1}{n}\mathrm{trace}(S_\lambda)\right)^2}. \tag{5.7}$$

Thus Equation 5.7 replaces s_{jj} in Equation 5.6 with the average of the diagonal elements of S_λ, which equals $\frac{1}{n}\mathrm{edf}_\lambda$. Thus a low value of edf_λ will deflate $Q_{GCV}(\lambda)$, making that value of λ more favourable.

In principle, OCV supplants GCV, but GCV is still used as it is numerically more stable. In particular, GCV is used in the `mgcv` package – this will be discussed in the next chapter.

6 General Additive Models

6.1 Overview

So far, we have considered the modelling of a response variable y in terms of a single response variable x. In particular, we have assumed that the form of this relationship is unknown but can be written as

$$y_i = f(t_i) + \epsilon_i, \qquad \epsilon_i \sim \mathrm{N}(0, \sigma^2)$$

where the ϵ_i are i.i.d. $\sim \mathrm{N}(0, \sigma^2)$ and $f(t)$ is assumed to be smooth. Given knot positions $\{t_i,\ i = 1, \dots, n\}$, we can estimate $f(t)$ with a smoothing spline $\hat{f}_\lambda(t)$ for given smoothing parameter λ and, further, we can estimate λ using ordinary or generalized cross validation. This approach is in contrast to simple linear regression where y is expressed as a linear function of the explanatory variable $y_i = \alpha + \beta x_i + \epsilon_i$ which enforces a very inflexible relationship.

In this chapter, we generalize the modelling to describe the dependence of the response variable y on a set of explanatory variables $\mathbf{x} = (x_1, x_2, \dots, x_p)$ where, conditionally on $\mathbf{x}$, observation y has a distribution which is not necessarily normal.

Just as with a generalized linear model, the *general additive model* relates a continuous or discrete response variable Y to a set of explanatory variables $\mathbf{x} = (x_1, x_2, \dots, x_p)$ and, again, the model contains three parts:

Random part: The probability (mass or density) function of Y is assumed to belong to the *two-parameter exponential family* of distributions with parameters θ and ϕ.

Systematic part: This is a *non-linear predictor equation*:

$$\eta = \sum_{j=1}^{p} f_j(x_j). \tag{6.1}$$

Link function: This is a one-to-one function providing the link between the predictor equation η and the mean $\mu = \mathrm{E}[Y]$:

$$\eta = g(\mu), \quad \text{and} \quad \mu = g^{-1}(\eta) = h(\eta). \tag{6.2}$$

Here, $g(\mu)$ is called the *link function*, and $h(\eta)$ is called the *inverse link function.*

6.2 Penalized deviance

The spline theory in the previous chapters has assumed Gaussian (normally distributed) data and the identity link function. For non-Gaussian data and/or a non-identity link function, we need to replace the penalized least-squares criterion of Equation 4.2 with a *penalized deviance*:

$$R_\nu(f, \lambda, \beta) = D(\mathbf{y}, f, \beta) + \lambda\, J_\nu(f), \tag{6.3}$$

where $D(\mathbf{y}, f, \beta)$ is the deviance for the vector $\mathbf{y}$ of observations modeled by a linear predictor that comprises a spline function $f(t)$ of order ν and possibly also covariate main-effects and interactions. The penalized deviance is then minimized with respect to the spline coefficients $\mathbf{a}$, $\mathbf{b}$ and regression parameters β, if any. Note that the fitted values for y are obtained by applying the inverse link function to the linear predictor $f(t)$, for example the logistic function when the link is logit.

When there are several smooth terms of order ν in the model, $\mathbf{f} = \{f_1, \dots, f_m\}$, each may be assigned its own roughness penalty, λ_h, and Equation 6.3 becomes

$$R_\nu(f_1, \dots, f_m, \lambda_1, \dots, \lambda_m, \beta) = D(\mathbf{y}, f_1, \dots, f_m, \beta) + \sum_{h=1}^{m} \lambda_h J_\nu(f_h) \tag{6.4}$$

or we may choose to write this as

$$R_\nu(\mathbf{f}, \lambda, \beta) = D(\mathbf{y}, \mathbf{f}, \beta) + \sum_{h=1}^{m} \lambda_h J_\nu(f_h).$$

6.3 GAMs in R

Fitting smoothing splines is straightforward in practice using R. At the beginning of each **R** session, the first step is to load the package `mgcv` which makes available a set of routines written by Simon Wood of the University of Bath.

The main command is `gam` which fits a smoothing spline (or, more generally, a general additive model). The `gam` function is an extension to the `glm` command for fitting generalized linear models, to allow nonparametric functions of explanatory variables.

The syntax of the `gam` command is similar to that of `glm`. Suppose `y` is a vector of length `n` containing observations of a dependent variable and `tt` is another vector of length `n` containing the times of those observations. Then each of the commands

fits a cubic smoothing spline to the dependent variable `y`. In the first version above, the user explicitly sets the smoothing parameter λ (here denoted `sp`) to the value 3.5. In the second version, the optimal value of λ is chosen by the routine to minimize the Generalized Cross-Validation criterion, GCV. The notation `s(tt)` means a smooth function (a cubic smoothing spline in this setting) of the explanatory variable `tt`. This notation can be viewed as an extension of the model notation used by the `glm` function. Setting `fx=FALSE` in function `s` specifies that the dimensionality of the spline should be free (see later notes).

Parameter `k` of function `s` specifies the *maximum* dimensionality of the spline, **and should be set according to the problem and data at hand** since the default value will not be appropriate in general. For example, if `tt` contains only 6 distinct values, then it would be appropriate to set `k=6`.

In each of the above examples, output from `gam` is stored in an object called `out`. This object contains several components of interest:

- `out$fitted.values` is a vector of length `n` containing the fitted values of the smoothing spline at the data values.
- `out$gcv.ubre` contains the value of the GCV criterion.
- `out$hat` contains the diagonal values of the smoothing matrix, also called the *hat* matrix. The total degrees of freedom in the model (including the intercept term) is given by `sum(out$hat)`.
- `out$sp` contains the smoothing parameter value.
- `summary.gam(out)` provides a summary of the smoothed model fit.
- `anova.gam(out)` provides an analysis of deviance for the model.

Thus, if `gam` is called without an explicit choice of `sp` (as in the second example above), the output from `gam` gives the optimal smoothing parameter value and the corresponding value of the GCV criterion.

To plot a smoothing spline:

- define a dense set of times spanning the data, at which to plot the smooth curve;
- use the `predict.gam` function to compute the cubic spline at these values.

For example, suppose `y` is a vector of length 20 containing the responses at times $1, \dots, 20$.

6.4 Coronary heart disease (CHD) in South Africa

The textbook *Elements of Statistical Learning*, by Hastie, Tibshirani and Friedman (2nd Edn, 2011), refers to a case–control study of coronary heart disease (CHD) in South Africa. The data are available from the authors' website, and may be read directly into **R** with the command:

Variable	Description
`tobacco`	cumulative tobacco consumption (kg)
`famhist`	family history of heart disease (Present, Absent)
`age`	age at onset of the disease (years)
`chd`	case–control status ($1 \Rightarrow$ CHD; $0 \Rightarrow$ no CHD).

The dependent variable in our models will be `chd`. We can consider the CHD status of each individual to be the result of an *experiment* in which the outcome is either CHD (*success*) or no CHD (*failure*). Thus we can model these data using the Binomial distribution, where the Binomial index is 1. The `glm` function allows such binary (0/1) dependent variables to be specified directly in the model formula, as in the example below, instead of via the usual two-column matrix of successes and failures.

We can examine CHD in relation to tobacco consumption, age and family history of heart disease, with the following **R** commands:

```
Call:
glm(formula = "chd~tobacco+age+famhist", family = "binomial")

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.8399  -0.8784  -0.4705   0.9698   2.3876

Coefficients:
                Estimate Std. Error z value Pr(>|z|)
(Intercept)    -3.620593   0.444576  -8.144 3.83e-16 ***
tobacco         0.083004   0.025712   3.228  0.00125 **
age             0.048812   0.009452   5.164 2.42e-07 ***
famhistPresent  0.974791   0.220023   4.430 9.41e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 596.11  on 461  degrees of freedom
Residual deviance: 495.39  on 458  degrees of freedom
AIC: 503.39

Number of Fisher Scoring iterations: 4
```

All variables in this model are statistically significant: disease is positively related to the amount of tobacco consumed, age and family history of heart disease. However, this model assumes that the logit of the probability of disease is *linearly* related to both tobacco consumption and age (logit being the default link for Binomial). We can explore more flexible tobacco-consumption and age trends with the following generalized additive model:

```
Loading required package: nlme
```

```
This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.
```

```
Family: binomial
```

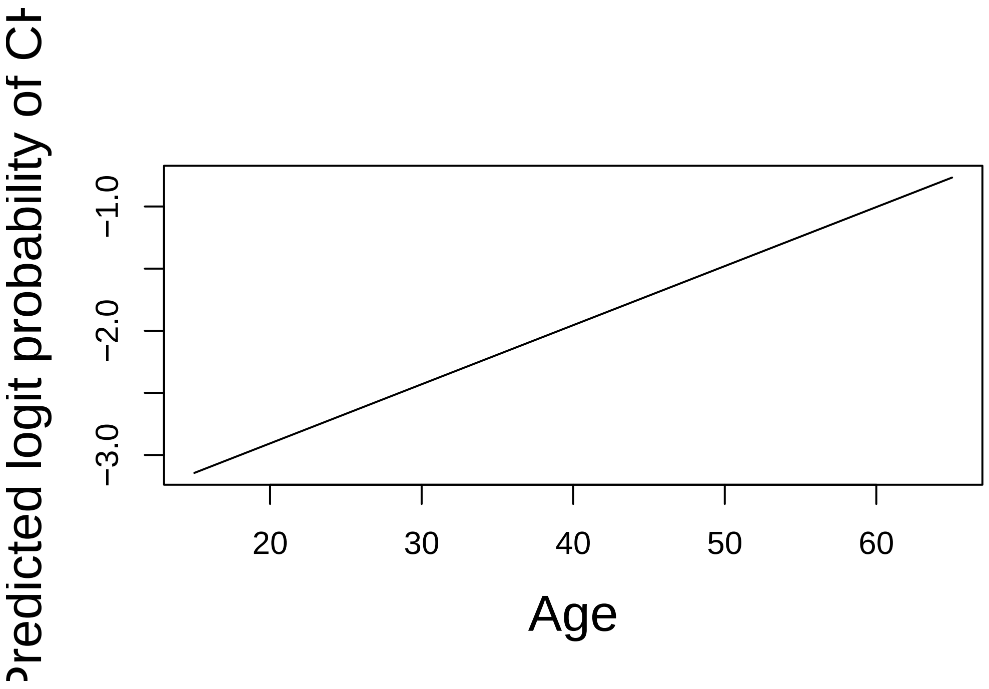
```
Link function: logit

Formula:
chd ~ s(tobacco, k = 20) + s(age, k = 20) + famhist

Parametric coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)     -1.2379     0.1631  -7.592 3.15e-14 ***
famhistPresent   0.9628     0.2233   4.311 1.62e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Approximate significance of smooth terms:
             edf Ref.df Chi.sq  p-value
s(tobacco) 6.080  7.573  17.89   0.0179 *
s(age)     1.002  1.003  24.11 9.53e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

R-sq.(adj) =  0.212   Deviance explained = 19.1%
UBRE = 0.083268  Scale est. = 1         n = 462
```

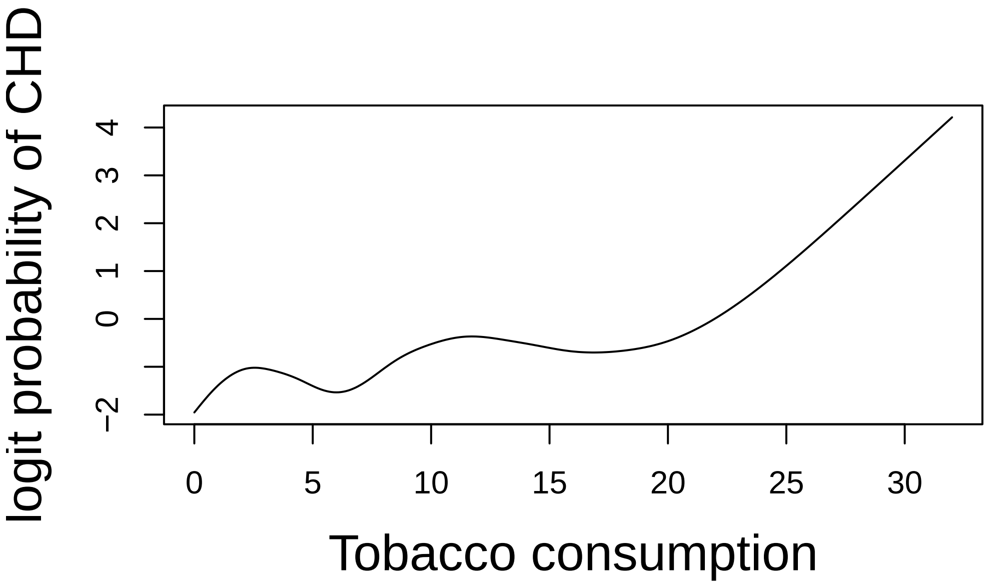
Here, the `s` function specifies a smooth (cubic spline) dependence. Parameter `k` of the `s` function specifies the maximum number of degrees of freedom to be allocated to this dependence. Setting `k` too small would restrict the set of basis functions used to construct the splines; setting `k` too large would increase the computational burden unnecessarily.

These results show that the fitted smoothing spline of the dependence of CHD on tobacco consumption has an *effective degrees of freedom* (edf) of 6.080, while the dependence on age has edf of only 1.002, implying an almost linear age-dependence because a linear age term would have exactly 1 degree of freedom. See Section Section 5.5 for further details on the calculation of edf.

The significance of each of these smooth terms is given by the p-value column in the above table. These p-values are computed from the χ^2 statistics in the previous column whose approximate degrees of freedom are given in the column headed `Ref.df`. For example, the p-value of 0.0196 for `s(tobacco)` is computed by referring 17.62 to the χ^2 distribution on 7.573 degrees of freedom. Note that this compares the above model with the model which omits `tobacco` completely.

The following **R** code was used to plot the fitted smooth functions of age and tobacco consumption:





The predicted dependence of the logit probability of CHD on smooth functions of age and tobacco consumption is shown above. We see an almost linear predicted age-dependence, in agreement with its edf. The curious predicted dependence on tobacco consumption might be explained by other factors correlated with tobacco consumption.