Exploiting smoothness in regression and joint models

Estimation, inference, and interpretation for models with smooth components

Alkeos Tsokos
Supervisor: Ioannis Kosmidis

A thesis presented for the degree of
Doctor of Philosophy

Statistical Science
University College London
United Kingdom

I, Alkeos Tsokos, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

In this thesis we explore various aspects of smooth modelling. We make contributions in two main areas. The first is in generalized additive modelling, for which we propose an approach that allows for the estimation of functions in a locally adaptive way that does not require the estimation of tuning parameters, and consequently scales well with the number of predictors. This is achieved through the use of a particular sparsity inducing prior on the coefficients of b-splines that are used to represent smooth functions. In addition, we propose a method to determine the individual and relative importance of predictors in generalized additive models, aiding in their interpretation and explanatory power.

The second topic we explore is a scenario in which multiple associated variables vary smoothly as a function of some argument, and the objective is to estimate the associations between them. To tackle this problem we propose a general framework that we name structural smooth modelling. Our approach allows us to model multiple stochastic processes jointly, estimating associations between them, without assuming that each process has been observed at the same set of argument values. The general model is flexible and potentially applicable in a variety of disciplines. As a use case we apply the model to data obtained from British Cycling, demonstrating strong potential for the framework to be used as a way to track athlete performance and estimate associations between performance in different types of training efforts.

Impact statement

The work in this thesis is mostly methodological in nature. As a result, potential impact lies in applications that the methodologies we develop may find, and perhaps in the development and improvement of further methodology as a result.

Generalized additive models have become a core part of the statistician's toolkit due to the fact that they are more flexible in the types of associations that they can model when compared to generalized linear models, while remaining more interpretable than supervised learning techniques that are focused solely on prediction. By developing scalable and adaptive methods to estimate generalized additive models, and tools to aid in their interpretation, we hope to expand their applicability and potential for explanatory and predictive power in any setting in which they may be used.

The development of the structural smooth modelling (SSM) framework that we present in this thesis was motivated by the structure of the British Cycling data that we had access to, and so immediate impact lies in the results of the applied analysis that we present, and the potential for further use within the context of British Cycling. Because it is a very general approach though, it could find applications in a variety of areas. As technology permeates everyday life to an ever increasing degree, the amount and types of data that are being collected in a variety of contexts is growing rapidly. The types of data sets that we envision SSM being applied to are large data sets in which many variables are being tracked over time, and there is interest in understanding how they co-evolve. Crucially, because we allow for different variables to be observed at different time points, SSM could be used in contexts where data from different sources are pooled together, or in situations where multiple variables are being tracked but it is not feasible to collect measurements for every variable simmultaneously. More specific use cases could include analysis of wearable sensor data that are longitudinal in nature or data in which health or well-being is tracked over time and there is a desire to test hypotheses regarding how variables influence eachother. Furthermore, there is much scope to extend the framework we have presented, and therefore further potential impact lies in the possibility of SSM being used as a starting point for further methodological development that can eventually become even more useful in applied settings.

Acknowledgements

This thesis would be incomplete without acknowledging the people who were instrumental in supporting its development.

I would like to start by thanking my parents, who instilled in me from a young age a sense of curiosity about the world and always stressed the importance of education. Their careers and achievements will always be an inspiration to me. Throughout my life and my studies they have supported me in every way imaginable and words will never be enough to express my gratitude. I would also like to sincerely thank my partner, Angeliki, for putting up with me during a writing up period that seemed as though it might never end, and for reminding me of what really matters at the end of the day.

The date of completion of this thesis marks nine years since I enrolled as an undergraduate in the Statistical Science department of University College London. With the exceptions of one year in the Computer Science department and one year as a study abroad student, these nine years have been spent in the Statistical Science department. During this time I have been lucky to have had some excellent lecturers and great coursemates. I will always be grateful to the department as a whole for making me feel welcome and supporting my studies. In particular, I would like to thank Yvo Pokern, who was one the best lecturers during my undergraduate degree, and who supervised my undergraduate project. I would also like to sincerely thank Deepti Jayawardena Wilkinson, who throughout my PhD sorted out any administrative issues I had at lightning speed, despite her busy schedule.

This PhD project was partially funded by the English Institute of Sport, and occurred in collaboration with British Cycling. Visits to the National Cycling Center were a great pleasure, and everyone from British Cycling was extremely helpful in facilitating the work that we did using British Cycling data.

A lot of my time during my PhD was spent at the Alan Turing Institute, first as an intern, then as an enrichment student, and finally as a visiting student. This gave me the opportunity to meet and interact with many great people and significantly improved the quality of my time as a PhD student.

Finally, and most importantly, I feel deeply grateful toward my PhD supervisor, Ioannis Kosmidis. For being an exceptional lecturer during my undergraduate degree. For supervising my MSc project. For allowing this PhD project to take place. For his technical expertise that provided guidance and support throughout. For the great generosity he has always shown with his time, and for genuinely caring about my progress as well as my well-being. For constantly finding, suggesting, and supporting opportunities for my development. For making my needs and interests a priority. For his patience and flexibility in working with me. And above all for gracefully managing to simmultaneously be a great supervisor but also a friend.

Contents

Chapter 1

Introduction

1.1 Preamble

This thesis is devoted to studying various aspects of modelling with smooth components. Generally this refers to models that involve functions whose shape is unknown, with the only assumption about the functions being that they are smooth. The quintessential example of such a model is scatterplot smoothing, in which one observes pairs of observations $(y_1, x_1), \ldots, (y_n, x_n)$. When plotted, with y_i on the y-axis and x_i on the x-axis, one may observe an association; as x varies, y may on average vary as well. The purpose of scatterplot smoothing is to estimate the functional form of this association without imposing constraints other than assuming that y varies smoothly as a function of x. There is a wealth of scenarios when a statistical model may include functions whose form is unknown. In this thesis we will focus on two such scenarios.

The first is in generalized additive modelling, which is an extension of scatterplot smoothing when there are multiple predictors. Generalized additive models have been studied extensively in the literature. We make two contributions to the existing literature. Firstly, we propose an approach that allows for the estimation of functions in a locally adaptive way that does not require the estimation of tuning parameters, and consequently scales well with the number of predictors, in contrast to existing methods in the literature. Secondly, we propose a method to determine the individual and relative importance of predictors in generalized additive models. To the best of our knowledge this problem has not previously been tackled.

The second scenario we consider is one in which multiple associated variables vary smoothly as a function of some argument, and the objective is to estimate the associations between them. To tackle this problem we propose a general framework that we name structural smooth modelling. Our approach allows us to model multiple stochastic processes jointly, estimating associations between them, without assuming that each process has been observed at the same set of argument values. The general model is flexible and potentially applicable in a variety of disciplines.

1.2 Motivation

Smooth modelling is an established research area in statistical science with a large number of applications. Methodological developments in this field can help applied researches in a variety of disciplines implement models that can be useful in both explanatory and predictive contexts. In our case, the practical applications that drove much of the methodology we developed stem from our partnership with British Cycling, and the

English Institute of Sport (EIS), which allowed this project to take shape by providing partial funding and data. EIS supports Olympic athletes by delivering services aimed at optimising training programmes and improving performance. Working with British Cycling, we gained access to a unique data set consisting of training efforts of the Great Britain Cycling Team. The data are longitudinal in nature, tracking the training efforts of multiple athletes over time, but also contain many unique features such as asynchronicity, in which different training efforts and physiological assessments occur at different points in time, without a fixed schedule. Developing models that are flexible but also explanatory is a challenge in this context, and some of the methods we develop aim to bridge this gap.

1.3 Overview of thesis

A popular approach to modelling with smooth components is to use basis functions to represent smooth functions, and then optimise a regularised likelihood to estimate the coefficients of the basis functions. Using basis functions, generalized additive models are reduced to regularised generalized linear models. With this in mind, we begin in Chapter 2 with a brief overview of linear and generalized linear modelling, followed by an overview of ℓ_2 and ℓ_1 regularisation techniques. In Chapter 3 we give an overview of modelling with spline functions, how splines are used in generalized additive models, and how regularisation plays a central role in using splines to model smooth functions. In Chapter 4 we introduce a method to achieve locally adaptive smoothness in generalized additive models, which in contrast to existing methods requires no tuning, is computationally efficient, and scales well with the number of predictors. In Chapter 5 we introduce a method to estimate the individual and relative importance of predictors in generalized additive models, creating a tool that can facilitate interpretation of generalized additive models. Finally, in Chapter 6 we present the structural smooth modelling framework along with examples of how it can be used using both simulated and real data.

Chapter 2

Generalized linear models and regularisation

2.1 Preample

Regularisation, and corresponding Bayesian interpretations, form a central part of this thesis. This chapter is devoted to introducing various regularisation techniques as well as associated optimisation algorithms and tuning parameter selection methods. We will begin with a very brief introduction to linear modelling, followed by an overview of the regularisation techniques that will feature in this thesis.

2.2 Introduction to linear modelling

The ordinary linear model

$$y_i = \sum_{j=1}^{p} \beta_j x_{ij} + \epsilon_i \quad (i = 1, \ldots, n) \ ,$$

models a response variable y as a noisy observation of a linear combination of a set of predictor variables $x_1, \ldots, x_p$. The error terms ϵ_i are often assumed to be independently and identically distributed normal random variables with finite variance and mean zero, implying that the responses are also normally distributed. Under the normality assumption, the negative log-likelihood for the parameter vector $\boldsymbol{\beta}$ is

$$-\ell(\boldsymbol{\beta}) = \frac{1}{2\sigma^2}(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta}) + c \ ,$$

where σ^2 is the variance of the errors, c is a constant that does not depend on $\boldsymbol{\beta}$, X is the model matrix, with (i, j)-th element x_{ij}, and $\boldsymbol{y}$ is the vector of responses with i-th element y_i. The maximum likelihood estimate (MLE) of $\boldsymbol{\beta}$ is the vector that minimises the negative log-likelihood. Differentiating and equating the gradient to zero gives

$$\hat{\boldsymbol{\beta}} = (X^T X)^{-1} X^T \boldsymbol{y} \ . \tag{2.1}$$

The covariance matrix of the MLE is

$$\begin{aligned}\operatorname{cov}(\hat{\boldsymbol{\beta}}) &= \operatorname{cov}\Big((X^TX)^{-1}X^T\boldsymbol{y}\Big) \\ &= \operatorname{cov}\Big((X^TX)^{-1}X^T(X\boldsymbol{\beta}+\boldsymbol{\epsilon})\Big) \\ &= \sigma^2(X^TX)^{-1} \ .\end{aligned}$$

While maximum likelihood estimators, under the model assumptions, enjoy many nice theoretical properties such as asymptotic normality and efficiency (Millar, 2011, Chapters 11, 12), there are scenarios in linear modelling where maximum likelihood estimators run into trouble. To see this we can re-express the MLE in terms of the eigendecomposition of X^TX. The eigendecomposition is comprised of an orthogonal matrix V and diagonal matrix D that satisfy the relation

$$X^TX = V^TDV \ , \tag{2.2}$$

with the columns of V referred to as eigenvectors and the elements of D referred to as eigenvalues. Plugging (2.2) into (2.1) we obtain

$$\hat{\boldsymbol{\beta}} = VD^{-1}V^TX^T\boldsymbol{y} \ ,$$

and similarly for the variance

$$\operatorname{cov}(\hat{\boldsymbol{\beta}}) = \sigma^2VD^{-1}V^T \ .$$

Now it becomes evident that if X^TX approaches rank-deficiency, implying that at least one eigenvalue will approach zero, the MLE and its variance will explode as they depend on the inverse of the eigenvalues. In the following sections we will introduce common techniques that can tackle this issue.

While the ordinary linear model is simple and its use widespread, it is inadequate when the response variable is not normally distributed, with common cases being binary, count, or non-negative responses. Generalized linear models (GLMs) (McCullagh and Nelder, 1989) extend the ordinary linear model by modelling a function of the expected value of the responses as a linear combination of predictors. Specifically, a GLM has

$$g\Big(E(y_i)\Big) = \sum_{j=1}^{p}\beta_jx_{ij} \quad (i=1,\dots,n) \ ,$$

where $g(\cdot)$ is a sufficiently smooth monotonic link function, and y_i has an exponential family distribution. Ordinary linear regression is a special case of a GLM in which the response is normally distributed and $g(\cdot)$ is the identity function. Logistic regression is one of the most popular GLMs in which a binary response variable is modelled and the logit link function is used. The binary responses $y_1,\dots,y_n$ are interpreted as Bernoulli random variables with probability of success $p_1,\dots,p_n$ respectively, which are modelled as

$$\log\left(\frac{p_i}{1-p_i}\right) = \sum_{j=1}^{p}\beta_jx_{ij} \ ,$$

or equivalently

$$p_i = \frac{\exp\left(\sum_{j=1}^{p}\beta_jx_{ij}\right)}{1+\exp\left(\sum_{j=1}^{p}\beta_jx_{ij}\right)} \ .$$

The logit link function allows the linear combination $\sum_j\beta_jx_{ij}$ to take on values on the real line while ensuring that p_i is a valid probability between zero and one. Whereas the negative log-likelihood can be minimised

analytically to obtain maximum likelihood estimates in the ordinary linear regression case, in the case of GLMs, iterative methods are typically required, with Newton's method or Fisher's scoring (Agresti, 2015, Chapter 4) being the most popular choices.

2.3 ℓ_2 regularisation

In Section 2.2 we saw that in cases when the model matrix approaches rank deficiency the MLE becomes unstable. Ridge regression (Hoerl and Kennard, 1970) deals with this by adding a quadratic penalty term to the negative log-likelihood. For a normal response variable the ridge estimator is

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{ (\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta}) \right\} , \tag{2.3}$$

where λ is a positive tuning parameter. Differentiating and solving for zero gives

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= (X^T X + \lambda I)^{-1} X^T \boldsymbol{y} \\ &= V^T (D + \lambda I)^{-1} V X^T \boldsymbol{y} , \end{aligned} \tag{2.4}$$

which has variance

$$\begin{aligned} \mathrm{cov}(\hat{\boldsymbol{\beta}}) &= \sigma^2 (X^T X + \lambda I)^{-1} X^T X (X^T X + \lambda I)^{-1} \\ &= V^T (D + \lambda I)^{-1} D (D + \lambda I)^{-1} V . \end{aligned} \tag{2.5}$$

From (2.4) and (2.5) it can be seen that even if $X^T X$ is rank-deficient, with at least one zero eigenvalue, the ridge estimator exists and has finite variance, which tends to zero as $\lambda \to \infty$. The intuition is that the optimisation in (2.3) compromises between finding a coefficient vector for which the likelihood is high but whose squared ℓ_2 norm is low. Seeking solutions with a low squared ℓ_2 norm may seem arbitrary, and indeed introduces bias since

$$E(\hat{\boldsymbol{\beta}}) = (X^T X + \lambda I)^{-1} X^T X \boldsymbol{\beta} .$$

However, the reduction in variance can make up for the bias introduced, resulting in a smaller mean squared error. In fact Hoerl and Kennard (1970) proved that there always exists a $\lambda > 0$ such that the ridge estimator achieves a lower mean squared error than the MLE, even though in practice it is not known what this λ is. In the context of GLMs the ridge estimator can be expressed as

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{ -\ell(\boldsymbol{\beta}) + \frac{\lambda}{2\phi} \boldsymbol{\beta}^T \boldsymbol{\beta}) \right\} , \tag{2.6}$$

where ϕ is the dispersion parameter of the exponential family distribution that is assumed for $y_1, \ldots, y_n$, and $-\ell(\boldsymbol{\beta})$ is the negative log-likelihood for $y_1, \ldots, y_n$. The solution of the minimisation problem in (2.6) cannot be expressed in closed form. Nevertheless, for general exponential families the same optimisation algorithms can be used as in ordinary GLMs, without much extra complication, because the squared ℓ_2 norm that is added to the negative log likelihood is a smooth, convex function.

More generally, ℓ_2 regularisation can take the form

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{ -\ell(\boldsymbol{\beta}) + \frac{1}{2\phi} \sum_{k=1}^{d} \lambda_k \boldsymbol{\beta}^T S_k \boldsymbol{\beta} \right\} , \tag{2.7}$$

which generalises (2.6) in two ways. Firstly, the penalisation is now on a quadratic form of $\boldsymbol{\beta}$, involving a positive-definite matrix $S = \sum_k \lambda_k S_k$, rather than on the squared ℓ_2 norm. Secondly, the matrix S is

composed of parts S_k each of which is weighted by its own tuning parameter λ_k. Whereas ordinary ridge regression seeks coefficients vectors with low squared ℓ_2 norm, generalised ridge regression seeks solutions for which the quadratic forms $(\boldsymbol{\beta}^T S_1 \boldsymbol{\beta}, \ldots, \boldsymbol{\beta}^T S_d \boldsymbol{\beta})$ are small. This can be a useful way to incorporate prior information into the estimation procedure. As will be seen in Chapter 3, the quadratic form can represent the roughness of a function, and adding it as a penalty term can result in function estimates that are smooth. For an in-depth treatment of regularised GLMs in the form of (2.7), including details on optimisation using Newton's method, see Wood (2000).

The estimator in (2.7) can also be viewed from a Bayesian perspective. If we set up a Bayesian GLM

$$\begin{aligned} \boldsymbol{\beta} &\sim \mathcal{N}(\mathbf{0}, \phi S^{-1}) \\ g(\mu_i) &= \sum_{j=1}^{p} \beta_j x_{ij} \\ y_i | \boldsymbol{x}_i, \boldsymbol{\beta}, \phi &\sim \pi(\mu_i, \phi) \ , \end{aligned}$$

where $\mu_i = E(y_i)$, and π is an exponential family distribution parametrised by its mean μ and dispersion parameter ϕ, then (2.7) emerges as the maximum a posteriori estimator

$$\begin{aligned} \hat{\boldsymbol{\beta}} &= \arg\max_{\boldsymbol{\beta}} p(\boldsymbol{\beta}|\boldsymbol{y}) \\ &= \arg\min_{\boldsymbol{\beta}} \Big\{ -\log p(\boldsymbol{\beta}|\boldsymbol{y}\Big) \Big\} \\ &= \arg\min_{\boldsymbol{\beta}} \Big\{ -\log p(\boldsymbol{y}|\boldsymbol{\beta}) - \log p(\boldsymbol{\beta}) \Big\} \\ &= \arg\min_{\boldsymbol{\beta}} \Big\{ -\ell(\boldsymbol{\beta}) + \frac{1}{2\phi} \sum_{k=1}^{d} \lambda_k \boldsymbol{\beta}^T S_k \boldsymbol{\beta} \Big\} \ . \end{aligned}$$

Intuitively the Bayesian view of regularisation reinforces the equivalence of regularisation and incorporation of prior information into the estimation procedure. For further details on the Bayesian interpretation of regularised GLMs see Wood (2011).

To compute the ℓ_2 regularised estimator one has to set the tuning parameters $(\lambda_1, \ldots, \lambda_d)$. This selection is crucial as if they are too small then regularisation is not strong enough and becomes ineffective. If, on the other hand, $(\lambda_1, \ldots, \lambda_d)$ are too large, then the contribution of the data to the model fit becomes too small. There are various approaches to selecting tuning parameters. Cross validation methods can be used to choose the tuning parameters that minimise an estimate of the out-of-sample prediction error. The resampling based variant, K-fold cross validation, involves splitting the data into K parts of roughly equal size, and estimating the model K times, each time with one part left out. Each time the model is estimated with one part left out, predictions are made for the data in the part that was left out, and an estimate of the out-of-sample prediction error is obtained. These estimates are then averaged over the K parts to obtain a final estimate of the out-of-sample prediction error. This process is repeated for several settings of the tuning parameters, and the tuning parameters with the lowest estimated out-of-sample prediction error are chosen. See Arlot and Celisse (2010) for a review of K-fold cross validation techniques.

Generalized cross validation (GCV), (Golub et al., 1979), is an analytic variant of cross validation, which

for a normal response variable, estimates the out-of-sample prediction error as

$$\mathrm{GCV}(\boldsymbol{\lambda}) = \frac{n\Big(\boldsymbol{y} - H\boldsymbol{y}\Big)^T\Big(\boldsymbol{y} - H\boldsymbol{y}\Big)}{\Big(n - \mathrm{tr}(H)\Big)^2} \ ,$$

where $H = X(X^TX + S)^{-1}X^T$. See Wood (2006, Page 173) for approximations of the GCV criterion for non-normal responses. The GCV approach has the advantage over K-fold cross validation that iterative methods can be used to find the optimal $\boldsymbol{\lambda}$ vector, in contrast to performing a grid search that can become prohibitively expensive for multiple tuning parameters in the case of K-fold cross validation. For details on implementing and optimising the GCV criterion see Wood (2004).

In the Bayesian interpretation of ℓ_2 regularisation the $\boldsymbol{\lambda}$ vector is a vector a hyperparameters, and standard Bayesian approaches are available for its estimation. One approach is empirical Bayes (Casella, 1985), which in this case corresponds to estimating tuning parameters by maximising the marginal log-likelihood

$$\ell_m(\boldsymbol{\lambda}, \phi) = \log p(\boldsymbol{y}; \boldsymbol{\lambda}, \phi) = \log \int_{\boldsymbol{\beta}} p(\boldsymbol{y}|\boldsymbol{\beta}; \boldsymbol{\lambda}, \phi) p(\boldsymbol{\beta}; \boldsymbol{\lambda}, \phi) d\boldsymbol{\beta} \ , \tag{2.8}$$

followed by estimation of $\boldsymbol{\beta}$ by

$$\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} p(\boldsymbol{\beta}|\boldsymbol{y}; \hat{\boldsymbol{\lambda}}, \hat{\phi}) \ .$$

In general (2.8) is analytically intractable, however it can be replaced by its Laplace approximation (Wood, 2011)

$$\ell_m(\boldsymbol{\lambda}, \phi) \approx \ell_c^{\mathrm{reg}}(\hat{\boldsymbol{\beta}}, \boldsymbol{\lambda}, \phi) + \frac{1}{2}\log|S/\phi|_+ - \frac{1}{2}\log|\mathcal{H}| + \frac{M_p}{2}\log(2\pi) \ ,$$

where

$$\ell_c^{\mathrm{reg}}(\boldsymbol{\beta}, \boldsymbol{\lambda}, \phi) = \ell_c(\boldsymbol{\beta}, \phi) - \frac{1}{2}\sum_{k=1}^{d} \lambda_k \boldsymbol{\beta}^T S_k \boldsymbol{\beta}/\phi$$

is the regularised conditional log-likelihood,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} -\ell_c^{\mathrm{reg}}(\boldsymbol{\beta}, \boldsymbol{\lambda}, \phi)$$

is the value of $\boldsymbol{\beta}$ that maximises it,

$$\mathcal{H} = -\frac{\partial^2 \ell_c^{\mathrm{reg}}(\boldsymbol{\beta}, \boldsymbol{\lambda}, \phi)}{\partial\boldsymbol{\beta}\partial\boldsymbol{\beta}^T}$$

is the negative Hessian of the regularised conditional log-likelihood, and M_p is the number of zero eigenvalues of S.

2.4 ℓ_1 regularisation

As previously mentioned, regularisation is a way of incorporating prior information into an estimation procedure. With ℓ_2 regularisation, it is possible to incorporate the information that some quadratic forms of the coefficient vector should not be large. In ℓ_1 regularisation ℓ_1 norms of the coefficient vector are added as penalties in order to enforce sparsity. For example, the LASSO (Tibshirani, 1996) solves

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \Big\{ -\ell(\boldsymbol{\beta}) + \frac{\lambda}{\phi}\|\boldsymbol{\beta}\|_1 \Big\} \ , \tag{2.9}$$

which depending on the size of λ may result in elements of $\hat{\boldsymbol{\beta}}$ that are exactly zero. Hence, ℓ_1 regularisation incorporates into the estimation procedure the information that some of the predictors do not influence the response at all, and the estimation procedure reflects this by estimating some coefficients to be exactly zero, essentially eliminating them from the model. Estimators that are capable of producing zero estimates are referred to as 'sparsity-inducing'. To see why the LASSO is sparsity-inducing, whereas ridge regression is not, it is easiest initially to consider the LASSO estimate for a one dimensional regression problem. In the case of one predictor and a normal response, the model is

$$y_i = \beta x_i + \epsilon_i \ , \quad (i = 1, \ldots, n) \ ,$$

and the LASSO objective function is

$$L(\beta) = \frac{1}{2}\sum_{i=1}^{n}(y_i - \beta x_i)^2 + \lambda|\beta| \ .$$

Now given that $L(\beta)$ is a sum of two convex functions, it is itself convex, and hence $\hat{\beta}$ is a minimiser of $L(\beta)$ if

$$\partial_- L(\beta)|_{\beta=\hat{\beta}} \leq 0 \leq \partial_+ L(\beta)|_{\beta=\hat{\beta}} \ ,$$

where $\partial_-(\cdot)$ and $\partial_+(\cdot)$ denote the derivative from the left and right respectively. In the case of differentiable functions this condition reduces to the ordinary requirement that at a minimum the derivative of a function must be zero (excluding boundary cases). For a non-differentiable function the condition states that a move towards the left will increase the objective function since the derivative from the left is negative, and similarly a move toward the right will also increase the objective function since the derivative from the right is positive, however the function need not be differentiable at the minimum. We have that

$$\partial_+ L(\beta)|_{\beta=0} = -\boldsymbol{x}^T\boldsymbol{y} + \lambda \ ,$$

and

$$\partial_- L(\beta)|_{\beta=0} = -\boldsymbol{x}^T\boldsymbol{y} - \lambda \ .$$

This implies that zero will be a minimiser of $L(\beta)$ if

$$-\boldsymbol{x}^T\boldsymbol{y} - \lambda \leq 0 \leq -\boldsymbol{x}^T\boldsymbol{y} + \lambda \ ,$$

which can be written more succinctly as

$$\lambda \geq |\boldsymbol{x}^T\boldsymbol{y}| \ .$$

For ridge regression on the other hand the penalty function with one predictor is simply $\lambda\beta^2$, whose derivative with respect to β at $\beta = 0$ is just zero, meaning that an infinitesimal move away from zero has no cost with respect to the ridge penalty. Now we have a condition under which the solution to the LASSO is zero. For $\lambda < |\boldsymbol{x}^T\boldsymbol{y}|$ we have

$$\partial L(\boldsymbol{\beta}) = -\boldsymbol{x}^T\boldsymbol{y} + \boldsymbol{\beta} + \lambda\text{sgn}(\boldsymbol{\beta}) \ ,$$

resulting in the estimating equation

$$\hat{\boldsymbol{\beta}} = \boldsymbol{x}^T\boldsymbol{y} - \lambda\text{sgn}(\hat{\boldsymbol{\beta}}) \ .$$

Noticing that $\text{sgn}(\hat{\boldsymbol{\beta}}) = \text{sgn}(\boldsymbol{x}^T\boldsymbol{y})$ since $\lambda < |\boldsymbol{x}^T\boldsymbol{y}|$, we obtain

$$\hat{\boldsymbol{\beta}} = \boldsymbol{x}^T\boldsymbol{y} - \lambda\text{sgn}(\boldsymbol{x}^T\boldsymbol{y}) \ .$$

Putting all this together we finally obtain

$$\hat{\beta} = \begin{cases} \boldsymbol{x}^T\boldsymbol{y} + \lambda & \text{if } \boldsymbol{x}^T\boldsymbol{y} < -\lambda \\ 0 & \text{if } -\lambda < \boldsymbol{x}^T\boldsymbol{y} < \lambda \\ \boldsymbol{x}^T\boldsymbol{y} - \lambda & \text{if } \boldsymbol{x}^T\boldsymbol{y} > \lambda \ . \end{cases} \tag{2.10}$$

The right hand side of equation (2.10) can be denoted by $\mathcal{S}_\lambda(\boldsymbol{x}^T\boldsymbol{y})$, and is commonly known as the soft-thresholding operator. In the case of many predictors, the solution cannot be found analytically, and iterative methods need to be used. Because the LASSO objective is not smooth, ordinary gradient based optimisation algorithms are not suitable. A popular method is coordinate descent, in which all the coefficients but one are held constant, and the solution is found for the one free coefficient. The algorithm then repeatedly cycles through coefficients like this until convergence. Because the solution can be found analytically in one dimension, this algorithm is simple and fast, and can be extended to GLMs by iteratively solving the LASSO with a quadratic approximation to the likelihood, in the spirit of Newton's method. See Friedman et al. (2010) for details.

ℓ_1 regularisation can also be interpreted from a Bayesian perspective, where instead of placing a normal prior on the coefficient vector as in ℓ_2 regularisation, a Laplacian prior is used. The hirearchical model is

$$\begin{aligned} \beta_j &\sim \text{Laplace}(0, \phi/\lambda) \\ g(\mu_i) &= \sum_{j=1}^{p} \beta_j x_{ij} \\ y_i|\boldsymbol{\beta}, \phi &\sim \pi(\mu_i, \phi) \ . \end{aligned}$$

The maximum a posterior estimator for $\boldsymbol{\beta}$ in the above modeel takes the form of (2.9). See Park and Casella (2008) for a fully Bayesian treatment of the LASSO.

ℓ_1 regularisation can also be used to achieve sparsity in linear combinations of the coefficient vector by modifying the LASSO objective to

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \Big\{ -\ell(\boldsymbol{\beta}) + \frac{\lambda}{\phi}\|D\boldsymbol{\beta}\|_1 \Big\} \ ,$$

where D is a penalty matrix. In this form the estimates $\hat{\boldsymbol{\beta}}$ will be such that some elements in $D\hat{\boldsymbol{\beta}}$ may be exactly zero. This is often referred to as the generalised LASSO, and it is in this form that ℓ_1 regularisation will appear in Chapter 3 with reference to smoothing problems. Tibshirani and Taylor (2011) provide a path algorithm that computes the solution along a sequence of λ values, while the proximal gradient method (Parikh and Boyd, 2013, Chapter 4) and the ADMM algorithm (Boyd et al., 2010, Chapter 3) can also be used.

Tuning parameter selection is more challenging for ℓ_1 regularisation than ℓ_2. This is because GCV optimisation and empirical Bayes which are the most popular choices for ℓ_2 regularisation are not easy to implement in the case of ℓ_1 regularisation. Because the degrees of freedom of ℓ_1 regularised fits do not vary smoothly as a function of the tuning parameters the GCV function is not smooth, and consequently ordinary gradient based optimisation algorithms would not be appropriate. For empirical Bayes the marginal likelihood would not exist in closed form and hence maximising it would be challenging. For these reasons tuning parameter selection is usually based on K-fold cross validation. While this may be alright for up to a couple of tuning parameters, with multiple tuning parameters performing a grid search with K-fold

cross validation becomes infeasible. These issues present a limitation in the flexibility of ℓ_1 regularisation compared to ℓ_2, for which tuning parameter selection with multiple tuning parameters has proven to be feasible.

Chapter 3

Modelling with splines

3.1 Preamble

In this chapter we give an overview of spline functions, and their use in modelling with smooth components. We begin by introducing a simple example of one dimensional smoothing to illustrate how spline functions can be used, followed by an overview of b-splines, regularisation, and knot selection. We then introduce generalized additive models as an instance of a more sophisticated modelling technique that can be implemented using b-splines.

3.2 Introduction

To illustrate the ideas in this chapter we begin with a simple example of scatterplot smoothing. Suppose we observe data $\{(y_i, x_i)\}_{i=1}^n$, and we wish to model each response y_i as noisy observations of some unknown smooth function $f(x_i)$. The model can be written as

$$y_i = f(x_i) + \epsilon_i \ ,$$

where $\epsilon_1, \ldots, \epsilon_n$ are independently identically distributed normal random variables with mean zero. The goal is to estimate the function $f(\cdot)$. For simplicity in what follows we will always assume that domain of $f(\cdot)$ is $[0, 1]$. In principle, any technique that performs some kind of supervised learning (Hastie et al., 2009) could be used to estimate $f(\cdot)$, however because we assume that it is smooth, ideally the method should result in estimates $\hat{f}(\cdot)$ that are smooth. Approaches that can be used to achieve this include locally estimated scatterplot smoothing (Cleveland and Devlin, 1998), kernel regression (Watson, 1964), and Gaussian process regression (Rasmussen, 2006). In this chapter we will focus on using splines to model $f(\cdot)$, and in particular using b-splines (de Boor, 2001) for estimation. Splines are piecewise polynomial functions that satisfy smoothness constraints. They are defined over a non-decreasing sequence of knots, which act as breakpoints over which each piecewise polynomial is defined. A spline function is said to be of order m if the degree of the piecewise polynomials is $m-1$. The piecewise polynomials are constrained such that the global spline function is $m-2$ times continuously differentiable. In other words, the values of adjacent piecewise polynomials (and their derivatives up to the $(m-2)$-th derivative must match at the breakpoints. Before moving on to describe how b-splines can be used to estimate $f(\cdot)$, it is worth giving some justification for why one may want to assume that $f(\cdot)$ is a spline at all. In estimating $f(\cdot)$, we seek a function that will fit

the data well. One way to define how well the function fits the data is by the sum of squared errors (SSE) between the response values and the function values, given by

$$\sum_i \Big(y_i - f(x_i)\Big)^2 \ .$$

We also assume that $f(\cdot)$ is smooth however, and so of all the functions $f(\cdot)$ that achieve a low SSE, we would prefer the smoothest. One way of measuring smoothness is through the integral of the squared second derivative of a function

$$\int_x \Big(f''(x)\Big)^2 dx \ ,$$

where $f''(\cdot)$ denotes the second derivative of $f(\cdot)$. Functions that are very 'wiggly' change rapidly, and hence will have large in magnitude second derivatives. With this definition of smoothness, the search for the smoothest function $\hat{f}(\cdot)$ that fits the data well can be written as the variational optimisation problem

$$\hat{f}(x) = \arg\min_f \Big\{ \sum_i \Big(y_i - f(x_i)\Big)^2 + \lambda \int_x \Big(f''(x)\Big)^2 dx \Big\} \ , \tag{3.1}$$

where λ is a positive tuning parameter that controls the trade-off between achieving a low SSE and a smooth fit, assuming that $f(\cdot)$ belongs to the class of twice differentiable functions. It turns out that for any $\lambda > 0$ the solution to this optimisation problem is the natural cubic spline with a knot at every x_i. Natural cubic splines are cubic spline functions with additional constraints on their boundaries. Specifically, natural cubic splines must be linear on the first and last interval in the knot sequence they are defined on. For a proof of this result and an introduction to splines, see Green and Silverman (1994, Chapter 2). The integrated squared second derivative is not the only measure of roughness that could be used. Another option is the total variation of some derivative of the function. Intuitively, total variation measures the total distance travelled along the y-axis as a point traverses the surface of a function along the x-axis. For a differentiable function, the total variation is given by

$$\mathrm{TV}\Big(f(x)\Big) = \int_x |f'(x)| dx \ ,$$

where $f'(x)$ is the first derivative of $f(\cdot)$. For a step function, the total variation is the sum of the magnitude of all the steps

$$\mathrm{TV}\Big(f(x)\Big) = \sum_j \Big|f(x_{j+1}) - f(x_j)\Big| \ ,$$

where $\{x_j\}$ is a set of values containing at least one point in each individual flat region of the step function. Measured this way, seeking the smoothest $f(\cdot)$ that fits the data well would amount to finding

$$\hat{f}(x) = \arg\min_f \Big\{ \sum_i \Big(y_i - f(x_i)\Big)^2 + \lambda \mathrm{TV}(f^{[m-1]}(x)) \Big\} \ , \tag{3.2}$$

where $f^{[m-1]}(\cdot)$ denotes the $(m-1)$-th derivative of $f(\cdot)$, and λ is a tuning parameter that controls the trade-off between achieving a low total variation in the $(m-1)$-th derivative and fitting the data well. Eilers and Marx (1996) considered the problem in (3.2) and showed that once again the solution is a spline function, of order m. Interestingly though, the knots of this spline function are data adaptive and depend on the tuning parameter λ. As λ grows larger, the spline function that solves (3.2) has fewer knots, meaning that the total variation penalty performs knot selection, in a similar fashion to which the LASSO performs variable selection in linear regression. While the positions of the knots cannot be found analytically, Eilers and Marx (1996) propose an algorithm to estimate them.

3.3 B-splines

B-splines (short for basis splines) are a particular basis system for representing arbitrary spline functions. They are themselves spline functions, but of compact support, meaning that each b-spline is positive over at most m adjacent intervals in the knot sequence, where m is the order of the spline. This gives them attractive computational qualities when used as a basis system for arbitrary spline functions. For a sequence of knots $(t_1, \ldots, t_K)$, the j-th b-spline of order m can be computed recursively through the formula

$$B_j^m(x) = \omega_1 B_j^{m-1}(x) + \omega_2 B_{j+1}^{m-1}(x) \tag{3.3}$$

with

$$B_j^1(x) = \begin{cases} 1 & t_j \leq x < t_{j+1} \\ 0 & \text{otherwise} \end{cases} ,$$

where

$$\omega_1 = \begin{cases} \frac{x - t_j}{t_{j+m-1} - t_j} & \text{if } t_{j+m-1} > t_j \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \omega_2 = \begin{cases} \frac{t_{j+m} - x}{t_{j+m} - t_{j+1}} & \text{if } t_{j+m} > t_{j+1} \\ 0 & \text{otherwise} \end{cases} .$$

For a knot sequence $(t_1, \ldots, t_K)$, $K - m$ b-spline functions are defined, and an arbitrary spline function $S(x)$ can be defined on the interval (t_{m-1}, t_{K-m+1}) by

$$S(x) = \sum_{i=1}^{K-m} \alpha_j B_j^m(x) ,$$

for a coefficient vector $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_{K-m})^T$. The reason $S(x)$ is not defined over the full interval (t_1, t_K) is that on the first and last intervals, (t_1, t_2) and (t_{K-1}, t_K), only one b-spline is non-zero, meaning that if the spline function were evaluated over those intervals it would be restricted to take the specific (scaled) shape of those individual b-splines. Similarly, on the second and second to last intervals only two b-splines would be non-zero, meaning that the spline function would still be less flexible than than in the middle region, which is not desirable. At any point within the interval (t_{m-1}, t_{K-m+1}) however, m splines will be non-zero, giving the spline function the same flexibility (not taking into account position of knots) throughout. Hence, if one wants to define a spline function over a sequence of knots, in order to construct this spline using b-splines, the b-splines must be constructed with the same knot sequence, but appended by $m - 1$ knots on both sides. A common strategy is to make these appended knots replicates of the boundary knots. In general, when knots are duplicated (anywhere in the knot sequence), the effect is to lose a degree of continuous differentiability. In other words, while at all break points over a knot sequence with no duplicates splines are $m - 2$ times continuously differentiable, a spline will be only $m - 3$ times continuously differentiable at a knot value that is duplicated. If a knot value is replicated $m - 1$ times, the spline is no longer continuous at that point. See Appendix A for a worked-out example of this situation.

B-splines provide a practical and efficient way to estimate spline functions because they reduce a functional estimation problem to a linear modelling problem. Returning to the one dimensional smoothing problem, by assuming that $f(x)$ is a spline function, which by definition can be written as some linear combination of b-splines, we have

$$\begin{aligned} y_i &= f(x_i) + \epsilon_i \\ &= \sum_{j=1}^{d} \alpha_j B_j^m(x_i) + \epsilon_i , \end{aligned} \tag{3.4}$$

where m is the order of the spline function we have chosen to model $f(\cdot)$, and d is the number of b-splines used. Now note that (3.4) is just an ordinary linear regression model, with predictors $\big(B_1^m(x), \ldots, B_d^m(x)\big)$, and coefficients $\alpha_1, \ldots, \alpha_d$. To illustrate, we define the function

$$f(x) = \begin{cases} \sin(5\pi x/4) & \text{if } 0 \leq x \leq 0.4 \\ 1 & \text{if } 0.4 < x \leq 0.6 \\ \sin(5\pi(x-0.2)/4) & \text{if } 0.6 < x \leq 1 \end{cases} . \tag{3.5}$$

This function is used to simulate a data set $\{(y_i, x_i)\}_{i=1}^{100}$, with $x_i \sim \text{Unif}(0,1)$, $\epsilon_i \sim \mathcal{N}(0, 0.1)$, and

$$y_i = f(x_i) + \epsilon_i \ . \tag{3.6}$$

We now set up b-splines of order m on an equidistant set of knots set up such that we end up with 8 piecewise polynomials of degree $m-1$ between 0 and 1. We do this for $m \in \{1, \ldots, 4\}$. For each set of b-splines, we estimate the coefficients using ordinary least squares and plot the resulting splines in red, along with $f(x)$ and the observed data in black in Figure 3.1. The dashed vertical lines are the cut-off points separating the piecewise polynomials. In this example the individual piecewise polynomials can be observed with the naked eye, especially in the case of $m = 1$ and $m = 2$, where the individual functions are constant and linear respectively. Modelling with b-splines in this way immediately raises the question of how to choose the number and positions of the knots. Using too many knots can result in overfitting, as the region that each piecewise function will correspond to may not have many data points, and consequently the individual functions may adapt to noise rather than signal. If too few knots are chosen on the other hand, the resulting spline function may not have enough flexibility to model the true function accurately. Furthermore, if the true function to be estimated has high curvature in some regions and low curvature in others, both of these problems can be inherited simultaneously if the knot positioning is not chosen carefully; in some regions there may be too many knots and in others too few. There is an extensive literature with a variety of approaches to deal with this problem. Dung and Tjahjowidodo (2017) provide a recent overview that covers both heuristic and optimisation based techniques. As will see in the following sections however regularisation techniques can be used to approach this issue.

3.4 Splines and ℓ_2 regularisation

One way to deal with the problem of selecting the number of knots is to select a large number, that when fitted using ordinary least squares would result in overfitting, however to add a penalty that penalises fits that are not smooth. One of the convenient computational qualities of b-splines is that the roughness penalty

$$\int_x \Big(f''(x)\Big)^2 dx \ ,$$

where $f(x) = \sum_j \alpha_j B_j^4(x)$ is a cubic spline function represented as a linear combination of b-splines, can be expressed as a quadratic form $\boldsymbol{\alpha}^T S \boldsymbol{\alpha}$, where $S_{lk} = \int_x \frac{\partial^2}{\partial x^2} B_l^4(x) \frac{\partial^2}{\partial x^2} B_k^4(x) dx$. Cubic splines are often used due to their theoretical justification as the solution to the variational problem in (3.1). The derivatives of b-splines can be computed analytically and because b-splines are of compact support, the matrix S will be banded, and consequently computationally efficient to deal with. Using a cubic spline, the solution to the smoothing problem can be written as

$$\hat{f}(x) = \sum_j \hat{\alpha}_j B_j^4(x) \ ,$$

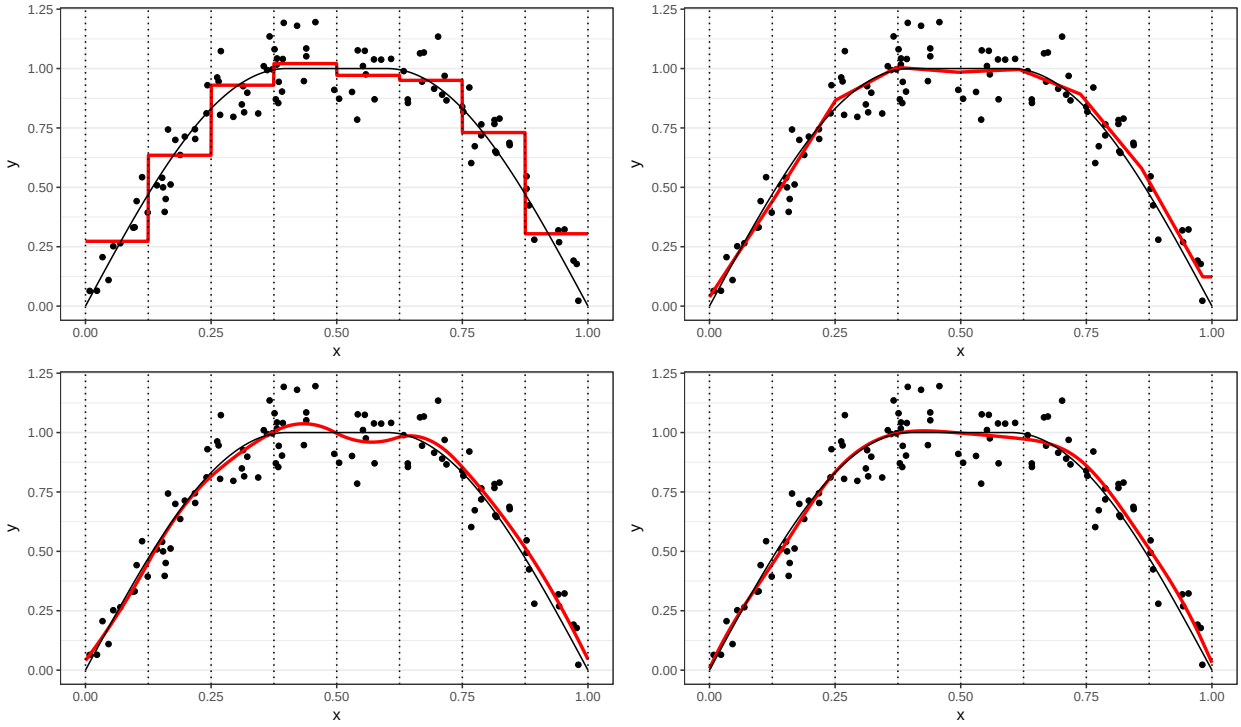

Figure 3.1: Fitted functions from model (3.6) using b-splines of order 1 (top left), 2 (top right), 3 (bottom left), and 4 (bottom right). Fitted functions are in red, the true function, expressed in equation (3.5) is in black, while the points are the simulated data used for estimation. The coefficients of the b-splines are estimated using ordinary least squares.

where

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_j \alpha_j B_j^4(x) \right)^2 + \lambda \boldsymbol{\alpha}^T S \boldsymbol{\alpha} \right\} ,$$

which is a standard ℓ_2 regularisation problem. This is the roughness penalty approach to smoothing (Green and Silverman, 1994, Chapters 2,3). Eilers and Marx (1996) proposed a different penalty to achieve smoothness in spline fits, penalising various order differences of successive coefficients of b-splines. If we let

$$D_p = \begin{pmatrix} 1 & 0 & 0 & \dots & 0 \\ 1 & -1 & 0 & \dots & 0 \\ 0 & 1 & -1 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 1 & -1 \end{pmatrix}$$

be the $p \times p$ first order differencing matrix, where p is the number of b-splines used, then the last $p-1$ elements of the vector $D_p\boldsymbol{\alpha}$ are the first order differences of $\boldsymbol{\alpha}$. By extension, D_p^l is the l-th order differencing matrix and the last $p-l$ elements of $D_p^l\boldsymbol{\alpha}$ represent the l-th order differences of $\boldsymbol{\alpha}$. Eilers and Marx (1996) propose as an estimator

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left\{ \sum_{i=1}^{n} \left(y_i - \sum_j \alpha_j B_j^m(x) \right)^2 + \lambda \sum_{k=l+1}^{p} (D_p^l \boldsymbol{\alpha})_k^2 \right\} ,$$

where $(D_p^l\boldsymbol{\alpha})_k$ is the k-th elements of $D_p^l\boldsymbol{\alpha}$, and the order m of the spline function and order l of differencing can be chosen freely. Splines estimated in this way are often referred to as p-splines. To see the effect of regularisation using both approaches we can return to our example of smoothing once again, this time choosing 40 equidistant knots and cubic splines. We estimate the coefficients without regularisation, with the integrated squared second derivative penalty, and with a second order differencing penalty. In the regularised cases the tuning parameter is chosen by maximum likelihood under the Bayesian formulation discussed in Section 2.3. The fits are plotted in Figure 3.2. The fit without any regularisation results in a very 'wiggly' function, as between each successive pair of knots the polynomial adapts to noise. In the regularised fits however a balance is found between fitting the data while also retaining a smooth fit, resulting in better estimates of the function. In this example the two regularised fits are indistinguishable to the naked eye.

3.5 Splines and ℓ_1 regularisation

ℓ_2 regularisation deals with the problem of selecting the number of knots by allowing one to choose a large number of knots while controlling the overall level of smoothness in the fit. While this approach may often perform well, the issue remains that the tuning parameter that is selected pertains to the entire function. In cases where a function is more smooth in some regions than others, controlling the smoothness with one parameter may not be adequate, as the resulting estimated function may be too smooth in some regions and not smooth enough in others. Mammen and Van De Geer (1997) addressed this issue by considering a total variation penalty on the $(m-1)$-th derivative of the function to be estimated, rather than the integrated squared second derivative. They showed that this penalty results in a spline with knots positioned in a way such that they adapt to local levels of smoothness. While Mammen and Van De Geer (1997) went on to

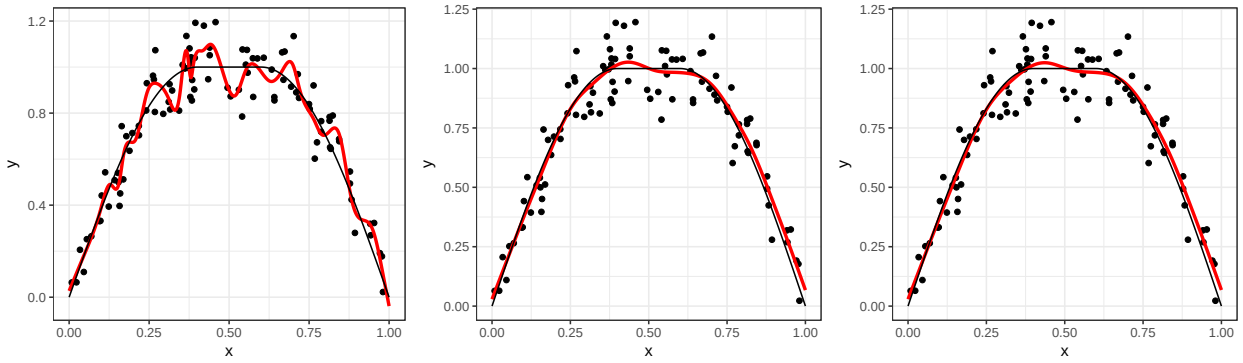

Figure 3.2: Fitted functions from model (3.6) using cubic b-splines with 40 knots. The fitted functions are in red, the true function, expressed in equation (3.5) is in black, while the points are the simulated data used for estimation. The left panel has no regularisation, the middle panel uses the integrated squared second derivative roughness penalty, and the right panel penalises squared second order differences of b-spline coefficients. Tuning parameters are selected by maximum marginal likelihood under the Bayesian interpretation of ℓ_2 regularisation, discussed in Section 2.3.

propose an algorithm that adds and deletes knots in a step-wise manor, it turns out that when using b-splines the total variation penalty can be expressed as an ℓ_1 regularisation problem. Because the derivatives of splines are themselves splines of lower order, the $(m-1)^{\text{th}}$ derivative of a spline of order m is just a step function, whose total variation is the sum of the magnitude of the steps. When using b-splines, the total variation of the m^{th} derivative of a spline can be expressed in closed form. See Jhong et al. (2017) for the closed form expressions for $m = 0, 1, 3$. In the case of equally spaced knots the expressions simplify, and the total variation of the $(m-1)^{\text{th}}$ derivative is just

$$c\|D_p^m \boldsymbol{\alpha}\|_1 \ ,$$

where c is a constant and $\boldsymbol{\alpha}$ are the coefficients of the b-splines used to represent the spline. Consequently Jhong et al. (2017) consider estimating b-spline coefficients as

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left\{ \sum_i \frac{1}{2}\Big(y_i - \sum_j \alpha_j B_j^m(x)\Big)^2 + \lambda \|D_p^m \boldsymbol{\alpha}\|_1 \right\} \ ,$$

where the constant c has been absorbed by the tuning parameter λ. This method resembles the approach of Eilers and Marx (1996), where the squared ℓ_2 norm is replaced with an ℓ_1 norm, and the order of differencing is the same as the order of the spline.

A closely related procedure to that of Jhong et al. (2017) is known as trend filtering (Kim et al., 2009; Tibshirani, 2014), in which exclusively b-splines of order 1 are used (i.e. a grid basis), with various order differences of the coefficients penalised. The procedure can be expressed as

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left\{ \sum_i \frac{1}{2}\Big(y_i - \sum_j \alpha_j B_j^1(x)\Big)^2 + \lambda \|D_p^m \boldsymbol{\alpha}\|_1 \right\} \ .$$

In the case of $m = 1$, trend filtering is equivalent to the approach of Jhong et al. (2017). For $m > 1$ the result is a piecewise constant function which can be thought of as a discrete approximation to a spline of order m, with adaptively placed knots.

3.6 Generalized additive models

Generalized additive models (GAMs) were introduced by Hastie and Tibshirani (1986) to extend one dimensional smoothing to multiple predictors. For an exponential family response variable the model can be written as

$$g\Big(E(y_i)\Big) = \sum_{j=1}^{p} f_j(x_{ij}) \quad (i = 1, \ldots, n) \ , \tag{3.7}$$

where $f_1, \ldots, f_p$ are arbitrary smooth functions, as in the one dimensional smoothing case. GLMs can be seen as a special case of GAMs in which $f_j(x_j) = \beta_j x_j$, essentially restricting each function to be a straight line with a free slope parameter. As introduced in Hastie and Tibshirani (1986), GAMs can be estimated using any method that performs one dimensional smoothing due to the backfitting algorithm, which consists of iteratively estimating one function at a time, while holding all the others constant. Nowadays, much attention has focused on reducing GAMs to regularised GLMs by using splines to model each $f_j(\cdot)$ and penalisation to enforce smoothness. Using m-th order b-splines to model each $f_j(\cdot)$, the model in (3.7) can succinctly be written as

$$g\Big(E(y_i)\Big) = \sum_{j=1}^{p} \boldsymbol{\alpha}_j^T \boldsymbol{b}_{ij}^m \ , \tag{3.8}$$

where $\boldsymbol{b}_{ij}^m$ is a vector with k-th element $B_k^m(x_{ij})$ and $\boldsymbol{\alpha}_j$ is a vector of coefficients that determines the shape of the j-th spline function. Using an integrated squared second derivative roughness penalty on each function $f_j(\cdot)$, estimation can be reduced to

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left\{ -\ell(\boldsymbol{\alpha}) + \sum_{j=1}^{p} \lambda_j \boldsymbol{\alpha}_j^T S_j \boldsymbol{\alpha}_j \right\} \ , \tag{3.9}$$

where $\boldsymbol{\alpha}$ collects all the vectors $\boldsymbol{\alpha}_j$ into one, S_j is a penalty matrix such that $\boldsymbol{\alpha}_j^T S_j \boldsymbol{\alpha}_j$ is the integrated squared second derivative of $f_j(x_j)$, and $-\ell(\boldsymbol{\alpha})$ is the negative log-likelihood for the observed data. For a detailed overview of this approach to estimating GAMs see Wood (2006), which also covers efficient ways to select the tuning parameters using GCV. For a description of how to select the tuning parameters using marginal likelihood under the Bayesian interpretation of (3.9) see Wood (2011).

Other authors have considered using a grid basis (i.e. b-splines of order 1) along with an ℓ_1 norm penalty on m^{th} order differences of the b-spline coefficients to estimate GAMs. Petersen et al. (2014) introduce the 'fused LASSO additive model' which consists of a grid basis for each function and an ℓ_1 norm on the first order differences of the coefficients of each grid basis. Sadhanala and Tibshirani (2018) introduce 'additive models with trend filtering', which is similar to the fused LASSO additive model but with m^{th} order differences of the coefficients of each grid basis. In this case the b-spline coefficient estimates are given by

$$\hat{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{\alpha}} \left\{ -\ell(\boldsymbol{\alpha}) + \sum_{j=1}^{p} \lambda_j \| D^m \boldsymbol{\alpha}_j \| \right\} \ , \tag{3.10}$$

where D^m is the m^{th} order differencing matrix. The rationale for these approaches is the same as in the one dimensional case, however there is a significant limitation when extending these methods to multiple predictors, and that is tuning parameter selection. While it has proven feasible to estimate multiple tuning parameters in the case of ℓ_2 regularisation, to the best of our knowledge estimating multiple tuning parameters with ℓ_1 regularisation remains a challenge without adequate solutions.

As a final note on GAMs we mention that with regard to estimation each function is typically identifiable only up to an additive constant. To see this, suppose we expressed a GAM as

$$g\Big(E(y_i)\Big) = \sum_{j=1}^{p} \big(f_j(x_{ij}) + c_j\big) \ ,$$

for some constants $c_1, \ldots, c_p$. For any configuration of constants that results in the same sum $\sum_{j=1}^{p} c_j = C$, the likelihood for the observed data would be the same, as the mean and variance of the responses would be identical. Consequently, only the global constant C is estimable. A typical approach to deal with this issue is to formulate GAMs with an intercept and 'centering' constraints on the functions. In this case, a GAM can be written as

$$\begin{aligned} g\Big(E(y_i)\Big) &= \beta_0 + \sum_{j=1}^{p} f_j(x_{ij}) \\ \sum_{i=1}^{n} f_j(x_{ij}) &= 0 \ , \end{aligned}$$

where β_0 is the estimable global intercept, and the constraints $\sum_{i=1}^{n} f_j(x_{ij}) = 0$ force each function to be centred around zero. For details on how the constraints can be implemented in practice when using splines to represent the functions, see Wood (2006, page 163).

Chapter 4

A new approach to estimating locally adaptive splines

4.1 Preamble

In this Chapter we discuss a sparsity inducing method called adaptive sparseness (Figueiredo, 2003), and apply it to estimation of GAMs. Adaptive sparseness was inspired by the LASSO but unlike the LASSO does not rely on tuning parameters. Our contributions are two-fold. Firstly, we provide some extensions for adaptive sparseness, as well as an efficient implementation of the EM algorithm used to obtain its parameter estimates. Secondly, we apply adaptive sparseness as an estimation procedure for GAMs, demonstrating its potential to estimate smooth functions in a locally adaptive way, without the need for tuning parameters, resulting in an efficient procedure that scales well with the number of predictors.

We begin by introducing adaptive sparseness in Section 4.2. In Section 4.3.1 we show that the adaptive sparseness coefficient estimates can be expressed in closed form, providing some intuition regarding how the method works. We then discuss some simple extensions to adaptive sparseness in Section 4.4. We provide an efficient implementation of the EM algorithm for adaptive sparseness in Section 4.5, and show how it can be used to estimated GAMs in Section 4.6. A related approach that was developed in parallel to ours is discussed in Section 4.7. In Section 4.8 we carry out simulation studies to examine the performance of a variety of estimation procedures for GAMs, comparing existing methods to our proposed method. Finally, in Section 4.9 we apply the proposed methodology to a real data set in which the age of abalones is predicted using various physical measurements of the abalones.

4.2 Adaptive sparseness

Figueiredo (2003) considers an alternative Bayesian formulation of the LASSO which in turn inspires a different sparsity inducing estimation technique that remarkably does not rely on any tuning parameters. The Laplace distribution can be written in a hierarchical fashion as an exponential mixture of normals. For

a normal response the Bayesian LASSO (Park and Casella, 2008) can be written as

$$\begin{aligned} p(\sigma^2) &\propto c \\ \tau_j &\sim \text{Exp}\Big(0, \frac{\gamma}{2}\Big) \\ \beta_j|\tau_j &\sim \mathcal{N}(0, \tau_j) \\ y_i|\boldsymbol{\beta}, \boldsymbol{x}_i, \sigma^2 &\sim \mathcal{N}\Big(\sum_{j=1}^{p} \beta_j x_{ij}, \sigma^2\Big) \ . \end{aligned} \tag{4.1}$$

Before introducing adaptive sparseness, it is worth noting that the formulation in (4.1) demonstrates an interesting conceptual link between ridge regression and the LASSO. The LASSO can be seen as a kind of adaptive ridge regression, in which instead of placing the same normal prior on each coefficient β_j, each coefficient instead has a normal prior with its own variance parameter, and all the variance parameters are drawn from an exponential distribution. The 'adaptivity' results from the fact that different coefficients receive differential shrinkage. Figueiredo (2003) takes adaptivity one step further and achieves two important goals simultaneously, considering the following formulation,

$$\begin{aligned} p(\sigma^2) &\propto c \\ p(\tau_j) &\propto 1/\tau_j \\ \beta_j|\tau_j &\sim \mathcal{N}(0, \tau_j) \\ y_i|\boldsymbol{\beta}, \boldsymbol{x}_i, \sigma^2 &\sim \mathcal{N}\Big(\sum_{j=1}^{p} \beta_j x_{ij}, \sigma^2\Big) \ , \end{aligned} \tag{4.2}$$

where the common prior on all the variance parameters τ_i in (4.1) has been replaced by individual, non-informative priors $p(\tau_j) \propto 1/\tau_j$. Firstly, this introduces more adaptivity because the variance parameters are no longer drawn from the same distribution, further decoupling the shrinkage that each coefficient receives relative to the others. Secondly, the hyperparameter γ has now been removed from the model, and hence does not have to be estimated. To estimate the coefficient vector $\boldsymbol{\beta}$ and noise variance σ^2, Figueiredo (2003) proposes the expectation conditional maximization (ECM) algorithm (Meng and Rubin, 1993) to obtain the maximum a posterior (MAP) estimate $p(\boldsymbol{\beta}, \sigma^2|\boldsymbol{y})$, treating the variance parameters $\tau_1, \ldots, \tau_p$ as missing data. The ECM algorithm is a generalisation of the expectation maximization (EM) algorithm (Dempster et al., 1977), which is a popular algorithm to estimate parameters in the presence of missing data. Specifically, if the variance parameters were known, then we would have

$$\begin{aligned} (\hat{\boldsymbol{\beta}}, \hat{\sigma}^2) &= \arg\max_{\boldsymbol{\beta}, \sigma^2} p(\boldsymbol{\beta}, \sigma^2|\boldsymbol{y}, \boldsymbol{\tau}) \\ &= \arg\max_{\boldsymbol{\beta}, \sigma^2} \big\{p(\boldsymbol{y}|\boldsymbol{\beta}, \sigma^2) p(\boldsymbol{\beta}|\boldsymbol{\tau})\big\} \\ &= \arg\max_{\boldsymbol{\beta}, \sigma^2} \big\{\log p(\boldsymbol{y}|\boldsymbol{\beta}, \sigma^2) + \log p(\boldsymbol{\beta}|\boldsymbol{\tau})\big\} \\ &= \arg\max_{\boldsymbol{\beta}, \sigma^2} \left\{-n \log \sigma^2 - \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} - \boldsymbol{\beta}^T W \boldsymbol{\beta}\right\} \ . \end{aligned} \tag{4.3}$$

where W is a diagonal matrix with $W_{jj} = 1/\tau_j$. Defining

$$L(\boldsymbol{\beta}, \sigma^2) = -n \log \sigma^2 - \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T (\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} - \boldsymbol{\beta}^T W \boldsymbol{\beta} \ ,$$

which is the function to be optimised in the last line of equation (4.3), the EM algorithm iterates between computing

1. $Q(\boldsymbol{\beta}, \sigma^2|\hat{\boldsymbol{\beta}}) = E_{\boldsymbol{\tau}|\hat{\boldsymbol{\beta}}}\Big(L(\boldsymbol{\beta}, \sigma^2)\Big)$
$$= E_{\boldsymbol{\tau}|\hat{\boldsymbol{\beta}}}\Big(- n\log\sigma^2 - \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} - \boldsymbol{\beta}^T W\boldsymbol{\beta}\Big)$$
$$= -n\log\sigma^2 - \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} - \sum_{j=1}^{p} E(\tau_j^{-1}|\hat{\beta}_j)\beta_j^2$$
2. $(\hat{\boldsymbol{\beta}}, \hat{\sigma}^2) = \arg\max_{\boldsymbol{\beta},\sigma^2} Q(\boldsymbol{\beta}, \sigma^2)$

The conditional expectation $E(\tau_j^{-1}|\hat{\beta}_j)$ can be found in close form since

$$\begin{aligned} E(\tau_j^{-1}|\hat{\beta}_j) &= \int_0^\infty \frac{1}{\tau_j} p(\tau_j|\beta_j = \hat{\beta}_j) d\tau_j \\ &= \frac{\int_0^\infty \frac{1}{\tau_j} p(\beta_j = \hat{\beta}_j|\tau_j)p(\tau_j)d\tau_j}{\int_0^\infty p(\beta_j = \hat{\beta}_j|\tau_j)p(\tau_j)d\tau_j} \\ &= \frac{\int_0^\infty \frac{1}{\tau_j^2} p(\beta_j = \hat{\beta}_j|\tau_j)d\tau_j}{\int_0^\infty \frac{1}{\tau_j} p(\beta_j = \hat{\beta}_j|\tau_j)d\tau_j} \\ &= \frac{1}{\hat{\beta}_j^2} \,. \end{aligned}$$

The optimisation $\arg\max_{\boldsymbol{\beta},\sigma^2} Q(\boldsymbol{\beta}, \sigma^2)$ cannot be computed in closed form, however it can be modified to consist of two conditional optimisation steps, resulting in the ECM algorithm, which alternates between

1. $Q(\boldsymbol{\beta}, \sigma^2|\hat{\boldsymbol{\beta}}) = -n\log\sigma^2 - \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} - \sum_{j=1}^{p} \frac{\beta_j^2}{\hat{\beta}_j^2}$
2. $\hat{\sigma}^2 = \arg\max_{\sigma^2} Q(\hat{\boldsymbol{\beta}}, \sigma^2|\hat{\boldsymbol{\beta}})$
$$= \frac{(\boldsymbol{y} - X\hat{\boldsymbol{\beta}})^T(\boldsymbol{y} - X\hat{\boldsymbol{\beta}})}{n}$$
3. $\hat{\boldsymbol{\beta}} = \arg\max_{\boldsymbol{\beta}} Q(\boldsymbol{\beta}, \hat{\sigma}^2|\hat{\boldsymbol{\beta}})$
$$= (X^T X + W)^{-1} X^T \boldsymbol{y} \,,$$

where W is a now a diagonal matrix with $W_{jj} = \frac{\hat{\sigma}^2}{\hat{\beta}_j^2}$. As we show in Section 4.3.1, the adaptive sparseness method is sparsity inducing. This implies that some of the weights W_{jj} may approach infinity, which in practice will lead to numerical problems when implementing the ECM algorithm. This can be dealt with by adding a very small regulariser ϵ to the denominator of the weights, resulting in weights $W_{jj} = \frac{\hat{\sigma}^2}{\hat{\beta}_j^2 + \epsilon}$. This results in a maximum value that the weights can take of $\hat{\sigma}^2/\epsilon$.

4.3 Properties of adaptive sparseness

4.3.1 Expression for adaptive sparseness coefficient estimates

To gain some intuition regarding the behaviour of the adaptive sparseness method, and why in fact it is sparsity inducing, it is illustrative to consider the simplified case in which the noise variance σ^2 is known,

and there is only one predictor. In this case, it turns out that the solution reached by the EM algorithm can be found in closed form. Specifically, at convergence, the solution must satisfy

$$\begin{aligned}\hat{\beta} &= \frac{\boldsymbol{x}^T\boldsymbol{y}}{\boldsymbol{x}^T\boldsymbol{x} + \hat{\tau}^{-1}\sigma^2} \\ \hat{\tau}^{-1} &= \frac{1}{\hat{\beta}^2} \ ,\end{aligned}$$

which can be re-written as

$$\hat{\beta} = \frac{\boldsymbol{x}^T\boldsymbol{y}}{\boldsymbol{x}^T\boldsymbol{x} + \sigma^2/\hat{\beta}^2}. \tag{4.4}$$

Assuming for now that $\hat{\beta} \neq 0$, equation (4.4) can be re-written as a quadratic equation with solutions

$$\hat{\beta} = \frac{\boldsymbol{x}^T\boldsymbol{y} \pm \sqrt{(\boldsymbol{x}^T\boldsymbol{y})^2 - 4\sigma^2\boldsymbol{x}^T\boldsymbol{x}}}{2\boldsymbol{x}^T\boldsymbol{x}} \ . \tag{4.5}$$

It remains to be decided which of the two solutions should be kept, and under which conditions the solution may be $\hat{\beta} = 0$. To answer these questions it is useful to shift focus to the solution found in terms of $\hat{\tau}^{-1}$, and examine the behavior of the EM algorithm. Let

$$\begin{aligned}f(\hat{\tau}^{-1}) &= \frac{1}{\hat{\beta}^2} \\ &= \left(\frac{\boldsymbol{x}^T\boldsymbol{x} + \hat{\tau}^{-1}\sigma^2}{\boldsymbol{x}^T\boldsymbol{y}}\right)^2 \ .\end{aligned}$$

The EM algorithm computes the sequence $\hat{\tau}_{k+1}^{-1} = f(\hat{\tau}_k^{-1})$, with $\hat{\tau}_0^{-1}$ set to be the starting value. At convergence, we must have $f(\hat{\tau}^{-1}) = \hat{\tau}^{-1}$. If the discriminant of $f(\hat{\tau}^{-1}) - \hat{\tau}^{-1}$ is negative,

$$f(\hat{\tau}^{-1}) > \hat{\tau}^{-1} \ \ \forall \ \hat{\tau}^{-1} > 0 \ ,$$

and the sequence $\{\hat{\tau}_k^{-1}\}_{k=1}^{\infty}$ diverges to infinity, giving final solution $\hat{\beta} = 0$. The discriminant is negative when $|\boldsymbol{x}^T\boldsymbol{y}| < 2\sigma\sqrt{\boldsymbol{x}^T\boldsymbol{x}}$. When the discriminant of $f(\hat{\tau}^{-1}) - \hat{\tau}^{-1}$ is positive, the quadratic equation $f(\hat{\tau}^{-1}) - \hat{\tau}^{-1} = 0$ has roots

$$\begin{aligned}\tau_+^{-1} &= (\boldsymbol{x}^T\boldsymbol{y})^2 - 2\sigma^2\boldsymbol{x}^T\boldsymbol{x} + \sqrt{(2\sigma^2\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y})^2 - 4\sigma^4(\boldsymbol{x}^T\boldsymbol{x})^2} \\ \tau_-^{-1} &= (\boldsymbol{x}^T\boldsymbol{y})^2 - 2\sigma^2\boldsymbol{x}^T\boldsymbol{x} - \sqrt{(2\sigma^2\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{x}^T\boldsymbol{y})^2 - 4\sigma^4(\boldsymbol{x}^T\boldsymbol{x})^2} \ ,\end{aligned}$$

and consequently it follows that

$$\begin{aligned}\hat{\tau}^{-1} < f(\hat{\tau}^{-1}) < \tau_-^{-1} \quad &\text{for} \quad 0 < \hat{\tau}^{-1} < \tau_-^{-1} \\ \tau_-^{-1} < f(\hat{\tau}^{-1}) < \hat{\tau}^{-1} \quad &\text{for} \quad \tau_-^{-1} < \hat{\tau}^{-1} < \tau_+^{-1} \\ \hat{\tau}^{-1} < f(\hat{\tau}^{-1}) \quad &\text{for} \quad \tau_+^{-1} < \hat{\tau}^{-1} \ .\end{aligned} \tag{4.6}$$

Because $f(\hat{\tau}^{-1})$ is positive and monotonically increasing, setting $\hat{\tau}^{-1} = \hat{\tau}_k^{-1}$ in the inequalities in (4.6) gives

$$\begin{aligned}\hat{\tau}_k^{-1} < \hat{\tau}_{k+1}^{-1} < \tau_-^{-1} \quad &\text{for} \quad 0 < \hat{\tau}_k^{-1} < \tau_-^{-1} \\ \tau_-^{-1} < \hat{\tau}_{k+1}^{-1} < \hat{\tau}_k^{-1} \quad &\text{for} \quad \tau_-^{-1} < \hat{\tau}_k^{-1} < \tau_+^{-1} \\ \hat{\tau}_k^{-1} < \hat{\tau}_{k+1}^{-1} \quad &\text{for} \quad \tau_+^{-1} < \hat{\tau}_k^{-1} \ .\end{aligned} \tag{4.7}$$

The inequalities in (4.7) in turn imply that as long as $\hat{\tau}_0^{-1} < \tau_+^{-1}$, the sequence $\{\hat{\tau}_k^{-1}\}_{k=1}^{\infty}$ will converge to the solution τ_-^{-1}, while if $\hat{\tau}_0^{-1} > \tau_+^{-1}$, the sequence $\{\hat{\tau}_k^{-1}\}_{k=1}^{\infty}$ will diverge to infinity. To find which solution $\hat{\beta}$ in equation (4.5) corresponds to τ_-^{-1}, it is sufficient to observe that the solution corresponding to τ_-^{-1} must be larger in magnitude compared to the solution corresponding to τ_+^{-1}, and that the solution with larger magnitude is

$$\hat{\beta} = \frac{\boldsymbol{x}^T\boldsymbol{y} + \text{sgn}(\boldsymbol{x}^T\boldsymbol{y})\sqrt{(\boldsymbol{x}^T\boldsymbol{y})^2 - 4\sigma^2\boldsymbol{x}^T\boldsymbol{x}}}{2\boldsymbol{x}^T\boldsymbol{x}} \ , \tag{4.8}$$

where $\text{sgn}(\boldsymbol{x}^T\boldsymbol{y})$ is the sign of $\boldsymbol{x}^T\boldsymbol{y}$. From a practical perspective setting $\hat{\tau}_0^{-1} > \tau_+^{-1}$ would not make sense as it would effectively be pre-determining the outcome of the algorithm to be $\hat{\beta} = 0$ by setting the starting value $\hat{\beta}^0$ so close to zero as to not allow it to escape. Consequently, we can conclude that when $|\boldsymbol{x}^T\boldsymbol{y}| > 2\sigma\sqrt{\boldsymbol{x}^T\boldsymbol{x}}$, the solution of interest is the value in (4.8), while when $|\boldsymbol{x}^T\boldsymbol{y}| < 2\sigma\sqrt{\boldsymbol{x}^T\boldsymbol{x}}$, the only solution is $\hat{\beta} = 0$.

With multiple predictors, we can still express the solution for each coefficient in closed form as a function of the solution reached for the other coefficients. Specifically, if the solution was known for all the coefficients except the j-th, then for the j-th coefficient we would have

$$\begin{aligned}\hat{\beta}_j &= \arg\min_{\boldsymbol{\beta}} \left\{ n\log\sigma^2 + \frac{(\boldsymbol{z}_j - \boldsymbol{x}_j\beta_j)^T(\boldsymbol{z}_j - \boldsymbol{x}_j\beta_j)}{\sigma^2} + \frac{\sigma^2\beta_j^2}{\hat{\beta}_j^2} \right\} \\ &= \frac{\boldsymbol{x}_j^T\boldsymbol{z}_j}{\boldsymbol{x}_j^T\boldsymbol{x}_j + \sigma^2/\hat{\beta}_j^2} \ ,\end{aligned} \tag{4.9}$$

where $\boldsymbol{z}_j = \boldsymbol{y} - X_{-j}\hat{\boldsymbol{\beta}}_{-j}$, $\boldsymbol{x}_j$ is the j-th column of X, X_{-j} is X with the j-th column removed, and $\hat{\boldsymbol{\beta}}_{-j}$ is $\hat{\boldsymbol{\beta}}$ with the j-th element removed. The last line of (4.9) is identical to the expression in (4.4) with $\boldsymbol{y}$ replaced by $\boldsymbol{z}_j$, and so we can therefore write $\hat{\beta}_j = 0$ if $|\boldsymbol{x}_j^T\boldsymbol{z}_j| < 2\sigma\sqrt{\boldsymbol{x}_j^T\boldsymbol{x}_j}$, and

$$\frac{\boldsymbol{x}_j^T\boldsymbol{z}_j + \text{sgn}(\boldsymbol{x}_j^T\boldsymbol{z}_j)\sqrt{(\boldsymbol{x}_j^T\boldsymbol{z}_j)^2 - 4\sigma^2\boldsymbol{x}_j^T\boldsymbol{x}_j}}{2\boldsymbol{x}_j^T\boldsymbol{x}_j} \tag{4.10}$$

otherwise.

4.3.2 Scale invariance

Expression (4.8) uncovers an appealing property of the coefficient estimates produced by adaptive sparseness. Suppose, in the case of one predictor $\boldsymbol{x}$, that we were to change its units resulting in a new vector $\tilde{\boldsymbol{x}} = c\boldsymbol{x}$. A natural question that arises in regression problems in which regularisation is used is how the solution responds to changes in scale of the predictors. Firstly, we examine whether the condition for setting a coefficient to zero changes if its scale changes. From Section 4.3.1 we have that $\hat{\beta} = 0$ if $|\boldsymbol{x}^T\boldsymbol{y}| > 2\sigma\sqrt{\boldsymbol{x}^T\boldsymbol{x}}$. Similarly, the coefficient estimate $\tilde{\beta}$, corresponding to the predictor $\tilde{\boldsymbol{x}}$ will be set to zero if

$$\begin{aligned}&|\tilde{\boldsymbol{x}}^T\boldsymbol{y}| > 2\sigma\sqrt{\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{x}}} \\ \iff &|c\boldsymbol{x}^T\boldsymbol{y}| > 2\sigma\sqrt{c^2\boldsymbol{x}^T\boldsymbol{x}} \\ \iff &|\boldsymbol{x}^T\boldsymbol{y}| > 2\sigma\sqrt{\boldsymbol{x}^T\boldsymbol{x}} \ ,\end{aligned}$$

from which we conclude that the scale of the predictor has no effect on the condition under which its coefficient is estimated as zero. For the estimate $\tilde{\beta}$ when it is not equal to zero, we have

$$\begin{aligned}
\tilde{\beta} &= \frac{\tilde{\boldsymbol{x}}^T\boldsymbol{y} + \text{sgn}(\tilde{\boldsymbol{x}}^T\boldsymbol{y})\sqrt{(\tilde{\boldsymbol{x}}^T\boldsymbol{y})^2 - 4\sigma^2\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{x}}}}{2\tilde{\boldsymbol{x}}^T\tilde{\boldsymbol{x}}} \\
&= \frac{c\boldsymbol{x}^T\boldsymbol{y} + \text{sgn}(c\boldsymbol{x}^T\boldsymbol{y})\sqrt{c^2(\boldsymbol{x}^T\boldsymbol{y})^2 - 4c^2\sigma^2\boldsymbol{x}^T\boldsymbol{x}}}{2c^2\boldsymbol{x}^T\boldsymbol{x}} \\
&= \frac{c\boldsymbol{x}^T\boldsymbol{y} + |c| \times \text{sgn}(c) \times \text{sgn}(\boldsymbol{x}^T\boldsymbol{y})\sqrt{(\boldsymbol{x}^T\boldsymbol{y})^2 - 4\sigma^2\boldsymbol{x}^T\boldsymbol{x}}}{2c^2\boldsymbol{x}^T\boldsymbol{x}} \\
&= \frac{c\boldsymbol{x}^T\boldsymbol{y} + c \times \text{sgn}(\boldsymbol{x}^T\boldsymbol{y})\sqrt{(\boldsymbol{x}^T\boldsymbol{y})^2 - 4\sigma^2\boldsymbol{x}^T\boldsymbol{x}}}{2c^2\boldsymbol{x}^T\boldsymbol{x}} \\
&= \hat{\beta}/c \ ,
\end{aligned}$$

implying that $\boldsymbol{x}\hat{\beta} = \tilde{\boldsymbol{x}}\tilde{\beta}$. Consequently, standardising predictors is not necessary when using adaptive sparseness to estimate regression coefficients.

4.3.3 Probability of estimating coefficients as zero

Finally, we consider from a frequentist perspective, in which we consider coefficients fixed, the probability of estimating any coefficient as zero. Unfortunately, this cannot be found in closed form for general design matrices, but in the special case of an orthogonal design this probability is very simple to compute and takes an intuitive form. Specifically, as we saw in Section 4.3.1, with an orthogonal design matrix, a coefficient is estimated as zero if

$$\begin{aligned}
&(\boldsymbol{x}_j^T\boldsymbol{y})^2 < 4\sigma^2\boldsymbol{x}_j^T\boldsymbol{x}_j \\
\iff &\left(\frac{\boldsymbol{x}_j^T\boldsymbol{y}}{\boldsymbol{x}_j^T\boldsymbol{x}_j}\right)^2 < \frac{4\sigma^2}{\boldsymbol{x}_j^T\boldsymbol{x}_j} \\
\iff &\left|\frac{\boldsymbol{x}_j^T\boldsymbol{y}}{\boldsymbol{x}_j^T\boldsymbol{x}_j}\right| < \frac{2\sigma}{\sqrt{\boldsymbol{x}_j^T\boldsymbol{x}_j}} \\
\iff &\left|\frac{\hat{\beta}_j^{\text{ols}}}{\text{sd}(\hat{\beta}_j^{\text{ols}})}\right| < 2 \ ,
\end{aligned}$$

where $\hat{\beta}_j^{\text{ols}}$ denotes the ordinary least squares estimate of β_j, which is normally distributed with mean β_j, and for orthogonal designs has standard deviation $\sigma/\sqrt{\boldsymbol{x}_j^T\boldsymbol{x}_j}$. If $\beta_j = 0$, we have

$$p(\hat{\beta}_j = 0) = p(|\phi| < 2) \approx 96\% \ ,$$

for all n, where ϕ is a standard normal random variable. If $\beta_j \neq 0$, $\hat{\beta}_j^{\text{ols}} \to \beta_j$ as $n \to \infty$, while $\text{sd}(\hat{\beta}_j^{\text{ols}}) \to 0$, and so $p(\hat{\beta}_j = 0) \to 0$ as $n \to \infty$.

4.4 Extensions to adaptive sparseness

4.4.1 Mixed sparsity

We now consider a few extensions to adaptive sparseness that will enable its use in estimation of locally adaptive splines. Firstly, suppose that we do not want to place the sparsity inducing prior on the entire

coefficient vector $\boldsymbol{\beta}$. If, for example, we know that some coefficients are non-zero, we can set the variance parameters in their priors to infinity, resulting in an improper flat prior for those coefficients. The model can then be written as

$$\begin{aligned} p(\sigma^2) &\propto c \\ p(\tau_j) &\propto 1/\tau_j \quad \forall j \in \mathcal{P} \\ \tau_j &= \infty \quad \forall j \notin \mathcal{P} \\ \beta_j|\tau_j &\sim \mathcal{N}(0, \tau_j) \\ y_i|\boldsymbol{\beta}, \boldsymbol{x}_i, \sigma^2 &\sim \mathcal{N}\Big(\sum_{j=1}^{p} \beta_j x_{ij}, \sigma^2\Big) \ , \end{aligned} \tag{4.11}$$

where $\mathcal{P}$ denotes the set of coefficients that are penalised. The ECM algorithm remains the same as before, simply with $W_{jj} = 0$ for every $j \notin \mathcal{P}$.

4.4.2 Sparsity on linear combinations

Next, assume we want sparsity in $D\boldsymbol{\beta}$, as opposed to $\boldsymbol{\beta}$, where D is some penalty matrix. Assume at first that D is a $p \times p$ matrix of full rank. A way to achieve sparsity in $D\boldsymbol{\beta}$ is by reparametrising to a new coefficient vector $\tilde{\boldsymbol{\beta}} = D\boldsymbol{\beta}$. The model

$$y = X\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

can then be re-written as

$$\begin{aligned} y &= X\boldsymbol{\beta} + \boldsymbol{\epsilon} \\ &= XD^{-1}D\boldsymbol{\beta} + \boldsymbol{\epsilon} \\ &= \tilde{X}\tilde{\boldsymbol{\beta}} + \boldsymbol{\epsilon} \ , \end{aligned}$$

where $\tilde{X} = XD^{-1}$. This is now an ordinary linear regression model, and equipping it with the adaptive sparseness prior on $\tilde{\boldsymbol{\beta}}$ will result in sparsity in $D\boldsymbol{\beta}$. Now suppose D is a $(p-q) \times p$ matrix of rank $p-q$, where $0 < q < p$. D is no longer invertible since it is not square. However we can define a new matrix

$$\tilde{D} = \begin{pmatrix} U \\ D \end{pmatrix} \ ,$$

where U is any $q \times p$ matrix such that $\tilde{D}$ is of rank p. We can now apply the same trick by defining $\tilde{\boldsymbol{\beta}} = \tilde{D}\boldsymbol{\beta}$ and $\tilde{X} = X\tilde{D}^{-1}$. We no longer seek sparsity in the entire vector $\tilde{\boldsymbol{\beta}}$, but instead only its last $p-q$ elements. As already described this can be achieved by applying a sparsity inducing prior on the last $p-q$ elements of $\tilde{\boldsymbol{\beta}}$, and a flat prior on the first q elements of $\tilde{\boldsymbol{\beta}}$.

4.4.3 Extending to generalized linear models

The treatment of adaptive sparseness thus far has assumed a normal response variable, however it can be extended to responses from the exponential family of distributions in a straightforward manner, as suggested

by Kiiveri (2003). The model can be adapted as

$$\begin{aligned} p(\phi) &\propto c \\ p(\tau_j) &\propto 1/\tau_j \quad \forall j \in \mathcal{P} \\ \tau_j &= \infty \quad \forall j \notin \mathcal{P} \\ \beta_j|\tau_j &\sim \mathcal{N}(0, \tau_j) \\ g(\mu_i) &= \sum_{j=1}^{p} \beta_j x_{ij} \\ y_i|\boldsymbol{\beta}, \phi, \boldsymbol{x}_i &\sim \pi(\mu_i, \phi) \ , \end{aligned} \tag{4.12}$$

where $g(\cdot)$ is a link function, π is an exponential family distribution, and ϕ is the dispersion parameter of the exponential family distribution that y_i follows, conditionally on $\boldsymbol{\beta}$. The E-step of the EM algorithm remains the same, since the terms involving τ_j are unchanged, however the updates for $\boldsymbol{\beta}$ are no longer available in closed form, with iterative methods required to compute them. Specifically, we now have

1. $Q(\boldsymbol{\beta}, \sigma^2|\hat{\boldsymbol{\beta}}) = \ell(\boldsymbol{\beta}, \phi) - \dfrac{1}{2}\displaystyle\sum_{j=1}^{p} \frac{\beta_j^2}{\hat{\beta}_j^2}$
2. $\hat{\phi} = \arg\max\limits_{\phi} Q(\hat{\boldsymbol{\beta}}, \phi|\hat{\boldsymbol{\beta}})$
3. $\hat{\boldsymbol{\beta}} = \arg\max\limits_{\boldsymbol{\beta}} Q(\boldsymbol{\beta}, \hat{\phi}|\hat{\boldsymbol{\beta}})$

where $\ell(\boldsymbol{\beta}, \phi)$ is the log-likelihood of the observed data implied by the last line of (4.12).

4.5 An efficient implementation of the ECM algorithm for adaptive sparseness

In this section we provide an efficient implementation of the ECM agorithm for adaptive sparseness introduced in Section 4.2, which provides significant time savings compared to a naive implementation. The algorithm is based on updating the QR decomposition of the design matrix X, as proposed by Gentleman (1974) and Miller (1992), and implemented in the `R` package `biglm` (Lumley, 2013). The QR decomposition of a square $n \times n$ matrix A is given by

$$A = QR$$

where Q is an $n \times n$ orthogonal matrix and R is an $n \times n$ upper triangular matrix with positive elements on the diagonal. The QR decomposition of A can be used to efficiently solve linear systems of the form $A\boldsymbol{x} = \boldsymbol{c}$ in two steps. Firstly, because Q is orthogonal, $QR\boldsymbol{x} = \boldsymbol{c}$ implies $R\boldsymbol{x} = Q^T\boldsymbol{c}$. Now, because R is upper triangular, the system of equations $R\boldsymbol{x} = Q^T\boldsymbol{c}$ can be efficiently solved using the back substitution algorithm. For a rectangular $n \times p$ matrix X the QR decomposition can be written as

$$X = \begin{pmatrix} Q_1 & Q_2 \end{pmatrix} \begin{pmatrix} R \\ \mathbf{0} \end{pmatrix} = Q_1 R \ ,$$

where $Q = (Q_1 \quad Q_2)$ is an $n \times n$ orthogonal matrix, R is a $p \times p$ upper triangular matrix, and $\mathbf{0}$ is an $(n-p) \times (n-p)$ matrix of zeros. A neat feature of the QR decomposition is that the same approach when

solving a linear system returns the least squares solution when applied to an over-determined system of equations

$$X\boldsymbol{\beta} = \boldsymbol{y} \ .$$

Specifically, the least squares solution is given by the system of equations

$$X^T X\boldsymbol{\beta} = X^T\boldsymbol{y} \ , \tag{4.13}$$

and substituting $X = Q_1 R$ into (4.13) gives

$$\begin{aligned} R^T Q_1^T Q_1 R\boldsymbol{\beta} &= R^T Q_1^T \boldsymbol{y} \\ R^T R\boldsymbol{\beta} &= R^T Q_1^T \boldsymbol{y} \\ R^T \boldsymbol{\beta} &= Q_1^T \boldsymbol{y} \ , \end{aligned}$$

implying that given the QR decomposition of the model matrix X, the least squares solution in an ordinary linear regression problem can be found by solving the linear system of p equations in p unknowns $R^T\boldsymbol{\beta} = Q_1^T\boldsymbol{y}$. Consequently, using the QR decomposition of X is one of the most popular approaches to obtaining least squares solutions since the computations are efficient and $X^T X$ never has to be formed. Working with X as opposed to $X^T X$ is preferable for conditioning reasons (Golub and Van Loan, 2012, Chapter 5). The QR decomposition can also be used to solve weighted ridge regression problems of the sort necessary in the ECM algorithm for adaptive sparseness. Specifically, we require repeated evaluations of

$$\hat{\boldsymbol{\beta}} = (X^T X + W)^{-1} X^T \boldsymbol{y} \ ,$$

where W is a diagonal matrix. If we let U be a diagonal matrix with $U_{jj} = \sqrt{W_{jj}}$, and

$$\tilde{\boldsymbol{y}} = \begin{pmatrix} \boldsymbol{y} \\ \boldsymbol{0} \end{pmatrix} \quad \text{and} \quad \tilde{X} = \begin{pmatrix} X \\ U \end{pmatrix} \ , \tag{4.14}$$

then

$$\hat{\boldsymbol{\beta}} = (\tilde{X}^T \tilde{X})^{-1} \tilde{X}^T \tilde{\boldsymbol{y}} \ ,$$

which is the ordinary least squares solution to a regression model with model matrix $\tilde{X}$ and response vector $\tilde{\boldsymbol{y}}$, and can be found using the QR decomposition of $\tilde{X}$. The idea behind our algorithm is that in each iteration, the QR decomposition of $\tilde{X}$ need not be found from scratch. Instead, if we pre-compute the QR decomposition of X, then much of the work towards find the new QR decomposition of $\tilde{X}$ has already been done. To see this, suppose we have already computed an orthogonal matrix Q and upper triangular matrix R such that $X = QR$. Now we seek an orthogonal matrix $\tilde{Q}$ and upper triangular matrix $\tilde{R}$ such that $\tilde{X} = \tilde{Q}\tilde{R}$. To start, one can write

$$\tilde{X} = \begin{pmatrix} Q & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} R \\ U \end{pmatrix} \ .$$

Now note that the matrix $\begin{pmatrix} Q & 0 \\ 0 & I \end{pmatrix}$ is already orthogonal, while $\begin{pmatrix} R \\ U \end{pmatrix}$ is upper triangular, except for the p diagonal elements of U which are non-zero. The general strategy to find $\tilde{Q}\tilde{R}$ is to find an orthogonal transformation matrix G such that $G \begin{pmatrix} R \\ U \end{pmatrix}$ is upper triangular. Then, setting

$$\tilde{Q} = \begin{pmatrix} Q & 0 \\ 0 & I \end{pmatrix} G^T \quad \text{and} \quad \tilde{R} = G \begin{pmatrix} R \\ U \end{pmatrix}$$

will be a QR decomposition for $\tilde{X}$ because $\tilde{Q}$ is an orthogonal matrix (as it is the product of two orthogonal matrices), $\tilde{R}$ is upper triangular due to the choice of G, and

$$\tilde{Q}\tilde{R} = \begin{pmatrix} Q & 0 \\ 0 & I \end{pmatrix} G^T G \begin{pmatrix} R \\ U \end{pmatrix} \tag{4.15}$$

$$= \begin{pmatrix} Q & 0 \\ 0 & I \end{pmatrix} \begin{pmatrix} R \\ U \end{pmatrix} \tag{4.16}$$

$$= \tilde{X} \ , \tag{4.17}$$

as required. The trick is now to find the orthogonal matrix G; this can be accomplished using a product of Givens rotation matrices (Golub and Van Loan, 2012, Chapter 5). A Givens matrix $G^{(i,j;\theta)}$ is a matrix with zeros on all off-diagonal elements, except for the i,j-th and j,i-th elements which are given by $\sin(\theta)$ and $-\sin(\theta)$ respectively, and ones on all the diagonal elements, except for the i,i-th and j,j-th elements, which are both equal to $\cos(\theta)$. The effect of multiplying a vector $\boldsymbol{v}$ by a Given's rotation matrix $G^{(i,j;\theta)}$ is that the i and j-th elements of the vector will be modified; specifically, $G^{(i,j;\theta)}\boldsymbol{v}$ will be a new vector whose i-th element is $v_j \sin(\theta) + v_i \cos(\theta)$ and j-th element is $v_j \cos(\theta) - v_i \sin(\theta)$. Consequently, i, j, and θ can be chosen such that an arbitrary element of $G^{(i,j;\theta)}\boldsymbol{v}$ is zero. For QR updating case we seek a matrix G such that $G \begin{pmatrix} R \\ U \end{pmatrix}$ is upper triangular. This can be achieved by considering a series of transformations such that each row of U is transformed into a vector of zeros, one row at a time. If we denote by $\boldsymbol{u}_i^T$ the i-th row of U, then $\boldsymbol{u}_i^T$ is a vector with zeros everywhere except for the i-th element, which is equal to $\sqrt{W_{jj}}$. The first product of Given's matrices is that which would make $\begin{pmatrix} R \\ \boldsymbol{u}_1^T \end{pmatrix}$ upper triangular. $\boldsymbol{u}_1^T$ has a one in the first element, and zeros everywhere. The first Given's matrix, $G^{(1,n+1,\theta_1)}$, can be used to make the first element in the vector $\boldsymbol{u}_1^T$ a zero, however this introduces non-zero elements in the rest of $\boldsymbol{u}_1^T$. Then a second Given's rotation matrix $G^{(2,n+1,\theta_2)}$ can be chosen to make the second element in $\boldsymbol{u}_1^T$ a zero. Note that for the second rotation, the indices $(2, n+1)$ are chosen, because the second row of R has a zero in the first position, and so the zero in the first position of $\boldsymbol{u}_2^T$ is maintained. This process is repeated for the entire row, resulting in a final matrix

$$\tilde{R}^{(1)} = G^{(n,n+1,\theta_1)} \dots G^{(2,n+1,\theta_1)} G^{(1,n+1,\theta_1)} R \ ,$$

which is upper triangular. Then an additional row can be added in exactly the same way, giving

$$\tilde{R}^{(2)} = G^{(n,n+2,\theta_1)} \dots G^{(2,n+2,\theta_1)} G^{(1,n+2,\theta_1)} \tilde{R}^{(1)} \ ,$$

and so on. The full QR decomposition of $\tilde{X}$ can be found by updating the QR decomposition of X with one row of U at a time. Note that the matrix Q need not at any point be formed explicitly. Recall that to solve for $\boldsymbol{\beta}$ we need to solve the rectangular system $\tilde{R}\boldsymbol{\beta} = \tilde{Q}^T\tilde{\boldsymbol{y}}$. Hence, only $\tilde{R}$ and $\tilde{Q}^T\tilde{\boldsymbol{y}}$ need to be formed. Since $\tilde{Q}^T$ simply encodes a series of orthogonal transformations, these can just directly be applied to $\tilde{\boldsymbol{y}}$, storing only $\tilde{Q}^T\tilde{\boldsymbol{y}}$, and not $\tilde{Q}$.

At each iteration, some coefficients β_j may be smaller in absolute value than a tolerance δ that we define to be an effective zero. Then, these coefficients will be estimated to be zero at every subsequent iteration, and hence the weights W_{jj} will not change for those coefficients. Consequently, for every coefficient that is estimated to be zero, we can update the QR decomposition with the respective weight of that coefficient, and need not update it again for that coefficient. Our approach to implementing the ECM algorithm for adaptive sparseness is given in Algorithm 1.

Data: y, X, W^{init},ϵ,γ,δ
Result: $\boldsymbol{\beta}$
$\{R, Q^T\boldsymbol{y}\} \leftarrow \text{qr}(X, \boldsymbol{y})$
$W \leftarrow W^{\text{init}}$
$\tilde{X}^T \leftarrow (X^T, W)^T$
$\tilde{\boldsymbol{y}}^T \leftarrow (\boldsymbol{y}^T, \mathbf{0}_p^T)$
$\{\tilde{R}, \tilde{Q}^T\tilde{\boldsymbol{y}}\} \leftarrow \text{qr}(\tilde{X}, \tilde{\boldsymbol{y}}|R, Q^T\boldsymbol{y})$
$\boldsymbol{\beta} \leftarrow \underset{\boldsymbol{\beta}}{\text{argsolve}}\{\tilde{R}\boldsymbol{\beta} = \tilde{Q}^T\tilde{\boldsymbol{y}}\}$
$\mathcal{Z} \leftarrow \varnothing$
$\Delta \leftarrow 1$
while $\Delta > \gamma$ **do**
  $\mathcal{Z}^{\text{old}} \leftarrow \mathcal{Z}$
  $\boldsymbol{\beta}^{\text{old}} \leftarrow \boldsymbol{\beta}$
  $\sigma^2 \leftarrow \frac{1}{n}(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta})$
  for $j \in \mathcal{Z}^{\complement}$ **do**
    $W_{jj} \leftarrow \sigma^2/(\beta_j + \epsilon)^2$
  end
  $\tilde{X}^T \leftarrow (X^T, W_{\mathcal{Z}^{\complement}})^T$
  $\tilde{\boldsymbol{y}}^T \leftarrow (\boldsymbol{y}^T, \mathbf{0}^T_{|\mathcal{Z}^{\complement}|})$
  $\{\tilde{R}, \tilde{Q}^T\tilde{\boldsymbol{y}}\} \leftarrow \text{qr}(\tilde{X}, \tilde{\boldsymbol{y}}|R, Q^T\boldsymbol{y})$
  $\boldsymbol{\beta} \leftarrow \underset{\boldsymbol{\beta}}{\text{argsolve}}\{\tilde{R}\boldsymbol{\beta} = \tilde{Q}^T\tilde{\boldsymbol{y}}\}$
  $\mathcal{Z} \leftarrow \{j : |\beta_j| < \delta\}$
  if $\mathcal{Z} \neq \mathcal{Z}^{old}$ **then**
    $X^T \leftarrow (X^T, W_{\mathcal{Z}})^T$
    $\boldsymbol{y}^T \leftarrow (\boldsymbol{y}^T, \mathbf{0}^T_{|\mathcal{Z}|})$
    $\{R, Q^T\boldsymbol{y}\} \leftarrow \text{qr}(X, \boldsymbol{y}|R, Q^T\boldsymbol{y})$
  end
  $\Delta \leftarrow (\boldsymbol{\beta}^{\text{old}} - \boldsymbol{\beta})^T(\boldsymbol{\beta}^{\text{old}} - \boldsymbol{\beta})/p$
end

Algorithm 1: QR updating implementation of the ECM algorithm for adaptive sparseness. In this algorithm, $\varnothing$ denotes the empty set, $\mathcal{Z}^{\complement}$ is the complement of $\mathcal{Z}$, $\mathbf{0}_p$ is a vector of zeros of length p, $|\beta_j|$ is the absolute value of β_j, $|\mathcal{Z}|$ is the size of $\mathcal{Z}$, and $\text{qr}(\tilde{X}, \tilde{\boldsymbol{y}}|R, Q^T\boldsymbol{y})$ denotes an algorithm that updates the QR decomposition $\{R, Q^T\boldsymbol{y}\}$ for the augmented data set $\{\tilde{X}, \tilde{\boldsymbol{y}}\}$. Implementations of the functions 'qr' and 'solve' exist in the `biglm` package in `R`.

4.6 Estimating locally adaptive splines using adaptive sparseness

In this section we describe how we can apply adaptive sparseness to knot selection when using B-splines. Consider a spline $S^{(m)}(x)$, of order m, with equally spaced knots, represented as a linear combination of b-splines. We have

$$S^{(m)}(x) = \sum_{j=1}^{k} \alpha_j B_j^{(m)}(x) \ .$$

We begin by showing that sparsity in m-th order differences of the coefficients $(\alpha_1, \ldots, \alpha_k)$ implies that $S^{(m)}(x)$ is equivalent to a spline with some knots removed, i.e. it is equivalent to a spline with fewer and unevenly spaced knots. Recall from Section 3.2 that a spline consists of piecewise polynomials, defined between the knots, with smoothness constraints at the knots. Hence, a somewhat cumbersome but illustrative way to express a spline is

$$S^{(m)}(x) = \begin{cases} \alpha_{1,m}x^m + \alpha_{1,m-1}x^{m-1} + \cdots + \alpha_{1,1}x + \alpha_{1,0} & \text{for } \ t_0 \le x < t_1 \\ \alpha_{2,m}x^m + \alpha_{2,m-1}x^{m-1} + \cdots + \alpha_{2,1}x + \alpha_{2,0} & \text{for } \ t_1 \le x < t_2 \\ \vdots & \vdots \\ \alpha_{k,m}x^m + \alpha_{k,m-1}x^{m-1} + \cdots + \alpha_{k,1}x + \alpha_{1,0} & \text{for } \ t_k \le x < t_1 \ , \end{cases} \tag{4.18}$$

where the coefficients $\alpha_{j,l}$ are subject to the smoothness constraints

$$\begin{aligned} \lim_{x \to t_j^+} S^{(m)}(x) &= \lim_{x \to t_j^-} S^{(m)}(x) \\ \lim_{x \to t_j^+} \frac{\partial S^{(m)}(x)}{\partial x} &= \lim_{x \to t_j^-} \frac{\partial S^{(m)}(x)}{\partial x} \\ \lim_{x \to t_j^+} \frac{\partial^2 S^{(m)}(x)}{\partial x^2} &= \lim_{x \to t_j^-} \frac{\partial^2 S^{(m)}(x)}{\partial x^2} \\ &\vdots \\ \lim_{x \to t_j^+} \frac{\partial^{(m-2)} S^{(m)}(x)}{\partial x^{(m-2)}} &= \lim_{x \to t_j^-} \frac{\partial^{(m-2)} S^{(m)}(x)}{\partial x^{(m-2)}} \ . \end{aligned}$$

Hence, the only parameter that is allowed to vary between adjacent polynomials is the coefficient of the leading power, $\alpha_{j,m}$. Consequently if, for a particular knot, we add in the constraint

$$\lim_{x \to t_j^+} \frac{\partial^{(m-1)} S^{(m)}(x)}{\partial x^{(m-1)}} = \lim_{x \to t_j^-} \frac{\partial^{(m-1)} S^{(m)}(x)}{\partial x^{(m-1)}} \ ,$$

the adjacent polynomials on either side of this knot will be constrained to be identical. If the two polynomials on either side of the knot are identical, then the spline is equivalent to a spline with that knot removed. Hence, to show that sparsity in m-th order differences of the coefficients of b-splines results in knot removal, it is sufficient to show that the $m-1$-th derivatives of some adjacent polynomials will be equal. Defining $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_k)^T$ and $\boldsymbol{b}^{(m)} = \left(B_1^{(m)}(x), \ldots, B_k^{(m)}(x)\right)^T$, we can re-express $S^{(m)}(x)$ as

$$S^{(m)}(x) = \boldsymbol{\alpha}^T \boldsymbol{b}^{(m)} \ .$$

With equal spacing between knots the derivative of a b-spline function (de Boor, 2001, page 115) is

$$\frac{\partial B_j^m(x)}{\partial x} = \begin{cases} \frac{1}{\Delta}\Big(B_j^{(m-1)}(x) - B_{j+1}^{(m-1)}(x)\Big) & \text{if} \quad j < k \\ \frac{1}{\Delta}B_j^{(m-1)}(x)) & \text{if} \quad j = k \ , \end{cases} \tag{4.19}$$

where Δ is the distance between knots. Assuming for simplicity of notation that the distance between knots is one, we can write the derivative of a spline function as

$$\begin{aligned} \frac{\partial S^{(m)}}{\partial x} &= \alpha_k B_k^{(m-1)} + \sum_{j=1}^{k-1} \alpha_j \Big(B_j^{(m-1)} - B_{j+1}^{(m-1)}\Big) \\ &= \boldsymbol{\alpha}^T D^T \boldsymbol{b}^{(m-1)} \ , \end{aligned} \tag{4.20}$$

where D is the differencing matrix defined in Chapter 3. Note that D is such that for some vector $\boldsymbol{v}$,

$$(D\boldsymbol{v})_j = \begin{cases} v_j & \text{for} \quad j = 1 \\ v_j - v_{j-1} & \text{for} \quad j > 1 \ , \end{cases}$$

whereas

$$(D^T\boldsymbol{v})_j = \begin{cases} v_j - v_{j+1} & \text{for} \quad j < d \\ v_j & \text{for} \quad j = d \ , \end{cases}$$

where d is the length of $\boldsymbol{v}$. Applying (4.19) and (4.20) recursively, higher order derivatives of splines can conveniently be expressed as

$$\frac{\partial^l S^{(m)}}{\partial x^l} = \boldsymbol{\alpha}^T (D^l)^T \boldsymbol{b}^{(m-l)} \ . \tag{4.21}$$

Equation (4.21) implies that derivatives of splines are also splines, but of lower order. This fact is illustrated in Figure 4.1, in which we display an arbitrary spline of order 4, along with its derivatives, and the b-splines that form a basis for them. In particular, for the $(m-1)$-th derivative of a spline of order m, we obtain

$$\begin{aligned} \frac{\partial^{m-1} S^{(m)}}{\partial x^{m-1}} &= \boldsymbol{\alpha}^T (D^{m-1})^T \boldsymbol{b}^{(1)} \\ &= (D^{(m-1)}\boldsymbol{\alpha})^T \boldsymbol{b}^{(1)} \end{aligned} \tag{4.22}$$

Recall from the definition of b-splines that b-splines of order 1 form a basis for a step function, and so (4.22) is a step function whose value in each individual region is the $(m-1)$-th derivative of each piecewise polynomial that makes up the spline function $S^{(m)}(x)$, and is given by the coefficient of the corresponding b-spline. All we needed to show is that if $D^m\boldsymbol{\alpha}$ is sparse, then some adjacent polynomials in the spline function $S^{(m)}(x)$ will have the same $(m-1)$-th derivative, implying that $S^{(m)}(x) = \boldsymbol{\alpha}^T\boldsymbol{b}^{(m)}$ would be equivalent to a spline with some knots removed. Looking at (4.21) we observe that for some $j > 1$,

$$(D^m\boldsymbol{\alpha})_j = (DD^{m-1}\boldsymbol{\alpha})_j = 0$$

implies that

$$(D^{m-1}\boldsymbol{\alpha})_j = (D^{m-1}\boldsymbol{\alpha})_{j-1} \ ,$$

which consequently means that the $(j-1)$-th and j-th polynomials in the spline $S^{(m)}(x)$ are identical.

When using splines to model unknown functions, the idea arises to use a sparsity inducing method that enforces sparsity on the vector $D^{(m)}\boldsymbol{\alpha}$. This provides the estimation procedure with the flexibility to choose which knots are necessary and which are not. While in theory any sparsity inducing could be used, we propose the adaptive sparseness method due its adaptive qualities and lack of tuning parameters.

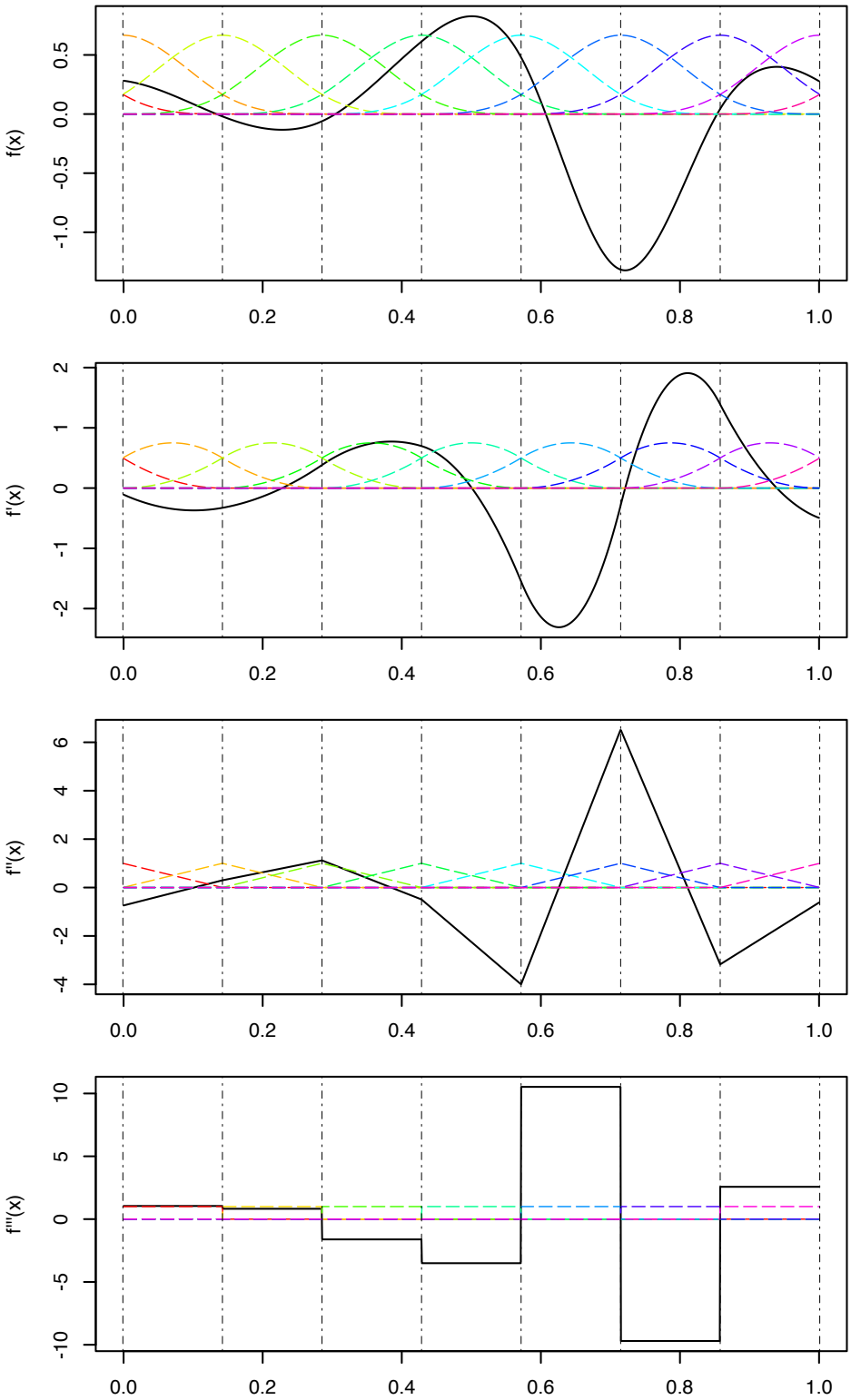

Figure 4.1: Derivatives, from the top to the bottom panel, ranging from order 0 to 3 of an arbitrary spline of order 4. The vertical dotted lines are the knot locations and the coloured dotted lines are b-splines that form a basis for each function.

4.7 Related methods

In work that occurred in parallel to ours, Goepp et al. (2018) consider a similar approach to estimation of locally adaptive splines. Their method depends on a sparsity inducing procedure termed 'adaptive ridge' regression, which was introduced by Frommlet and Nuel (2016). Adaptive ridge regression computationally ends up being very similar to the adaptive sparseness method of Figueiredo (2003), however it is arrived at from a different perspective. Frommlet and Nuel (2016) begin by considering best subset variable selection, which consists of running regression models with all possible combinations of variables included, and using a model selection method to choose which performs best. For linear regression with a normal response, the best subset estimator can be written as the following penalised likelihood estimator,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} \left\{ \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} + \lambda \|\boldsymbol{\beta}\|_0 \right\} , \tag{4.23}$$

where $\|\boldsymbol{\beta}\|_0$ denotes the number of non-zero elements in $\boldsymbol{\beta}$, and λ is a tuning parameter to be selected using a model selection technique. Because the penalty $\|\cdot\|$ is not continuous, the optimisation in (4.23) is difficult to perform, with the only guaranteed way of achieving a global minimum being running the regression model for every combination of variables, and checking which combination achieves a minimum. Running so many regression models can be prohibitively expensive even for a small number of predictors however, rendering best subset selection infeasible in most practical scenarios. As an alternative, Frommlet and Nuel (2016) consider solving (4.23) by local quadratic approximation, in which given a current estimate $\hat{\boldsymbol{\beta}}^{(k)}$ of $\boldsymbol{\beta}$, $\|\boldsymbol{\beta}\|_0$ is approximated by $\sum_{j=1}^{p} \beta_j^2/(\hat{\beta}_j^{(k)} + \epsilon)^2$, where ϵ is some very small number. An updated estimator is then obtained by solving an approximation to (4.23), given by

$$\hat{\boldsymbol{\beta}}^{(k+1)} = \arg\min_{\boldsymbol{\beta}} \left\{ \frac{(\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta})}{\sigma^2} + \lambda \sum_{j=1}^{p} \beta_j^2/(\hat{\beta}_j^{(k)} + \epsilon)^2 \right\} , \tag{4.24}$$

which is just a ridge regression problem. Equation (4.24) is then iterated until convergence. Note that equation (4.24) would be identical to the M-step of the EM algorithm for adaptive sparseness discussed in Section 4.2 if the tuning parameter λ was set to one, and the noise variance σ^2 was estimated iteratively at each step by

$$\hat{\sigma}^{2(k+1)} = (\boldsymbol{y} - X\boldsymbol{\beta}^{(k)})^T(\boldsymbol{y} - X\boldsymbol{\beta}^{(k)})/n \ .$$

Goepp et al. (2018) consider estimating locally adaptive splines by using the adaptive ridge procedure to enforce sparsity on the m-th order differences of the coefficients of b-splines that form a basis for an m-th order spline. In their implementation, they do not estimate the noise variance σ^2, and use model selection techniques to set the tuning parameter λ. Note by allowing a free of choice of λ, it can be argued that the noise variance becomes irrelevant, since (4.24) can be reparametrised as

$$\hat{\boldsymbol{\beta}}^{(k+1)} = \arg\min_{\boldsymbol{\beta}} \left\{ (\boldsymbol{y} - X\boldsymbol{\beta})^T(\boldsymbol{y} - X\boldsymbol{\beta}) + \tilde{\lambda} \sum_{j=1}^{p} \beta_j^2/(\hat{\beta}_j^{(k)} + \epsilon)^2 \right\} ,$$

where $\tilde{\lambda} = \sigma^2\lambda$, with selection of $\tilde{\lambda}$ amounting to simultaneous selection of λ and σ^2, without the need for individual estimates for each. In practice however there are still reasons why estimating the noise variance is useful. When selecting $\tilde{\lambda}$ by model selection techniques, a range of values to try must be selected. Given that $\tilde{\lambda}$ depends on σ^2, there is more uncertainty regarding the range of values that the optimal $\tilde{\lambda}$ would lie in. Secondly, as shown in Section 4.2, the Bayesian formulation of adaptive sparseness provides a motivation to simply set λ to one, and forgo tuning parameter selection altogether.

4.8 Additive models: empirical assessment of methods

In this section we carry out simulation studies to compare the performance of the various methods we have discussed in terms of estimation of components of generalized additive models. Below we give a list of the methods we compare along with the name we use to refer to them, and the implementation of the method that we use. In all cases we use b-splines of order 4 (in order to obtain cubic spline fits) with evenly spaced knots to represent the unknown functions, and the only thing that changes between methods is the approach to regularisation, or in the Bayesian contexts considered the form of the prior used for the coefficients of the b-splines.

1. Our proposed method, which consists of using the adaptive sparseness prior on 4-th order differences of the coefficients of 4-th order b-splines. We refer to this with the acronym AS, which stands for adaptive sparseness. We use our own implementation.

2. The LASSO penalty on fourth order differences of the coefficients as proposed by Jhong et al. (2017). We refer to this as LASSO. We use the `R` package `glmnet` to fit the LASSO regularisation path, and our own implementation of the GCV criterion to choose the tuning parameter.

3. The method of Goepp et al. (2018), which consists of using the adaptive ridge penalty on fourth order differences of the coefficients, with tuning parameters selected either by AIC, BIC, or EBIC. We refer to this as AR / AIC, AR / BIC, and AR / EBIC respectively, where the AR stands for adaptive ridge. We use the implementation available at Goepp (2018).

4. The integrated squared second derivative roughness penalty with tuning parameter selection either by GCV or maximum likelihood (ML) as described in Section 2.3 . We refer to this as CR / GCV and CR / ML, where the CR stands for cubic regression spline. This approach, with both tuning parameter selection methods, is implemented in the `R` package `mgcv` (Wood, 2006).

5. A ridge penalty on fourth order differences of the coefficients with tuning parameter selection both by GCV and ML. We refer to this as PS / GCV and PS / ML respectively, where the PS stands for p-spline. This approach is also implemented in the `R` package `mgcv`, with both tuning parameter selection methods.

6. The integrated squared second derivative roughness penalty with multiple tuning parameters for a single function in order to achieve adaptivity. Tuning parameter selection is carried out either with GCV or ML. We refer to this as AD / GCV and AD / ML, where the AD stands for adaptive. This approach, with both tuning parameter selection methods, is implemented in the `R` package `mgcv`.

To compare the methods, we define five functions $f_1(\cdot), \ldots, f_5(\cdot)$ defined on the domain $[0, 1]$. The first simulation scenario is one of an additive model with only one predictor, which is equivalent to one-dimensional smoothing. For each function, we define the model

$$y_i = f(x_i) + \epsilon_i \ , \tag{4.25}$$

where $\epsilon_i \sim \mathcal{N}(0, 0.2^2)$. To create the values of the predictor $\{x_i\}_{i=1}^n$, we draw n values from the uniform distribution between 0 and 1. To evaluate the performance of the different methods in estimating $f(\cdot)$, we simulate $M = 10,000$ response vectors $\{y_i\}_{i=1}^n$ conditionally on the predictors, and for each vector obtain an estimate $\hat{f}(\cdot)$ of $f(\cdot)$. We then use the M estimates of $f(\cdot)$ to estimate the following performance criteria:

1. Integrated squared error (ISE): $\int_0^1 E\left\{\left(\hat{f}(x) - f(x)\right)^2\right\} dx$
2. Integrated absolute error (IAE): $\int_0^1 E\left\{|\hat{f}(x) - f(x)|\right\} dx$
3. Integrated squared bias (ISB): $\int_0^1 \left[E\left\{\hat{f}(x) - f(x)\right\}\right]^2 dx$
4. Integrated absolute bias (IAB): $\int_0^1 \left|E\left\{\hat{f}(x) - f(x)\right\}\right| dx$
5. Integrated variance (IV): $\int_0^1 E\left\{\left[\hat{f}(x) - E\Big(\hat{f}(x)\Big)\right]^2\right\} dx$
6. Average time (AT): $\frac{1}{M}\sum_{m=1}^{M} t_m$, where t_m is the time taken to obtain an estimate $\hat{f}(\cdot)$.

To estimate the performance criteria, we use the samples $\{\hat{f}^{(m)}\}_{m=1}^{M}$ to estimate the expectations, and numerical integration using the trapezoid rule to approximate the integrals. For instance, the estimate of the ISE is

$$\frac{\Delta}{M}\sum_{g=1}^{G-1}\sum_{m=1}^{M}\left(\hat{f}^{(m)}(g\Delta) - f(g\Delta)\right)^2 + \frac{\Delta}{2M}\sum_{g=0}^{1}\sum_{m=1}^{M}\left(\hat{f}^{(m)}(g) - f(g)\right)^2 ,$$

where G is the number of points used in the numerical integration, and $\Delta = 1/G$.

In the second simulation scenario we include various combinations of functions in the additive model and evaluate all the performance criteria for each individual function as well as for the sum of the functions. For instance, in the model

$$y_i = f_1(x_{1i}) + f_2(x_{2i}) + \epsilon_i ,$$

we evaluate the performance criteria for $f_1(x_1)$ and $f_2(x_2)$ separately, but also for $f(x_1, x_2) = f_1(x_1) + f_2(x_2)$. The only difference in how the criteria are defined in the multidimensional case is in how we approximate the integral, which is now multidimensional. The ISE for example is given by

$$\int_0^1\int_0^1 E\left\{\left(\hat{f}(x_1, x_2) - f(x_1, x_2)\right)^2\right\} dx_1 dx_2 .$$

We approximate the integrals by averaging over the observed data points, giving an estimator for the ISE

$$\frac{1}{NM}\sum_{i=1}^{N}\sum_{m=1}^{M}\left(\hat{f}^{(m)}(x_{1i}, x_{2i}) - f\big(x_{1i}, x_{2i}\big)\right)^2 .$$

4.8.1 Individual functions

Sine with increasing period

The first function we consider is given by

$$f_1(x) = \sin\left(17.5x^4\right) . \tag{4.26}$$

We set up b-splines of order 4, with 40 equally spaced knots to model $f_1(\cdot)$. In Figure 4.2 we display the function (4.26) with an example data set generated from (4.25), along with fitted functions using some of the methods we compare in the simulation study. Table 4.1 displays estimates of the various performance criteria we consider for each of the methods compared. The function $f_1(x)$ is an example of a function that

could benefit from locally adaptive estimation due to the fact that it has almost constant curvature in the region $x \in [0, 0.5]$, however in the region $x \in [0.5, 1]$ the curvature varies a lot. This appears in the results of the simulation studies as the best performing methods are the ones that explicitly aim to provide adaptive fits, with AD / ML, AD / GCV, and AS performing best both in terms of ISE and IAE. This also appears visually in Figure 4.2 in which the fit provided by CR / ML displays 'wiggliness' around the true function in the region $x \in [0, 0.5]$, which is alleviated in the adaptive fits. Interestingly, AS performs better than AR in all three tuning parameter selection cases, and is also faster given that it does not need to perform tuning parameter selection. Table 4.1 also illustrates that while remaining competitive with AD / ML and AD / GCV, AS requires much less time to run.

Method	IAB	ISB	IV	ISE	IAE	AT
AS	14.70	0.85	4.63	5.48	55.26	39.48
LASSO	53.86	6.56	3.70	10.27	72.44	149.11
AR / AIC	6.62	0.46	10.62	11.08	78.02	180.99
AR / BIC	9.25	0.57	6.34	6.90	61.02	180.99
AR / EBIC	10.30	0.60	6.24	6.84	60.86	180.99
CR / GCV	13.62	0.99	7.45	8.44	71.69	16.60
PS / GCV	12.44	1.03	7.72	8.75	72.39	18.82
CR / ML	12.00	0.80	7.65	8.45	71.92	29.97
PS / ML	13.00	1.14	7.29	8.43	71.20	34.50
AD / GCV	14.63	0.36	4.74	5.10	53.00	168.70
AD / ML	11.44	0.36	4.21	4.58	50.33	198.92

Table 4.1: Performance of methods in estimation of $f_1(x)$, defined in equation (4.26). All values are multiplied by 1000 to enhance readability. $n = 150$

Broken sine

The next function we consider is given by

$$f_2(x) = \begin{cases} 2\sin\left(\frac{\pi x}{1.5}\right) & \text{if } 0 \leq x < 0.75 \\ 2 & \text{if } 0.75 \leq x \leq 1 \end{cases}, \tag{4.27}$$

and we set up an identical simulation as we did for $f_1(x)$, using b-splines of order 4, with 40 knots. Figure 4.3 displays the function, along with example data sets and fitted functions, and Table 4.2 displays the performance of the various methods. Unlike $f_1(x)$, $f_2(x)$ is almost polynomial and we would not expect to see a benefit in performance due to locally adaptive estimation. Indeed the best performing methods in terms of ISE are now CR / ML and PS / ML. Interestingly, in this example, when comparing CR and PS it appears that the tuning parameter selection matters more than the penalty, as CR / ML and PS / ML are very close in performance, as are CR / GCV and PS / GCV. In this example, AS, CR / GCV, PS / GCV, and AD / ML all perform similarly, with AS performing slightly better relatively. Once again, AS performs better than AR across tuning parameter selection methods.

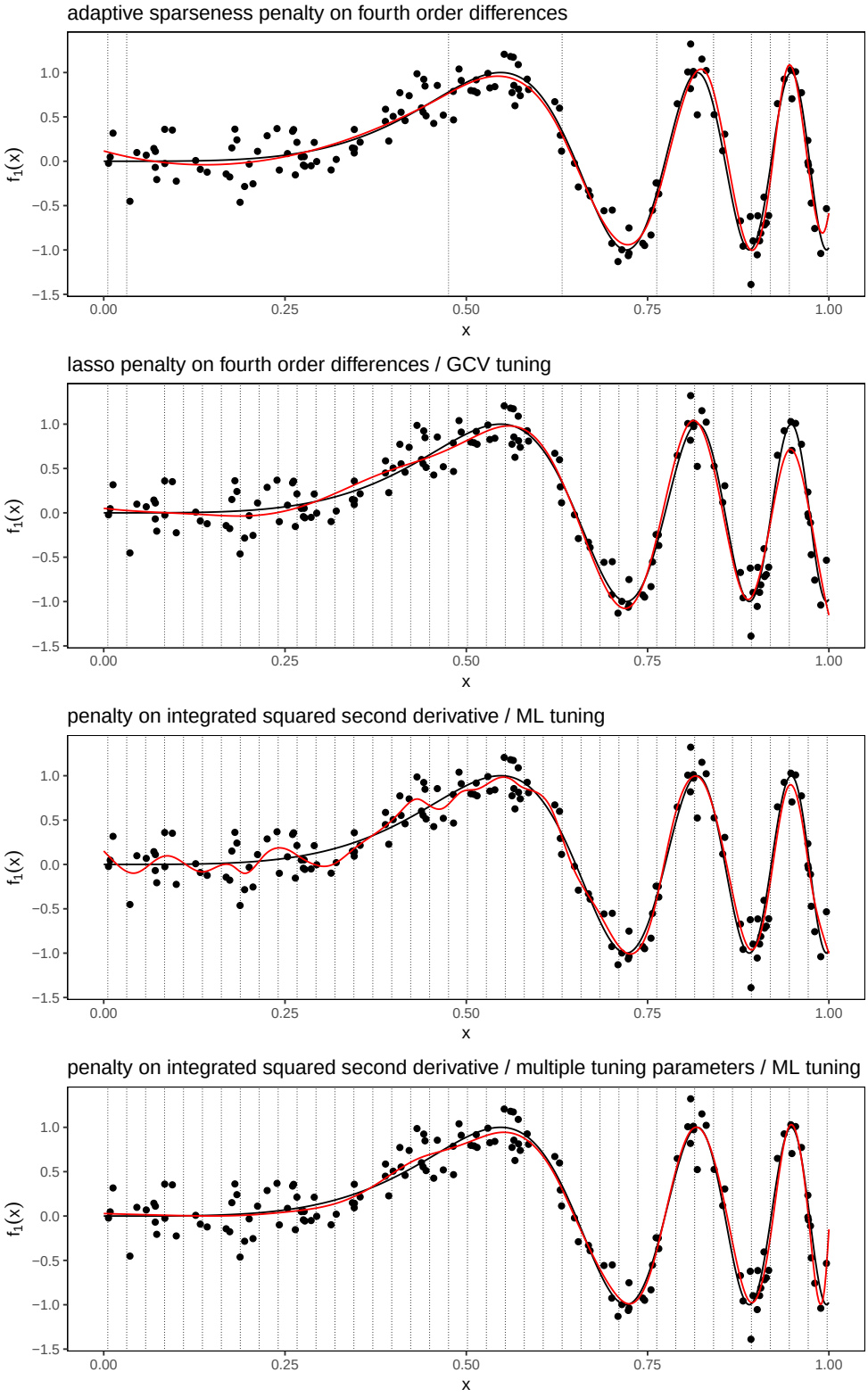

Figure 4.2: Estimates of $f_1(x)$, defined in equation (4.26), using a variety of methods, for an example data set. The points are the simulated data, the red curves are the fitted curves, the black curve is $f_1(x)$, and the vertical dotted lines are the final knot locations.

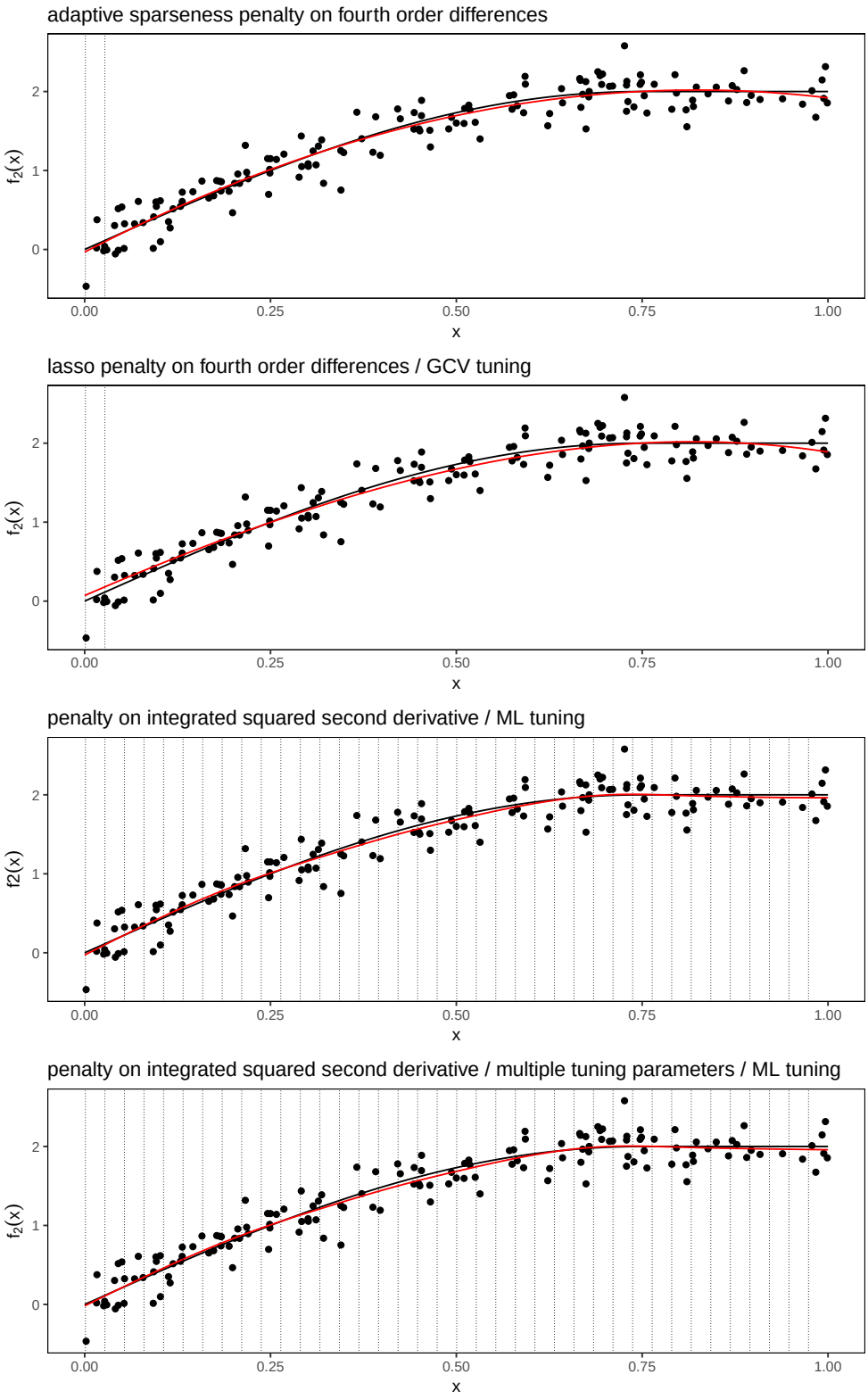

Figure 4.3: Estimates of $f_2(x)$, defined in equation (4.27), using a variety of methods, for an example data set. The points are the simulated data, the red curves are the fitted curves, the black curve is $f_2(x)$, and the vertical dotted lines are the final knot locations.

	IAB	ISB	IV	ISE	IAE	AT
AS	9.65	0.13	1.49	1.61	30.89	34.31
LASSO	22.84	0.74	1.24	1.99	34.83	10.27
AR / AIC	3.20	0.01	9.67	9.69	69.83	175.62
AR / BIC	8.21	0.09	2.15	2.24	34.43	175.62
AR / EBIC	13.54	0.24	1.45	1.69	31.71	175.62
CR / GCV	8.62	0.10	1.52	1.62	30.49	18.24
PS / GCV	8.64	0.10	1.53	1.63	30.56	18.97
CR / ML	3.93	0.02	1.32	1.34	28.63	38.73
PS / ML	3.95	0.03	1.31	1.34	28.56	40.42
AD / GCV	8.94	0.11	2.23	2.34	35.24	74.98
AD / ML	7.88	0.08	1.60	1.68	31.92	566.86

Table 4.2: Performance of methods in estimation of $f_2(x)$, defined in equation (4.27). All values are multiplied by 1000 to enhance readability. $n = 150$

Linear combinations of Gaussian densities

For the third and fourth functions we use a linear combination of Gaussian densities centred at a uniformly distributed sequence of points between 0 and 1. We then standardise the functions to have a range of 2, so that they have the same range as f_1 and f_2. Specifically, we set

$$\begin{aligned}
\tilde{f}_3(x) &= \sum_{j=1}^{10} \alpha_j^{(1)} \mathcal{N}(x; t_j, 0.07) \\
\tilde{f}_4(x) &= \sum_{j=1}^{10} \alpha_j^{(2)} \mathcal{N}(x; t_j, 0.07) \\
f_3(x) &= 2\tilde{f}_3(x)/(\sup_x \tilde{f}_3(x) - \inf_x \tilde{f}_3(x)) \\
f_4(x) &= 2\tilde{f}_4(x)/(\sup_x \tilde{f}_4(x) - \inf_x \tilde{f}_4(x)) \ ,
\end{aligned} \tag{4.28}$$

where $\sup_x$ denotes the supremum and $\inf_x$ denotes the infimum. We draw the coefficients $\alpha_1^{(1)}, \ldots, \alpha_{10}^{(1)}$ and $\alpha_1^{(2)}, \ldots, \alpha_{10}^{(2)}$ independently from a standard normal distribution, and the locations $t_1^{(1)}, \ldots, t_{10}^{(1)}$ and $t_1^{(2)}, \ldots, t_{10}^{(2)}$ from the uniform distribution. The performance of the various methods are summarised in Table 4.3 for f_3 and Table 4.4 for f_4, while example fits and data are given in Figures 4.4 and 4.5. For both functions both CR and PS perform well with both tuning parameter selection methods. AS falls a little bit behind however still performs better than AR with all tuning parameter selection methods and considerably better than LASSO.

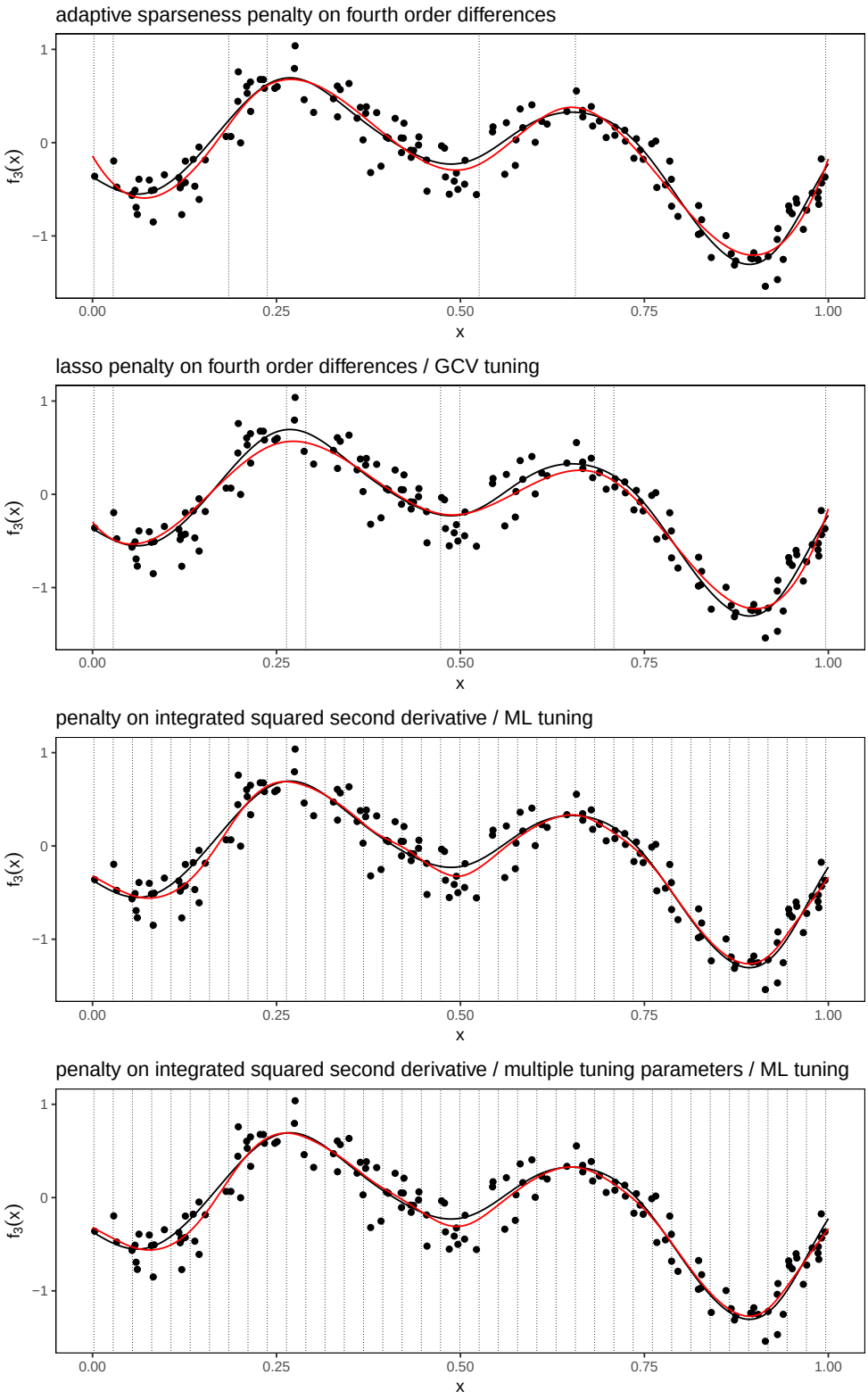

Figure 4.4: Estimates of $f_3(x)$, defined in equations (4.28), using a variety of methods, for an example data set. The points are the simulated data, the red curves are the fitted curves, the black curve is $f_3(x)$, and the vertical dotted lines are the final knot locations.

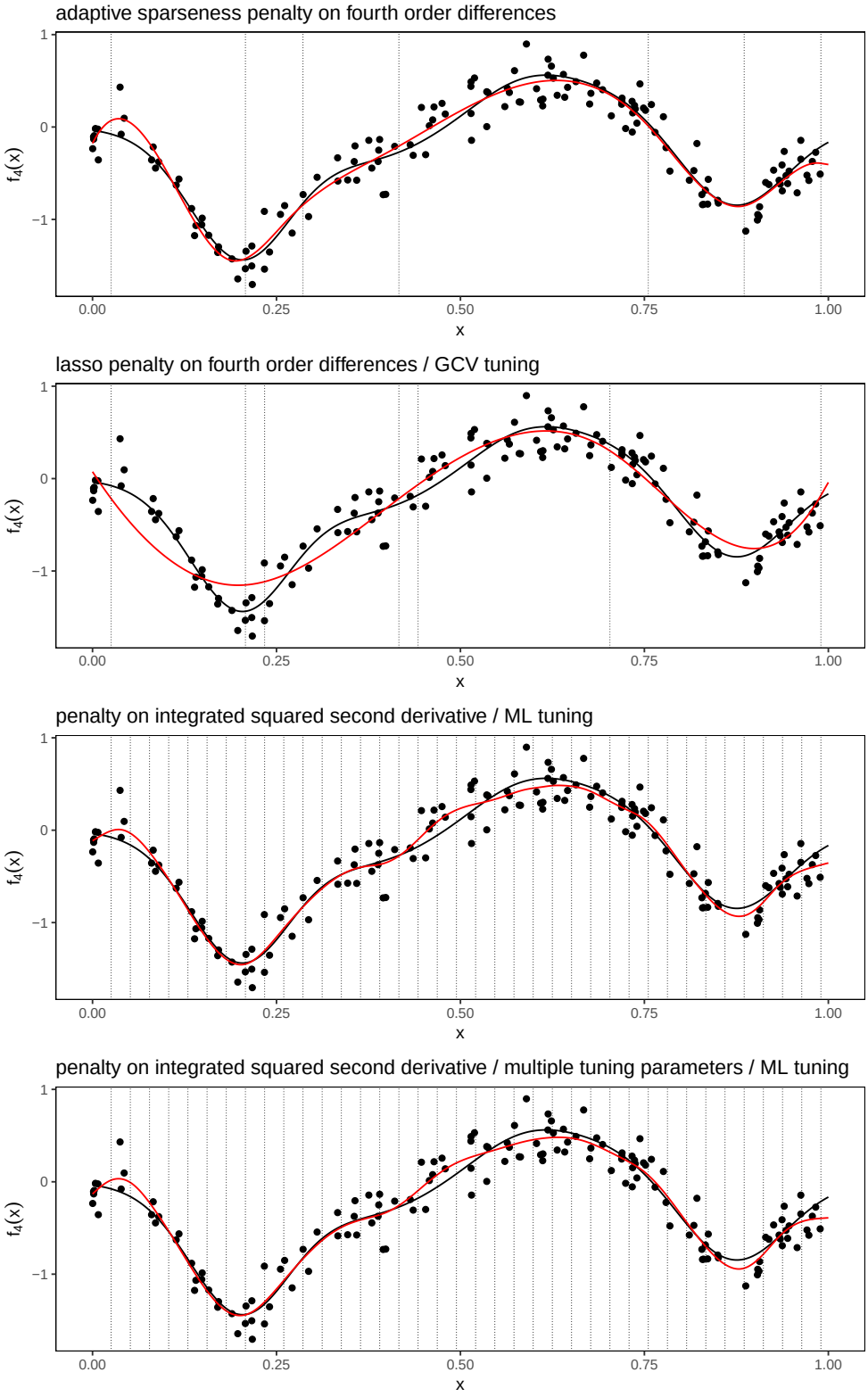

Figure 4.5: Estimates of $f_4(x)$, defined in equations (4.28), using a variety of methods, for an example data set. The points are the simulated data, the red curves are the fitted curves, the black curve is $f_4(x)$, and the vertical dotted lines are the final knot locations.

Method	IAB	ISB	IV	ISE	IAE	AT
AS	15.79	0.37	3.99	4.35	51.21	40.81
LASSO	57.52	5.02	2.10	7.12	66.87	39.91
AR / AIC	5.98	0.12	21.38	21.50	79.00	161.05
AR / BIC	13.22	0.31	6.55	6.86	57.33	161.05
AR / EBIC	16.39	0.45	5.32	5.78	57.66	161.05
CR / GCV	15.44	0.44	3.33	3.77	47.49	16.36
PS / GCV	14.87	0.41	3.30	3.71	47.13	16.97
CR / ML	9.28	0.18	3.59	3.76	47.85	34.52
PS / ML	9.42	0.18	3.48	3.66	47.18	36.27
AD / GCV	15.31	0.40	3.83	4.23	49.70	78.63
AD / ML	10.39	0.21	3.55	3.76	47.46	161.14

Table 4.3: Performance of methods in estimation of $f_3(x)$, defined in equations (4.28). All values are multiplied by 1000 to enhance readability. $n = 150$

Step function

For the fifth function, we consider a step-function whose jump points are unevenly spaced, given by

$$f_5(x) = \begin{cases} 0.2 & \text{if } 0 \leq x < 0.15 \\ -1 & \text{if } 0.15 \leq x < 0.60 \\ -0.2 & \text{if } 0.60 \leq x < 0.80 \\ 1 & \text{if } 0.80 \leq x \leq 1 \end{cases} . \tag{4.29}$$

We use a spline of order 1 to model $f_5(x)$, and only consider sparsity inducing estimation methods for this function, with a penalty on first order differences of the coefficients. The performance of the methods we consider is given in Table 4.5 while an example data set and fits are given in Figure 4.6. For this function, AS and LASSO perform similarly, and both do considerably better than AR.

4.8.2 Multiple functions

We now evaluate the performance of AS and CR / ML in the estimation of additive models with multiple functions. We leave out AR and LASSO because these methods require tuning parameter selection which becomes impractical in the presence of multiple tuning parameters. We also leave out AD which requires multiple tuning parameters per function to be estimated as this also becomes quite computationally costly in the presence of multiple functions. Furthermore, we compare AS only to CR / ML for simplicity as CR / ML was the method that generally appeared to perform best in the simulation studies with individual functions. In the example data sets and fits we show the data plotted are partial residuals. For an additive model

$$y_i = \sum_{j=1}^{p} f_j(x_{ji}) + \epsilon_i \ ,$$

the i-th partial residual for predictor x_j is given by $\hat{f}_j(x_{ij}) + e_i$, where $e_i = y_i - \sum_{j=1}^{p} \hat{f}_j(x_{ij})$.

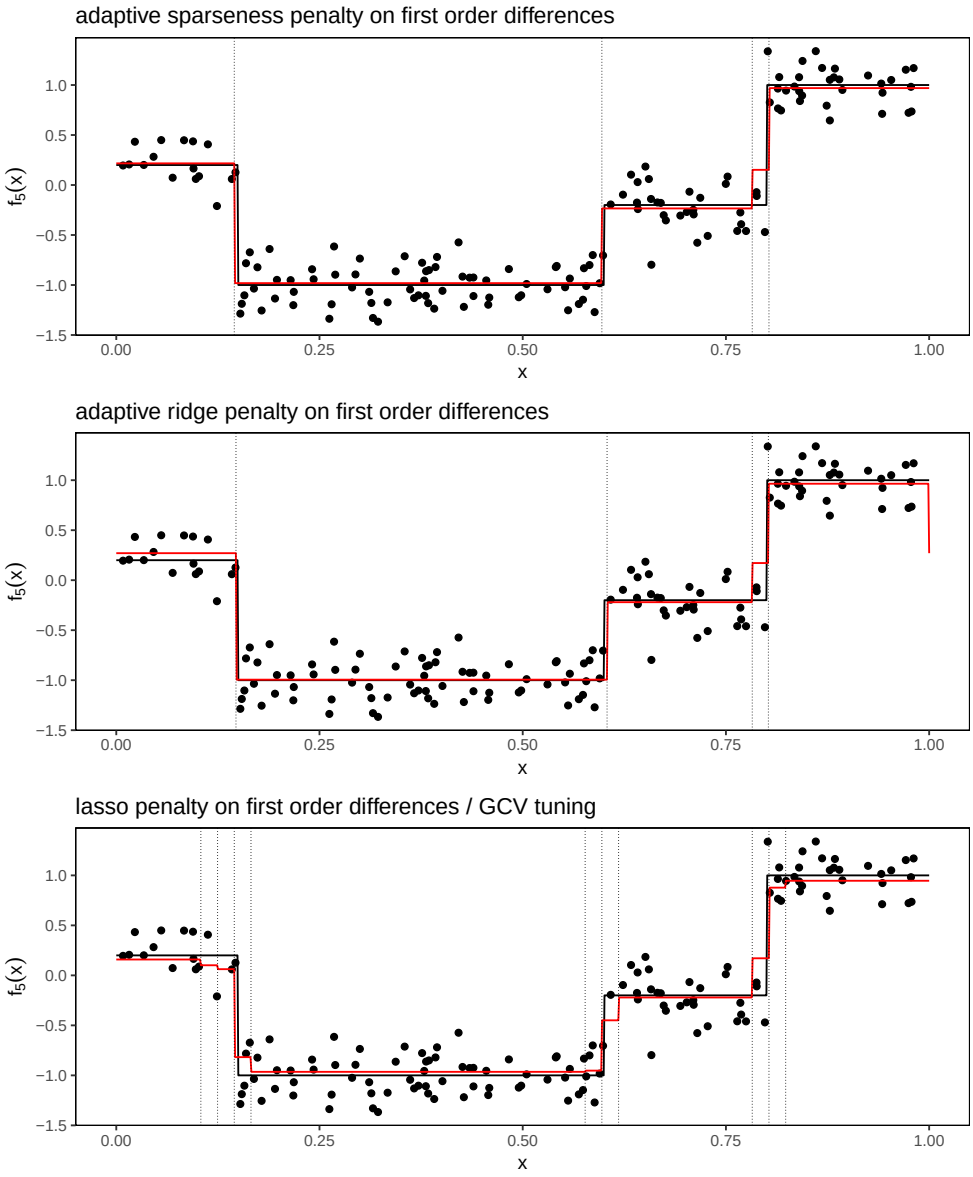

Figure 4.6: Estimates of $f_5(x)$, defined in equation (4.29), using a variety of methods, for an example data set. The points are the simulated data, the red curves are the fitted curves, the black curve is $f_5(x)$, and the vertical dotted lines are the final knot locations.

Method	IAB	ISB	IV	ISE	IAE	AT
AS	30.78	1.34	4.38	5.72	58.71	44.61
LASSO	111.36	22.09	1.88	23.98	117.93	17.13
AR / AIC	13.05	0.43	14.26	14.69	81.88	171.66
AR / BIC	20.36	0.70	6.90	7.61	62.43	171.66
AR / EBIC	25.71	1.01	6.54	7.55	62.70	171.66
CR / GCV	20.64	0.73	3.41	4.13	50.85	16.23
PS / GCV	20.03	0.68	3.35	4.04	50.33	16.14
CR / ML	18.65	0.61	3.29	3.90	49.47	34.18
PS / ML	18.96	0.63	3.22	3.84	49.12	33.96
AD / GCV	19.18	0.61	4.10	4.72	53.44	75.36
AD / ML	18.18	0.56	3.34	3.91	49.42	152.82

Table 4.4: Performance of methods in estimation of $f_4(x)$, defined in equations (4.28). All values are multiplied by 1000 to enhance readability. $n = 150$

Method	IAB	ISB	IV	ISE	IAE	AT
AS	26.44	12.57	3.03	15.60	51.47	55.40
LASSO	52.59	13.94	3.02	16.96	66.44	11.89
AR / AIC	43.61	17.79	8.79	26.58	84.70	238.26
AR / BIC	45.08	18.03	6.40	24.43	66.00	238.26
AR / EBIC	46.36	18.35	6.63	24.98	62.86	238.26

Table 4.5: Performance of methods in estimation of $f_5(x)$, defined in equation (4.29). All values are multiplied by 1000 to enhance readability. $n = 150$

Sine functions together

We now consider an additive model with two predictors, and use f_1 and f_2 as the functions to be estimated. The performance metrics are given in Table 4.6, and example data and fits in Figure 4.7. Similarly as when estimated individually, AS performs better than CR / ML for f_1 in terms of ISE. The performance is more comparable for f_2, however AS still performs slightly better.

Linear combinations of Gaussian densities together

We now consider an additive model with two predictors using f_3 and f_4. The performance metrics are given in Table 4.7 and example data and fits in Figure 4.8. CR / ML performs better than AS in terms of ISE for f_3, while for f_4 the ISE of the two methods is very close.

Sine functions and linear combinations of Gaussian densities together

Finally, we consider an additive model with four predictors, and use functions $f_1, \ldots, f_4$, to examine how well the methods perform in the presence of more predictors. In addition, we run the simulation experiment with $n = 150$, $n = 250$, and $n = 350$, to see how the methods respond to an increasing sample size. The performance metrics are given in Tables 4.8, 4.9, and 4.10. For $n = 250$ and $n = 350$, AS continues to

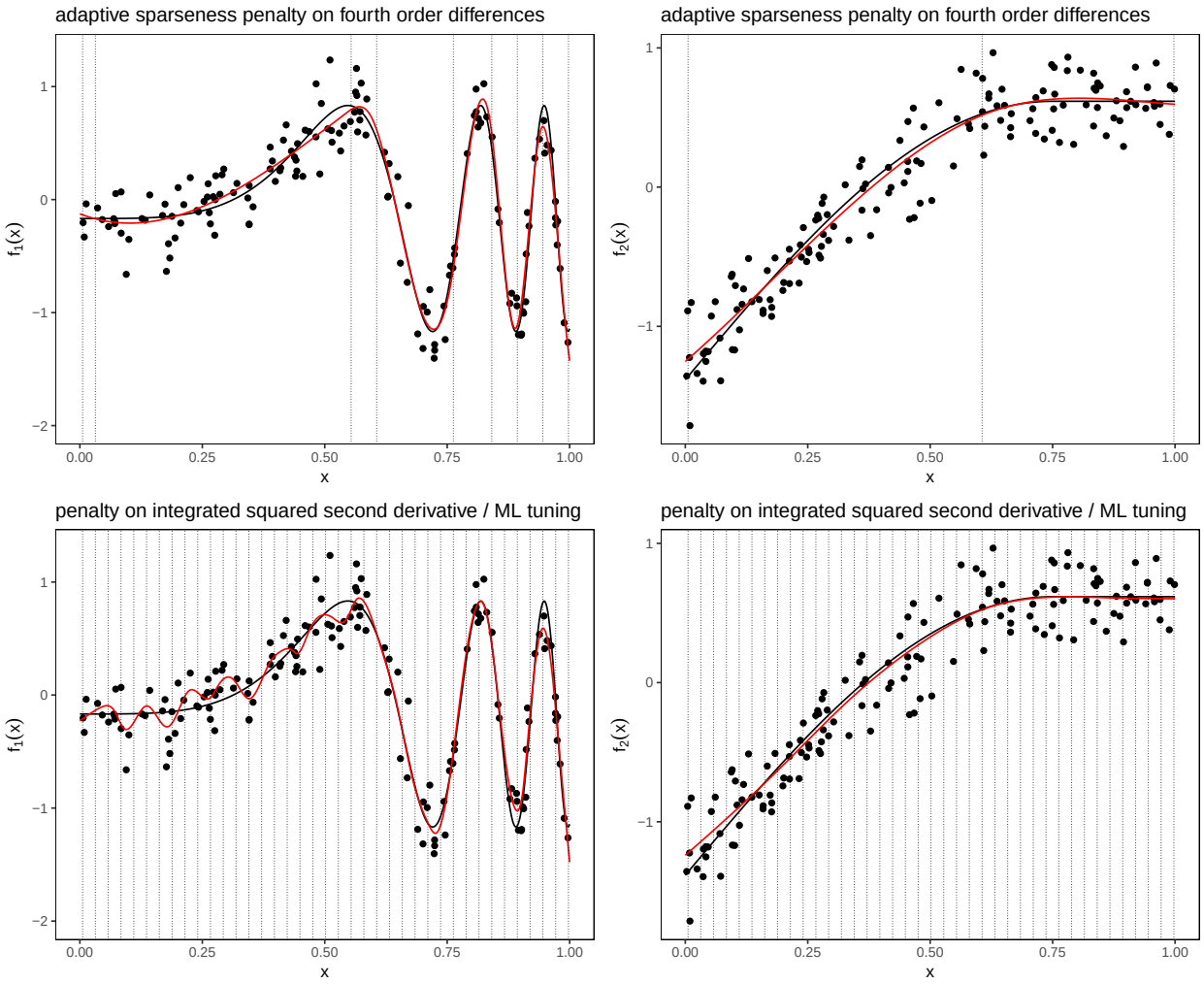

Figure 4.7: Estimates of $f_1(x)$ and f_2, defined in equations (4.27) and (4.27), using the adaptive sparseness method and penalisation on the integrated squared second derivative with ML tuning. The points are the partial residuals for each predictor, the red curves are the fitted curves, the black curves are $f_1(x)$ (left) and $f_2(x)$ (right), and the vertical dotted lines are the final knot locations.

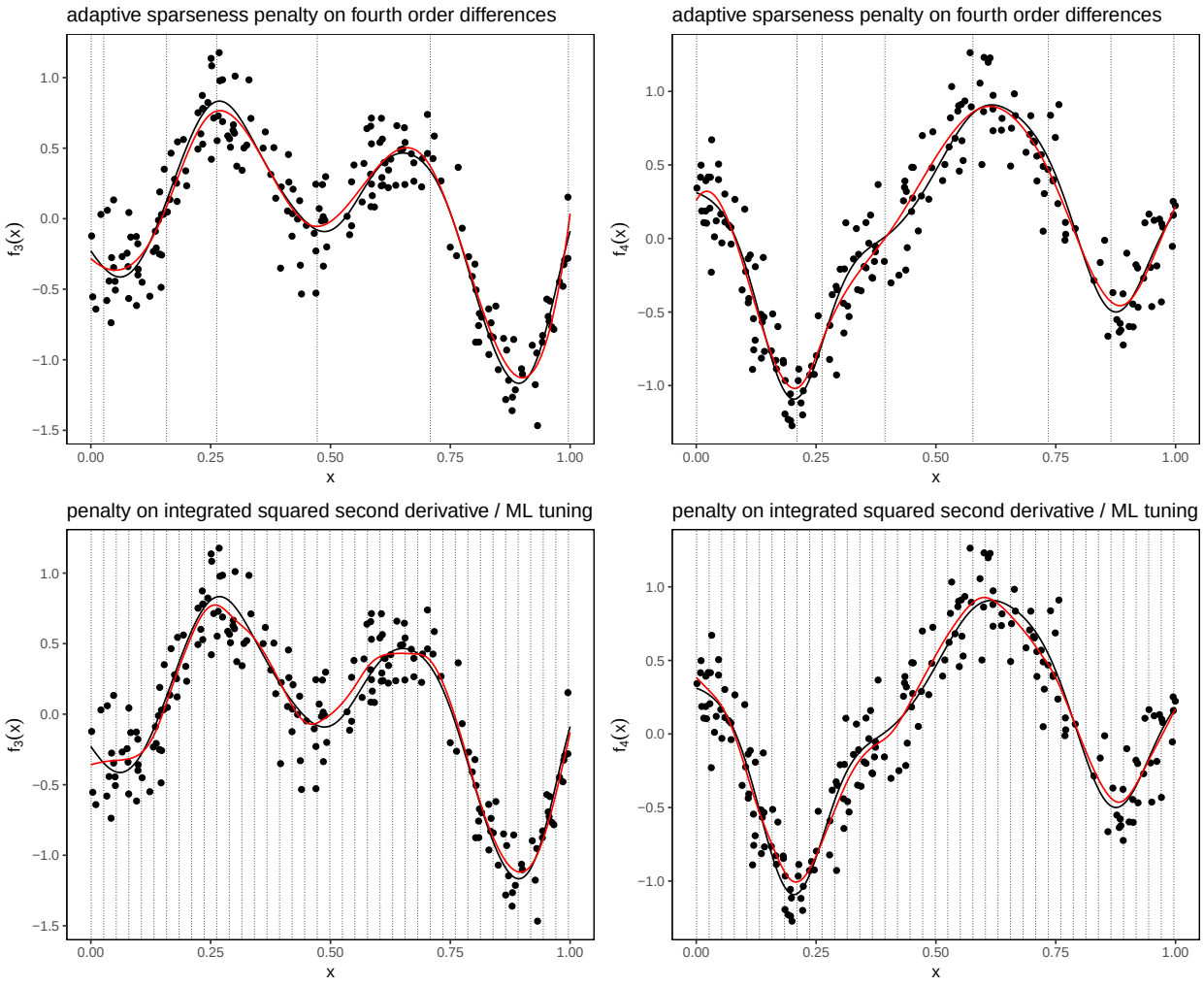

Figure 4.8: Estimates of $f_3(x)$ and f_4, defined in equation (4.28), using the adaptive sparseness method and penalisation on the integrated squared second derivative with ML tuning. The points are the partial residuals for each predictor, the red curves are the fitted curves, the black curves are $f_1(x)$ (left) and $f_2(x)$ (right), and the vertical dotted lines are the final knot locations.

Method		IAB	ISB	IV	ISE	IAE	AT
AS	f_1	16.55	0.99	4.44	5.43	54.62	64.59
	f_2	8.40	0.10	1.33	1.43	28.90	64.59
	$f_1 + f_2$	21.05	1.07	5.68	6.75	63.27	64.59
CR / ML	f_1	14.28	1.02	7.35	8.37	71.40	85.95
	f_2	6.75	0.08	1.46	1.54	28.48	85.95
	$f_1 + f_2$	17.38	1.05	7.71	8.76	73.49	85.95

Table 4.6: Performance of methods in estimation of $f_1(x)$ and $f_2(x)$ together. All values are multiplied by 1000 to enhance readability. $n = 150$

Method		IAB	ISB	IV	ISE	IAE	AT
AS	f_3	13.82	0.29	3.01	3.30	44.82	71.87
	f_4	9.10	0.13	3.10	3.22	43.97	71.87
	$f_3 + f_4$	14.94	0.35	5.53	5.88	60.41	71.87
CR / ML	f_3	14.30	0.36	2.51	2.87	41.83	81.27
	f_4	14.33	0.41	2.78	3.20	44.01	81.27
	$f_3 + f_4$	19.15	0.65	4.87	5.52	58.59	81.27

Table 4.7: Performance of methods in estimation of $f_3(x)$ and $f_4(x)$ together. All values are multiplied by 1000 to enhance readability. $n = 150$

perform well, achieving comparable performance for f_2, f_3, f_4 in terms of ISE, better performance for f_1, and better performance for the estimation of $\sum_{j=1}^{4} f_j$. For $n = 150$, CR / ML performs marginally better.

Method		IAB	ISB	IV	ISE	IAE	AT
AS	f_1	40.82	8.54	6.65	15.20	77.09	129.38
	f_2	15.08	0.33	1.85	2.18	35.23	129.38
	f_3	24.16	0.93	4.90	5.83	60.30	129.38
	f_4	22.80	0.83	6.13	6.97	61.22	129.38
	$\sum_j f_j$	42.52	4.06	13.31	17.37	101.15	129.38
CR / ML	f_1	32.38	4.84	9.16	14.01	86.65	613.41
	f_2	19.90	0.57	2.32	2.89	39.54	613.41
	f_3	17.65	0.43	4.63	5.06	55.01	613.41
	f_4	22.22	1.11	5.11	6.22	58.86	613.41
	$\sum_j f_j$	27.82	1.66	13.93	15.58	98.51	613.41

Table 4.8: Performance of methods in estimation of $f_1(x)$, $f_2(x)$, $f_3(x)$, and $f_4(x)$ together. All values are multiplied by 1000 to enhance readability. $n = 150$

Method		IAB	ISB	IV	ISE	IAE	AT
AS	f_1	12.73	0.76	3.81	4.58	47.65	164.22
	f_2	9.73	0.12	0.97	1.09	25.34	164.22
	f_3	13.25	0.23	2.58	2.81	41.14	164.22
	f_4	10.95	0.20	2.83	3.03	42.65	164.22
	$\sum_j f_j$	21.20	0.78	8.29	9.07	74.78	164.22
CR / ML	f_1	18.97	1.61	5.00	6.61	62.15	810.31
	f_2	9.10	0.12	1.00	1.12	25.01	810.31
	f_3	13.80	0.39	2.30	2.68	40.17	810.31
	f_4	16.55	0.55	2.58	3.12	43.65	810.31
	$\sum_j f_j$	26.00	1.34	8.87	10.20	79.95	810.31

Table 4.9: Performance of methods in estimation of $f_1(x)$, $f_2(x)$, $f_3(x)$, and $f_4(x)$ together. All values are multiplied by 1000 to enhance readability. $n = 250$

Method		IAB	ISB	IV	ISE	IAE	AT
AS	f_1	11.07	0.59	2.54	3.13	39.82	161.03
	f_2	6.19	0.05	0.74	0.79	21.12	161.03
	f_3	12.51	0.24	1.93	2.17	36.32	161.03
	f_4	6.95	0.07	1.93	2.00	34.19	161.03
	$\sum_j f_j$	17.79	0.62	6.15	6.77	64.50	161.03
CR / ML	f_1	15.56	0.85	3.56	4.40	51.23	836.97
	f_2	5.02	0.04	0.72	0.75	20.46	836.97
	f_3	12.29	0.24	1.60	1.84	33.64	836.97
	f_4	15.22	0.44	1.73	2.17	35.89	836.97
	$\sum_j f_j$	21.33	0.86	6.59	7.45	68.44	836.97

Table 4.10: Performance of methods in estimation of $f_1(x)$, $f_2(x)$, $f_3(x)$, and $f_4(x)$ together. All values are multiplied by 1000 to enhance readability. $n = 350$

4.8.3 Conclusions from simulation studies

The results of the simulation studies presented in this section show promise for our proposed method of using adaptive sparseness to estimate spline functions in a locally adaptive way. No single method performed best across all simulation settings but adaptive sparseness remained competitive with the other methods in all the simulation studies we considered. In particular, adaptive sparseness almost always performed better in terms of integrated squared error than the LASSO and the adaptive ridge method of Goepp et al. (2018). It was also shown that when estimating functions that could benefit from locally adaptive estimation, adaptive sparseness performed very well, while also taking less time to run than competing methods. In addition, adaptive sparseness was shown to perform well when estimating multiple functions, comparing favourably to cubic regression splines with an integrated squared second derivative roughness penalty and tuning parameters selected by maximum likelihood.

4.9 Abalone data

To demonstrate our methodology on a real data set we consider a data set that was first presented by Warwick J and Marine Research Laboratories (1994) and subsequently studied by Waugh (1995). The data contain over 4,000 observations on the age and various physical measurements of abalones. The age of an abalone can be determined through a time consuming process that involves cutting the shell and counting the number of rings under a microscope. The problem considered by Waugh (1995) was that of predicting the age of abalones based on other physical measurements that are easier to obtain. The data are freely available in the UCI Machine Learning Repository Lichman (2013), and contain information on the following variables:

1. Length (longest shell measurement, millimeters)
2. Diameter (perpendicular to length, millimeters)
3. Height (with meat in shell, millimeters)
4. Whole weight (grams)
5. Shucked weight (weight of meat, grams)
6. Viscera weight (gut weight after bleeding, grams)
7. Shell weight (after being dried, grams)
8. Sex (male, female, or infant)
9. Rings (+1.5 gives the age in years of the abalone)

In this section we consider an additive model for the rings given by

$$y_i = \beta_0 + \beta_1 z_{1i} + \beta_2 z_{2i} + \sum_{j=1}^{7} f_j(x_{ji}) + \epsilon_i \quad (i = 1, \ldots, 4177) \ , \tag{4.30}$$

where $z_{11}, \ldots, z_{1n}$ are indicator variables taking the value 1 if the corresponding abalone is male and 0 otherwise, and $z_{21}, \ldots, z_{2n}$ are indicator variables taking the value 1 if the corresponding abalone is an

infant and 0 otherwise. The variables $x_1, \ldots, x_7$ represent all the continuous predictors available, with x_{ji} denoting the i-th case of the j-th predictor. We represent the functions $f_1, \ldots, f_7$ using cubic splines with 40 equally spaced knots. We fit the model using the adaptive sparseness prior on fourth order differences of the coefficients of the b-splines as well as with an integrated squared second derivative penalty on the spline with tuning parameter selection by GCV. The estimated functions are shown in Figure 4.9 along with partial residuals for each variable. Table 4.11 displays the estimates of the coefficients $\beta_0, \beta_1, \beta_2$ using each method along with the time taken to compute the fits. The parameter estimates and fitted functions are similar using both approaches, however adaptive sparseness take a fraction of the time to run. We will return to the abalone data in Chapter 5, where we will explore the importance of each continuous predictor in explaining variability in the age of abalones.

		CR / GCV	AS
coefs	intercept	10.109	10.111
	male	0.021	0.013
	infant	-0.571	-0.566
time		5.893	0.753

Table 4.11: Estimates of the coefficients in model (4.30) along with time taken to obtain the estimates.

4.10 Conclusions and discussion

In this chapter we proposed the use of adaptive sparseness for the estimation of locally adaptive splines, and showed how it can be used to estimate additive models. Enforcing sparsity in m-th order differences of b-spline coefficients appears to be an effective way to achieve local adaptivity when estimating smooth functions, and of all the sparsity inducing methods we tested, adaptive sparseness was the best performing across all the simulation studies we ran. Remarkably, adaptive sparseness is tuning parameter free, and yet in the simulation studies we considered it performed better than the LASSO and the method of Goepp et al. (2018), both of which used tuning parameter selection techniques. The proposed method also performed competitively with some of the methods implemented in the `R` package `mgcv`. While not as general in its approach as some of the ℓ_2 regularisation based methods available in `mgcv`, the lack of tuning parameters in adaptive sparseness means that it can easily be applied to additive models with a relatively large number of predictors, which other estimation methods might struggle to handle. Paired with good performance in terms of integrated squared error, and the ability to provide locally adaptive fits, there are cases where adaptive sparseness could be a very useful alternative to existing methods.

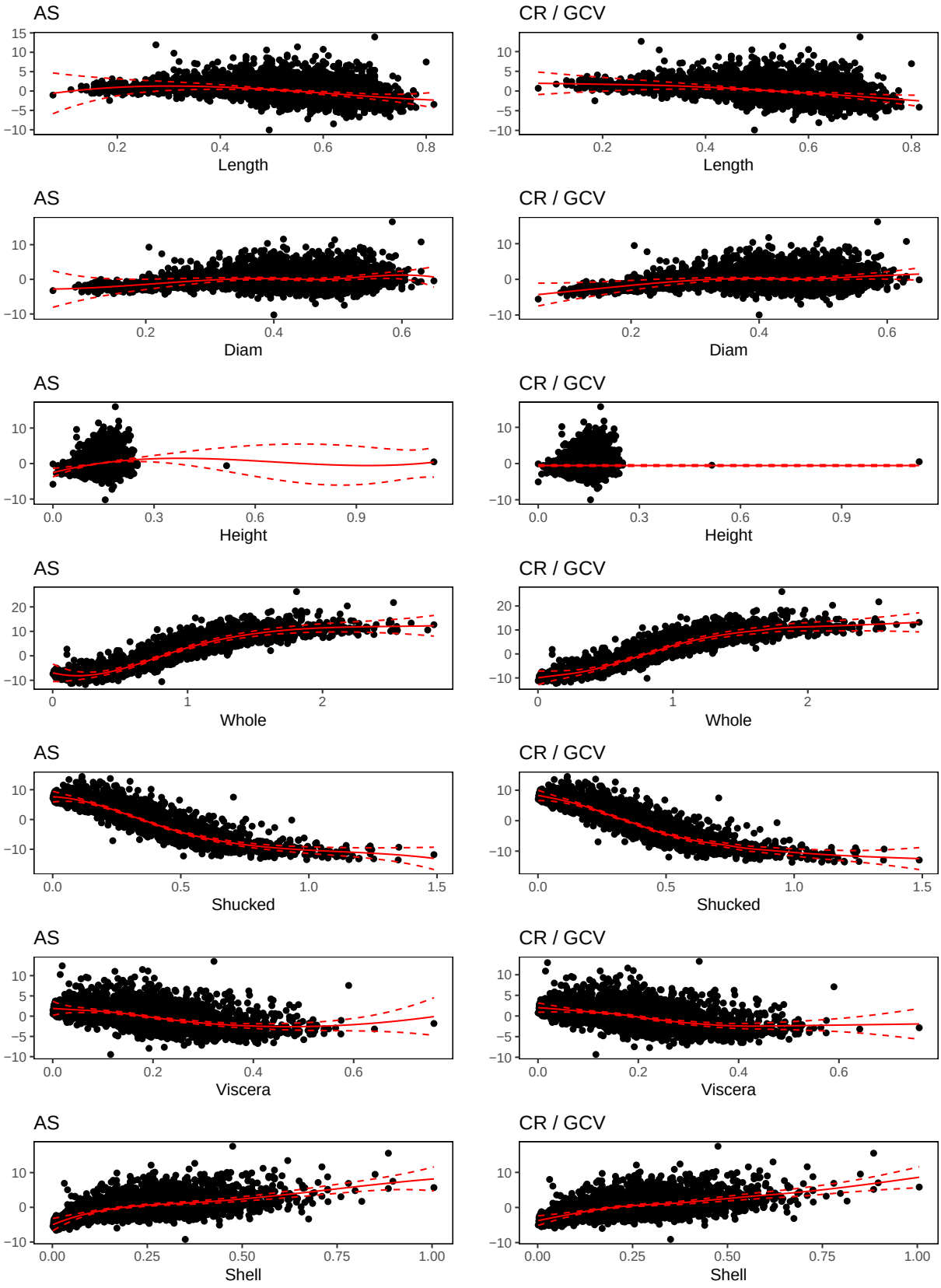

Figure 4.9: Estimates of the functions in model (4.30) along with the partial residuals for each variable.

Chapter 5

Relative importance of terms in models with smooth components

5.1 Preamble

In Chapters 3 and 4 we explored the use of b-splines in estimating smooth functions, focusing on various approaches to the estimation of the b-spline coefficients. In this chapter we shift focus and address a different problem. Given a model with smooth components that we have estimated, it may be of interest to either rank the importance of the functional terms or compare the relative importance of two functional terms. To approach this problem, we build on previous work examining the relative importance of groups of predictors in GLMs, expanding on it to apply in the context of GAMs estimated using the methods introduced in Chapters 3 and 4. We begin by introducing the problem in Section 5.2, and introduce our proposed method in Section 5.3. In Section 5.4 we return to the abalone data that we introduced in Chapter 4, and use our methodology to estimate the individual importance of each explanatory variable in predicting the age of abalones. Finally, in Section 5.5 we introduce a data set in which perceived prestige in a variety of occupations is recorded along with average education and income levels in each occupation, and we use our methodology to estimate the relative importance of education over income in explaining perceived prestige of occupations. For both data sets, we also carry out a simulation study in which we examine the performance, in a frequentist context, of the Bayesian inferential procedures we propose.

5.2 Introduction

Consider a GLM

$$g\Big(E(y_i)\Big) = \boldsymbol{x}_{1i}^T\boldsymbol{\beta}_1 + \boldsymbol{x}_{2i}^T\boldsymbol{\beta}_2 \quad (i = 1, \ldots, n) \ ,$$

where $\boldsymbol{x}_{1i}$ and $\boldsymbol{x}_{2i}$ are centered (i.e. $\sum_i \boldsymbol{x}_{1i}$ and $\sum_i \boldsymbol{x}_{2i}$ are both vectors of zeros) p_1- and p_2-dimensional vectors of predictors respectively, y_i is a scalar response that follows an exponential family distribution with dispersion parameter ϕ, $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ are p_1- and p_2-dimensional vectors of parameters to be estimated, and g is a sufficiently smooth monotonic link function. Now suppose a researcher is interested in quantifying which group of predictors, $\{\boldsymbol{x}_{1i}\}_{i=1}^n$ or $\{\boldsymbol{x}_{2i}\}_{i=1}^n$, is more important in terms of explaining the variability present in

$E(\boldsymbol{y}) = E\Big((y_1, \ldots, y_n)^T\Big)$. Towards this end, Silber et al. (1995) propose the relative importance statistic

$$\omega = \frac{\boldsymbol{\beta}_1^T X_1^T X_1 \boldsymbol{\beta}_1}{\boldsymbol{\beta}_2^T X_2^T X_2 \boldsymbol{\beta}_2} \ , \quad \text{where } X_j = \begin{pmatrix} \boldsymbol{x}_{j1}^T \\ \vdots \\ \boldsymbol{x}_{jn}^T \end{pmatrix} , \quad j = 1, 2 \ ,$$

which is the ratio of the variances of the linear predictors $X_1\boldsymbol{\beta}_1$ and $X_2\boldsymbol{\beta}_2$ respectively. An ω value of 1 indicates equal importance, $\omega > 1$ indicates that X_1 is more important than X_2, and $\omega < 1$ indicates that X_2 is more important than X_1. Given estimates $\hat{\boldsymbol{\beta}}_1$ and $\hat{\boldsymbol{\beta}}_2$ of the parameter vectors, the estimated relative importance is

$$\hat{\omega} = \frac{\hat{\boldsymbol{\beta}}_1^T X_1^T X_1 \hat{\boldsymbol{\beta}}_1}{\hat{\boldsymbol{\beta}}_2^T X_2^T X_2 \hat{\boldsymbol{\beta}}_2} \ .$$

Because $\hat{\omega}$ is strictly positive, Silber et al. (1995) approximate the distribution of $\log \hat{\omega}$ under the model as a Gaussian whose mean and variance can be approximated using the delta method (Cramer, 1999, page 353) as

$$\log \hat{\omega} \ \dot{\sim} \ \mathcal{N}(\log \omega, w^T \Sigma_{\hat{\boldsymbol{\beta}}} w) \ , \tag{5.1}$$

where $\dot{\sim}$ denotes asymptotic distribution,

$$w = 2 \begin{pmatrix} \frac{X_1^T X_1 \boldsymbol{\beta}_1}{\boldsymbol{\beta}_1^T X_1^T X_1 \boldsymbol{\beta}_1} \\ \frac{-X_2^T X_2 \boldsymbol{\beta}_2}{\boldsymbol{\beta}_2^T X_2^T X_2 \boldsymbol{\beta}_2} \end{pmatrix} \ , \tag{5.2}$$

and $\Sigma_{\hat{\boldsymbol{\beta}}}$ is the covariance matrix of

$$\hat{\boldsymbol{\beta}} = \begin{pmatrix} \hat{\boldsymbol{\beta}}_1 \\ \hat{\boldsymbol{\beta}}_2 \end{pmatrix} \ .$$

Given an estimate $\hat{\Sigma}_{\hat{\boldsymbol{\beta}}}$ of $\Sigma_{\hat{\boldsymbol{\beta}}}$, the quantity $w^T \Sigma_{\hat{\boldsymbol{\beta}}} w$ can be estimated as $\hat{w}^T \hat{\Sigma}_{\hat{\boldsymbol{\beta}}} \hat{w}$, where $\hat{w}$ consists of replacing the unknown quantities in (5.2) by their respective estimates. Using the distribution in (5.1), approximate hypothesis tests and confidence intervals can be conducted for ω which are only asymptotically exact.

We propose to extend the work of Silber et al. (1995) to the context of GAMs. When the functional components of GAMs are assumed to be linear combinations of b-splines, the GAM reduces to a GLM with a particular grouping of coefficients. Using the same formulation as in Section 3.6, a GAM can be written as

$$g\Big(E(y_i)\Big) = \sum_{j=1}^{p} \boldsymbol{\alpha}_j^T \boldsymbol{b}_{ij} \ , \tag{5.3}$$

where $\boldsymbol{b}_{ij}$ is the vector of b-splines corresponding to the i-th case of the j-th predictor, and $\boldsymbol{\alpha}_j$ is the vector of b-spline coefficients for the j-th function. With a GAM expressed as (5.3) the individual importance of predictor j can be defined to be

$$\psi_j = \boldsymbol{\alpha}_j^T B_j^T B_j \boldsymbol{\alpha}_j \ ,$$

where B_j is a matrix with i-th row $\boldsymbol{b}_{ij}$. Similarly, the relative importance of predictor j over predictor l can be defined to be $\omega_{jl} = \psi_j / \psi_l$. In this chapter we discuss inference for ψ_j and ω_{jl} when the GAM in (5.3) is estimated either using Bayesian ℓ_2 regularisation, or adaptive sparseness.

5.3 Proposed methodology

In contrast to the approach in Silber et al. (1995), in which the maximum likelihood estimator was used for the coefficients of a GLM, the estimation procedures we consider for GAMs are Bayesian. To carry out inference for the individual and relative importance of predictors in GAMs, we consider approximations to the posterior distributions $\psi_j|\boldsymbol{y}$ and $\omega_{jl}|\boldsymbol{y}$ respectively. More specifically, consider the estimator

$$\hat{\boldsymbol{\alpha}} = \arg\max_{\boldsymbol{\alpha}} \log p(\boldsymbol{\alpha}|\boldsymbol{y}) \ ,$$

where $\boldsymbol{\alpha}^T = (\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_p)$, and $p(\boldsymbol{\alpha}|\boldsymbol{y})$ arises from the specific Bayesian formulation used (in our case either ℓ_2 regularisation or adaptive sparsity). Assume that asymptotically we have

$$\boldsymbol{\alpha}|\boldsymbol{y} \mathrel{\dot\sim} \mathcal{N}\big(\hat{\boldsymbol{\alpha}}, \Sigma\big) \ .$$

Then, using the delta method, for the individual importances we obtain

$$\log\big(\psi_j|\boldsymbol{y}\big) \mathrel{\dot\sim} \mathcal{N}\big(\log(\hat{\psi}_j), \boldsymbol{w}_j^T \Sigma_j \boldsymbol{w}_j\big), \tag{5.4}$$

where

$$\log(\hat{\psi}_j) = \log\big(\hat{\boldsymbol{\alpha}}_j^T B_j^T B_j \hat{\boldsymbol{\alpha}}_j\big) \ ,$$

and we define

$$\boldsymbol{w}_j = \frac{2 B_j^T B_j \boldsymbol{\alpha}_j}{\boldsymbol{\alpha}_j^T B_j^T B_j \boldsymbol{\alpha}_j} \ .$$

and

$$\Sigma_j = \mathrm{cov}(\boldsymbol{\alpha}_j|\boldsymbol{y}) \ .$$

Similarly, for the relative importance of predictor j over predictor l, we obtain

$$\log\big(\omega_{jl}|\boldsymbol{y}\big) \mathrel{\dot\sim} \mathcal{N}(\log(\hat{\omega}_{jl}), \boldsymbol{w}_{jl}^T \Sigma_{jl} \boldsymbol{w}_{jl}), \tag{5.5}$$

where

$$\log\big(\hat{\omega}_{jl}\big) = \log(\hat{\psi}_j) - \log(\hat{\psi}_l) \ ,$$

and we have now defined

$$\boldsymbol{w}_{jl} = \begin{pmatrix} \boldsymbol{w}_j \\ -\boldsymbol{w}_l \end{pmatrix} \ .$$

and

$$\Sigma_{jl} = \mathrm{cov}(\boldsymbol{\alpha}_j, \boldsymbol{\alpha}_l|\boldsymbol{y}) \ .$$

The idea now is to use the distributions in (5.4) and (5.5) to carry out approximate inference for ψ_j and ω_{jl}. For example, approximate 95% credible intervals for $\log \psi_j$ can be computed as

$$\Big[\log \hat{\psi}_j - 1.96\sqrt{\boldsymbol{w}_j^T \Sigma_j \boldsymbol{w}_j}, \log \hat{\psi}_j + 1.96\sqrt{\boldsymbol{w}_j^T \Sigma_j \boldsymbol{w}_j}\ \Big] \ .$$

To estimate the asymptotic covariance Σ of $\boldsymbol{\alpha}|\boldsymbol{y}$, Wood et al. (2016) provides an approximation based on a Taylor expansion for the distribution $p(\boldsymbol{\alpha}|\boldsymbol{y})$ in the context of Bayesian ℓ_2 regularisation. The expressions are somewhat cumbersome and we therefore do not report them here, however the reader may refer to Wood et al. (2016, Section 4) for further details.

In the context of adaptive sparseness, we propose the inverse Hessian matrix of $-\log p(\boldsymbol{\alpha}|\boldsymbol{y})$. We do not directly have access to the log posterior $\log p(\boldsymbol{\alpha}|\boldsymbol{y})$, however given that the EM algorithm is used to maximise it, we can use methods that have been developed to compute the Hessian using only quantities necessary for the EM algorithm. Specifically, Oakes (1999) gives the equation

$$\frac{\partial \log p(\boldsymbol{\alpha}|\boldsymbol{y})}{\partial \boldsymbol{\alpha} \partial \boldsymbol{\alpha}^T} = \left\{ \frac{\partial Q(\boldsymbol{\alpha}|\hat{\boldsymbol{\alpha}})}{\partial \boldsymbol{\alpha} \partial \boldsymbol{\alpha}^T} + \frac{\partial Q(\boldsymbol{\alpha}|\hat{\boldsymbol{\alpha}})}{\partial \boldsymbol{\alpha} \partial \hat{\boldsymbol{\alpha}}^T} \right\}_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}} .$$

With the Q function used in the EM algorithm for adaptive sparseness, with a normally distributed response and model matrix X, we have

$$\begin{aligned} \frac{\partial Q(\boldsymbol{\alpha}|\hat{\boldsymbol{\alpha}})}{\partial \boldsymbol{\alpha} \partial \boldsymbol{\alpha}^T}\Big|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}} &= -X^T X/\hat{\sigma}^2 - W \\ \frac{\partial Q(\boldsymbol{\alpha}|\hat{\boldsymbol{\alpha}})}{\partial \boldsymbol{\alpha} \partial \hat{\boldsymbol{\alpha}}^T}|_{\boldsymbol{\alpha}=\hat{\boldsymbol{\alpha}}} &= 2W \end{aligned}$$

from which we obtain

$$\frac{\partial \log p(\boldsymbol{\alpha}|\boldsymbol{y})}{\partial \boldsymbol{\alpha} \partial \boldsymbol{\alpha}^T} = -X^T X/\hat{\sigma}^2 + W ,$$

where W is a diagonal matrix with j-th diagonal element $1/\hat{\alpha}_j^2$.

5.4 Abalone data

We now return to the abalone data introduced in Chapter 4. Recall the model we considered was given by

$$y_i = \beta_0 + \beta_1 z_{1i} + \beta_2 z_{2i} + \sum_{j=1}^{7} f_j(x_{ji}) + \epsilon_i \quad (i = 1, \ldots, 4177) , \tag{5.6}$$

where y_i is the number of rings in the shell of the i-th abalone, $z_{11}, \ldots, z_{1n}$ is an indicator variable taking the value 1 if the corresponding abalone is male and 0 otherwise, $z_{21}, \ldots, z_{2n}$ is an indicator variable taking the value 1 if the corresponding abalone is an infant and 0 otherwise. The variables $x_1, \ldots, x_7$ represent all the continuous predictors available, given by

1. Length (longest shell measurement, millimeters)
2. Diameter (perpendicular to length, millimeters)
3. Height (with meat in shell, millimeters)
4. Whole weight (grams)
5. Shucked weight (weight of meat, grams)
6. Viscera weight (gut weight after bleeding, grams)
7. Shell weight (after being dried, grams)

Once the model has been fitted, we can rank the importance of each continuous predictor. Table 5.1 displays the log-importance, along with approximate 95% credible intervals for each continuous predictor, when the model is estimated using Bayesian ℓ_2 regularisation with an integrated squared second derivative roughness penalty on each function. Similarly, Table 5.2 displays the individual importance, along with

approximate 95% credible intervals, when the model is estimated using adaptive sparseness. With both methods, height, diameter, and length are deemed the least important predictors, with all the weight related variables appearing to be more important. The whole weight is estimated to be the most important of the continuous predictors.

CR / ML	log importance	lower 95%	upper 95%
Whole	12.05	11.79	12.32
Shucked	11.63	11.46	11.79
Shell	9.22	8.65	9.79
Viscera	8.83	8.30	9.36
Length	8.07	6.91	9.22
Diameter	7.52	5.79	9.25
Height	6.88	6.07	7.69

Table 5.1: Ranking the importance of each continuous predictor, along with approximate 95% credible intervals, in model (5.6), when using ℓ_2 regularisation with tuning parameters selected by maximum likelihood.

AS	log importance	lower 95%	upper 95%
Whole	12.03	11.76	12.30
Shucked	11.61	11.44	11.77
Shell	9.33	8.79	9.87
Viscera	8.95	8.43	9.47
Length	7.94	6.83	9.04
Diameter	7.27	5.60	8.93
Height	7.07	6.36	7.79

Table 5.2: Ranking the importance of each continuous predictor, along with approximate 95% credible intervals, in model (5.6), when using adaptive sparseness as the estimation procedure.

We now carry out a simulation study in which we examine the frequentist performance of the Bayesian credible intervals we consider. For both methods, we assume that the estimated values are the true values of the model, and we simulate $10,000$ new response vectors from the conditional model

$$p(\boldsymbol{y}|\boldsymbol{\gamma}, \boldsymbol{z}_1, \boldsymbol{z}_2, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_7) \ ,$$

where $\boldsymbol{\gamma}$ is the entire vector of parameters estimated. For each response vector we compute approximate 95% credible intervals for the log-importance of each continuous predictor. We then compute the coverage probability of the intervals produced. The results for both methods are given in Table 5.3. We observe that for both methods and for all the continuous predictors the approximate 95% credible intervals achieve close to 95% coverage.

5.5 Occupational prestige data

We now consider a data set in which the perceived prestige (Pineo-Porter score) of 102 occupations is recorded, along with average education (years) and income levels (Canadian dollars) of sampled employees

	Length	Diam	Height	Whole	Shucked	Viscera	Shell
CR / ML	0.937	0.935	0.964	0.944	0.948	0.956	0.944
AS	0.957	0.930	0.956	0.953	0.951	0.960	0.932

Table 5.3: Frequentist coverage of approximate 95% credible intervals for the log importance of each continuous predictor in the model (5.6).

in each occupation, in Canada, 1971. The data are available in the `R` package `car` (Fox and Weisberg, 2011). We fit the additive model

$$y_i = \beta_0 + f_1(x_{1i}) + f_2(x_{2i}) + \epsilon_i \ , \tag{5.7}$$

where y_i represents perceived prestige, x_{1i} the average income, and x_{2i} the average education, associated with the i^{th} occupation respectively. f_1 and f_2 are unknown smooth functions to be estimated, i uniquely enumerates the recorded occupations, and $\epsilon_i \sim \mathcal{N}(0, \sigma^2)$. As for the abalone data, we estimate the model using both Bayesian ℓ_2 regularisation and adaptive sparseness. The fitted functions plus and minus 2 standard deviations, are given in Figure 5.1. The estimated functions appear to be similar using both methods.

The estimated relative importance of education over income in explaining variability in perceived prestige of occupations, along with approximate 95% confidence intervals, for both estimation methods, is given in Table 5.4. According to both methods, average education levels are nearly three times more important than average income levels in explaining variability in perceived prestige of occupations, with similar approximate 95% credible intervals as well.

	relimp education/income	lower 95%	upper 95%
CR / ML	2.96	1.34	6.53
AS	2.82	1.31	6.06

Table 5.4: Estimates, along with approximate 95% credible intervals, of the relative importance of education over income in explaining variability in perceived prestige of occupations.

We now carry out a similar simulation study as the one we carried out for the abalone data in Section 5.4, simulating, for each method, $10,000$ response vectors from the conditional model

$$p(\boldsymbol{y}|\boldsymbol{\gamma}, \boldsymbol{x}_1, \boldsymbol{x}_2) \ ,$$

where $\boldsymbol{\gamma}$ denotes the full estimated parameter vector. Instead of computing approximate credible intervals for the relative importance of education over income though, we compute p-values for the null hypothesis $\omega = \omega_{\text{true}}$, where ω_{true} is the estimated relative importance that we take to be the true value for the purpose of the simulation. The p-values are computed as

$$2\left[1 - \Phi\Big(\big|(\log\hat{\omega} - \log\omega_{\text{true}})/\sigma\big|\Big)\right] \ ,$$

where $\sigma = \sqrt{\boldsymbol{w}_{12}^T \Sigma_{12} \boldsymbol{w}_{12}}$ as discussed in Section 5.3, and $\Phi(\cdot)$ denotes the cumulative distribution function of the standard normal distribution. Histograms of the $10,000$ p-values produced in this way are given in in Figure 5.2. For both methods, the histograms indicate adequate similarity to the uniform distribution, implying that the test for the null hypothesis ω_{true} has approximately the correct size at every significance level.

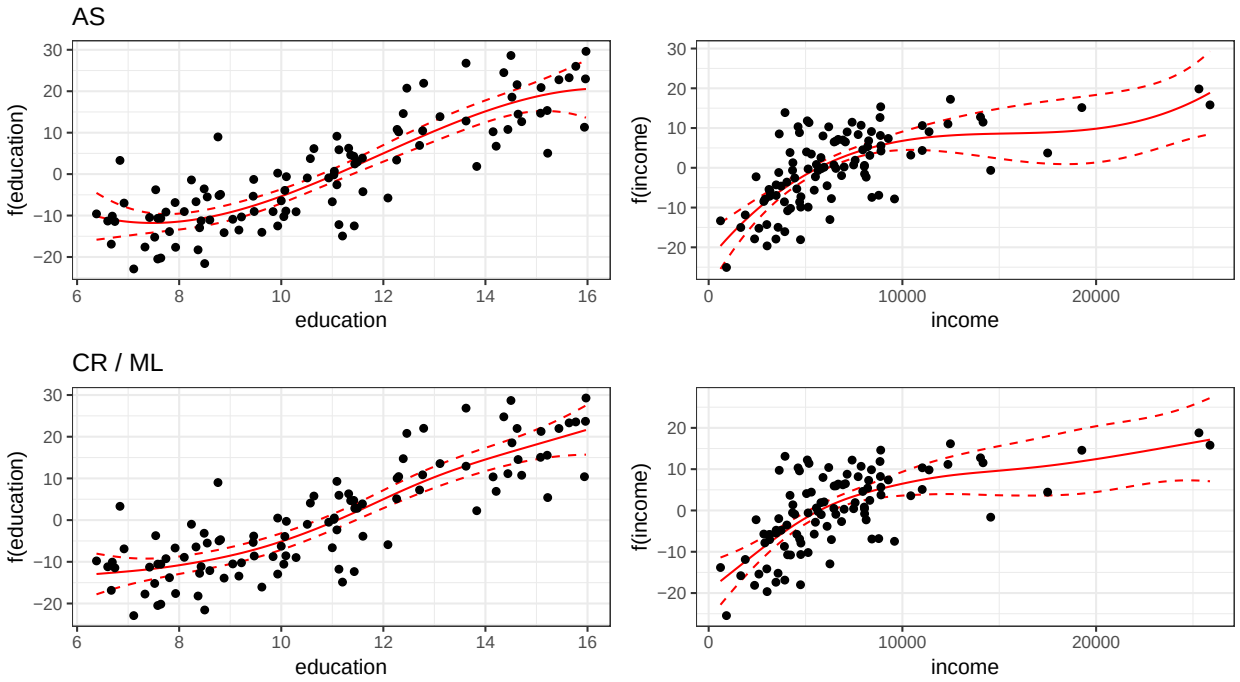

Figure 5.1: Estimated functions (solid lines) from model (5.7) ± 2 standard deviations (dotted lines), along with partial residuals (points).

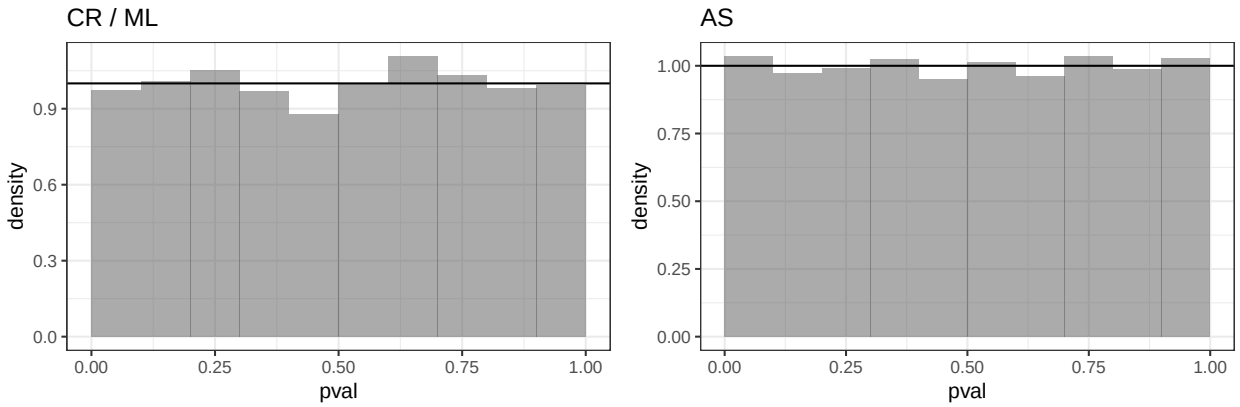

Figure 5.2: Empirical distribution of p-values corresponding to the hypothesis test $\omega = \omega^{\text{true}}$.

5.6 Conclusion & discussion

In this chapter we explored estimation and inference for the individual and relative importance of terms in GAMs that are estimated either using Bayesian ℓ_2 regularisation or adaptive sparseness. We demonstrated the use of the method using two data sets, in which the method was able to aid interpretation of the models estimated. For the abalone data, we were able to conclude that weight related variables are more important than size related variables in explaining variability in age between abalones. For the occupational prestige data we were able to conclude that average education levels are more important than average income levels in explaining variability in perceived prestige between occupations.

Although the method we proposed to estimate and perform inference for the individual and relative importance of predictors is Bayesian, in the simulation studies we considered based on the abalone data and occupational prestige data frequentist inferential procedures based on the Bayesian estimators were shown to perform well. Estimating the individual and relative importance of predictors can be a simple way to gain further insight into the results produced by GAMs, aiding their interpretability and usefulness.

Chapter 6

Structural smooth modelling

6.1 Preamble

In the previous chapters of this thesis we explored techniques for estimation and inference in models with smooth components, focusing on generalized additive models. GAMs are a type of regression model, in which there is a dependent variable whose mean is a function of some predictor variables. The predictors are usually assumed to be fixed, and the process that generated them is not of interest. In this chapter we propose a different kind of model that is more flexible than ordinary regression in that i) all the observed variables are jointly modelled, ii) the variables are assumed to vary smoothly as a function of some argument, and iii) each variable may be observed at a distinct set of argument values. We name the new approach structural smooth modelling (SSM). We begin in Section 6.2 by introducing structural equation modelling and Gaussian processes, which are key ideas for our development. In Section 6.2.3 we discuss asynchronous data, which are the type of data that motivate our model. We introduce SSM in 6.3, and discuss learning and inference for it in Section 6.4. We then show some example uses of SSM and present simulation studies that examine its performance in Section 6.5, followed by an analysis of data obtained from British Cycling in Section 6.6.

6.2 Background

6.2.1 Structural equation modelling

Structural equation modelling refers to a broad class of models that developed with contributions from various fields over the years. Broadly speaking, structural equation models emerge as the synthesis of path analysis, latent variable, and measurement models. Path analysis was pioneered by Wright (1918, 1934, 1960) and consists of models in which multiple variables depend on each other through a set of regression equations, departing from ordinary regression in which there is a clear distinction between a single dependent variable and one or more independent variables. The other essential building block of structural equation models is factor analysis, which examines the relationships between observed variables and latent variables. One of the most common uses of factor analysis is in measurement models, where one wants to infer properties of variables that cannot be measured directly (latent variables), but instead through a set of indicators (observed variables). Structural equation models combine path and factor analysis by allowing a set of

latent variables to be linked through a set of multiple regression equations, with the latent variables linked to a set of observable variables through a measurement model. Today, the model that many practitioners refer to as a structural equation model stems from the work of Jöreskog (1970), Wiley (1973), and Keesling (1972), who contributed to a general formulation that is implemented in the LISREL program (Jöreskog, 2001). The general model can be written as

$$\boldsymbol{\eta} = \mathrm{B}\boldsymbol{\eta} + \Gamma\boldsymbol{\xi} + \boldsymbol{\zeta} \tag{6.1}$$

$$\boldsymbol{y} = \Lambda_y\boldsymbol{\eta} + \boldsymbol{\epsilon} \tag{6.2}$$

$$\boldsymbol{x} = \Lambda_x\boldsymbol{\xi} + \boldsymbol{\delta}\ . \tag{6.3}$$

Equation (6.1) describes the relationships between the vector of latent variables $\boldsymbol{\eta}$, which are usually the variables of interest in the model. These variables can be thought of as dependent variables, as in ordinary regression, with the difference being that there now multiple dependent variables that depend on eachother. The matrix B contains coefficients that describe how each latent dependent variable depends on the others. $\boldsymbol{\xi}$ is a vector of latent covariates, or independent variables, as any dependencies that they have on other variables are not part of the model. The matrix Γ contains coefficients that describe how the dependent variables depend on the independent ones, and $\boldsymbol{\zeta}$ is a vector of mean zero normally distributed errors. Setting aside for a moment that $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ are latent (i.e. not directly measurable) variables, equation (6.1) can be thought of as a system of regression equations, in which each dependeent variable is a noisy observation of a linear combination of some of the other dependent variables and some of the independent variables.

Equations (6.2) and (6.3) are factor analysis models that describe how the latent variables $\boldsymbol{\eta}$ and $\boldsymbol{\xi}$ are measured respectively, where $\boldsymbol{y}$ and $\boldsymbol{x}$ are the variables that are observed. The factor models posit that each observed variable is a noisy observation of a linear combination of the latent variables, with the coefficients of the linear combinations given in the matrices Λ_y and Λ_x. $\boldsymbol{\epsilon}$ and $\boldsymbol{\delta}$ are vectors of mean zero normally distributed errors.

The unknown parameters to be estimated are the elements of the matrices $\mathrm{B}, \Gamma, \Lambda_y$, and Λ_x, as well as the variances of the error vectors $\boldsymbol{\zeta}, \boldsymbol{\epsilon}$, and $\boldsymbol{\delta}$. The data that are actually observed are matched realisations of the vectors $\boldsymbol{y}$ and $\boldsymbol{x}$, i.e. a set $\{(\boldsymbol{y}_1, \boldsymbol{x}_1), \ldots, (\boldsymbol{y}_n, \boldsymbol{x}_n)\}$. With regard to estimation the central quantity of interest is the sample covariance matrix, which is compared to the population covariance matrix as a function of the parameters. Parameter estimates can be chosen to minimise some measure of distance between the observed covariance and implied covariance matrices. For an introduction to structural equation and latent variable models, see Bollen (1989).

6.2.2 Gaussian processes

A Gaussian process is a stochastic process, any finite collection of realisations from which are jointly normally distributed. Gaussian processes are often used as a distributional assumption regarding an unknown function. For example, a function $f(\boldsymbol{x})$ is a Gaussian process if the collection of random variables $\{f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\}$ are jointly normally distributed, for any $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$.

Gaussian processes are usually parametrised by a covariance function (or kernel) $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ which specifies the covariance between any two points in the process, and a mean function. Assuming for now that the mean function is zero, the covariance function fully specifies the distribution of any collection of observations of the Gaussian process. A commonly used covariance function is the squared exponential, given by

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j) = \sigma_f^2 \exp\left\{-\frac{1}{2l^2}\|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2^2\right\}\ ,$$

where σ_f^2 and l are parameters of the kernel. A Gaussian process can be denoted by

$$f(x) \sim GP\Big(0, k(\cdot,\cdot)\Big) \ . \tag{6.4}$$

Expression (6.4) encodes the information that

$$\Big(f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)\Big)^T \sim \mathcal{N}(\mathbf{0}, K) \ ,$$

where K is an $n \times n$ matrix with i,j-th element $k(\boldsymbol{x}_i, \boldsymbol{x}_j)$, for any set of argument values $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. K is referred to as the Gram matrix of the kernel $k(\cdot,\cdot)$ with respect to the inputs $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$. Kernel functions used for Gaussian processes must be such that the resulting Gram matrix is symmetric and positive semi-definite, as otherwise it would not be a valid covariance matrix. Gaussian processes are often used for regression by assuming that the function $f(\boldsymbol{x})$ in a regression model

$$y_i = f(\boldsymbol{x}_i) + \epsilon_i$$

is a Gaussian process. With a covariance function such as the squared exponential, Gaussian process regression encodes the intuitive idea that the responses corresponding to predictor vectors that are close in l_2 norm should be highly correlated (i.e. relatively close in value) compared to predictor vectors that are far in l_2 norm. Inference for the function $f(\boldsymbol{x})$ consists of finding the posterior distribution

$$p\Big(f(x)|\boldsymbol{y}\Big) \ ,$$

which can be found analytically using properties of the normal distribution, assuming that the parameters σ_f^2 and l^2 are known. In practice, these need to be selected. One popular approach is to estimate σ_f^2 and l^2 by maximising the marginal log-likelihood

$$\ell(\sigma_f^2, l^2) = \log p(\boldsymbol{y}; \sigma_f^2, l^2) = \log \mathcal{N}(\boldsymbol{y}; \mathbf{0}, K + \sigma_\epsilon^2 I) \ ,$$

where K is the Gram matrix, σ_ϵ^2 is the variance of the errors, and I is the identity matrix.

6.2.3 Asynchronous data

A standard format for multivariate data is a matrix in which multiple observed cases (rows) of a set of variables (columns) are recorded. The rows usually represent observations of the variables matched in some way, for example by individual, or by time. Many multivariate statistical techniques depend directly on the observation of such a matrix. Not many techniques have been developed however for the case when a set of variables is recorded but cannot be matched in an obvious way. For example, as is the focus of this paper, a set of variables may be recorded longitudinally, however not at the same time points. More generally, variables may be recorded at mismatched values of some vector, such as space. We refer to such data as asynchronous. In these settings, standard methods such as linear regression or factor analysis cannot be directly applied, and extensions need to be devised that can take into account the asynchronous nature of the data. In what follows we assume that asynchronicity is with respect to time.

Rehfeld et al. (2011) compare various approaches to estimating the correlation between two asynchronously observed vectors. These include binning techniques, in which data are artificially aligned, interpolation techniques, and kernel weighting schemes. Cao et al. (2015, 2016); Chen and Cao (2017) propose various approaches to longitudinal regression with asynchronous data. The main idea in these approaches

is to consider generalized estimating equations for longitudinal regression in which various combinations of responses and predictors are weighted by a kernel that takes into account the time asynchronicity between them. Crucially however, in these works the predictors are all assumed to be observed at the same time points. Duncker and Sahani (2018) consider a factor analysis for asynchronous longitudinal data in which the factors are Gaussian processes, while also modelling the observed time points using point processes.

In this chapter we aim to combine Gaussian process factor analysis with path models to form a structural model for asynchronous data that can flexibly handle a variety of modelling scenarios, while handling asynchronicity naturally.

6.3 Structural smooth modelling

We aim to construct a joint model for a set of stochastic processes $\{y_1(t), \ldots, y_p(t)\}$, each of which has been observed at a different vector of time points $\{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_p\}$. We begin by assuming that each process $y_j(t)$ is a shifted, noisy observation of a mean zero latent Gaussian process $\eta_j(t)$, and that the Gaussian processes are structurally related according to the set of equations

$$\boldsymbol{\eta}(t) = B\boldsymbol{\eta}(t) + \boldsymbol{v}(t) \ , \tag{6.5}$$

where $\boldsymbol{\eta}(t) = \Big(\eta_1(t), \ldots, \eta_p(t)\Big)^T$, $\boldsymbol{v}(t) = \Big(v_1(t), \ldots, v_p(t)\Big)^T$ is a vector of independent Gaussian processes with kernels that depend respectively on a set of parameter vectors $\{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_p\}$, and B is a strictly lower triangular matrix that depends on a vector of unknown parameters $\boldsymbol{\beta}$. This structure is depicted graphically in Figure 6.1. In the graph, dashed lines pointing at a node indicate that the pointing nodes are parameters of the distribution of the process being pointed at; solid lines pointing at a node indicate linear dependence of the process represented by that node on the processes from which the arrows are pointing; finally, bold lines indicate linear dependence with a coefficient of one. Circular nodes represent latent processes, while square nodes represent processes that have been observed at sets of distinct time points. Intuitively, equation (6.5) posits that for each j, the stochastic process $\eta_j(t)$ is a linear combination of other stochastic processes $\{\eta_1, \ldots, \eta_{j-1}\}$, plus an independent Gaussian process $v_j(t)$.

Equation (6.5) can be re-written as

$$\begin{aligned} \boldsymbol{\eta}(t) &= B\boldsymbol{\eta}(t) + \boldsymbol{v}(t) \\ \Rightarrow \boldsymbol{\eta}(t) - B\boldsymbol{\eta}(t) &= \boldsymbol{v}(t) \\ \Rightarrow (I - B)\boldsymbol{\eta}(t) &= \boldsymbol{v}(t) \\ \Rightarrow \boldsymbol{\eta}(t) &= (I - B)^{-1}\boldsymbol{v}(t) \ , \end{aligned} \tag{6.6}$$

where $(I-B)$ is guaranteed to be invertible because B is strictly lower triangular, implying that $I-B$ is lower triangular with invertible entries on the diagonal. The equation in (6.6) implies that each stochastic process $\eta_j(t)$ is a linear combination of the independent Gaussian processes $\{v_1(t), \ldots, v_p(t)\}$, and consequently is also a Gaussian process. The full generative model can be written as

$$\begin{aligned} \epsilon_j(t) &\sim \mathcal{N}(0, \phi_j) \\ v_j(t) &\sim GP(0, k_{\boldsymbol{\lambda}_j}) \ , \\ \boldsymbol{\eta}(t) = B\boldsymbol{\eta}(t) + \boldsymbol{v}(t) \Leftrightarrow \boldsymbol{\eta}(t) &= (I - B)^{-1}\boldsymbol{v}(t) \ , \\ y_j(t) &= \mu_j + \eta_j(t) + \epsilon_j(t) \ , \end{aligned} \tag{6.7}$$

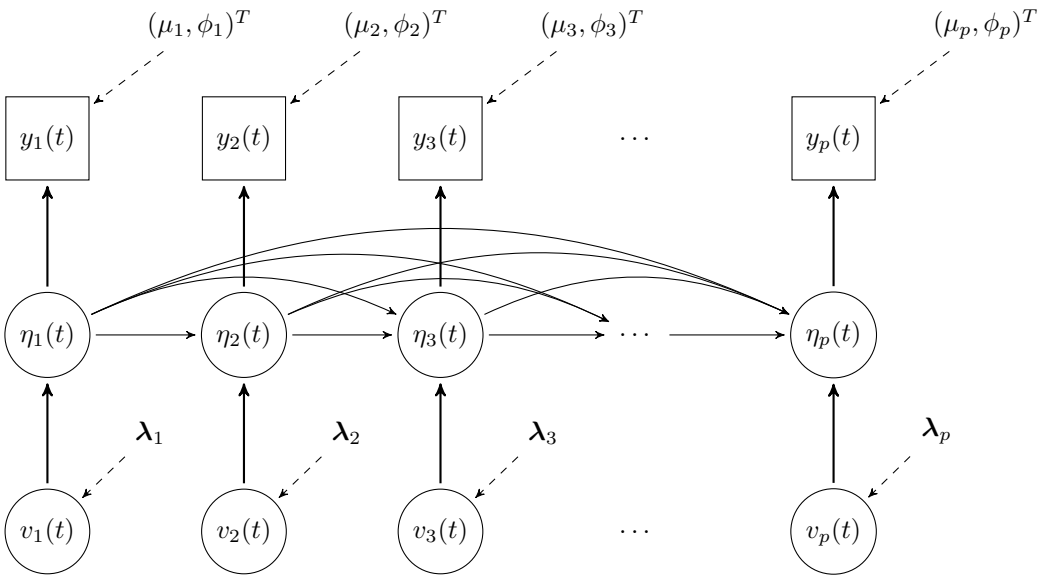

Figure 6.1: A graphical representation of the general structural model.

for $j = 1, \ldots, p$, and $t \in \mathcal{T}$, where $\mathcal{T}$ denotes some time interval. The formulation in (6.7) allows us to model multiple stochastic processes jointly without assuming that they have been observed synchronously.

6.4 Estimation, inference and prediction

6.4.1 Parameter estimation and inference

When working with a hierarchical model such as (6.7), there are three tasks, relevant to learning and inference, that one might consider. The first is that of parameter estimation, or learning. The second task is that of constructing confidence intervals and carrying out hypothesis tests for the estimated parameters, which we refer to as inference, in the traditional frequentist sense. Finally, one may wish to compute the posterior distribution of latent variables given the observed data. We refer to this as prediction.

The parameters to be estimated in model (6.7) consist of the means $\mu_1, \ldots, \mu_p$ of the observed processes, which we denote by $\boldsymbol{\mu}$; the coefficients $\boldsymbol{\beta}$ that make up the matrix B, which determines the relationships between the latent processes $\{\eta_1(t), \ldots, \eta_p(t)\}$; the collection of kernel parameters $\{\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_p\}$, which we concatenate into a vector $\boldsymbol{\lambda}$; and the noise variance parameters $\{\phi_1, \ldots, \phi_p\}$, which we denote by the vector $\boldsymbol{\phi}$. Finally, we denote the covariance parameters by $\boldsymbol{\gamma}^T = (\boldsymbol{\beta}^T, \boldsymbol{\lambda}^T, \boldsymbol{\phi}^T)$, and the entire coefficient vector by $\boldsymbol{\delta}^T = (\boldsymbol{\mu}^T, \boldsymbol{\gamma}^T)$. We consider maximum likelihood estimation for $\boldsymbol{\delta}$, and consequently the majority of this section is devoted to expressing the likelihood for observed data, as well as the gradient of the likelihood, for optimisation purposes. We assume that the observed data consist of vectors $\{y_1(\boldsymbol{t}_1), \ldots, y_p(\boldsymbol{t}_p)\}$, where $y_j(\boldsymbol{t}_j)$ denotes the vector $\Big(y_j(t_{j1}), \ldots, y_j(t_{jq_j})\Big)^T$, and $\{\boldsymbol{t}_1, \ldots, \boldsymbol{t}_p\}$ are distinct, length $\{q_1, \ldots, q_p\}$ respectively, vectors of time points at which each process is observed. Because the observed stochastic processes are linear combinations of jointly normally distributed quantities, the observations of the processes are also jointly normally distributed. Consequently, to express the likelihood function for the observed data, it is sufficient to find the mean and covariance matrix of the observations. We denote by α_{ij} the i, j-th element

of $(I-B)^{-1}$. Then, for any two arbitrary time points t_1^* and t_2^*, we have

$$\eta_i(t_1^*) = \sum_{k=1}^{p} \alpha_{ik} v_k(t_1^*) \ ,$$
$$\eta_j(t_2^*) = \sum_{l=1}^{p} \alpha_{jl} v_l(t_2^*) \ .$$

Firstly, it is easy to see that

$$E\Big(\eta_i(t_1^*)\Big) = E\Big(\eta_j(t_2^*)\Big) = 0 \ ,$$

since $E\Big(v_1(t)\Big) = \cdots = E\Big(v_p(t)\Big) = 0$. We therefore have

$$\begin{aligned}
\mathrm{cov}\Big(\eta_i(t_1^*), \eta_j(t_2^*)\Big) &= E\left[\left(\sum_{k=1}^{p} \alpha_{ik} v_k(t_1^*)\right)\left(\sum_{l=1}^{p} \alpha_{jl} v_l(t_2^*)\right)\right] \\
&= \sum_{k=1}^{p} E\left[\alpha_{ik} v_k(t_1^*) \sum_{l=1}^{p} \alpha_{jl} v_l(t_2^*)\right] \\
&= \sum_{k=1}^{p} E\Big[\alpha_{ik} v_k(t_1^*) \alpha_{jk} v_k(t_2^*)\Big] \\
&= \sum_{k=1}^{p} \alpha_{ik}\alpha_{jk} \mathrm{cov}\Big(v_k(t_1^*), v_k(t_2^*)\Big) \\
&= \sum_{k=1}^{p} \alpha_{ik}\alpha_{jk} k_{\boldsymbol{\lambda}_k}(t_1^*, t_2^*) \ .
\end{aligned} \tag{6.8}$$

Similarly, $E\Big(y_i(t_1^*)\Big) = \mu_i$, and therefore

$$\begin{aligned}
y_i(t_1^*) - E\Big(y_i(t_1^*)\Big) &= \sum_{k=1}^{p} \alpha_{ik} v_k(t_1^*) + \epsilon_i(t_1^*) \ , \\
y_j(t_2^*) - E\Big(y_j(t_2^*)\Big) &= \sum_{l=1}^{p} \alpha_{jl} v_l(t_2^*) + \epsilon_j(t_2^*) \ ,
\end{aligned} \tag{6.9}$$

from which we conclude

$$\mathrm{cov}\Big(y_i(t_1^*), y_j(t_2^*)\Big) = \sum_{k=1}^{p} \alpha_{ik}\alpha_{jk} k_{\boldsymbol{\lambda}_k}(t_1^*, t_2^*) + \mathbb{1}_{\{i=j, t_1^*=t_2^*\}} \phi_i \ ,$$

where $\mathbb{1}_{\{i=j, t_1^*=t_2^*\}}$ takes the value 1 if $i=j$ and $t_1^*=t_2^*$, and 0 otherwise.

For ease of notation, we define $\boldsymbol{\eta}_j = \eta_j(\boldsymbol{t}_j)$, $\boldsymbol{\eta}^T = (\boldsymbol{\eta}_1^T, \ldots, \boldsymbol{\eta}_p^T)$, $\boldsymbol{y}_j = y_j(\boldsymbol{t}_j)$, and $\boldsymbol{y}^T = (\boldsymbol{y}_1^T, \ldots, \boldsymbol{y}_p^T)$. Let K be the $Q \times Q$ covariance matrix of $\boldsymbol{y}$, where $Q = \sum_j q_j$. K can be described in a block-wise fashion, with the i,j-th block denoting the cross-covariance $\mathrm{cov}(\boldsymbol{y_i}, \boldsymbol{y_j})$. We then have

$$K^{(i,j)} = \sum_{k} \alpha_{ik}\alpha_{jk} k_{\boldsymbol{\lambda}_k}(\boldsymbol{t}_i, \boldsymbol{t}_j) + \phi_i I_{[i=j]} \ ,$$

where $I_{[i=j]}$ is the $q_i \times q_i$ identity matrix if $i=j$, and 0 otherwise, and $k_{\boldsymbol{\lambda}_k}(\boldsymbol{t}_i, \boldsymbol{t}_j)$ denotes the Gram matrix of the kernel $k_{\boldsymbol{\lambda}_k}$ with respect to the inputs $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$. Consequently we can write the log-likelihood for the

observed data as

$$\begin{aligned}\ell(\boldsymbol{\delta}) &= \log p(\boldsymbol{y};\boldsymbol{\delta}) \\ &= -\frac{1}{2}\log|K| - \frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu}^{\text{rep}})^T K^{-1}(\boldsymbol{y}-\boldsymbol{\mu}^{\text{rep}}) + c \ ,\end{aligned}$$

where c is a constant that does not depend on the parameters to be estimated, $|K|$ denotes the determinant of K, and

$$\boldsymbol{\mu}^{\text{rep}} = \left(\mu_1 \mathbf{1}_{q_1}^T, \ldots, \mu_p \mathbf{1}_{q_p}^T\right)^T \ ,$$

where $\mathbf{1}_{q_j}$ is a vector or ones of length q_j. To optimise the log-likelihood, we propose the BFGS algorithm (Fletcher, 1987, Chapter 3).

As a quasi-Newton method, the BFGS algorithm requires only evaluations of the function to be optimised, along with its gradient, and builds up an approximation to the Hessian as it progresses iteratively. This makes it convenient for our case as we need not compute the Hessian analytically, but we still obtain an approximation to it at convergence.

We now give expressions for the derivative of the log-likelihood with respect to each parameter. For the mean parameters, we have

$$\frac{\partial \ell(\boldsymbol{\delta})}{\partial \mu_k} = (\boldsymbol{y}-\boldsymbol{\mu}^{\text{rep}})^T K^{-1} \frac{\partial \boldsymbol{\mu}^{\text{rep}}}{\partial \mu_k} \ ,$$

where

$$\frac{\partial \boldsymbol{\mu}^{\text{rep}}}{\partial \mu_k} = \left(\mathbf{0}_{q_1}^T, \ldots, \mathbf{1}_{q_k}^T, \ldots, \mathbf{0}_{q_p}^T\right)^T \ ,$$

with $\mathbf{0}_{q_j}$ denoting a vector of zeros of length q_j. For the variance parameters we have

$$\frac{\partial \ell(\boldsymbol{\delta})}{\partial \gamma_k} = -\frac{1}{2}\text{tr}\left[K^{-1}\frac{\partial K}{\partial \gamma_k}\right] + \frac{1}{2}(\boldsymbol{y}-\boldsymbol{\mu}^{\text{rep}})^T K^{-1}\frac{\partial K}{\partial \gamma_k}K^{-1}(\boldsymbol{y}-\boldsymbol{\mu}^{\text{rep}}) \ ,$$

and so all that remains to fully express the gradient of the log-likelihood is the derivative of K with respect to each parameter. Firstly, it is useful to obtain

$$\begin{aligned}\frac{\partial \alpha_{is}}{\partial \beta_{kl}} &= \frac{\partial [(I-B)^{-1}]_{is}}{\partial \beta_{kl}} \\ &= \Big(-(I-B)^{-1}\frac{\partial (I-B)}{\partial \beta_{kl}}(I-B)^{-1}\Big)_{is} \\ &= [(I-B)^{-1}]_{ik}[(I-B)^{-1}]_{ls} = \alpha_{ik}\alpha_{ls} \ ,\end{aligned}$$

where β_{kl} is the coefficient in the k,l-th position of B. Similarly, we denote by λ_{kl} the l-th element of the

vector $\boldsymbol{\lambda}_k$. We can then write

$$\begin{aligned}
\frac{\partial K^{(i,j)}}{\partial \beta_{kl}} &= \frac{\partial \sum_{s=1}^{p} \alpha_{is}\alpha_{js} k_{\lambda_s}(\boldsymbol{t}_i, \boldsymbol{t}_j)}{\partial \beta_{kl}} \\
&= \sum_{s=1}^{p} \Big(\frac{\partial \alpha_{is}}{\partial \beta_{kl}} \alpha_{js} + \alpha_{is} \frac{\partial \alpha_{js}}{\partial \beta_{kl}} \Big) k_{\lambda_s}(\boldsymbol{t}_i, \boldsymbol{t}_j) \\
&= \sum_{s=1}^{p} (\alpha_{ik}\alpha_{ls}\alpha_{js} + \alpha_{is}\alpha_{jk}\alpha_{ls}) k_{\lambda_s}(\boldsymbol{t}_i, \boldsymbol{t}_j) \\
\frac{\partial K^{(i,j)}}{\partial \lambda_{kl}} &= \frac{\partial \sum_{s=1}^{p} \alpha_{is}\alpha_{js} k_{\boldsymbol{\lambda}_s}(\boldsymbol{t}_i, \boldsymbol{t}_j)}{\partial \lambda_{kl}} \\
&= \alpha_{ik}\alpha_{jk} \frac{\partial k_{\boldsymbol{\lambda}_k}(\boldsymbol{t}_i, \boldsymbol{t}_j)}{\partial \lambda_{kl}} \\
\frac{\partial K^{(i,j)}}{\partial \phi_k} &= I_{[i=j=k]} \ ,
\end{aligned}$$

where $I_{[i=j=k]}$ is the appropriately sized identity matrix if $i = j = k$, and a matrix of zeros otherwise. Once parameter estimates $\hat{\boldsymbol{\delta}}$ are obtained by finding $\hat{\boldsymbol{\delta}} = \arg\max_{\boldsymbol{\delta}} \ell(\boldsymbol{\delta})$, using the BFGS algorithm, an approximate covariance matrix for $\hat{\boldsymbol{\delta}}$ can be computed as

$$\hat{\Sigma} = \hat{H}^{-1} \ ,$$

where $\hat{H}$ is the approximate Hessian of the log-likelihood, returned at convergence of the BFGS algorithm. Approximate inference for $\boldsymbol{\delta}$ can then be carried out, for example by constructing approximate 95% confidence intervals

$$\hat{\delta}_k \pm 1.96\sqrt{\hat{\Sigma}_{kk}} \ .$$

6.4.2 Prediction

Once maximum likelihood estimates $\hat{\boldsymbol{\delta}}$ have been obtained, it may be of interest to compute the posterior distribution of the latent variables

$$p\Big(\boldsymbol{\eta}(t^*)|\boldsymbol{y}; \hat{\boldsymbol{\delta}}\Big) \ ,$$

and point predictions of the form

$$\hat{\boldsymbol{\eta}}(t^*) = E\Big(\boldsymbol{\eta}(t^*)|\boldsymbol{y}; \hat{\boldsymbol{\delta}}\Big) \ ,$$

for some arbitrary time point t^*. To compute the conditional distribution of $\boldsymbol{\eta}(t^*)|\boldsymbol{y}$, we first need to compute the joint distribution, from which the conditional distribution will follow given the properties of the normal distribution. The joint distribution can be written as

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{\eta}(t^*) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \boldsymbol{\mu}^{\text{rep}} \\ \mathbf{0} \end{bmatrix}, \begin{bmatrix} K & K^* \\ K^{*T} & K^{**} \end{bmatrix} \right) \ ,$$

where K^* denotes the cross-covariance $\text{cov}\big(\boldsymbol{y}, \boldsymbol{\eta}(t^*)\big)$ and K^{**} denotes the covariance of $\boldsymbol{\eta}(t^*)$. We then have

$$\boldsymbol{\eta}(t^*)|\boldsymbol{y} \sim \mathcal{N}\Big(K^*K^{-1}(\boldsymbol{y} - \boldsymbol{\mu}^{\text{rep}}), K^{**} - K^*K^{-1}K^*\Big)$$

and therefore,

$$\hat{\boldsymbol{\eta}}(t^*) = K^*K^{-1}(\boldsymbol{y} - \boldsymbol{\mu}^{\text{rep}}) \ .$$

K^* and K^{**} can be computed in a similar manner to the way K was computed in Section 6.4.1, using equations (6.8) and (6.9)

6.5 Example applications and simulation studies

In this section we examine the performance of SSM using simulated data. We begin with an example of a regression type scenario, followed by factor analysis.

6.5.1 Regression

For the regression scenario, we consider a case of three observed processes $y_1(t), y_2(t), y_3(t)$. The generative model we consider can be written as

$$\begin{aligned} \eta_1(t) &= v_1(t) \\ \eta_2(t) &= v_2(t) \\ \eta_3(t) &= \beta_{31}\eta_1(t) + \beta_{32}\eta_2(t) + v_3(t) \\ y_1(t) &= \mu_1 + \eta_1(t) + \epsilon_1(t) \\ y_2(t) &= \mu_2 + \eta_2(t) + \epsilon_2(t) \\ y_3(t) &= \mu_3 + \eta_3(t) + \epsilon_3(t) \ , \end{aligned} \tag{6.10}$$

where $v_1(t), v_2(t), v_3(t)$ are independent Gaussian processes and $\epsilon_1(t), \epsilon_2(t), \epsilon_3(t)$ are independent noise processes (i.e. $\epsilon_j(t) \sim \mathcal{N}(0, \phi_j)$). We use the squared exponential kernel for each Gaussian process. We set all the parameter values arbitrarily and simulate from the model in the following way. We first generate three uniformly distributed vectors of time points, $\boldsymbol{t}_1, \boldsymbol{t}_2, \boldsymbol{t}_3$, between zero and one. We then generate realisations of the Gaussian processes $v_1(t), v_2(t), v_3(t)$, followed by realisations of the noise processes $\epsilon_1(\boldsymbol{t}_1), \epsilon_2(\boldsymbol{t}_2), \epsilon_3(\boldsymbol{t}_3)$. We then compute the compete vector of observed data $\boldsymbol{y}^T = \Big(y_1^T(\boldsymbol{t}_1), y_2^T(\boldsymbol{t}_2), y_3^T(\boldsymbol{t}_3)\Big)$. To examine the performance of maximum likelihood estimation we repeat the data generation procedure $N = 1,000$ times, and compute the maximum likelihood estimate for each data set. We then compute, for the parameters of interest, the expected value of each parameter, its variance, and the expected value of the estimator of its variance. Figure 6.2 is a graphical representation of the model, while Figure 6.3 displays one example data set, with the top panel showing the realisations of $\Big(\mu_1 + \eta_1(t), \mu_2 + \eta_2(t), \mu_3 + \eta_3(t)\Big)$ that were used to generate the observed data, and the bottom panel displaying the the fitted values $\Big(\hat{\mu}_1 + \hat{\eta}_1(t), \hat{\mu}_2 + \hat{\eta}_2(t), \hat{\mu}_3 + \hat{\eta}_3(t)\Big)$, with $\hat{\eta}_j(t) = E(\eta_j(t)|\boldsymbol{y})$. We repeat this entire simulation study two times, changing the number of observations obtained from each observed stochastic process. The first time, we obtain 40 observations from each process, while the second time we obtain 80. Tables 6.1 and 6.2 display the respective results. In both simulation cases the expected value of the estimators are close to the true values, with the difference decreasing for larger n. The variance estimator is shown to perform well, especially in the larger n case. Furthermore, the variance of the estimators decreases with n.

	μ_1	μ_2	μ_3	β_{31}	β_{32}
expected value of MLE	0.00	-0.17	-1.28	-2.00	1.97
true value	0.02	-0.18	-1.37	-2.00	2.00
expected value of variance estimator	0.26	0.26	2.04	0.22	0.22
simulation based variance	0.27	0.29	3.27	0.24	0.26

Table 6.1: Results of the simulation study using model (6.10) as the data generating process, with $n = 40$.

	μ_1	μ_2	μ_3	β_{31}	β_{32}
expected value of MLE	0.01	-0.19	-1.35	-1.98	2.00
true value	0.02	-0.18	-1.37	-2.00	2.00
expected value of variance estimator	0.26	0.25	2.00	0.11	0.10
simulation based variance	0.30	0.26	3.15	0.11	0.10

Table 6.2: Results of the simulation study using model (6.10) as the data generating process, with $n = 80$.

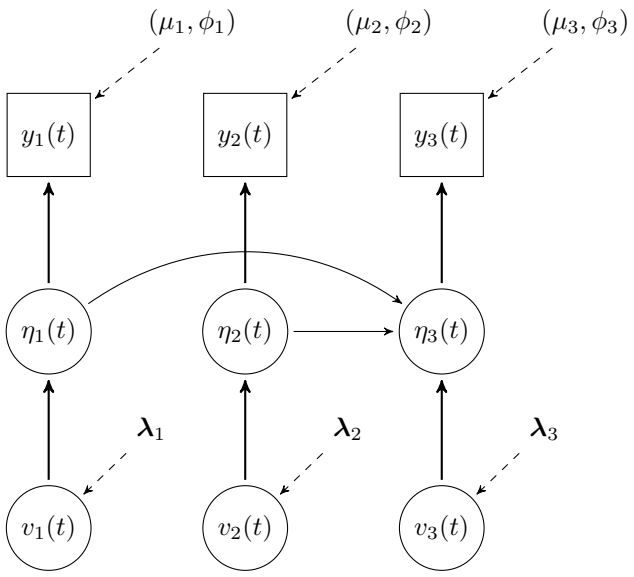

Figure 6.2: Graphical representation of the model formulated in (6.10).

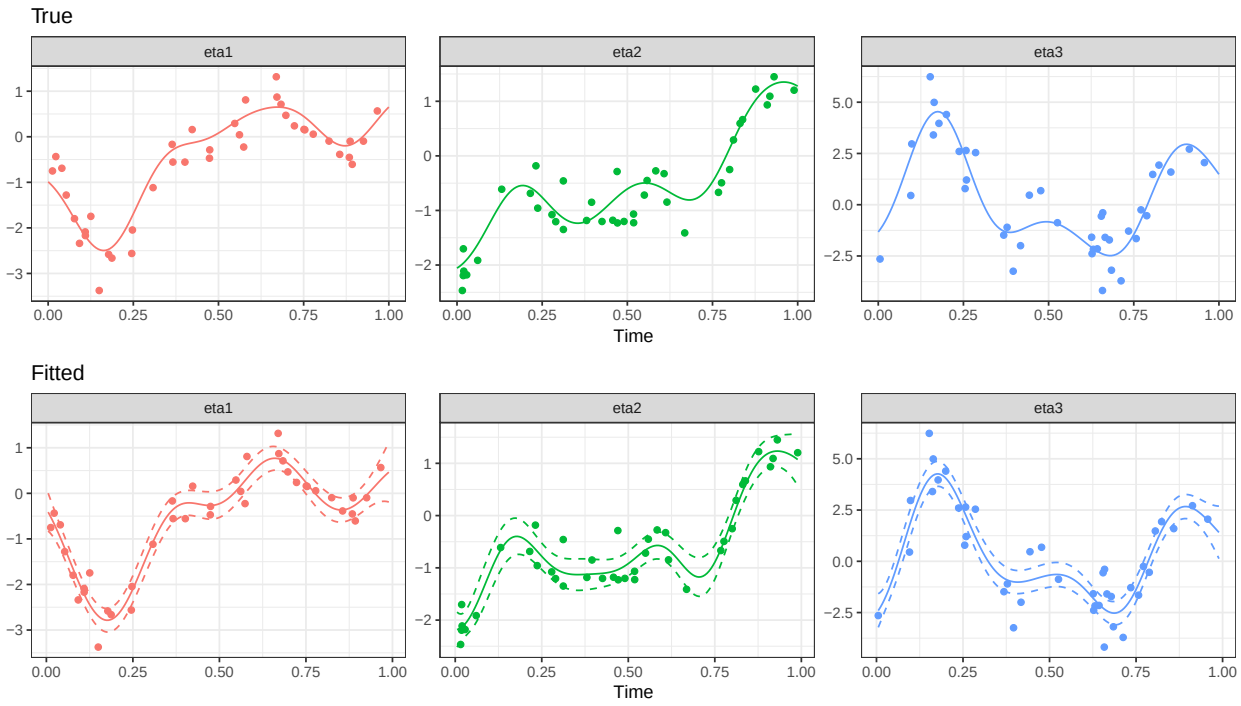

Figure 6.3: Example data set generated from (6.10), with the top panel showing the realisations of $\Big(\mu_1 + \eta_1(t), \mu_2 + \eta_2(t), \mu_3 + \eta_3(t)\Big)$ that were used to generate the observed data, and the bottom panel displaying the fitted values $\Big(\hat{\mu}_1 + \hat{\eta}_1(t), \hat{\mu}_2 + \hat{\eta}_2(t), \hat{\mu}_3 + \hat{\eta}_3(t)\Big)$ (solid line), plus and minus 2 standard deviations of $\hat{\eta}_j(t)$ (dashed line). The true coefficients are $\beta_{31} = -2$ and $\beta_{32} = 2$ while the estimates are $\hat{\beta}_{31} = -1.60$ and $\hat{\beta}_{32} = 1.60$.

6.5.2 Factor analysis

Now we show how the general SSM formulation can also be used to perform a kind of factor analysis. The generative model we consider is given by

$$\begin{aligned}
\eta_1(t) &= v_1(t) \\
\eta_2(t) &= \beta_{21}\eta_1(t) + v_2(t) \\
\eta_3(t) &= \beta_{31}\eta_1(t) + v_3(t) \\
y_2(t) &= \mu_2 + \eta_2(t) + \epsilon_2(t) \\
y_3(t) &= \mu_3 + \eta_3(t) + \epsilon_3(t) \ .
\end{aligned} \tag{6.11}$$

The crucial point to make in this case is that we do not observe a stochastic process $y_1(t)$ that depends on $\eta_1(t)$ only, instead observing $y_2(t)$ and $y_3(t)$, which in turn depend on $\eta_1(t)$. $\eta_1(t)$ can be thought of as representing shared variability between $\eta_2(t)$ and $\eta_3(t)$, and hence is a way to introduce association between $\eta_2(t)$ and $\eta_3(t)$. The structure in (6.11) is shown graphically in Figure 6.4. Note that if, for example, a squared exponential kernel is used to model $v_1(t)$, an identifiability issue is created. To see this, we denote the observed data by $(\boldsymbol{y}_2^T, \boldsymbol{y}_3^T) = \left(y_2(\boldsymbol{t}_2^T), y_3(\boldsymbol{t}_3^T)\right)$ and obtain

$$\operatorname{cov}\begin{bmatrix}\boldsymbol{y}_2 \\ \boldsymbol{y}_3\end{bmatrix} = \begin{bmatrix}\beta_{21}^2 k_{\boldsymbol{\lambda}_1}(\boldsymbol{t}_2,\boldsymbol{t}_2) + k_{\boldsymbol{\lambda}_2}(\boldsymbol{t}_2,\boldsymbol{t}_2) & \beta_{21}\beta_{31}k_{\boldsymbol{\lambda}_1}(\boldsymbol{t}_2,\boldsymbol{t}_3) \\ \beta_{21}\beta_{31}k_{\boldsymbol{\lambda}_1}(\boldsymbol{t}_3,\boldsymbol{t}_2) & \beta_{31}^2 k_{\boldsymbol{\lambda}_1}(\boldsymbol{t}_3,\boldsymbol{t}_3) + k_{\boldsymbol{\lambda}_3}(\boldsymbol{t}_3,\boldsymbol{t}_3)\end{bmatrix} \ .$$

With the squared exponential kernel, we have

$$k_{\boldsymbol{\lambda}_1}(t_1^*, t_2^*) = \lambda_{11}\exp\left\{-\frac{(t_1^* - t_2^*)^2}{2\lambda_{12}}\right\} \ ,$$

where t_1^* and t_2^* are arbitrary time points, and λ_{11} and λ_{21} are positive. It now becomes clear that for any positive constant c, setting $\beta_{21}^* = \sqrt{c}\beta_{21}$, $\beta_{31}^* = \sqrt{c}\beta_{31}$, $\lambda_{11}^* = \lambda_{11}/c$ would yield exactly the same covariance matrix as β_{21}, β_{31}, and λ_{11}. In addition, interchanging the signs of β_{21} and β_{31} also results in the same covariance matrix. Both of these issues can be dealt with by setting the value of either β_{21} or β_{31}. In this example we set $\beta_{21} = 1$. We now carry out the same simulation as in Section 6.5.1, generating $N = 1,000$ data sets from the generative model, once with 40 observations per observed process and once with 80. The results are given in Tables 6.3 and 6.4. In both cases the maximum likelihood estimator performs well, with negligible bias and good variance estimates. With larger n the variance of the estimators decreases and the variance estimators improve.

	μ_1	μ_2	β_{21}	β_{31}
expected value of MLE	2.00	2.98	1.00	-2.01
true value	2.00	3.00	1.00	-2.00
expected value of variance estimator	0.54	1.30	-	0.21
simulation based variance	0.63	1.32	-	0.25

Table 6.3: Results of the simulation study using model (6.11) as the data generating process, with $n = 40$.

	μ_1	μ_2	β_{21}	β_{31}
expected value of MLE	1.97	3.01	1.00	-1.99
true value	2.00	3.00	1.00	-2.00
expected value of variance estimator	0.56	1.13	0.00	0.10
simulation based variance	0.60	1.08	0.00	0.09

Table 6.4: Results of the simulation study using model (6.11) as the data generating process, with $n = 80$.

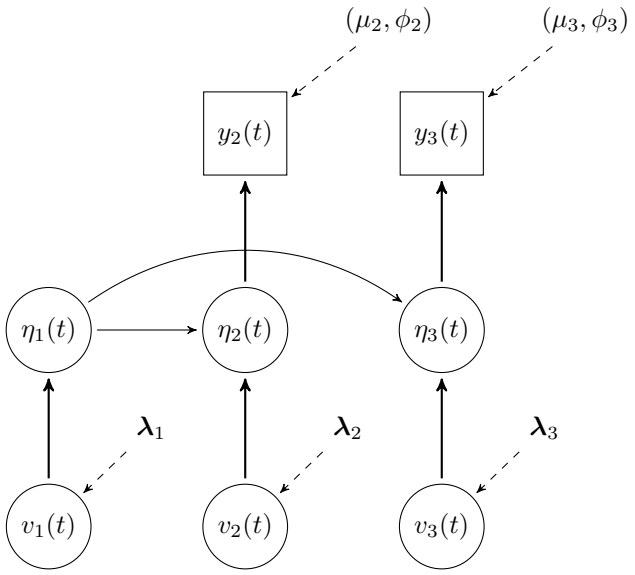

Figure 6.4: Graphical representation of the model formulated in (6.11).

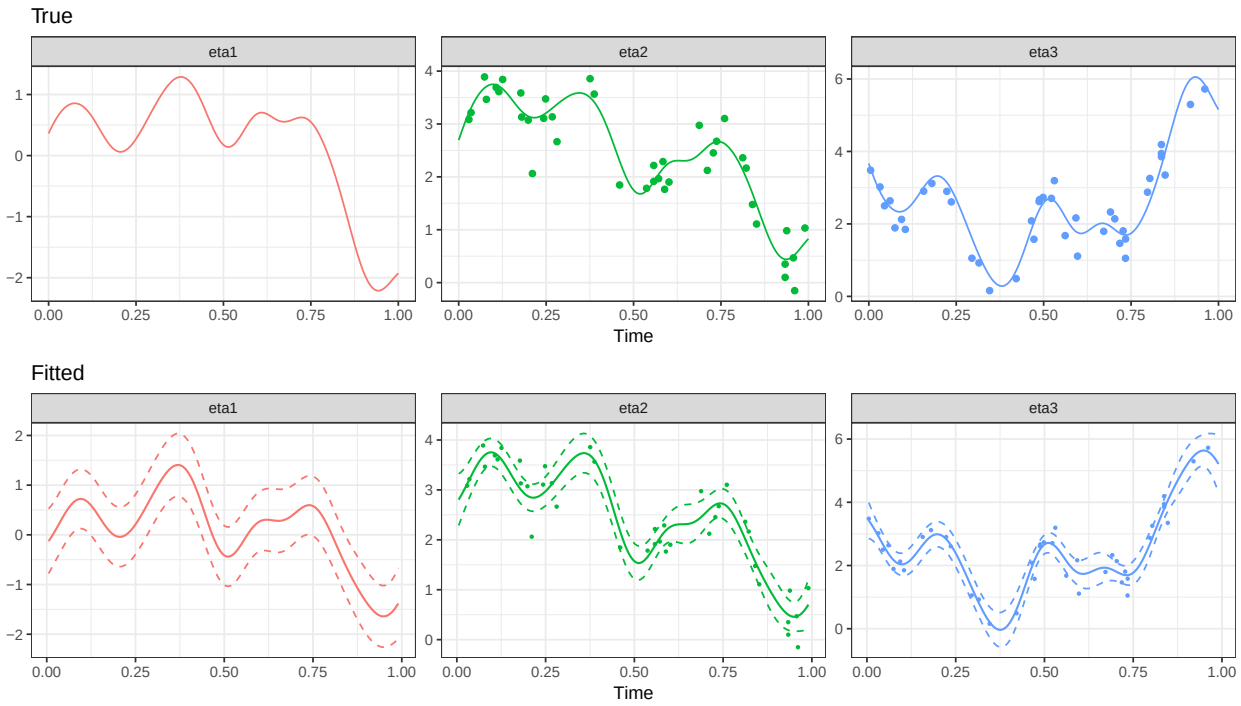

Figure 6.5: Example data set generated from (6.11), with the top panel showing the realisations of $\Big(\eta_1(t), \mu_2 + \eta_2(t), \mu_3 + \eta_3(t)\Big)$ that were used to generate the observed data, and the bottom panel displaying the fitted values $\Big(\hat{\eta}_1(t), \hat{\mu}_2 + \hat{\eta}_2(t), \hat{\mu}_3 + \hat{\eta}_3(t)\Big)$ (solid line), plus and minus 2 standard deviations of $\hat{\eta}_j(t)$ (dashed line). The true coefficients are $\beta_{21} = 1$ and $\beta_{31} = -2$. $\hat{\beta}_{21}$ is set to 1 for identifiability, while $\hat{\beta}_{31} = -1.60$.

6.6 British Cycling data

We now present the data that motivated the SSM framework introduced in this chapter, which comes from British Cycling, the organisation that supports Olympic cyclists in the UK. The data are comprised of measurements taken from various types of training efforts undertaken by the athletes over a period of a few years. In the analyses we present, we will focus on three types of training efforts. The first occurs in the gym, on a bicycle ergometer. A bicycle ergometer is a stationary bicycle that allows cyclists to train in the gym while also recording their performance. The ergometer measurement that we have access to is the peak torque produced by the athlete during each effort. The second type of training effort is cycling on the road, from which we have access to the highest average power produced over one, two, five, and ten minutes. Finally we consider training efforts that occur in the velodrome, which consist of efforts over various distance (in laps) with different starting speeds. While there are many measurements taken for efforts in the velodrome, we will focus on the peak power produced during each training effort. While there are many challenges in working with a data set like this one, the most challenging aspect is that different training efforts occur at different time points. Consequently, it is not obvious how one may attempt to associate measurements taken from different training efforts, as they are not matched in time. This is what motivated the development of a model that can jointly model various processes without assuming that they have been measured at the same time points. Other challenges include lack of information regarding the circumstances under which a training effort took place, e.g. whether an athlete was returning from injury, or whether training efforts took place with a particular intention that might vary the results produced. Finally, the frequency with which efforts occur is variable, and it is not clear whether this is because not all efforts were recorded, or because of injury, or because the focus of training changed and consequently the types of efforts varied. This is particularly challenging because for many athletes there is not much overlap between the timings of different types of efforts, making the task of associating these measurements particularly difficult. For this reason we focus on only the three athletes who have the most recorded efforts in the data.

6.6.1 Data pre-processing

To analyse the BC data using the SSM approach, we had to make a small ad-hoc modification to the data to make them more amenable to analysis by SSM. For all the models we consider with the BC data, we use squared exponential kernels for the Gaussian processes. Because the squared exponential kernel is designed for use with continuous inputs, problems can be created if some inputs are identical or very close together. In the BC data dates are measured in days, and for some athletes and types of training, efforts occur on the same day with large spaces between days with training efforts. To deal with this, for each athlete, and for each type of training effort, we add small perturbations to the dates at which efforts are recorded. The perturbations are all drawn independently from a normal distribution with mean zero and variance ten.

6.6.2 Tracking velodrome performance over time

In this section we examine whether performance in different training efforts in the velodrome over time are associated, and whether these changes over time can be explained by a single factor. The velodrome training efforts are flying lap, standing lap, and standing half. Flying lap refers to a lap of the velodrome starting from high speed. Standing lap refers to a lap in the velodrome starting from rest. Similarly standing half is the same as standing lap but for only half a lap. The measurement we consider for each training effort is

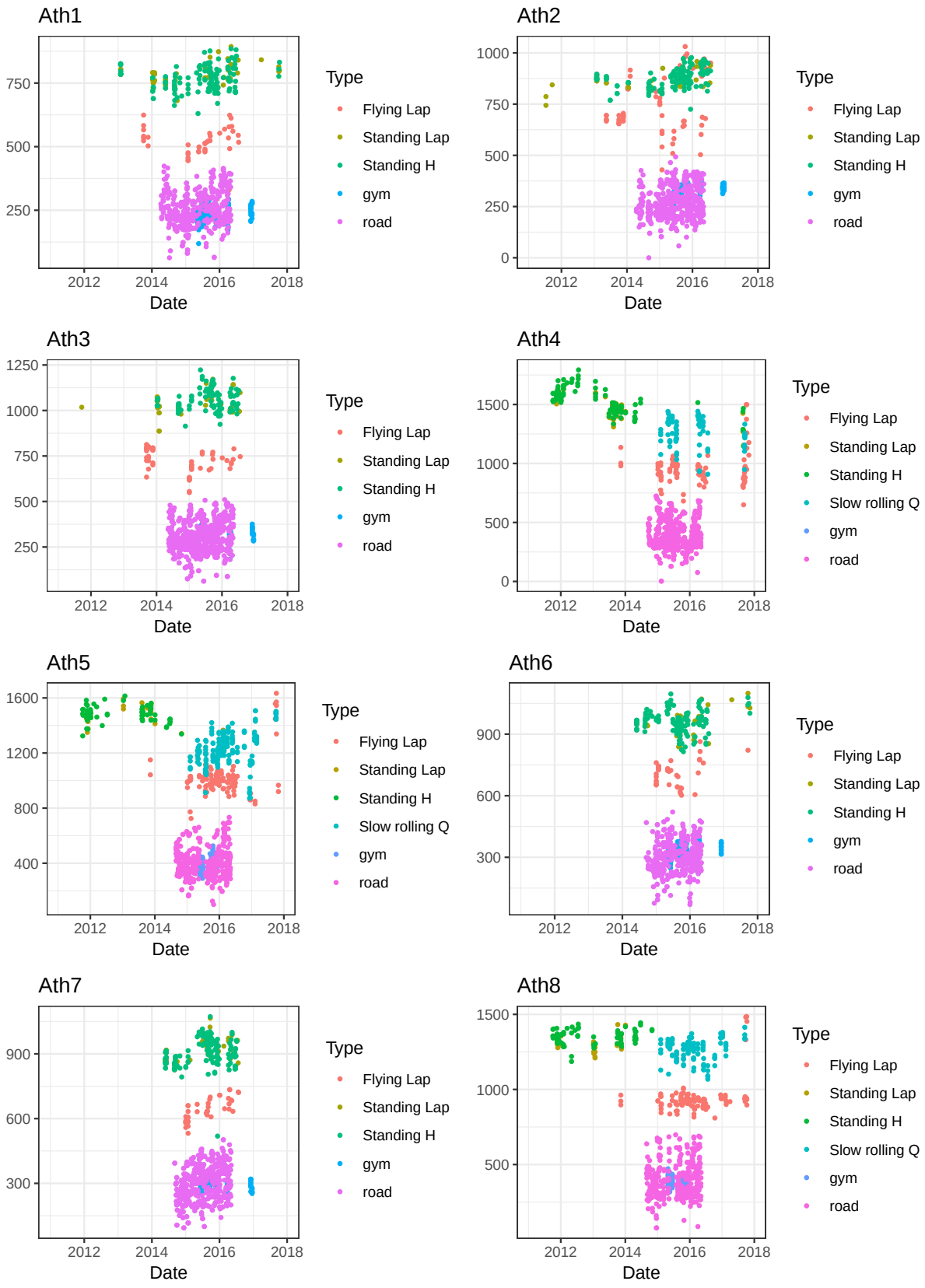

Figure 6.6: Plot of training efforts for top eight athletes with most recorded efforts.

the peak power produced by the athlete. We use the same type of single factor model that we used for the simulation study in Section 6.5.2. The model can be written as

$$\begin{aligned}
\eta_1(t) &= v_1(t) \\
\eta_2(t) &= \beta_{21}\eta_1(t) + v_2(t) \\
\eta_3(t) &= \beta_{31}\eta_1(t) + v_3(t) \\
\eta_4(t) &= \beta_{41}\eta_1(t) + v_4(t) \\
y_2(t) &= \mu_2 + \eta_2(t) + \epsilon_2(t) \\
y_3(t) &= \mu_3 + \eta_3(t) + \epsilon_3(t) \\
y_4(t) &= \mu_4 + \eta_4(t) + \epsilon_4(t) \ ,
\end{aligned} \tag{6.12}$$

where $\eta_1(t)$ is the factor that explains shared variability between $\eta_2(t)$, $\eta_3(t)$, and $\eta_4(t)$. $y_2(t)$, $y_3(t)$, and $y_4(t)$ denote the peak power achieved in flying lap, standing lap, and standing half lap training efforts at time t. We estimate the model for each athlete individually, using a squared exponential kernel for all Gaussian processes. The data, along with the shifted posterior curves $\hat{\mu}_j + \eta_j(t)|\boldsymbol{y}$, are shown for each athlete in Figure 6.7. The plots indicate that a single factor fits the data well for all three athletes, implying that a global indicator of changes in peak power performance in the velodrome can be measured by finding the shared variability between different types of velodrome training efforts. The parameter estimates of the means of each observed process along with the coefficients of the factor for each athlete are given in Table 6.5. The parameter β_{21} was set to one for identifiability.

		μ_2	μ_3	μ_4	β_{21}	β_{31}	β_{41}
Ath1	estimate	531.69	793.35	774.97	1.00	0.57	0.49
	standard error	23.79	13.82	11.64	-	0.15	0.09
Ath2	estimate	742.34	859.21	855.82	1.00	0.69	0.59
	standard error	30.80	18.80	16.40	-	0.10	0.09
Ath3	estimate	700.89	1047.77	1057.37	1.00	1.38	0.95
	standard error	15.06	19.52	13.54	-	0.26	0.08

Table 6.5: Parameter estimates, along with standard errors, from model (6.12).

6.6.3 Associating standing half lap with ergometer and road measurements

The next type of model we consider is a regression model similar to the one used in the simulation study in Section 6.5.1. We aim to examine whether changes in ergometer peak torque and road average power are associated with changes in standing half lap performance. We choose standing half among flying lap and standing lap because there are more measurements for standing half. The model can be written as

$$\begin{aligned}
\eta_1(t) &= v_1(t) \\
\eta_2(t) &= v_2(t) \\
\eta_3(t) &= \beta_{31}\eta_1(t) + \beta_{32}\eta_2(t) + v_3(t) \\
y_1(t) &= \mu_1 + \eta_1(t) + \epsilon_1(t) \\
y_2(t) &= \mu_2 + \eta_2(t) + \epsilon_2(t) \\
y_3(t) &= \mu_3 + \eta_3(t) + \epsilon_3(t) \ ,
\end{aligned} \tag{6.13}$$

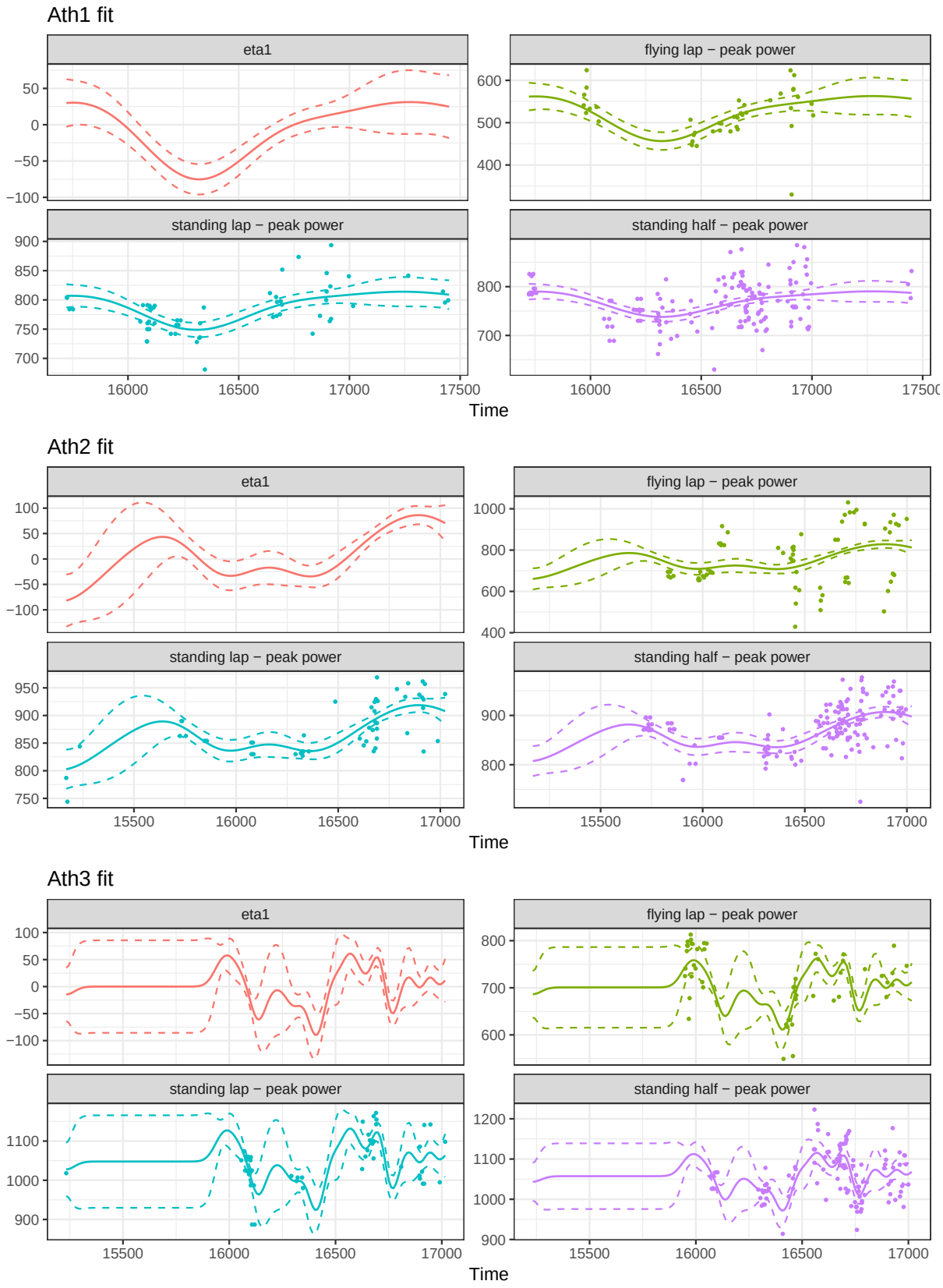

Figure 6.7: Fits obtained from the factor model formulated in (6.12). As shown in the plots a single factor does a good job of explaining over time changes in the peak power produced in flying lap, standing lap, and standing half lap efforts for Ath1, Ath2, and Ath3. 85

where $y_1(t)$, $y_2(t)$ and $y_3(t)$ are ergometer peak torque, road average power, and standing half peak power, respectively, at time t. The data, along with fitted curves, are shown for each athlete in Figure 6.8, while the parameter estimates, along with standard errors, are shown in Table 6.6. For all three athletes, β_{32} is small relative to its standard error, and hence no evidence is provided for a link between average power on the road and peak power in standing half laps. On the contrary, for all three athletes β_{31} is positive and larger than twice its standard error, and so evidence is provided for a positive association between peak torque produced on the ergometer and peak power produced in standing half laps.

		μ_1	μ_2	μ_3	β_{31}	β_{32}
Ath1	estimate	236.33	257.78	767.73	1.09	0.32
	standard error	8.58	9.16	9.27	0.35	0.30
Ath2	estimate	270.46	265.45	863.26	0.51	0.01
	standard error	37.62	8.11	18.82	0.12	0.22
Ath3	estimate	311.93	314.16	773.12	2.05	0.34
	standard error	7.28	11.24	285.86	0.46	0.67

Table 6.6: Parameter estimates, along with standard errors, from model (6.13).

6.6.4 Associating velodrome performance with ergometer and road measurements

The last model we consider with the BC data is a a combination of regression and factor analysis. The factor analysis model in Section 6.6.2 indicated that over time variability in peak power produced in flying laps, standing laps, and standing half laps could effectively be explained by a single factor. The regression model in Section 6.6.3 indicated that over time variability in peak torque produced on the ergometer could explain over time variability in peak power produced in standing half laps. In this section we assume that peak power produced in flying laps, standing laps, and standing half laps are fully determined by a single factor, which we assume represents overall velodrome performance, and allow this factor to depend on ergometer and road measurements. The model can be written as

$$
\begin{aligned}
\eta_1(t) &= v_1(t) \\
\eta_2(t) &= v_2(t) \\
\eta_3(t) &= \beta_{31}\eta_1(t) + \beta_{32}\eta_2(t) + v_3(t) \\
\eta_4(t) &= \beta_{43}\eta_3(t) \\
\eta_5(t) &= \beta_{53}\eta_3(t) \\
\eta_6(t) &= \beta_{63}\eta_3(t) \\
y_1(t) &= \mu_1 + \eta_1(t) + \epsilon_1(t) \\
y_2(t) &= \mu_2 + \eta_2(t) + \epsilon_2(t) \\
y_4(t) &= \mu_4 + \eta_4(t) + \epsilon_4(t) \\
y_5(t) &= \mu_5 + \eta_5(t) + \epsilon_5(t) \\
y_6(t) &= \mu_6 + \eta_6(t) + \epsilon_6(t) \ ,
\end{aligned}
\tag{6.14}
$$

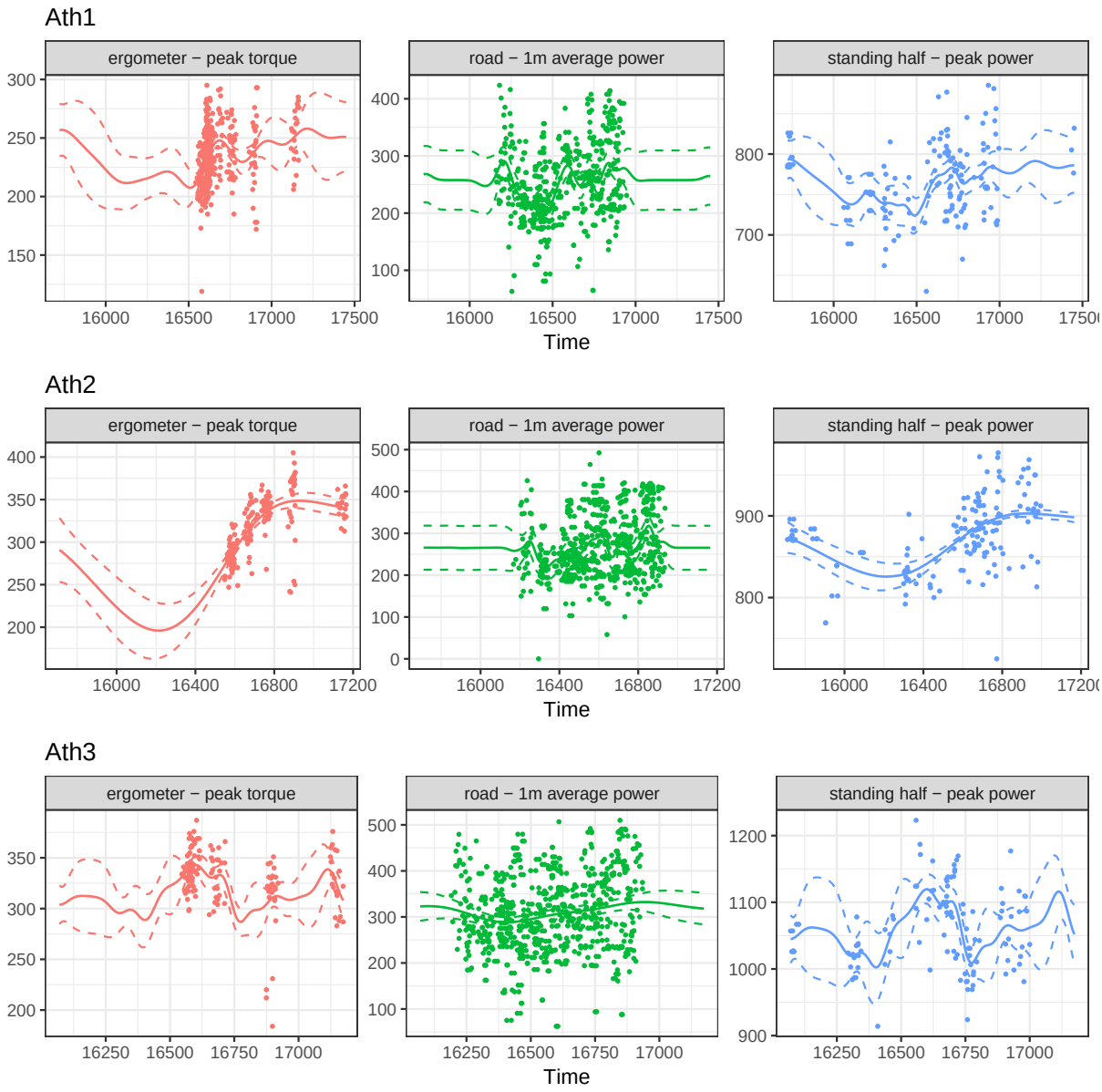

Figure 6.8: Fits obtained from the regression model formulated in (6.13). For all three athletes, the shape of the fitted curve for standing half peak power is almost completely determined by the shape of the fitted curve for ergometer peak torque.

where $y_1(t)$ represents peak torque on the ergometer, $y_2(t)$ is average power on the road, $y_4(t)$ is peak power in flying laps, $y_5(t)$ is peak power in standing laps, and $y_6(t)$ is peak power in standing half laps. $\eta_3(t)$ is the factor that we take to represent overall velodrome performance. The data, along with fitted curves, are plotted in Figure 6.9 while the parameter estimates along with standard errors are shown in Table 6.7. Unfortunately, because the Hessian of the log-likelihood produced at convergence by the BFGS algorithm is only an approximation, it is not guaranteed to be positive-definite and so occasionally it may happen that negative variance estimates are produced. This was the case for some of the parameters for Ath3, and so the standard errors for those parameters are left blank. Similarly to the regression of Section 6.6.3, the results from the combined regression and factor analysis indicate that over time changes in ergometer peak torque can explain over time changes in velodrome performance as a whole.

		μ_1	μ_2	μ_3	μ_4	μ_5	β_{31}	β_{32}	β_{43}	β_{53}	β_{63}
Ath1	estimate	238.19	258.18	522.98	791.45	771.77	1.26	0.31	1.00	0.89	0.77
	standard error	8.91	9.39	13.13	10.18	8.89	0.20	0.23	-	0.20	0.14
Ath2	estimate	276.03	265.58	603.62	776.01	786.39	1.37	-0.12	1.00	0.61	0.51
	standard error	25.58	8.16	138.98	89.39	75.11	0.45	0.36	-	0.19	0.16
Ath3	estimate	316.40	254.14	618.21	931.86	980.79	3.36	1.33	1.00	1.36	0.91
	standard error	3.58							-	0.30	0.19

Table 6.7: Parameter estimates, along with standard errors, from the combined factor analysis and regression model formulated in (6.14).

6.7 Conclusions and future work

In this chapter we introduced the structural smooth modelling (SSM) framework , and demonstrated its applicability through simulation studies and the BC data. The strength of SSM is that it jointly models a set of stochastic processes that have been observed at mis-matched argument values. In the simulations we conducted the model and associated estimation procedures appear to work as expected, and its application to the BC data shows great potential as a way to track changes in athlete's performance and examine associations between different types of training.

Further work with SSM could focus on three areas. Firstly, as presented, SSM is only applicable when the observed data are continuous, and hence can at least be approximated as noisy observations of Gaussian processes. In other applications researchers may wish to use the SSM framework with observed processes that are not continuous, for example with binary or count data. SSM could be extended to accommodate this scenario by allowing responses to be drawn from an exponential family distribution. The model could then be formulated as

$$\begin{aligned} \epsilon_j(t) &\sim \mathcal{N}(0, \phi_j) \\ v_j(t) &\sim GP(0, k_{\boldsymbol{\lambda}_j}) \ , \\ \boldsymbol{\eta}(t) = B\boldsymbol{\eta}(t) + \boldsymbol{v}(t) &\Leftrightarrow \boldsymbol{\eta}(t) = (I - B)^{-1}\boldsymbol{v}(t) \ , \\ g_j\Big(y_j(t)\Big) &= \mu_j + \eta_j(t) \ , \end{aligned} \tag{6.15}$$

where $g_j(\cdot)$ is a monotonic link function and $y_j(t)$ is assumed to have an exponential family distribution with dispersion parameter ϕ_j. The main complication that arises is that the likelihood of the observed data,

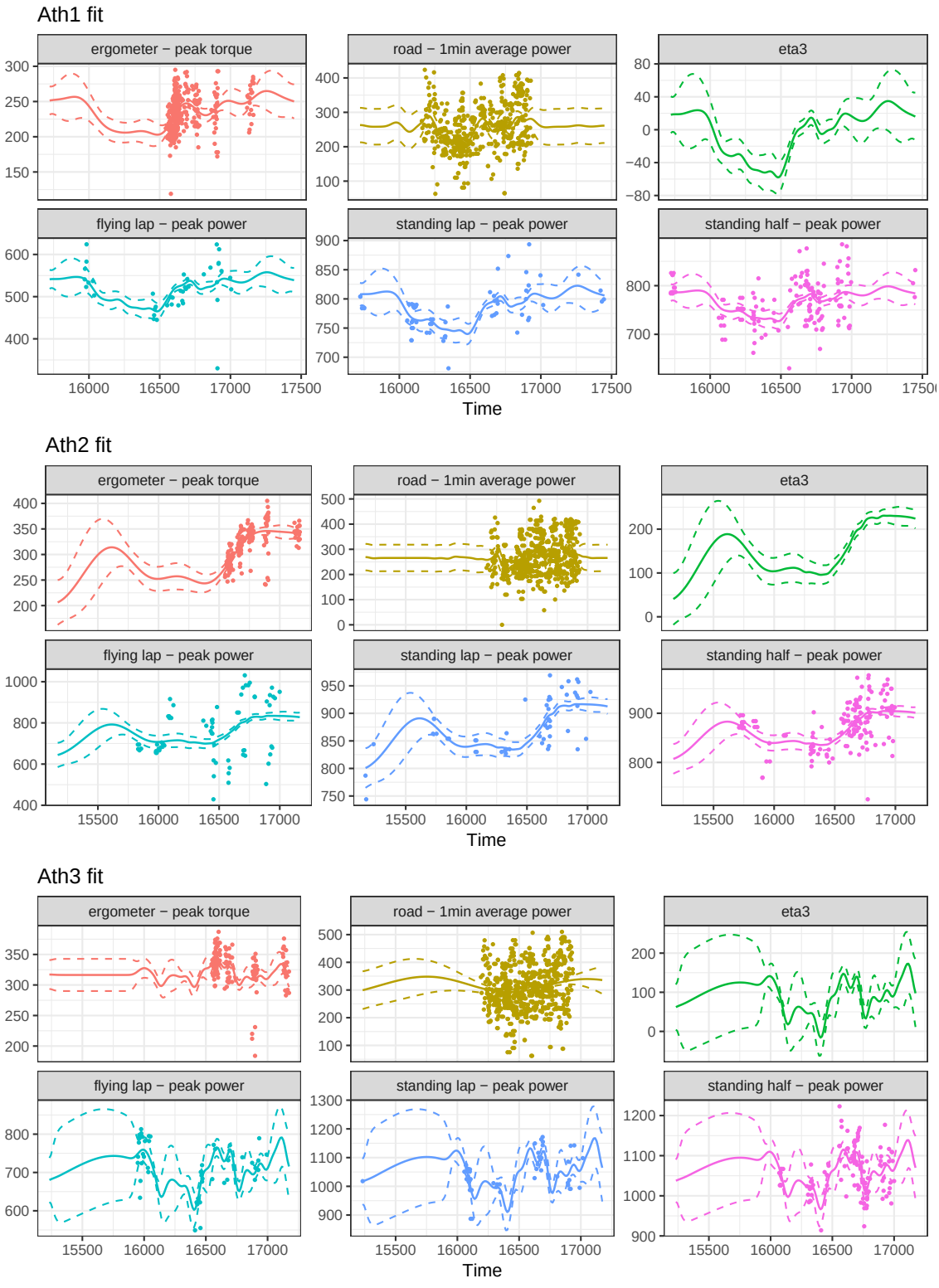

Figure 6.9: Fits obtained from the combined factor analysis and regression model formulated in (6.14). The fits show that a single factor seems to explain over time variability in flying lap, standing lap, and standing half peak power measurements, and the shape of this factor is mostly determined by the shape of the ergometer peak torque curve.

given by

$$\begin{aligned}\ell(\boldsymbol{\delta}) &= \log p\Big(y_1(\boldsymbol{t}_1), \ldots, y_p(\boldsymbol{t}_p); \boldsymbol{\delta}\Big) \\ &= \log \int p\Big(\eta_1(\boldsymbol{t}_1), \ldots, \eta_p(\boldsymbol{t}_p); \boldsymbol{\beta}, \boldsymbol{\lambda}\Big) \prod_{j=1}^{p} \pi_j\Big(y_j(\boldsymbol{t}_j)|\eta_j(\boldsymbol{t}_j); \mu_j, \phi_j\Big) d\eta_1(\boldsymbol{t}_1) \cdots \eta_p(\boldsymbol{t}_p) \ ,\end{aligned}$$

where $\pi_j(\cdot)$ is the exponential family density of $y_j(t)|\eta_j(t)$, is analytically intractable. Consequently, approximate inference techniques would have to be used. One option is to use a Laplace approximation to the integral, as is often done with Gaussian process classification (Rasmussen, 2006, Chapter 3).

The second area in which SSM could be expanded is in scalability. The likelihood function requires the inversion of the covariance matrix of the entire vector of observed data. This causes scaling issues when the number of data points grows, and is especially challenging for SSM in the presence of many observed processes. One of the approaches used to deal with this issue in Gaussian process regression is with the use of so called inducing points, through which a low rank approximation to the entire covariance matrix is obtained. This was the approach used in (Duncker and Sahani, 2018), and a similar approach could be used for SSM.

Lastly, it would be interesting to examine the properties of SSM from a theoretical perspective, establishing the precise conditions under which the parameters are identifiable and the maximum likelihood estimator is consistent. The study of the maximum likelihood estimator is complicated by the fact that the joint distribution of the entire data vector is modelled, and hence the log-likelihood is not a sum of independently identically distributed terms.

Appendix A

Worked example of splines

Suppose we compute b-spline functions of order 3 (i.e. piecewise quadratic polynomials) over the following two knot sequences:

1. $t^{(1)}$ = (0,1,2,3,4,5)
2. $t^{(2)}$ = (0,1,2,2,2,3,4,5)

In the first case, there are no repeating knots and hence all the b-splines will be $m-1=1$ times continuously differentiable. In the second scenario, the first spline will be continuous, but will not be differentiable at $x=2$. The second and third splines will be not be continuous at 2. The fourth will be continuous but not differentiable at 2, while the fifth will be continuous and differentiable everywhere. We can use equation (3.3) to actually compute these b-splines and verify this. For the first case, dropping the argument x for ease of notation, we have:

$$B_1^1 = \mathbb{1}_{(0\le x<1)}, \quad B_2^1 = \mathbb{1}_{(1\le x<2)}, \quad B_3^1 = \mathbb{1}_{(2\le x<3)}, \quad B_4^1 = \mathbb{1}_{(3\le x<4)}, \quad B_5^1 = \mathbb{1}_{(4\le x<5)} \ .$$

$$\begin{aligned}
B_1^2 &= x\mathbb{1}_{(0\le x<1)} + (2-x)\mathbb{1}_{(1\le x<2)} \\
B_2^2 &= (x-1)\mathbb{1}_{(1\le x<2)} + (3-x)\mathbb{1}_{(2\le x<3)} \\
B_3^2 &= (x-2)\mathbb{1}_{(2\le x<3)} + (4-x)\mathbb{1}_{(3\le x<4)} \\
B_4^2 &= (x-3)\mathbb{1}_{(3\le x<4)} + (5-x)\mathbb{1}_{(4\le x<5)} \ .
\end{aligned}$$

$$\begin{aligned}
B_1^3 &= \frac{x^2}{2}\mathbb{1}_{(0\le x<1)} + (-x^2+3x-3/2)\mathbb{1}_{(1\le x<2)} + \frac{(3-x)^2}{2}\mathbb{1}_{(2\le x<3)} \\
B_2^3 &= \frac{(x-1)^2}{2}\mathbb{1}_{(1\le x<2)} + (-x^2+5x-11/2)\mathbb{1}_{(2\le x<3)} + \frac{(4-x)^2}{2}\mathbb{1}_{(3\le x<4)} \\
B_3^3 &= \frac{(x-2)^2}{2}\mathbb{1}_{(2\le x<3)} + (-x^2+7x-23/2)\mathbb{1}_{(3\le x<4)} + \frac{(5-x)^2}{2}\mathbb{1}_{(4\le x<5)} \ .
\end{aligned}$$

We can now check, for example, that B_1^3 is continuous and differentiable at $x=2$. We have:

$$\begin{aligned}
\lim_{x\to 2^+} B_1^3(x) &= -2^2 + 3*2 - 3/2 = 1/2 \\
\lim_{x\to 2^-} B_1^3(x) &= (3-2)^2/2 = 1/2 \ .
\end{aligned}$$

Similarly,

$$\lim_{x\to 2^-} dB_1^3(x)/dx = -2*2+3 = -1$$
$$\lim_{x\to 2^+} dB_1^3(x)/dx = -(3-2) = -1 \ .$$

Now for the second knot sequence, we have:

$$B_1^1 = \mathbb{1}_{(0\le x<1)}, \quad B_2^1 = \mathbb{1}_{(1\le x<2)}, \quad B_3^1 = 0 \quad B_4^1 = 0 \quad B_5^1 = \mathbb{1}_{(2\le x<3)}, \quad B_6^1 = \mathbb{1}_{(3\le x<4)}, \quad B_7^1 = \mathbb{1}_{(4\le x<5)} \ .$$

$$\begin{aligned}
B_1^2 &= x\mathbb{1}_{(0\le x<1)} + (2-x)\mathbb{1}_{(1\le x<2)} \\
B_2^2 &= (x-1)\mathbb{1}_{(1\le x<2)} \\
B_3^2 &= 0 \\
B_4^2 &= (3-x)\mathbb{1}_{(2\le x<3)} \\
B_5^2 &= (x-2)\mathbb{1}_{(2\le x<3)} + (4-x)\mathbb{1}_{(3\le x<4)} \\
B_6^2 &= (x-3)\mathbb{1}_{(3\le x<4)} + (5-x)\mathbb{1}_{(4\le x<5)} \ .
\end{aligned}$$

$$\begin{aligned}
B_1^3 &= \frac{x^2}{2}\mathbb{1}_{(0\le x<1)} + (-3x^2/2+4x-2)\mathbb{1}_{(1\le x<2)} \\
B_2^3 &= \frac{(x-1)^2}{2}\mathbb{1}_{(1\le x<2)} \\
B_3^3 &= \frac{(3-x)^2}{2}\mathbb{1}_{(2\le x<3)} \\
B_4^3 &= (-3x^2/2+8x-10)\mathbb{1}_{(2\le x<3)} + \frac{(4-x)^2}{2}\mathbb{1}_{(3\le x<4)} \\
B_5^3 &= \frac{(x-2)^2}{2}\mathbb{1}_{(2\le x<3)} + (-x^2+7x-23/2)\mathbb{1}_{(3\le x<4)} + \frac{(5-x)^2}{2}\mathbb{1}_{(4\le x<5)} \ .
\end{aligned}$$

It is now easy to verify that the second are third b-splines are not continuous at $x=2$, since $(2-1)^2 = 1 \neq 0$ and $(3-2)^2 = 1 \neq 0$. Similarly, we can see that the first and fourth b-splines are not differentiable at 2, since $-3*2+4 \neq 0$ and $-3*2+8 \neq 0$. We can also visually verify these facts and see what these functions look like by plotting all the b-splines in both cases, shown in Figure A.1.

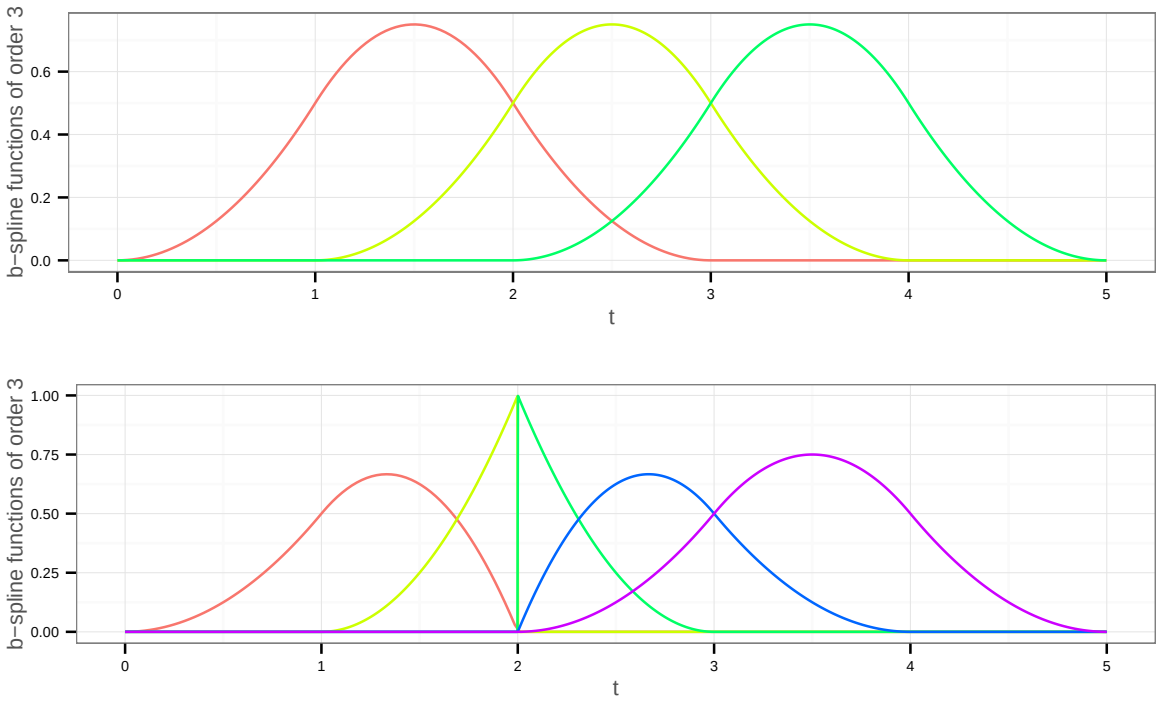

Figure A.1: plots of b-spline functions of order 3. The plot on top uses knot sequence $t^{(1)}$ while the plot on the bottom uses sequence $t^{(2)}$.

Bibliography

Agresti, A. (2015). *Foundations of Linear and Generalized Linear Models*. Wiley.

Arlot, S. and Celisse, A. (2010). A survey of cross-validation procedures for model selection. *Statistics Surveys*, 4:40–79.

Bollen, K. A. (1989). *Structural Equations with Latent Variables*.

Boyd, S., Parikh, N., E Chu, B. P., and Eckstein, J. (2010). Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122.

Cao, H., Li, J., and Fine, J. P. (2016). On last observation carried forward and asynchronous longitudinal regression analysis. *Electronic Journal of Statistics*, 10:1155–1180.

Cao, H., Zeng, D., and Fine, J. P. (2015). Regression analysis of sparse asynchronous longitudinal data. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 77(4):755–776.

Casella, G. (1985). An Introduction to Empirical Bayes Data Analysis. *The American Statistician*, 39(2).

Chen, L. and Cao, H. (2017). Analysis of asynchronous longitudinal data with partially linear models. *Electronic Journal of Statistics*, 11(1):1549–1569.

Cleveland, W. S. and Devlin, S. J. (1998). Locally Weighted Regression : An Approach to Regression Analysis by Local Fitting. *Journal of the American Statistical Association*, 83(403):596–610.

Cramer, H. (1999). *Mathematical Methods of Statistics*. Princeton University Press.

de Boor, C. (2001). *A practical guide to splines*. Springer.

Dempster, A., Laird, N., and Rubin, D. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal Of The Royal Statistical Society Series B-Methodological*, 39(1):1–38.

Duncker, L. and Sahani, M. (2018). Temporal alignment and latent Gaussian process factor inference in population spike trains. *bioRxiv*, page 331751.

Dung, V. T. and Tjahjowidodo, T. (2017). A direct method to solve optimal knots of B- spline curves: An application for non-uniform B-spline curves fitting. *PLoS ONE*, 12(2):1–24.

Eilers, P. H. C. and Marx, B. D. (1996). Flexible smoothing with B-splines and penalties. *Statistical Science*, 11(2):89–121.

Figueiredo, M. a. T. (2003). Adaptive Sparseness for Supervised Learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(9):1150–1159.

Fletcher, R. (1987). *Practical Method of Optimization*. Wiley & Sons.

Fox, J. and Weisberg, S. (2011). *An R Companion to Applied Regression*. Sage, Thousand Oaks CA, second edition.

Friedman, J., Hastie, T., and Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software*, 33(1):1–22.

Frommlet, F. and Nuel, G. (2016). An Adaptive Ridge Procedure for L0 Regularization. *PLOS ONE*, 11.

Gentleman, M. W. (1974). Algorithm AS 75: Basic Procedures for Large, Sparse or Weighted Linear Least Problems Author(s):. *Journal of the Royal Statistical Society. Series C: Applied Statistics*, 23(3):338–345.

Goepp, V. (2018). aspline: Fitting Adaptive Splines. *https://github.com/goepp/aspline*.

Goepp, V., Bouaziz, O., and Nuel, G. (2018). Spline Regression with Automatic Knot Selection. *¡hal-01853459¿*.

Golub, G. and Van Loan, C. (2012). *Matrix Computations (4th Ed.)*. Johns Hopkins University Press.

Golub, G. H., Heath, M., and Wahba, G. (1979). Generalized Cross-Validation as a Method for Choosing Good Ridge Parameter. *Technometrics*, 21(2).

Green, P. J. and Silverman, B. W. (1994). *Nonparametric Regression and Generalized Linear Models: A Roughness Penalty Approach*. Chapman & Hall.

Hastie, T. and Tibshirani, R. (1986). Generalized Additive Models. *Statistical Science*, 1(3):297–310.

Hastie, T., Tibshirani, R., and Friedman, J. (2009). *The Elements of Statistical Learning*, volume 1.

Hoerl, A. E. and Kennard, R. W. (1970). Ridge Regression: Biased Estimation for Nonorthogonal Problems. *Technometrics*, 12(1):55–67.

Jhong, J.-H., Koo, J.-Y., and Lee, S.-W. (2017). Penalized B-spline estimator for regression functions using total variation penalty. *Journal of Statistical Planning and Inference*.

Jöreskog, K. (1970). A general method for estimating a linear structural equation system. *ETS Research Bulletin Series*.

Jöreskog, K. G. (2001). LISREL 8.50.

Keesling, J. (1972). Maximum Likelihood Approaches to Causal Analysis. *Ph.D. dissertation, Department of Education, University of Chicago*.

Kiiveri, H. T. (2003). A Bayesian approach to variable selection when the number of variables is very large. *Lecture Notes-Monograph Series*, 40(2003):127–143.

Kim, S.-j., Koh, K., Boyd, S., and Gorinevsky, D. (2009). ℓ_1 Trend Filtering. *SIAM Review*, 51(2):339–360.

Lichman, M. (2013). UCI Machine Learning Repository.

Lumley, T. (2013). biglm: bounded memory linear and generalized linear models.

Mammen, E. and Van De Geer, S. (1997). Locally Adaptive Regression Splines. *The Annals of Statistics*, 25(1):387–413.

McCullagh, P. and Nelder, J. (1989). *Generalized Linear Models*. Chapman and Hall.

Meng, X.-L. and Rubin, D. B. (1993). Maximum Likelihood Estimation via the ECM Algorithm: A General Framework. *Biometrika*, 80(2):267–278.

Millar, R. B. (2011). *Maximum Likelihood Estimation and Inference: With Examples in R, SAS and ADMB*. Wiley.

Miller, A. J. (1992). Algorithm AS 274: Least Squares Routines to Supplement Those of Gentleman. *Journal of the Royal Statistical Society. Series C: Applied Statistics*, 41(2):338–345.

Oakes, D. (1999). Direct calculation of the information matrix via the EM algorithm. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 61(2):479–482.

Parikh, N. and Boyd, S. (2013). Proximal Algorithms. *Foundations and Trends in Optimization*, 1(3):123–231.

Park, T. and Casella, G. (2008). The Bayesian Lasso. *Journal of the American Statistical Association*, 103(482):681–686.

Petersen, A., Witten, D., and Simon, N. (2014). Fused Lasso Additive Model. *Journal of Computational and Graphical Statistics*, 25(4).

Rasmussen, C. E. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Rehfeld, K., Marwan, N., Heitzig, J., and Kurths, J. (2011). Comparison of correlation analysis techniques for irregularly sampled time series. *Nonlinear Processes in Geophysics*, 18(3):389–404.

Sadhanala, V. and Tibshirani, R. J. (2018). Additive Models with Trend Filtering. *preprint at arXiv:1702.05037*.

Silber, J. H., Rosenbaum, P. R., and Ross, R. N. (1995). Comparing the Contributions of Groups of Predictors: Which Outcomes Vary With Hospital Rather Than Patient Characterstics. *Journal of the American Statistical Association*, 90(429):7–18.

Tibshirani, R. (1996). Regression Shrinkage and Selection via the Lasso. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 58(1):267–288.

Tibshirani, R. J. (2014). Adaptive piecewise polynomial estimation via trend filtering 1. *The Annals of Statistics*, 42(1):285–323.

Tibshirani, R. J. and Taylor, J. (2011). The solution path of the generalized lasso. *Annals of Statistics*, 39(3):1335–1371.

Warwick J, N. and Marine Research Laboratories, T. (1994). *The Population biology of abalone (Haliotis species) in Tasmania*. Taroona, Tas: Sea Fisheries Division, Marine Research Laboratories.

Watson, G. S. (1964). Smooth Regression Analysis. *Sankhyā: The Indian Journal of Statistics*, 26(4).

Waugh, S. G. (1995). *Extending and Benchmarking Cascade-Correlation*. PhD thesis.

Wiley, D. (1973). The identification problem for structural equation models with unmeasured variables. *Econometrica*, 40.

Wood, S. (2004). Stable and efficient multiple smoothing parameter estimation for generalized additive models. *Journal of the American Statistical Association*, 99:673–686.

Wood, S. N. (2000). Modelling and smoothing parameter estimation with multiple quadratic penalties. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 62(2):413–428.

Wood, S. N. (2006). *Generalized Additive Models: an introduction with R*. CRC Press.

Wood, S. N. (2011). Fast stable restricted maximum likelihood and marginal likelihood estimation of semi-parametric generalized linear models. *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, 73(1):3–36.

Wood, S. N., Pya, N., and Säfken, B. (2016). Smoothing parameter and model selection for general smooth models. 1459(March).

Wright, S. (1918). On the Nature of Size Factors. *Genetics*, 3(4).

Wright, S. (1934). The method of path coefficients. *Annals of Mathematical Statistics*, 5.

Wright, S. (1960). Path coefficients and path regressions: Alternative or complementary concepts? *Biometrics*, 16.