THE UNIVERSITY OF
SYDNEY

IMPROVED MONOTONE POLYNOMIAL FITTING WITH APPLICATIONS AND VARIABLE SELECTION

KEVIN MURRAY

A thesis submitted in fulfilment of
the requirements for the degree of
Doctor of Philosophy

School of Mathematics and Statistics
Faculty of Science
The University of Sydney

10 July 2015

ABSTRACT

In many regression settings it is known, for example from some underlying physical or economic theory, that the regression curve is monotone. While many fully parametric monotone regression models exist, in particular in the growth curve literature, such models may miss important features such as the number and location of roots, extrema and inflection points of (higher order) derivatives of a regression function. Given the many useful properties of monotone polynomials, for example that they are naturally strictly monotone and have easily identifiable derivatives, we investigate the use of monotone polynomials as a tool for exploring isotonic relationships in data.

Initially in Chapter 2 we revisit Hawkins (1994) algorithm for fitting monotone polynomials and discuss two major practical issues that we encountered using this algorithm; namely when fitting high degree polynomials, and situations with a sparse design matrix but multiple observations per x-value. As an alternative, we describe two new approaches to fitting monotone polynomials to data. The first approach is based on the different characterisations of monotone polynomials, and extensions of these, proposed by Elphinstone (1983) and by using a Levenberg–Marquardt type algorithm. We consider the effectiveness of these different parameterisations, examine effective starting values for the non-linear algorithms, and discuss some limitations. The second method provides an approach for fitting monotone polynomials to data which markedly improves on the algorithms of both Hawkins (1994) and those created based on the Elphinstone parameterisations. We use a sum of squared polynomials parameterisation to achieve this which in turn allows us to impose monotonicity constraints to be over either a compact interval or a semi-compact interval of the form $[a, \infty)$, which is in contrast to the previous approaches that imposed monotonicity over the whole real line. This is the first time such algorithms have been made available and the ramifications and potential benefits of this are discussed. Material described in this chapter form parts of Murray, Müller & Turlach (2013, 2015).

In the subsequent chapters we consider inference for monotone polynomials using our new methodology. Estimates of variation in monotone polynomials has received no attention in the literature to date with the main focus being on trying to find effective algorithms to fit monotone polynomials to data. In Chapter 3 we provide extensive investigations into estimates of variance for the fitted values and to estimates of variance around the parameter estimates. We use Monte-Carlo simulations and compare several bootstrapping algorithms to not only demonstrate the need for such techniques, but to identify situations in which these may need adjusting. We initially start by demonstrating consistency of monotone polynomials and provide, through Monte-Carlo sampling, empirical results with justifications for considering alternative approaches to standard least squares unconstrained model fitting for monotone polynomials. We describe situations where the use of standard bootstrap methodology for polynomials under the monotonicity constraints produces particularly low coverage probabilities in certain areas of the fitted curve, dependent on how close to the boundary of the cone of monotone polynomials the underlying function is. Consequently we describe alternative methodologies to address this issue and show that an adjustment by using either

the m out of n bootstrap or a post hoc symmetrisation of the confidence bands is necessary to achieve more uniform coverage probabilities over the whole range of the curve. We show that using standard bootstrap techniques to generate point wise *prediction intervals* does not appear to suffer from the same under coverage problems as the corresponding confidence intervals.

In Chapter 4 we provide some comparisons of monotone polynomials to existing shape constrained regression methods and also look at speed comparisons. We show through Monte-Carlo simulations that the bias and variance produced from monotone polynomial curve fitting are at least comparable to those from the non-parametric smoothing techniques, and for certain functions monotone polynomials provide better results. This is demonstrated not only for data which are generated from polynomial functions but also for those that originate from a sigmoidal or trigonometric function.

We compare several of the existing methods available to fit monotone polynomials to data, described in Murray, Müller & Turlach (2013), and demonstrate that the sums of squares formulation is much faster than previous monotone formulations and comparable to the semi-infinite programming approach of Hawkins (1994), with the sums of squares formulation proving much more flexibility, for example through the ability to fit over a compact or semi-compact interval. When compared to smoothing spline approaches, we demonstrate the monotone polynomial model fitting appears to perform comparatively well for small sample sizes, but is more effective with smaller run times when the sample sizes are larger. We show that the efficiency of our methodology is pertinent to the ability to use computationally intensive techniques such as the bootstrap in variance estimation. Furthermore, we demonstrate that in many instances monotone polynomials are at least comparable in terms of bias and variance to readily available constrained regression techniques. Material described in this chapter form part of Murray, Müller & Turlach (2015).

In Chapter 5 we switch focus to model selection techniques, and the visualisation of the model selection process. This chapter describes graphical methods that assist in the selection of models and comparison of many different selection criteria. Specifically, we describe for logistic regression initially, how to visualize measures of description loss and of model complexity to facilitate the model selection dilemma. We advocate the use of the bootstrap to assess the stability of selected models and to enhance our graphical tools. We demonstrate which variables are important using *variable inclusion plots* and show that these can be invaluable plots for the model building process. We show with two general case studies how these proposed tools are useful to learn more about important variables in the data and how these tools can assist the understanding of the model building process. Furthermore, we extend these techniques to model selection using monotone polynomials, identify some of the issues with model selection in such constrained scenarios, and provide a further two real world examples to illustrate this process. The majority of this chapter forms the basis for the published article by Murray, Heritier & Müller (2013) and is based on analysis carried out by Murray in Rosenwax, McNamara, Murray *et al.* (2011) and Lawrance, Murray, Batman *et al.* (2013).

Finally in Chapter 6 we describe methods for detecting features of curves from fitted models, for example derivatives or inflection points, through fitting monotone polynomials to data. We demonstrate how simple this process now becomes and how effective this technique is for such a problem and describe through our comparative examples with other techniques for this particular problem, how other such techniques have

many difficulties associated with identifying such features. We show through simulated data and real world examples how effective using monotone polynomials can be when the aim is to identify features of curves. We finalise the thesis by demonstrating the use of monotone polynomials with a well known, and analysed, adolescent growth curve data set and demonstrate the simplicity and effectiveness of monotone polynomials curve fitting using our newly developed methods and techniques.

All monotone polynomial algorithms and methods discussed are available in the latest version of our newly created R (R Core Team, 2015) package `MonoPoly` (version 0.3-6 or later). The graphical model selection work has already been further developed and now forms part of the `mplot` R package.

PUBLICATIONS, PRESENTATIONS AND SOFTWARE FROM THIS THESIS

Some ideas and figures have appeared previously in the following publications:

- Murray, K., Heritier, S. & Müller, S. (2013). Graphical tools for model selection in generalized linear models. Statistics in Medicine 32, 4438–4451. (Chapter 5)
- Murray, K., Müller, S. & Turlach, B.A. (2013). Revisiting fitting monotone polynomials to data. Computational Statistics 28, 1989–2005. (Chapter 2)
- Murray, K., Müller, S. & Turlach, B.A. (2015 under review). Flexible and fast monotone polynomial fitting. Submitted . (Chapter 2, 3, 4)

The methods developed in this thesis are based on analyses carried out by Murray in:

- Rosenwax, L.K., McNamara, B.A., Murray, K., McCabe, R.J., Aoun, S.M. & Currow, D.C. (2011). Hospital and emergency department use in the last year of life: A baseline for future modifications to end-of-life care. Medical Journal of Australia. 194, 570–573.
- Lawrance, I.C., Murray, K., Batman, B., Gearry, R.B., Grafton, R., Krishnaprasad, K., Andrews, J.M., Prosser, R., Bampton, P.A., Cooke, S.E., Mahy, G., Radford-Smith, G., Croft, A. & Hanigan, K. (2013). Crohns disease and smoking: Is it ever too late to quit? Journal of Crohns and Colitis 7, e665–e671.

Furthermore, the following manuscript is in preparation:

- Murray, K., Müller, S. & Turlach, B.A. (2015 in preparation). Using monotone polynomials to detect inflection points. (Chapter 6)

The work from this thesis has also been presented at the following conferences:

- A new algorithm for fitting monotone polynomials to data, *International Sri Lankan Statistical Conference*, 2010, Columbo, Sri Lanka.
- Fitting monotone polynomials to data, *Young Statisticians Conference*, 2011, Brisbane, Australia.
- A comparison of monotone polynomials and monotone smoothing splines in the detection of inflection points, *Young Statisticians Conference*, 2013, Melbourne, Australia.
- Using monotone polynomials and monotone smoothing splines to detect inflection points, *International Biometric Conference*, 2014, Florence, Italy.

We have produced an R package entitled `MonoPoly` to accompany this thesis and an R package entitled `mplot` has been produced by other authors utilising our work in Chapter 5 of this thesis.

To find out what happens when you change something,
it is necessary to change it.

Box, Hunter and Hunter (Box, Hunter & Hunter, 1978)

ACKNOWLEDGMENTS

During the time enrolled at the University of Sydney I have been fortunate enough to have the generous support of colleagues, family, friends and others I have met along the way. I would like to express my gratitude to all of you through this acknowledgement.

Firstly, to my supervisors, Samuel Müller from the University of Sydney and Berwin Turlach from the University of Western Australia; I simply could not have asked for a better team to guide me through. You have both been an inspiration to me. However, it has not gone unnoticed that my written English now has a some what German tinge to it!

Samuel, I will never forget your unwavering support, encouragement and general all-round knowledge. You have guided me through this journey which has taken many twists and turns and at each difficult junction you have provided sensible advice, suggestions and encouragement that has kept me going, even when the end seemed so far away. You have always been extremely generous with your time and I have always had the utmost confidence in any advice you have provided me. Although, with hindsight, blindly following you on to a somewhat questionable form of transport during the Columbo conference was probably not one of my better decisions!

Berwin — what can I say? Well for one I now know the difference between emdash and endash in LaTeX 2ε, even if I still manage to get them wrong! Of course, your keen eye for detail has not gone unnoticed and whilst I express frustration at the millions of tiny things you pick up every time I present something to you, it is also with great appreciation that you go to such extremes. However, whilst this an appreciable quality in a supervisor, it is your all round knowledge of mathematics, statistics and programming that has made my life easier throughout this thesis and for as long as I can remember before. It has always been a pleasure working with you, and I will always remember to bring coffee to any meetings!

To both of you — you have my utmost respect and gratitude and I sincerely hope you have both enjoyed the ride and that there are many good collaborative years ahead of us.

I have been lucky enough to have had many travel opportunities and I would like to thank the University of Sydney School of Mathematics and Statistics for supporting me on these endeavours. I would also like to thank the generous support the University of Western Australia has provided in terms of time away from my usual roles to pursue this endeavour.

During my early days working at the University of Western Australia, I was fortunate enough to have a wonderful group of people, both statisticians and those non-statistically minded, around me who were full of encouragement, both before I started and during my candidature. The list is too long to mention but I would like to give special mention to two of my earlier mentors in Kaipillil Vijayan and Martin Hazelton. Not only were you there at the start of my long statistical journey in Australia but have been a source of encouragement and expertise ever since. My all round statistical knowledge is forever in your debt. I would also like to thank Charley Budgeon, who has not only provided me with extensive support, particularly in the later stages of my

candidature, with numerous reviews and proof readings, but has been a source of general enthusiasm throughout — it's difficult to see how this would have been completed without our regular cups of coffee and your extensive advice on graphics!

I would like to thank my parents for being a constant support in all my endeavours, from a child to the only marginally less child like person I am now — your encouragement and pride in everything I do is appreciated now more than ever.

To my wife Emma — you have stuck by me through this journey and for that you will have my eternal gratitude. You have never once bemoaned the time I have put into my work and the travel I have done, and have provided an endless source of encouragement and support throughout. This is something never to be forgotten — you are truly a wonderful person and a great wife and mother to our children. It, also hasn't gone unnoticed that during my candidature you have not only had to put up with me, but have managed to give birth to our two wonderful children. A truly magnificent effort for which I am forever in your debt.

Finally, my children, Aidan and Holly — there are no others who have impacted on my life as much as you. You are my world and I dedicate this thesis with eternal love to you both.

CONTENTS

1 Introduction and background 1
1.1 Background on monotonic curve estimation . . . 1
1.2 Uses for monotone regression models . . . 3
1.3 The aims and outline of this thesis . . . 5
2 Fitting monotone polynomials to data 10
2.1 Introduction . . . 11
2.2 Background . . . 13
2.3 Hawkins' approach . . . 14
2.4 Isotonic parameterisations . . . 17
2.4.1 Elphinstone type . . . 17
2.4.2 Sum of squared polynomials . . . 23
2.5 Optimising the objective function . . . 27
2.5.1 Levenberg–Marquardt routine . . . 28
2.5.2 Starting values . . . 29
2.5.3 Stopping criteria . . . 36
2.6 Miscellaneous comments . . . 37
2.7 Numerical experiments and results . . . 38
2.7.1 Simulated data examples . . . 38
2.7.2 Optimisation speed . . . 43
2.7.3 Random starting values . . . 45
2.7.4 Rank deficient designs . . . 45
2.8 Conclusions . . . 47
3 Inference for monotone polynomials 49
3.1 Introduction . . . 50
3.2 Simulated data and motivation . . . 51
3.3 Theoretical properties . . . 53
3.3.1 Consistency . . . 53
3.3.2 Bias and variance . . . 54
3.4 Empirical properties . . . 55
3.4.1 Consistency . . . 55
3.4.2 Bias and variance . . . 57
3.5 Bootstrapping . . . 60
3.5.1 Bootstrap in regression . . . 63
3.5.2 Notation and bootstrap algorithms . . . 64
3.6 Bootstrapping - numerical examples . . . 67
3.6.1 Estimating the variance of regression parameters . . . 69
3.7 Bootstrap confidence intervals and prediction intervals . . . 74
3.7.1 Confidence bands for polynomials using standard least squares estimation . . . 74
3.7.2 Confidence bands and prediction intervals for monotone polynomials . . . 76
3.7.3 Coverage probabilities . . . 77
3.8 Conclusions . . . 84

4 A comparison of monotone regression techniques 86
4.1 Introduction . . . 86
4.2 Comparisons with previous parameterisations . . . 88
4.3 Monte-Carlo simulations to compare constrained regression techniques . 91
4.4 Speed comparisons with different approaches to fitting monotone curves 94
4.5 Discussion . . . 97
5 Graphical tools for model selection 99
5.1 Introduction . . . 99
5.2 Terminology and graphical methods . . . 103
5.2.1 Generalised linear model framework . . . 103
5.2.2 Measuring description loss and complexity in GLM . . . 103
5.3 Case study I – Simulated logistic regression example . . . 104
5.3.1 Visualising L_n and p_α . . . 105
5.3.2 Assessing model stability through bootstrapping . . . 106
5.3.3 The maximum enveloping lower convex curve . . . 109
5.3.4 The variable inclusion plot . . . 110
5.4 Results and further applications . . . 112
5.4.1 Case study II – Cross-sectional palliative care study . . . 112
5.4.2 Case study III – Smoking in patients with Crohns disease . . . 117
5.5 Model selection plots with Monotone Polynomials . . . 121
5.5.1 Simulated data example . . . 121
5.5.2 Case study IV - Brain function data . . . 124
5.5.3 Case study V - Estimation of dental age . . . 127
5.6 Additional uses for model selection plots . . . 129
5.6.1 Correlated predictors . . . 130
5.6.2 Outliers in data . . . 132
5.6.3 Other uses for model selection plots . . . 132
5.7 Discussion . . . 133
6 Using monotone polynomials for detecting inflection points 135
6.1 Introduction . . . 136
6.2 Inflection points . . . 137
6.3 Simulated data examples . . . 138
6.4 Numerical experiments - results . . . 140
6.4.1 Selecting the degree of monotone polynomial . . . 141
6.4.2 Number and location of inflection points . . . 144
6.5 Real world examples . . . 150
6.5.1 Firmin et al. brain function data . . . 150
6.5.2 Growth curve examples from Berkley Guidance Study . . . 155
6.6 Conclusions . . . 158
A Appendix A: Evaluating objective function and its derivatives for models (2.7) and (2.8) 161
A.1 Model parameterisation (2.7) . . . 161
A.2 Model parameterisation (2.8) . . . 163
B Appendix B: Further results from simulation studies carried out in Chapter 3 166
C Appendix C: Additional example of curvature plot from Chapter 3 169

BIBLIOGRAPHY 170

LIST OF FIGURES

Figure 1.1 Example data from Berkley Guidance study . . . 7
Figure 2.1 Description of the Levenberg-Marquardt algorithm . . . 30
Figure 2.2 Data sets used for numerical experiments . . . 39
Figure 2.3 Parameter estimates changes through the iterative stages I . . . 41
Figure 2.4 Parameter estimates changes through the iterative stages II . . . 42
Figure 2.5 Processing time for algorithms to converge by degree . . . 44
Figure 2.6 RSS for final converged iteration by degree . . . 45
Figure 2.7 Monotone fits to subset of $W2$ data . . . 46
Figure 3.1 Monotone polynomial fitted curves for data generated from polynomial functions . . . 56
Figure 3.2 Monotone polynomial fitted curves for data generated from sigmoidal functions . . . 57
Figure 3.3 Standard unconstrained linear model fits and monotone polynomial fits to data generated from polynomial function I . . . 61
Figure 3.4 Standard unconstrained linear model fits and monotone polynomial fits to data generated from polynomial function II . . . 62
Figure 3.5 Histograms of non-parametric bootstrap with overlaid kernel density estimates I . . . 71
Figure 3.6 Histograms of non-parametric bootstrap with overlaid kernel density estimates II . . . 73
Figure 3.7 Coverage probabilities for 80% confidence intervals for three different polynomial functions . . . 75
Figure 3.8 Coverage probabilities for 80% confidence intervals for data generated from polynomial functions . . . 78
Figure 3.9 Non-parametric bootstrap results for the underlying cubic polynomial . . . 79
Figure 3.10 Prediction interval plots for cubic and quintic polynomials . . . 80
Figure 3.11 Coverage probabilities for 80% confidence intervals for the quintic monotone polynomial . . . 81
Figure 3.12 Fitted monotone polynomial based on simulated quintic data with adjusted 80% and 95% bootstrap confidence bands with adjusted coverage probabilities . . . 83
Figure 4.1 Data generating curves and different monotone polynomial fits . . . 89
Figure 4.2 Average time by sample size for three approaches to fitting monotone polynomials . . . 90
Figure 4.3 Average time by polynomial degree for three approaches to fitting monotone polynomials . . . 91
Figure 4.4 Bias, variance and mean squared error comparing the isotonic regression techniques for cubic function . . . 93
Figure 4.5 Bias, variance and mean squared error comparing the isotonic regression techniques for quintic function . . . 94

Figure 4.6 Bias, variance and mean squared error comparing the isotonic regression techniques for sigmoidal function . . . 95
Figure 4.7 Bias, variance and mean squared error comparing the isotonic regression techniques for trigonometric function . . . 96
Figure 4.8 Comparison of time to fit models for different sample sizes using different isotonic regression techniques and sigmoidal data . . . 96
Figure 5.1 Enriching the scatter plot - L_n versus dimension by model type . . 106
Figure 5.2 Assessing model stability using weighted bootstrapped probabilities . . . 108
Figure 5.3 Constructing the maximum enveloping lower convex curve . . . 108
Figure 5.4 Variable inclusion plot for simulated data . . . 111
Figure 5.5 Basic and enriched scatter plot with MELCC for case study II . . . 113
Figure 5.6 Model stability for case study II with weighted bootstrapped proportions . . . 115
Figure 5.7 Variable inclusion plot for case study II . . . 116
Figure 5.8 Simple model selection plots for case study III . . . 119
Figure 5.9 Variable inclusion plot for case study III . . . 120
Figure 5.10 L_n vs dimension for polynomial model selection with model stability plots . . . 123
Figure 5.11 Different degree 7 polynomial fits to brain function data of an individual . . . 125
Figure 5.12 Selection probabilities for the m out of n bootstrap by degree of polynomial and m for brain function data . . . 127
Figure 5.13 Selection probabilities for the m out of n bootstrap for dental maturity example . . . 129
Figure 5.14 Model stability plot for simulated correlated data . . . 130
Figure 5.15 Variable inclusion plot for simulated correlated data . . . 131
Figure 6.1 Simulated data, first and second derivatives for inflection point simulations . . . 141
Figure 6.2 Model selection probabilities for simulated data with monotonicity constraint lower bounded at 0 . . . 142
Figure 6.3 Model selection probabilities for simulated data with unbounded monotonicity constraint . . . 143
Figure 6.4 Fitted models over the whole real line and semi-compact interval with inflection points . . . 147
Figure 6.5 Distribution of inflection points for simulated data from model (6.5) . . . 149
Figure 6.6 Comparing two inflection point estimations for example patient X 152
Figure 6.7 Different monotone polynomial fits for patient X . . . 152
Figure 6.8 Model selection and fitted curve for patient X . . . 153
Figure 6.9 Bootstrap distribution of inflection points . . . 154
Figure 6.10 Comparison of inflection points for patient X . . . 154
Figure 6.11 Comparison of inflection points for selection of patients . . . 155
Figure 6.12 Comparison of inflection points for two children from the Berkley Guidance Study . . . 157
Figure 6.13 Comparison of inflection points . . . 157
Figure B.1 Comparison of different bootstrap methods $n = 50$ . . . 167

Figure B.2 Comparison of different bootstrap methods $n = 1,000$ 168
Figure C.1 Curvature and coverage probabilities for $p_4(x) = (x - 0.5)^5$ monotone polynomial using residuals bootstrap, with $m = n$. 169

LIST OF TABLES

Table 2.1 Objective function value RSS/n $\times 10,000$, by parameterisation for the four data sets 40
Table 3.1 Estimated proportion of monotone fits for a range of sample sizes and different polynomial functions 53
Table 3.2 Monte-Carlo estimates and standard errors on simulated data . . . 59
Table 3.3 Mean coverage probabilities over the range of x for 80% confidence intervals 76
Table 3.4 Summaries of coverage probabilities from different m out of n bootstrapping algorithms 82
Table 5.1 Full model, AIC and BIC best model, forward/backward best model, parameter estimates and standard errors 114
Table 5.2 Best models by dimension showing decrease in L_n for Case Study III 118
Table 5.3 Brain function data: Proportion of models selected using m out of n bootstrap by degree q and m 127
Table 5.4 Estimation of dental age data: Proportion of models selected using m out of n bootstrap 129
Table 6.1 Description of simulated data for inflection point detection. 139
Table 6.2 Distribution of number of inflection points 148

1

INTRODUCTION AND BACKGROUND

The work presented in this thesis combines work from two different projects worked on in parallel. The first being a larger project which has resulted in three chapters of this thesis focussing on estimation of, inference for, and applications of monotone polynomials. The second is a project on model selection, with a particular focus on graphical methods, which has resulted in one chapter of this thesis. In this introduction we provide a background and outline for the former of these projects with a detailed background and explanation on the latter provided in Chapter 5. The sixth and final chapter is an applications chapter.

1.1 BACKGROUND ON MONOTONIC CURVE ESTIMATION

In many regression settings it is known, for example from some underlying physical or economic theory, that the regression curve is monotone. Further examples include, but are not limited to, calibration problems, estimation of monotone transformations (for example to transform a variable to normality), growth curves, and dose-response curves. While many fully parametric monotone regression models exist, in particular in the growth curve literature, such models may miss important features such as the number and location of roots, extrema and inflection points of (higher order) derivatives of the regression function.

A researcher who needs to fit a monotone regression curve typically has to make the choice between a parametric nonlinear regression model (Ratkowsky, 1990), in particular those models developed in the growth curve literature (see, for example, Mirman, 2014; Panik, 2014), or a shape constrained smoothing technique. Popular methods for the latter approach involve the use of either spline smoothing or kernel smoothing techniques. Incorporating shape constraints into spline smoothing has been well studied

and, for reviews of the existing literature, we refer to the introductory sections of Turlach (2005), Hazelton & Turlach (2011) or Meyer (2008, 2012). By way of contrast, kernel smoothing techniques for (general) shape constraints (Marron, Turlach & Wand, 1997; Mammen, Marron, Turlach *et al.*, 2001) are somewhat less often used, but there is some notable work on monotone kernel smoothers (Friedman & Tibshirani, 1984; Mammen, 1991; Hall & Huang, 2001; Dette, Neumeyer & Pilz, 2006; Dette & Pilz, 2006).

Whilst both the parametric nonlinear regression models and the shape constrained smoothing techniques are extremely effective for many applications they are not without their drawbacks. Unfortunately approaches to monotone regression that use parametric nonlinear models may suffer from the problem that many parametric monotone regression models depend only on a few parameters and, hence, such models may miss important features such as the number and location of roots, extrema and inflection points of (higher order) derivatives of a regression function. On the other hand, most approaches that impose monotonicity on nonparametric smoothing techniques can be problematic due to the estimated regression curve having flat stretches, and monotone regression smoothing "has been criticized because practitioners do not believe in all those flat spots" (Dette, Neumeyer & Pilz, 2006). Having to contend with estimated regression functions with many (spurious) flat spots is highly undesirable, especially in situations where the estimation of derivatives (and features thereof) is important, as these would translate to increased variability of such estimates. Furthermore, while the approaches of Ramsay (1998), Dette, Neumeyer & Pilz (2006), and Hazelton & Turlach (2011) lead to estimated regression functions that are strictly monotone, the functional form of the estimated regression function does not lend itself readily to a subsequent analysis to aspects of its derivatives.

This suggests that there is a need for some methodology that occupies the middle ground between the two approaches discussed, that is there is a need for models that are more flexible than parametric monotone regression curves, but which lend themselves more easily to post-processing and further calculations than monotone nonparametric smoothing techniques. Monotone polynomials provide such a methodology and we note that these polynomials also exhibit many useful properties, for example that they are naturally strictly monotone, have easily identifiable derivatives, and have the

major advantage of only having one tuning parameter to estimate, that is the degree of the polynomial, as opposed to spline smoothing, which is similar in nature to a monotone polynomial, but has potentially many tuning parameters, such as the number and location of knots and the particular choice of the basis function.

1.2 USES FOR MONOTONE REGRESSION MODELS

In many real world situations there is a necessity to not only identify a model or set of models that adequately describe the underlying phenomenon, but that also have underlying features that are important for the application of interest. A well known example of the latter are growth curve models (see, for example, Mirman, 2014; Panik, 2014), in which many different models have been postulated. Specifically, human growth curves have been examined extensively and modelled in numerous different ways. In these situations the modeller is usually led by the well documented trajectories of adolescent growth; that is by four distinct periods prior to reaching final adult height around 18-19: rapid growth during infancy, steady growth in childhood, rapid growth during adolescence, and very slow growth approaching adulthood. These lead to the change in y (height) differing throughout the growth period, hence generating (multiple) inflection points. To model the whole human growth parametrically has been attempted and models such as the Preece-Baines models (Preece & Baines, 1978) for example have been developed.

However, in other situations there may be little or no knowledge of the underlying physical set of mathematical functions, but there are still specific aims. For example searching for the LD50 in a toxicological study, that is the individual dose required to kill 50% of a population of test subjects, would usually be based on the knowledge that the proportion survived (y) has a monotone decreasing relationship with the dose of the drug (x), but little more is known about the specific shape. In these instances one can either assume a known parametric from, for example the sigmoidal function, or adapt a more liberal approach by not specifying a functional form for the model explicitly.

In order to solve these types of problem many researchers have turned to a flexible set of non-parametric smoothing spline approaches, see, for example, the work by Ramsay (Ramsay, 1988, 1998; Ramsay & Silverman, 2002, 2006; Heckman & Ramsay, 2000; Ramsay, Wickham, Graves *et al.*, 2013), which has been used extensively. However, given the numerous decisions that need to be made when fitting smoothing splines, the optimisation process can become difficult, which in many instances leads to subjective choices of parameter values, thus in turn leading to non-robust solutions.

An example of the use of such methodology is described in Firmin, Müller & Rösler (2011, 2012) in which the motor evoked potentials in the brain is modelled in terms of a stimulus applied to an individual for different delay times. As the delay of the stimulus increases the motor evoked potential decreases defining a underlying phenomenon which is monotonic. In this research the number and location of inflection points is of paramount interest as it would enable further classification of individuals, for example to identify patients with diseases such as multiple sclerosis. In these situations, flexible models, with easily defined inflection points are needed as details on the specific locations of these features are very rarely known at the individual level, with many individuals having only one inflection point, some having them early, some having them late and many having much more than one inflection point. This obviously increases the difficulty of the modelling problem and further suggests that alternatives to the usual methods would be useful.

1.3 THE AIMS AND OUTLINE OF THIS THESIS

Our main thesis aims can be concisely put into two objectives:

> **OBJECTIVE 1:** Provide a major contribution to the research area of monotone curve estimation
>
> **OBJECTIVE 2:** Contribute to the model selection literature through the visualisation of the model selection process for generalised linear models

In order to achieve these objectives we describe the five major areas contributing to this thesis:

1. Investigate and develop the use of monotone polynomials for modelling the monotone relationship between x and y, in particular:
   - Develop new methodology for fitting monotone polynomials for data;
   - Expand on the existing methodology for fitting monotone polynomials to data;
   - Consider different optimisation routines, starting values, stopping rules;
   - Contribute to the development of an R package for these purposes.
2. Develop methodology for estimation of standard errors, and the calculation of confidence and prediction intervals for monotone polynomials;
3. Provide a comparison to previously developed methodology for explaining the monotone relationship between a response and a predictor;
4. Further develop work on model selection in general, with a focus on providing graphical techniques to aid in model selection, and further develop the model selection methods for monotone polynomials;

5. Provide some real world applications and comparisons with monotone smoothing splines when the aim of the research is to develop methods for detecting inflection points, or points arising from higher order derivatives.

To this end, in this thesis, we have achieved all of this, and a simple example that demonstrates the achievements of this thesis can be made with example data taken from the Berkley Guidance Study (Tuddenham & Snyder, 1954), a famous and well used data set, which examined the growth of boys and girls from the age of one through to adulthood. We show one such individual girl's growth curve data in the top left panel of Figure 1.1, and show three different models fitted to this data in the top right panel. These curves demonstrate the extent to which our monotone polynomial methodology (green line) provides a similar fit to the smoothing spline approach proposed by Ramsay (1998) and Ramsay & Silverman (2002, 2006) (black line), both of which are only marginally different from using the Constrained Generalized Additive Model (CGAM) framework described in Meyer (2008, 2012) (blue line). We note that all three techniques provide a very similar fitting curve in this instance, highlighted by their overlapping nature, but the monotone polynomial approach is the only methodology that has the added simplicity of only one tuning parameter (the degree of polynomial). This suggests monotone polynomials are potentially beneficial in constrained regression situations. Furthermore, with the methodology we developed, we see in the bottom left panel of Figure 1.1, both confidence bands and prediction bands for the fitted monotone polynomial curve. These intervals are based on bootstrap methodology developed in this thesis and described in Chapter 3. Finally, in the bottom right panel we show the bootstrap distribution for the location of inflection points in our growth data, noting several peaks in the distribution, denoting the location of the inflection points for this individual. The latter problem, is made particularly easy using monotone polynomials due to their known parametric from and simplicity of their higher order derivatives.

From this focus our thesis is outlined as follows: We will initially investigate new methodologies for fitting monotone polynomials to data with the aim of making the fits more robust and more efficient by decreasing the run-time. In Chapter 2 we con-

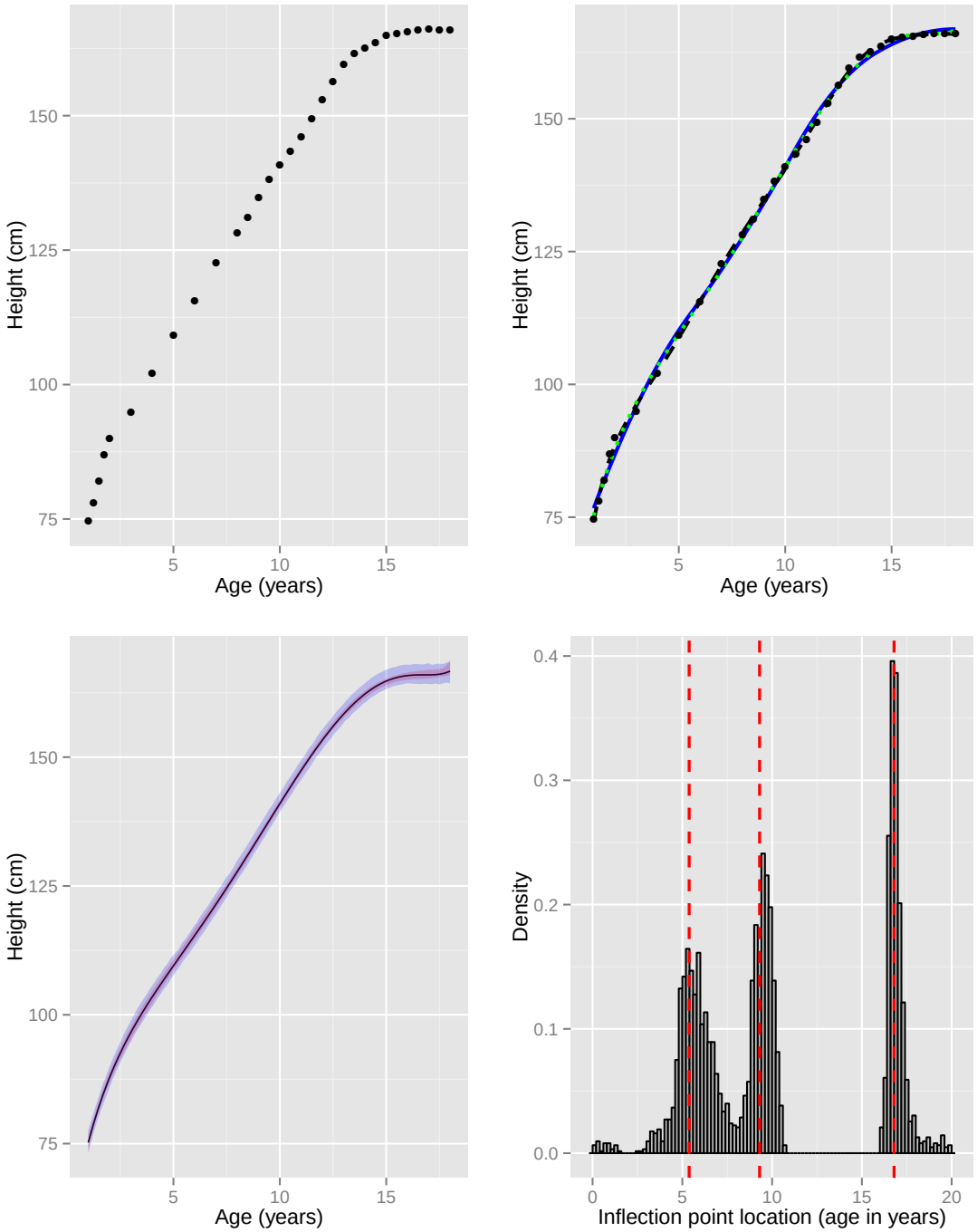

Figure 1.1: Data from Berkley Guidance Study. Top left panel: actual growth data of one individual; top right panel: different constrained regression fits to data with monotone polynomial fit (red line), CGAM fit (blue line) and monotone smoothing spline fit (black line); bottom left panel: confidence and prediction intervals using monotone polynomials; bottom right panel: histogram of location of inflection points based on bootstrap distribution.

sider the existing methodology for fitting monotone polynomials to data through the methods described by Elphinstone (1983), Hawkins (1994), and Heinzmann (2008). We develop new methodologies, expanding on existing work, by introducing the use of the Levenberg-Marquardt algorithm for the non-linear optimisation. We describe problems associated with the use of the previous methodologies, in particular for the situation where the number of design points is sparse and the aim is to estimate the location of features of fitted curves, for example inflection points. Furthermore, we introduce a novel algorithm for fitting monotone polynomials to data, which makes use of a sum of squared polynomials approach. We provide some extensive simulations to compare all of our newly proposed methods, making recommendations on which conditions would suit each method.

In Chapter 3 we consider consistency of monotone polynomial estimators and provide some numerical experiments to complement our theoretical argument. In addition we look at the variability associated with monotone polynomials, and provide Monte-Carlo simulations to demonstrate the difference between these and those obtained using a standard unconstrained least squares approach. We outline the necessity to determine different methods for estimating confidence bands for monotone polynomial fits and describe bootstrap methodology to obtain such estimates. We propose adaptations of the standard bootstrap methodology for monotone polynomials and describe through simulations the effectiveness of such adaptations. Finally, we consider prediction intervals and demonstrate that the problem associated with confidence intervals is not as pronounced when calculating prediction intervals, indicating that such intervals could be derived using existing bootstrap techniques.

In Chapter 4 we take a look at the new methodology developed in Chapters 2 and 3, and provide a comparison of these approaches to the existing methodology in Hawkins (1994), along with some comparisons to two of the more commonly used monotone smoothing spline techniques described in Ramsay (1998),Ramsay & Silverman (2002, 2006) and Meyer (2008, 2012). Comparisons are quantified in terms of both bias and variance and some run time comparisons are provided.

Chapter 5 presents material on the model selection problem for monotone polynomials, along with the more general model selection problem for any generalised linear

models, the latter of which was a parallel project to the monotone polynomial work. We make extensive use of the bootstrap methodology discussed in Chapter 3 and consider several different approaches including an information criteria approach to general model selection, with the development of model selection plots to aid in the model selection process. We examine the effectiveness of these with the more simple model selection problem for monotone polynomials, that is determining the most effective degree of monotone polynomial. We demonstrate these methodologies with several real world problems from Medicine, Forensic science and Neurophysiology.

Finally, in Chapter 6, we address our initial problem that motivated the further development of methods for fitting monotone polynomials to data, of detecting inflection points in monotone increasing or monotone decreasing functions. We revisit the model selection problem discussed in Chapter 5 and carry out extensive simulations, investigating the use of various methodologies. We give a comparison of two alternative approaches for detecting inflection points and show the effectiveness of each methodology. We finalise Chapter 6 by demonstrating these methodologies with the Neurophysiology data and the growth curve data described earlier in this chapter.

At the start of each chapter we will include a brief introduction which goes beyond the general introduction in this chapter, and at the end of each chapter we will provide some general discussion and conclusions. Throughout the thesis our aim is to keep an emphasis on visualisation of our techniques, and results generated using them, and as a consequence we provide numerous visual aids to assist in the understanding and the interpretation of our results.

2

FITTING MONOTONE POLYNOMIALS TO DATA

SUMMARY

In this chapter we revisit Hawkins' (1994) algorithm for fitting monotone polynomials and discuss two major practical issues that we encountered using this algorithm, namely when fitting high degree polynomials and situations with a sparse x design but multiple observations per x-value. As an alternative, we describe two new approaches to fitting monotone polynomials to data.

The first approach is based on the different characterisations of monotone polynomials, and extensions of these, proposed by Elphinstone (1983) and by using a Levenberg–Marquardt type algorithm. We consider the effectiveness of these different parameterisations, examine effective starting values for the non-linear algorithms, and discuss some limitations.

The second method provides an approach for fitting monotone polynomials to data which markedly improves on the algorithms of both Hawkins (1994) and those created based on the Elphinstone parameterisations. We use a sum of squared polynomials parameterisation to achieve this which in turn allows us to impose monotonicity constraints to be over either a compact interval or a semi-compact interval of the form $[a, \infty)$, which is in contrast to these previous approaches that imposed monotonicity over the whole real line. This is, to our knowledge, the first time such algorithms have been made available and the ramifications and potential benefits of this are discussed.

All algorithms discussed in this chapter are available in the R (R Core Team, 2015) package `MonoPoly` (version 0.3-6 or later).

2.1 INTRODUCTION

Given the many useful properties of monotone polynomials, for example that they are naturally strictly monotone and have easily identifiable derivatives, we revisit the fitting of monotone polynomials to data. Our initial motivation was to improve upon the methodology used in Firmin, Müller & Rösler (2011, 2012) who examined monotone decreasing relationships in Neuroscience. They used a triple stimulation technique (TST) to study motor evoked potentials (MEP) in the brain. Specifically this technique is a collision technique, whereby the degree of MEP desynchrinization may be suppressed (Magistris, Rösler, Truffert *et al.*, 1998, 1999). It is performed by first delivering a magnetic stimulus followed by an electrical stimulation of the peripheral nerve of the wrist, after an appropriate *delay*. A third stimulus is delivered to Erb's point, eliciting a motor response that can be assessed. They fitted a monotone decreasing function into the measured motor evoked potentials (TST amplitude) as a function of stimulation delay on more than 40 different data sets, one from each participating patient in their study. Firmin, Müller & Rösler (2011, 2012) used smoothing splines via the smooth.monotone function which is part of the R-package `fda` (Ramsay, Wickham, Graves *et al.*, 2013) and extracted those inflection points (roots of the second derivative) from the fitted curve that relate to local maxima of the first derivative. These estimated modes are of relevance in Neuroscience. The purpose of this chapter is not to demonstrate that using monotone polynomials in this context outperforms "full-blown" nonparametric smoothing techniques, this will be investigated in Chapter 6, but to present improved algorithms for fitting monotone polynomials.

Recently, fitting monotone polynomials to data has been considered in a Bayesian framework by Curtis & Ghosh (2011), and previously in a frequentist setting by Hawkins (1994), whose algorithm we revisit in Section 2.3. While Hawkins' algorithm is fast and effective, we identify situations in which it cannot be used. For these situations, and for use in general, we consider several parameterisations of monotone polynomials, and show how the best fitting monotone polynomial, using any of these parameterisations, can be fitted to data using a Levenberg–Marquardt modified Newton–Raphson algorithm.

We continue this chapter by describing broad methodology for fitting monotone polynomials to data in Section 2.2 and describe our experience with Hawkins' (1994) algorithm in Section 2.3. Furthermore by closer investigation of the parameterisations and algorithms, based on the earlier formulations proposed by Elphinstone (1983) and a semi-definite programming algorithm by Hawkins (1994), we identify two situations in which improvements could be made on the existing methodology: (i) when q, the degree of the polynomial, is large and leads to problems with the QR factorisation of the design matrix, or (ii) when q is large relative to the number of unique points in the x design. In Section 2.4 the various parameterisations of monotone polynomials based on formulations described in Elphinstone (1983) and modifications of these parameterisations, along with a sum of squared polynomial approach is presented, together with a discussion on how to evaluate the objective functions and its derivatives. We show that one of the drawbacks of earlier methodologies proposed for fitting monotone polynomials to data, is the inability to constrain the polynomial to be monotone over compact or semi-compact intervals. To date algorithms for fitting monotone polynomials to data, when the function is only constrained to be monotone over a compact or semi-compact region, are to our knowledge not available, and we consider this by developing a methodology to fit such polynomials using a sums of squares parametrisation, which also allows fitting over the entire real line. We believe this is key to generalise the fitting of monotone polynomials to these regions, as well as to allow the fitting of constrained even degree polynomials, and we examine the impact of this sums of squares parametrisation and its effectiveness. For all methodologies presented, we describe optimising the objective function through the use of a Levenberg-Marquardt optimisation routine in Section 2.5 and describe methods for selecting effective starting values and stopping criteria. In Section 2.6 we describe our experiences of the different parameterisations and, in Section 2.7, results from our numerical experiments are illustrated. Finally, we provide some discussion and conclusions in Section 2.8.

2.2 BACKGROUND

The usual parameterisation for a polynomial regression function is

$$p(x) = p(x;\boldsymbol{\beta}) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_q x^q \tag{2.1}$$

where q is the degree of the polynomial and $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_q)^T$, with the β_js being the regression parameters in the linear regression model

$$Y = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_q x^q + \epsilon. \tag{2.2}$$

and ϵ is the error term. For a random sample from model (2.2) the errors $\epsilon_1, \ldots, \epsilon_n$ are for many applications assumed to be independent and identically distributed. Note that such a polynomial, $p(x)$, can be monotone over the whole real line, only if its degree $q = 2K + 1$ is odd, where K is some non-negative integer. To fit a polynomial regression curve to given data (x_i, y_i), $i = 1, \ldots, n$, one usually minimises the residual sum of squares (RSS), which under the assumption of Gaussian errors in (2.2) achieves the same as maximising the likelihood. That is, the fit is determined by minimising

$$\mathrm{RSS}(\boldsymbol{\beta}) = \sum_{i=1}^{n} (y_i - p(x_i))^2. \tag{2.3}$$

Note that this objective function is strictly convex in $\boldsymbol{\beta}$ if the number of distinct x-values exceeds q. Moreover, the set of monotone increasing polynomials, $\mathcal{I} = \{p(x) : \forall x, p'(x) \geq 0\}$, and the set of monotone decreasing polynomials, $\mathcal{D} = \{p(x) : \forall x, p'(x) \leq 0\}$, are both convex, closed sets. Thus, minimising (2.3) over either $\mathcal{I}$ or $\mathcal{D}$ is a convex optimisation problem, with a unique solution, which should be easy to solve. However, while it would be relatively easy to find starting values in $\mathcal{I}$ or $\mathcal{D}$, for example by fitting a simple linear regression and setting $\beta_j = 0$ for $j \geq 2$ or fitting the function $\beta_0 + \beta_q x^q$ to the data, finding descent directions is extremely difficult at best.

2.3 HAWKINS' APPROACH

Hawkins (1994) shows that the solution to minimising $\text{RSS}(\boldsymbol{\beta})$ subject to $p \in \mathcal{I}$ must be a polynomial with derivative $p'(x;\boldsymbol{\beta}) > 0$ for all but a finite number of points, say, x_m^* with $p'(x_m^*;\boldsymbol{\beta}) = p''(x_m^*;\boldsymbol{\beta}) = 0$, $m = 1,\dots M$ (with $M = 0$ a possibility), which he calls "hips" (short for "horizontal inflection points"). This insight shows that the optimal monotonic (increasing) polynomial is the saddle-point (that is, a stationary point in the range of the function which does not correspond to a local extrema) of the Lagrangian

$$\text{RSS}(\boldsymbol{\beta}) - \sum_{m=1}^{M} \lambda_m p'(x_m^*;\boldsymbol{\beta}).$$

Thus, if M and the location of the hips were known, then the optimal monotonic polynomial is given by the solution to the quadratic program

$$\text{minimise}_{\boldsymbol{\beta}}\, \text{RSS}(\boldsymbol{\beta}) \quad \text{subject to} \quad p'(x_m^*;\boldsymbol{\beta}) = 0, \quad m = 1,\dots,M. \tag{2.4}$$

This led Hawkins (1994) to propose the following algorithm:

1. Set $M = 0$ and solve the unconstrained problem $\text{minimise}_{\boldsymbol{\beta}}\, \text{RSS}(\boldsymbol{\beta})$.

2. Check whether the first derivative $p'(x,\boldsymbol{\beta})$ of the fitted polynomial is non-negative everywhere. If it is, then the fitted polynomial is monotonic already, and therefore solves the problem.

   However, if the resulting polynomial is not monotonic, then it will be necessary to find an additional candidate hip at which to force the polynomial's slope to be zero. To do this, increase M by 1, adding an additional candidate hip and forcing a zero derivative there. Solve the resulting equality-constrained problem.

   The additional constraint at the new candidate hip may make one of the existing constraints redundant, a fact that will be detected by its Lagrange multiplier λ_m becoming negative. If this occurs, remove that candidate hip from the active set. Repeat the algorithm from step 2.

To complete the description of his algorithm, Hawkins (1994) suggests (with slight modifications to adapt to the notation used here):

> The test for monotonicity can be made quickly and efficiently, finding the minimum of $p'(x)$ by solving $p''(x) = 0$ [...] which is easily done using standard packages for manipulating polynomials. The test for monotonicity is made by evaluating $p'(x)$ at each real root of $[p''(x)]$—by definition the x values at which $p'(x)$ takes on its most extreme values. If any of these derivatives is negative the polynomial is not monotonic. The converse is not true; $p'(x)$ may have positive extreme values but be negative for all sufficiently large $|x|$; for this reason while the primary test for monotonicity is based on the values of $p'(x)$ at the roots of $p''(x) = 0$, we supplement this test with a direct evaluation of $p'(x)$ at some relative extreme value of x.
>
> Where the current polynomial is not monotonic, the algorithm needs to select an additional candidate hip. A good choice is the x-value at which the largest negative slope occurs, a value which is the byproduct of the suggested method for checking monotonicity. This value is then used as the candidate additional hip in step 2.

and describes how (2.4) can be solved for a given set of hips.

In attempting to implement this algorithm, we frequently experienced two or more Lagrange multipliers becoming negative in step 2. It appears that one should still drop only one constraint during this step, which may require a judicious selection. More problematic were situations in which two, or more, of the Lagrange multipliers became close to zero; that is having an absolute value smaller than typically employed tolerance levels for numerical calculations. In such situations it is difficult to determine whether the corresponding constraints are inactive and should be dropped, and if so, in what order. However, not dropping such constraints leads to an increasing set of candidate hips and (2.4) being solved with an increasing number of equality constraints which may create numerical problems.

2.3.0.1 *Improving Hawkins' Approach*

To improve the algorithm provided in Hawkins (1994) we choose to implement step 2 by using the R package `quadprog` (Turlach & Weingessel, 2011) which implements the quadratic programming algorithm of Goldfarb & Idnani (1982, 1983). In doing so, we replaced the equality constraints in (2.4) by inequality constraints ($p'(x_m^*;\boldsymbol{\beta}) \geq 0$, $m = 1,\ldots,M$) and removed all candidate hips for which the Lagrangian parameter was non-positive at the solution. Finally, to cope with different levels of numerical precision for the polynomial root finder (function `polyroot()` in R), the evaluation of polynomials and the quadratic programming solver, we found it necessary to declare a polynomial to be monotone if $p'(x) > -\varepsilon$ at the roots of $p''(x)$, where ε can be chosen to be in the order of 10^{-7} to 10^{-12}, and $p'(x) > 0$ at some relative extreme value of x. This correlates with Hawkins' (1994) warning that "Care is needed [since] it is not always trivial to decide whether a polynomial is really non-monotonic, or just seems so because of roundoff noise". The choice of ε obviously influences the number of iterations of the algorithm, but we found our implementation of the algorithm to be fast and reliable for values of ε in the indicated range.

We note that for numerical stability a QR factorisation of the design matrix

$$D = \begin{pmatrix} 1 & x_1 & x_1^2 & x_1^3 & \ldots & x_1^q \\ 1 & x_2 & x_2^2 & x_2^3 & \ldots & x_2^q \\ 1 & x_3 & x_3^2 & x_3^3 & \ldots & x_3^q \\ \vdots & \vdots & \vdots & \vdots & \ldots & \vdots \\ 1 & x_n & x_n^2 & x_n^3 & \ldots & x_n^q \end{pmatrix} \tag{2.5}$$

should be used as recommended by Hawkins (1994), where n is the number of observations. However, it becomes increasingly difficult to calculate a QR factorisation of the design matrix as q increases, irrespective of the x-design. The reason is that the approach of Hawkins uses unorthogonalised polynomials which pose severe numerical challenges at higher degrees and the calculations may fail with default settings

of numerical routines. Consequently judicious choice of numerical tolerances becomes necessary.

While Hawkins (1994) surmises that "Polynomials of degree 1, 3, 5, 7 and 9 were fitted—[to test] the method when using a polynomial of degree far above what most people would normally consider using for data modeling", the application that motivated our research, namely the search for inflection points, may necessitate the use of polynomials of (much) higher degrees. With the methods described in the subsequent sections we are able to fit much higher degree monotone polynomials to data than can be achieved using Hawkins' approach.

Furthermore, in situations with a sparse design but multiple observations per x-value, such as our example data $W2$ below in Section 2.7, one might want to fit a monotone polynomial of order q in a situation where one has only q, or even slightly less, unique x-values. In such circumstances the algorithm of Hawkins (1994) will fail as it starts off at the unconstrained solution.

These two drawbacks, which limit the usefulness of Hawkins' approach for our application of interest, lead to the consideration of other approaches for fitting monotone polynomials which are based on isotonic parameterisations.

2.4 ISOTONIC PARAMETERISATIONS

We describe in this section two different approaches to parameterising monotone polynomials and the benefits and drawbacks of each.

2.4.1 *Elphinstone type*

As an alternative approach to overcome the problem that (2.3) cannot be easily minimised, over $\mathcal{I}$ or $\mathcal{D}$, when the parameterisation (2.1) is used, we consider here several parameterisations of isotonic polynomials. The first parameterisation is due to Elphinstone (1983) who realised that a polynomial of degree $q = 2K + 1$ is isotonic if, and

only if, all real roots of its derivative have an even multiplicity. This insight leads to the following parameterisation

$$p_1(x) = \delta + \alpha \int_0^x \prod_{j=1}^{K} \left\{ \left(c_j^2 + b_j^2 \right) + 2b_j t + t^2 \right\} dt \tag{2.6}$$

where α, δ, b_j and c_j, are arbitrary real values, $j = 1, \ldots, K$; the sign of α determines whether the polynomial is monotone increasing or monotone decreasing. In this parameterisation $\alpha/(2K+1)$ is the coefficient of the x^q term, and the roots of the derivative of $p_1(x)$ are given by $-b_j \pm c_j \iota$, where $\iota = \sqrt{-1}$.

Elphinstone (1983) also proposed the following parameterisation, subsequently mentioned by Hawkins (1994) and used by Heinzmann (2008),

$$p_2(x) = \delta + \alpha \int_0^x \prod_{j=1}^{K} \left\{ 1 + 2b_j t + \left(b_j^2 + c_j^2 \right) t^2 \right\} dt, \tag{2.7}$$

and motivated it by observing that "when stepping from $K-1$ to K the starting values [...] would be the solution which [minimise] the criterion for $K-1$, and the two new parameters [...] would be set to zero". We shall comment on this observation later in this section, but note for now that unlike parameterisation (2.6), this parameterisation cannot be used for all isotonic polynomials, for example $p(x) = x^q$ with $q = 3$ cannot be represented in the form (2.7). The reason being that the roots of the derivative of $p_2(x)$ are given by $-b_j/(b_j^2 + c_j^2) \pm c_j/(b_j^2 + c_j^2)\,\iota$. For example with $q = 3$, that is $K = 1$, then (2.7) becomes

$$p_2(x) = \delta + \alpha \left\{ x + \frac{b_1 x^2}{2} + \frac{(b_1^2 + c_1^2)x^3}{3} \right\},$$

which will always have a linear term included so long as α is non-zero, hence $p_2(x)$ will never be x^3. As in (2.6) the sign of α determines whether $p_2(x)$ lies in $\mathcal{I}$ or $\mathcal{D}$.

The final parameterisation that we consider, proposed by Penttila (2006), is a modification of (2.7) and, to the best of our knowledge, was published by Murray, Müller & Turlach (2013) for the first time:

$$p_3(x) = \delta + \alpha \int_0^x \prod_{j=1}^{K} \left\{ b_j^2 + 2b_j t + \left(1 + c_j^2\right) t^2 \right\} dt. \tag{2.8}$$

The roots of the derivative of $p_3(x)$ are given by $-b_j/(1+c_j^2) \pm b_j c_j/(1+c_j^2)\,\imath$, which shows that this parameterisation can also not represent all isotonic polynomials; the derivative of $p_3(x)$ cannot have roots of the form $0 \pm e\,\imath$ with $e \neq 0$. Again the sign of α determines whether $p_3(x)$ is monotonic increasing or decreasing. It should be noted that with parameterisations (2.6) and (2.8) the fitted polynomial will either be constant or of degree q, while for (2.7) and for Hawkins' (1994) method, described in Section 2.3, the fitted polynomial can potentially have degree less than q.

For all three parameterisations we denote the vector of parameters generically by $\boldsymbol{\theta} = (\delta, \alpha, b_1, c_1, \ldots, b_K, c_K)^T = \left(\delta, \alpha, \tilde{\boldsymbol{\theta}}^T\right)^T$. Note that all these parameterisations have the form

$$p(x) = \delta + \alpha \tilde{p}(x) = \delta + \alpha \tilde{p}(x; \tilde{\boldsymbol{\theta}}), \tag{2.9}$$

where $\tilde{p}(x)$ depends only on the b_js and c_js. Finally, for the remainder of this chapter, it will be clear from the context whether we regard the residual sum of squares as a function of $\boldsymbol{\beta}$ or as a function of $\boldsymbol{\theta}$, and we avoid the subscript of p.

2.4.1.1 *Evaluating the objective function and its derivatives*

To evaluate the RSS using any of the parameterisations described previously in this section, we first calculate the $\boldsymbol{\beta}$ which corresponds to the given $\boldsymbol{\theta}$. This allows an easy evaluation of $p(x)$ for arbitrary values of x, using for example the Horner scheme for numerical stability (Fausett, 2003), and to use (2.3) for the calculation of the RSS.

We illustrate the necessary calculations using parameterisation (2.6), the other two parameterisations can be handled analogously, and are shown in Appendix A. To calculate $\boldsymbol{\beta}$ for a given $\boldsymbol{\theta}$, we first build triples $(c_j^2 + b_j^2, 2b_j, 1)$, which are the coefficients

for the quadratic functions appearing in (2.6). By convoluting these triplets, we can calculate the coefficients $\gamma = (\gamma_0, \ldots, \gamma_{2K})^T$ of the polynomial

$$\gamma_0 + \gamma_1 t + \cdots + \gamma_{2K} t^{2K} = \prod_{j=1}^{K} \left\{ \left(c_j^2 + b_j^2 \right) + 2 b_j t + t^2 \right\}.$$

From γ we can readily calculate $\boldsymbol{\beta}$ as

$$\boldsymbol{\beta} = \left(\delta, \alpha\gamma_0, \alpha \frac{\gamma_1}{2}, \ldots, \alpha \frac{\gamma_{2K}}{2K+1} \right)^T.$$

Note that $\left(0, \gamma_0, \frac{\gamma_1}{2}, \ldots, \frac{\gamma_{2K}}{2K+1}\right)^T$ is the vector that contains the coefficient of the polynomial $\tilde{p}(x)$ in (2.9).

To minimise RSS numerically we also need first and second derivatives for a derivative based optimisation algorithm. These derivatives can easily be calculated in a similar manner. In general using (2.3) we see

$$\frac{\partial}{\partial \boldsymbol{\theta}} \text{RSS} = -2 \sum_{i=1}^{n} (y - p(x_i)) \frac{\partial}{\partial \boldsymbol{\theta}} p(x_i) \tag{2.10}$$

and from (2.9) we have

$$\frac{\partial}{\partial \delta} \text{RSS} = -2 \sum_{i=1}^{n} (y - p(x_i)), \tag{2.11a}$$

$$\frac{\partial}{\partial \alpha} \text{RSS} = -2 \sum_{i=1}^{n} (y - p(x_i)) \tilde{p}(x_i), \tag{2.11b}$$

$$\frac{\partial}{\partial \tilde{\theta}_k} \text{RSS} = -2\alpha \sum_{i=1}^{n} (y - p(x_i)) \frac{\partial}{\partial \tilde{\theta}_k} \tilde{p}(x_i), \tag{2.11c}$$

where $\tilde{\theta}_k$ is a component of the vector $\tilde{\boldsymbol{\theta}}$, that is one of the b_js or c_js. Previously we have discussed how $\tilde{p}(x_i)$ can be evaluated once γ is determined. The partial derivatives $\frac{\partial}{\partial \tilde{\theta}_k} \tilde{p}(x_i)$ can be evaluated similarly. For example, if $\tilde{\theta}_k$ is b_{j_0}, then we build the triples $(c_j^2 + b_j^2, 2b_j, 1)$ for $j \neq j_0$ and the triple $(2b_{j_0}, 2, 0)$. After convoluting these K triples, the coefficients of the polynomial $\frac{\partial}{\partial \tilde{\theta}_k} \tilde{p}(\cdot)$ can be readily calculated. Similarly, if $\tilde{\theta}_k$ is c_{j_0}, then we build the triples $(c_j^2 + b_j^2, 2b_j, 1)$ for $j \neq j_0$ and the triple $(2c_{j_0}, 0, 0)$. After convoluting these K triples, the coefficients of the polynomial $\frac{\partial}{\partial \tilde{\theta}_k} \tilde{p}(\cdot)$ can be readily calculated.

Using (2.10) it follows that the Hessian is

$$\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}\mathrm{RSS}(\boldsymbol{\theta}) = 2\sum_{i=1}^{n}\left(\frac{\partial}{\partial\boldsymbol{\theta}}p(x_i)\right)\left(\frac{\partial}{\partial\boldsymbol{\theta}}p(x_i)\right)^T - 2\sum_{i=1}^{n}(y-p(x_i))\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}p(x_i) \quad (2.12)$$

and combining with (2.11) allows us to find formulae for the second partial derivatives of RSS, that is of $\frac{\partial^2}{\partial\theta_k\,\partial\theta_l}\mathrm{RSS}$. Some of these second derivatives involve second derivatives of the polynomial $\tilde{p}(x)$. Again, the coefficients of $\frac{\partial^2}{\partial\theta_k\,\partial\theta_l}\tilde{p}(x;\tilde{\boldsymbol{\theta}})$ can be determined by convoluting appropriately constructed triples. Though if either θ_l or θ_k is either α or δ, then, trivially, $\frac{\partial^2}{\partial\theta_k\,\partial\theta_l}\tilde{p}(x;\tilde{\boldsymbol{\theta}}) \equiv 0$. Also note that $\frac{\partial^2}{\partial b_j\partial c_j}\tilde{p}(x;\tilde{\boldsymbol{\theta}}) \equiv 0$, $j = 1,\ldots,K$.

These second partial derivatives are shown here for completeness. First, we note the obvious relationship

$$\frac{\partial^2}{\partial\theta_k\,\partial\theta_l}\mathrm{RSS} \equiv \frac{\partial^2}{\partial\theta_l\,\partial\theta_k}\mathrm{RSS}.$$

From (2.11a) we have

$$\frac{\partial^2}{\partial\delta^2}\mathrm{RSS} \equiv 2n, \quad (2.13a)$$

$$\frac{\partial^2}{\partial\delta\partial\alpha}\mathrm{RSS} = 2\sum_{i=1}^{n}\tilde{p}(x_i), \quad (2.13b)$$

$$\frac{\partial^2}{\partial\delta\partial\tilde{\theta}_k}\mathrm{RSS} = 2\alpha\sum_{i=1}^{n}\frac{\partial}{\partial\tilde{\theta}_k}\tilde{p}(x_i). \quad (2.13c)$$

From (2.11b) we have

$$\frac{\partial^2}{\partial\alpha^2}\mathrm{RSS} = 2\sum_{i=1}^{n}\tilde{p}(x_i)^2, \quad (2.14a)$$

$$\frac{\partial^2}{\partial\alpha\partial\tilde{\theta}_k}\mathrm{RSS} = -2\sum_{i=1}^{n}\left\{(y_i-p(x_i)) - \alpha\tilde{p}(x_i)\right\}\frac{\partial}{\partial\tilde{\theta}_k}\tilde{p}(x_i). \quad (2.14b)$$

From (2.11c) we have

$$\frac{\partial^2}{\partial\tilde{\theta}_k{}^2}\mathrm{RSS} = -2\alpha\sum_{i=1}^{n}\left\{-\alpha\left(\frac{\partial}{\partial\tilde{\theta}_k}\tilde{p}(x_i)\right)^2 + (y_i-p(x_i))\frac{\partial^2}{\partial\tilde{\theta}_k{}^2}\tilde{p}(x_i)\right\}, \quad (2.15a)$$

$$\frac{\partial^2}{\partial\tilde{\theta}_k\partial\tilde{\theta}_j}\mathrm{RSS} = -2\alpha\sum_{i=1}^{n}\left\{-\alpha\frac{\partial}{\partial\tilde{\theta}_j}\tilde{p}(x_i)\frac{\partial}{\partial\tilde{\theta}_k}\tilde{p}(x_i) + (y_i-p(x_i))\frac{\partial^2}{\partial\tilde{\theta}_k\partial\tilde{\theta}_j}\tilde{p}(x_i)\right\}, \text{where } j\neq k. \quad (2.15b)$$

From these equations we see immediately some potential problems associate with using a derivative based optimisation algorithm and make some comments in the next section.

2.4.1.2 *Some remarks on these isotonic parameterisations*

Earlier we noted that the RSS is strictly convex in $\boldsymbol{\beta}$ if the regressor variable has sufficiently many distinct values. However, RSS as a function of $\boldsymbol{\theta}$ is no longer convex. It is easy to see that RSS is a quartic polynomial in the b_js and c_js and this leads to potential non-convexity and the possibility of local extrema in $\mathrm{RSS}(\boldsymbol{\theta})$.

In fact, note that for each of the parameterisations considered in Section 2.4 the polynomial $\tilde{p}(x;\tilde{\boldsymbol{\theta}})$ depends on the c_js only through c_j^2. Generalising an observation by Heinzmann (2008), we could replace in all the parameterisations considered here c_j^2 by $\tau(c_j)$, where $\tau(z)$ is a function with $\tau(0) = 0$ and whose range is the non-negative reals. If additionally $\tau(z)$ is differentiable then $\tau'(0) = 0$ since zero is a global minimum of $\tau(z)$. With this generalisation, (2.11c) implies

$$\frac{\partial}{\partial c_j}\mathrm{RSS} = -2\alpha \sum_{i=1}^{n} (y - p(x_i)) \left(\frac{\partial}{\partial c_j} \tilde{p}(x_i) \right) \tau'(c_j). \tag{2.16}$$

Moreover, if θ_k is a component of $\boldsymbol{\theta}$ different from c_j then

$$\frac{\partial^2}{\partial c_j\, \partial \theta_k}\mathrm{RSS} = -2\alpha \sum_{i=1}^{n} \left\{ (y - p(x_i)) \left(\frac{\partial^2}{\partial c_j\, \partial \theta_k} \tilde{p}(x_i) \right) - \left(\frac{\partial}{\partial \theta_k} p(x_i) \right) \left(\frac{\partial}{\partial c_j} \tilde{p}(x_i) \right) \right\} \tau'(c_j). \tag{2.17}$$

These results have undesirable consequences for derivative based optimisation algorithms such as the Newton–Raphson algorithm and its variants. If during the iterative optimisation a c_j becomes zero, (2.16) implies that the corresponding entry in the gradient of RSS will be zero. More seriously, (2.17) implies that in the Hessian matrix all elements in the row and column corresponding to c_j equal zero, with the possible exception of the diagonal element. This, in turn, implies that the step direction calculated from this Hessian matrix for the next iteration will have a zero in the component corresponding to c_j. In other words, if during the iterations a c_j becomes zero, it will remain so.

Similar results hold for smooth loss functions other than the RSS. Thus, we discourage following the proposition in Elphinstone (1983), that is to change from (2.6) to (2.7) so that when stepping from $K-1$ to K the values which minimise the criterion for $K-1$ can be used as starting values with the two new parameters set to zero. One of these new parameters would be c_K. Setting c_K to zero when a derivative based optimisation routine is used ensures that c_K remains at zero. Admittedly, Elphinstone (1983) does not discuss the optimisation algorithm that he uses. By way of contrast, Heinzmann (2008) appears to use a Levenberg–Marquardt modification of the Newton–Raphson algorithm, that is a second-derivative based algorithm, but does not discuss the starting values that he uses. We note that neither of these authors discussed the problem with local extrema in the parameterisation of isotonic polynomials and that c_js can get trapped at zero.

In this chapter we also propose to use a Levenberg–Marquardt modification of the Newton–Raphson algorithm, described in Section 2.5. We demonstrated though our numerical experiments that all three parameterisations of isotonic polynomials indeed suffer under the problem of local extrema when one, or more, of the c_js become fixed at zero. Thus, we also investigate replacing c_j^2 in each of the parameterisations by c_j and optimising the RSS under the constraints $c_j \geq 0$, $j = 1, \ldots, K$.

2.4.2 *Sum of squared polynomials*

A major limitation of all previously consider parameterisations is that they can only fit monotone polynomials over the whole real line, that is $\mathcal{R} = (-\infty, \infty)$. In this section we describe a methodology that allows the fitting of monotone polynomials over a compact or semi-compact interval. We start by working with the usual parametrisation for a polynomial regression function as described in (2.1) and (2.2). We again consider fitting a polynomial regression curve to given data (x_i, y_i), $i = 1, \ldots, n$, using least squares with the residual sum of squares (RSS) being minimised which, under the assumption of Gaussian errors in (2.2), achieves the same as maximising the likelihood.

We reiterate the point that this objective function is strictly convex in $\boldsymbol{\beta}$ if the number of distinct x-values exceeds q. However, as noted in Section 2.2, this parametrisation

is not convenient to use if there are monotonicity constraints on the polynomial over a set $\mathcal{R} \subseteq \mathbb{R}$. To date we have only considered the case $\mathcal{R} = (-\infty, \infty)$ and revisited two approaches for fitting monotone polynomials, namely the semi-indefinite programming approach by Hawkins (1994) and various isotonic parametrisation based on work by Elphinstone (1983). In this section we now propose to use another isotonic parametrisation which will allow us to specify, in addition to $(-\infty, \infty)$, more general regions, namely semi-compact intervals $[a, \infty)$ and compact intervals $[a, b]$ for finite $a, b \in \mathbb{R}$, on which the fitted polynomial satisfies a monotonicity constraint.

2.4.2.1 *Isotonic parameterisation – sum of squared polynomials*

As in Section 2.2 we require an alternative approach to overcome the problem that (2.3) cannot be easily minimised under monotonicity constraints when the parametrisation (2.1) is used. We again consider here parametrisation of isotonic polynomials of the form

$$p(x) = \delta + \alpha \int_0^x \check{p}(u)\, du, \tag{2.18}$$

where $\check{p}(u)$ is required to be non-negative on $\mathcal{R}$. However, in this instance we wish to ensure that fitting monotone polynomials over a compact or semi-compact interval is achievable. We note again that from Equation (2.18) it follows immediately that the first derivative of the polynomial $p'(x) = \alpha\check{p}(x)$, ensuring that $p(x)$ is monotone increasing or monotone decreasing on $\mathcal{R}$ depending on whether α is positive or negative, respectively.

The isotonic parameterisations considered in Section 2.2, based on work by Elphinstone (1983), write $\check{p}(x)$ as a product of quadratic polynomials where each of these quadratics has either conjugate complex roots or a real root of multiplicity two. Finding the parameter vector that minimises RSS is typically slow when using these parameterisations, although they allow the fitting of monotone polynomials in some situations in which the semi-indefinite programming approach fails. Moreover, without awkward case distinctions, the approaches in 2.2 cannot be readily extended to more general forms of $\mathcal{R}$.

In this section we propose another isotonic parameterisation for $\check{p}(u)$ which is based on the following proposition.

Proposition 2.4.1. *A polynomial $\check{p}(x)$ of degree q is non-negative;*

1. *On $\mathcal{R} = (-\infty, \infty)$ if and only if $q = 2K$ and it can be written as the sum of two squared polynomials*

$$\check{p}(x) = p_1(x)^2 + p_2(x)^2, \quad \forall x \in \mathbb{R}, \tag{2.19}$$

   where $p_1(x)$ and $p_2(x)$ are polynomials whose degrees are at most K.

2. *On $\mathcal{R} = [a, \infty)$ if and only if it can be written as*

$$\check{p}(x) = p_1(x)^2 + (x-a)p_2(x)^2, \quad \forall x \in \mathbb{R}, \tag{2.20}$$

   where, if $q = 2K$, $p_1(x)$ and $p_2(x)$ are polynomials whose degrees are at most K and $K-1$, respectively, and, if $q = 2K+1$, both degrees are at most K.

3. *On $\mathcal{R} = [a, b]$ if and only if it can be written as*

   a) if $q = 2K$:

$$\check{p}(x) = p_1(x)^2 + (x-a)(b-x)p_2(x)^2, \quad \forall x \in \mathbb{R}, \tag{2.21}$$

   where $p_1(x)$ and $p_2(x)$ are polynomials whose degrees are at most K and $K-1$, respectively.

   b) if $q = 2K+1$:

$$\check{p}(x) = (x-a)p_1(x)^2 + (b-x)p_2(x)^2, \quad \forall x \in \mathbb{R}, \tag{2.22}$$

   where $p_1(x)$ and $p_2(x)$ are polynomials with their degree at most K.

Proposition 2.4.1 can be proved using the theory of Tchebycheff systems (Karlin & Studden, 1966) or the theory of canonical moments (Dette & Studden, 1997). A proof that does not utilise such deep mathematical theories can be found in Brickman & Steinberg (1962) or Papp (2011).

We use Proposition 2.4.1, by fixing α in (2.18) to be either -1 or 1, depending on whether the polynomial should be monotone decreasing or monotone increasing respectively over $\mathcal{R}$. We denote the coefficients of $p_1(x)$ and $p_2(x)$ in (2.19)–(2.22) by $\boldsymbol{\beta}_1 = (\beta_{01}, \beta_{11}, \ldots, \beta_{q_1 1})^T$ and $\boldsymbol{\beta}_2 = (\beta_{02}, \beta_{12}, \ldots, \beta_{q_2 2})^T$, respectively, with $q_1, q_2 \in \{K-1, K\}$. This allows us to write the vector of parameters generically by $\boldsymbol{\theta} = \left(\delta, \boldsymbol{\beta}_1^T, \boldsymbol{\beta}_2^T\right)^T$. Again, it will be clear from the context whether we regard the residual sum of squares as a function of $\boldsymbol{\beta}$ or as a function of $\boldsymbol{\theta}$.

2.4.2.2 *Evaluating the objective function and its derivatives*

To evaluate the RSS we first calculate the $\boldsymbol{\beta}$ which corresponds to the given $\boldsymbol{\theta}$. This allows an easy evaluation of $p(x)$ for arbitrary values of x, using for example the Horner scheme for numerical stability (Fausett, 2003), and to use (2.3) for the calculation of the RSS.

We illustrate the necessary calculations for a polynomial that is monotone on $\mathcal{R} = (-\infty, \infty)$; other choices of $\mathcal{R}$ can be handled analogously. To calculate $\boldsymbol{\beta}$ for a given $\boldsymbol{\theta}$, we convolve $\boldsymbol{\beta}_1$ and $\boldsymbol{\beta}_2$ each with themselves to obtain the coefficients of the polynomials $p_1(x)^2$ and $p_2(x)^2$, respectively. Adding these two sets of coefficients, we obtain the coefficients $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{q-1})^T$ of the polynomial

$$\check{p}(t) = \gamma_0 + \gamma_1 t + \cdots + \gamma_{q-1} t^{q-1}.$$

From $\boldsymbol{\gamma}$ we can readily calculate $\boldsymbol{\beta}$ as

$$\boldsymbol{\beta} = \left(\delta, \alpha\gamma_0, \alpha\frac{\gamma_1}{2}, \ldots, \alpha\frac{\gamma_{q-1}}{q}\right)^T.$$

To minimise RSS numerically, using a derivative based optimisation algorithm, requires first and second derivatives. These derivatives can easily be calculated in a similar manner to how the objective function is evaluated. Using (2.18) we see clearly,

$\partial p(x_i)/\partial\delta \equiv 1$. For other components of $\boldsymbol{\theta}$, say β_{ji}, for $i \in \{1,2\}$ and some $j \in \{0,1,\ldots,q_i\}$, we find

$$\frac{\partial}{\partial\beta_{ji}}p(x) = \frac{\partial}{\partial\beta_{ji}}\alpha\int_0^x \check{p}(u)\,du = \alpha\int_0^x \frac{\partial}{\partial\beta_{ji}}\check{p}(u)\,du = \alpha\int_0^x 2p_i(u)u^j\,du \tag{2.23a}$$

$$= \alpha\int_0^x 2\left(\sum_{k=0}^{q_i}\beta_{ki}u^k\right)u^j\,du = \frac{2\alpha}{k+j+1}\left(\sum_{k=0}^{q_i}\beta_{ki}x^{k+j+1}\right), \tag{2.23b}$$

The two equations in (2.23) show that the coefficients of the polynomials appearing in (2.10) are easily determined.

Furthermore, from (2.10) it follows that the Hessian matrix of RSS is (2.12) as defined previously. And from (2.23) it follows that

$$\frac{\partial^2}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}^T}p(x) = \begin{pmatrix} 0 & \mathbf{0}^T & \mathbf{0}^T \\ \mathbf{0} & \mathbf{H} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{H} \end{pmatrix} \tag{2.24}$$

where $\mathbf{H}$ is a $(q_i+1)\times(q_i+1)$ matrix with (k,j)th entry being $\frac{2\alpha}{k+j+1}x^{k+j+1}$, with $k,j = 0,\ldots,q_i$.

2.5 OPTIMISING THE OBJECTIVE FUNCTION

We considered various approaches for minimising RSS($\boldsymbol{\theta}$). Initial experiments using derivative free methods (Nelder & Mead, 1965; Powell, 2009; Bates, Mullen, Nash *et al.*, 2011) showed that such algorithms struggle with minimising RSS($\boldsymbol{\theta}$). Coordinate descent algorithms that were recently applied quite successfully to other computational statistics problems (see, among others, Friedman, Hastie, Höfling *et al.*, 2007; Friedman, Hastie & Tibshirani, 2010) proved to be somewhat more successful but required a large number of iterations, and long run-times, to achieve convergence to the optimiser of RSS($\boldsymbol{\theta}$). Here, we describe a Levenberg–Marquardt modification to the Newton–Raphson algorithm, which differs from the one described by Heinzmann (2008) and is more in the spirit of Osborne (1976), that proved to be effective for minimising RSS($\boldsymbol{\theta}$).

2.5.1 *Levenberg–Marquardt routine*

In the following description $\nabla \text{RSS}(\boldsymbol{\theta})$ denotes the gradient vector of RSS and $\text{H}(\boldsymbol{\theta})$ the Hessian matrix, calculated as outlined in Sections 2.4.1 and 2.4.2. The algorithm that we propose to use for optimising $\text{RSS}(\boldsymbol{\theta})$ is the following:

1: Set $\lambda = 0.1$, $t = 0$ and initialise $\boldsymbol{\theta}^{(t)}$
2: **repeat**
3: Calculate $\text{RSS}_t = \text{RSS}\left(\boldsymbol{\theta}^{(t)}\right)$, $\boldsymbol{g}_t = \nabla \text{RSS}\left(\boldsymbol{\theta}^{(t)}\right)$ and $\text{H}_t = \text{H}\left(\boldsymbol{\theta}^{(t)}\right)$
4: Set H_D to be a diagonal matrix with i^{th} diagonal entry being h_{ii}, if $h_{ii} > 0$, and 1 otherwise; where h_{ii} is the i^{th} diagonal entry of H_t
5: Set $l = 1$
6: **loop**
7: Calculate $\boldsymbol{\theta}^c = \boldsymbol{\theta}^{(t)} - (\text{H}_t + \lambda \text{H}_D)^{-1} \boldsymbol{g}_t$
8: Calculate $\text{RSS}_c = \text{RSS}\left(\boldsymbol{\theta}^c\right)$
9: **if** $\text{RSS}_c \leq \text{RSS}_t$ **then**
10: **if** $l = 1$ **then**
11: Set $\lambda = \lambda / 10$
12: **end if**
13: Set $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^c$
14: Set $t = t + 1$ and exit from loop
15: **else**
16: Set $\lambda = 10 * \lambda$ and $l = l + 1$
17: **if** $\lambda > 10^{10}$ **then**
18: Terminate algorithm with failure to converge
19: **end if**
20: **end if**
21: **end loop**
22: **until** convergence criteria are met

Note that H_D in step 4 is always positive definite and will typically contain information on the curvature of RSS with respect to each component of $\boldsymbol{\theta}$. Thus, for large λ, the

algorithm essentially takes steps according to a steepest descent algorithm where the components of the gradient vector are rescaled by the curvature. Such steps are typically taken when the current value of $\boldsymbol{\theta}$ is far away from the optimal value and, hence, $\mathrm{H}(\boldsymbol{\theta})$ may be neither positive definite nor useful for determining a descent direction. The typical scenario, as the iterates approach a (local) minimum, is that each proposal $\boldsymbol{\theta}^c$ is immediately accepted ($l = 1$), λ becomes very small, and the algorithm turns into a Newton–Raphson algorithm.

In the variation of the parameterisations that use c_j instead of c_j^2, the algorithm is much the same. Except in step 7 the proposal is calculated as $\boldsymbol{\theta}^c = \boldsymbol{\theta}^{(t)} + \boldsymbol{\delta}^{(t)}$ where $\boldsymbol{\delta}^{(t)}$ is determined by solving the following quadratic program:

$$\text{minimise}_{\boldsymbol{\delta}} \frac{1}{2}\boldsymbol{\delta}^T(\mathrm{H}_t + \lambda \mathrm{H}_D)\boldsymbol{\delta} + \boldsymbol{g}_t^T\boldsymbol{\delta}$$

where the imposed constraints are such that all entries in $\boldsymbol{\theta}^c$ that correspond to any of the c_js will be non-negative. Figure 2.1 describes pictorially this algorithm.

To complete the description of the algorithm we discuss in the following sections how $\boldsymbol{\theta}^{(0)}$ is initialised and the stopping criteria.

2.5.2 *Starting values*

In order to have efficient and effective optimisation routines it is extremely important to have a sensible approach to produce starting values for the parameter estimates in the optimisation routine. In light of this we describe extensively the methods used to produce a general set of rules to derive a reasonable set of starting values for our optimisation routines.

2.5.2.1 *Elphinstone*

With parameterisations (2.6), (2.7) and (2.8) our experience has suggested that for numerical stability a rescaling of the x- and y-values is beneficial. For these parameterisations we advocate the rescaling of both x and y so that their minimum and maximum values are -1 and 1 respectively.

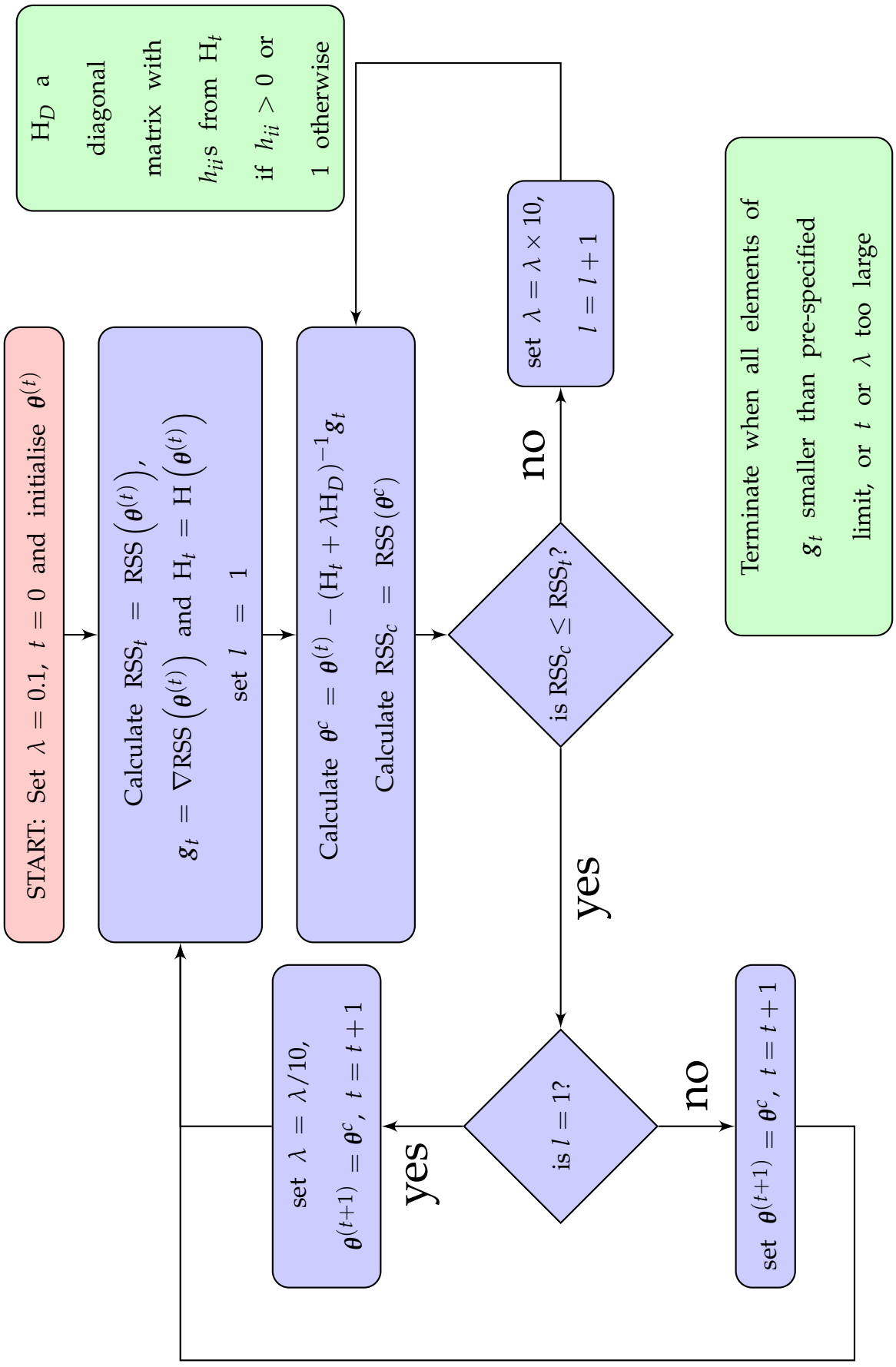

Figure 2.1: Description of the Levenberg-Marquardt algorithm

During our numerical experiments, discussed in more detail in Section 2.7, the following procedure seemed to yield a reliable and satisfactory initialisation of $\boldsymbol{\theta}^{(0)}$. For any parameterisation, we begin by setting the b_js to zero. Next, for (2.6) and (2.7) the c_js are initialised to $c_j = 0.1 + (j-1)\frac{0.9}{K-1}$, $j = 1, \ldots, K$, and for (2.8) to $c_j = 1 + (j-1)\frac{1}{K-1}$, $j = 1, \ldots, K$. Finally, based on (2.9), α and δ are determined by regressing the responses y_i on the predictor $\tilde{p}(x_i)$, which completes the initialisation of $\boldsymbol{\theta}$.

2.5.2.2 *Sum of squared polynomials*

As with the Elphinstone parameterisations we rescale the x and y values. In this instance this not only aids numerical stability but also assists in providing a more general methodology to determine starting values when looking at fitting monotone polynomials over a (semi-)compact interval. Specifically for the three variations of $\mathcal{R}$, that is Case 1: $\mathcal{R} = (-\infty, \infty)$, Case 2: $\mathcal{R} = [a, \infty)$, Case 3: $\mathcal{R} = [a, b]$, we have:

CASE 1 For fitting monotone polynomials over the whole real line, that is $\mathcal{R} = (-\infty, \infty)$, we rescale x and y values as per the Elphinstone parameterisations.

CASE 2 For fitting monotone polynomials over the semi-compact interval, that is $\mathcal{R} = [a, \infty)$, we rescale such that $a \to 0$ and $\max x \to 1$. We rescale y as per Elphinstone parameterisations.

CASE 3 For fitting monotone polynomials over the compact interval, that is $\mathcal{R} = [a, b]$, we rescale such that $a \to 0$ and $b \to 1$. We rescale y as per Elphinstone parameterisations.

These rescalings, based on the definition of $\mathcal{R}$, enable easier working with the formulations to determine effective starting values.

To start the optimisation routine, we fit a polynomial using a constant, linear and cubic term to those points (x_i, y_i) for which x_i falls into $\mathcal{R}$ if the monotone polynomial to be fitted is of odd degree, otherwise we use an initial polynomial fit up to and including the quadratic term.

From this initial fit starting values are determined. If the initial fit is monotone, one can calculate its representation using the appropriate isotonic parameterisation (2.19)–

(2.22). Otherwise, it is possible to determine the isotonic parameterisation of a polynomial whose coefficients equals the coefficients of the initial fit in absolute value. Finally, α is automatically chosen as either -1 or 1 based on the sign of the correlation of those points (x_i, y_i) for which x_i falls into $\mathcal{R}$.

We provide details of these calculations below.

2.5.2.3 *Starting values – Case 1*

If we are fitting monotone polynomials over $\mathcal{R} = (-\infty, \infty)$, using (2.19), then only odd degree polynomials can be fit. We take the following initialisation steps:

1: Fit unconstrained polynomial of the form $y^* = A + Bx^* + Cx^{*3}$ where (x^*, y^*) are the rescaled (x, y) pairs.
2: Initialise $\delta = \hat{A}$
3: Initialise $\beta_{01} = \sqrt{|\hat{B}|}$
4: Initialise $\beta_{12} = \sqrt{3|\hat{C}|}$
5: Initialise all other $\beta_{ij} = 0$.

Putting these initial starting values with the rescaling into our monotone polynomial in (2.18) and using (2.19) gives:

$$\begin{aligned} p(x^*) &= \delta + \alpha \int_0^{x^*} \check{p}(u)\, du \\ &= \hat{A} + \alpha \int_0^{x^*} p_1(u)^2 + p_2(u)^2\, du \\ &= \hat{A} + \alpha|\hat{B}|x + \alpha|\hat{C}|x^{*3} \end{aligned}$$

Once α is determined we see this is then the fitted unconstrained polynomial described in the algorithm above (up to the sign of $|\hat{B}|$).

2.5.2.4 *Starting values – Case 2*

If we are fitting monotone polynomials over $\mathcal{R} = [a, \infty)$ as described in (2.20) then our starting values depend on whether the degree of the polynomial is odd or even. If the degree is even and hence the derivative is odd we take the following initialisation steps:

1: Fit unconstrained polynomial of the form $y^* = A + Bx^* + Cx^{*2}$ where (x^*, y^*) are the rescaled (x, y) pairs.
2: Initialise $\delta = \hat{A}$
3: Initialise $\beta_{01} = \sqrt{|\hat{B}|}$
4: Initialise $\beta_{02} = \sqrt{2|\hat{C}|}$
5: Initialise all other $\beta_{ij} = 0$.

Putting these initial starting values with the rescaling into our monotone polynomial in (2.18) using (2.20) gives

$$\begin{aligned} p(x^*) &= \hat{A} + \alpha \int_0^{x^*} p_1(u)^2 + u p_2(u)^2 \, du \\ &= \hat{A} + \alpha|\hat{B}|x^* + \alpha|\hat{C}|x^{*2} \end{aligned}$$

Again, once α is determined we see this is then the fitted unconstrained polynomial described in the algorithm above (up to the sign of $|\hat{B}|$).

If the degree is odd and hence the derivative even, still considering monotone polynomials over $\mathcal{R} = [a, \infty)$ as described in (2.20), the initialisation becomes:

1: Fit unconstrained polynomial of the form $y^* = A + Bx^* + Cx^{*3}$ where (x^*, y^*) are the rescaled (x, y) pairs.
2: Initialise $\delta = \hat{A}$
3: Initialise $\beta_{01} = \sqrt{|\hat{B}|}$
4: Initialise $\beta_{11} = -\sqrt{3|\hat{C}|}$
5: Initialise $\beta_{02} = -\sqrt{2|\beta_{01}\beta_{11}|}$
6: Initialise all other $\beta_{ij} = 0$.

We note that $\beta_{01} \geq 0$ and $\beta_{11} \leq 0$ hence the product of these is ≤ 0. Using this information and inserting these initial starting values with the rescaling into our monotone polynomial in (2.18) using sums of squares formulation in (2.20) gives:

$$\begin{aligned} p(x^*) &= \hat{A} + \alpha \int_0^{x^*} p_1(u)^2 + u p_2(u)^2 \, du \\ &= \hat{A} + \alpha \int_0^{x^*} \{\beta_{01}^2 + 2\beta_{01}\beta_{11}u + \beta_{11}^2 u^2\} + \{2|\beta_{01}\beta_{11}|u\} \, du \\ &= \hat{A} + \alpha \int_0^{x^*} \beta_{01}^2 + \beta_{11}^2 u^2 \, du \\ &= \hat{A} + \alpha|\hat{B}|x + \alpha|\hat{C}|x^{*3} \end{aligned}$$

Again, once α is determined we see this is then the fitted unconstrained polynomial described in the algorithm above (up to the sign of $|\hat{B}|$).

2.5.2.5 *Starting values – Case 3*

If we are fitting monotone polynomials over $\mathcal{R} = [a, b]$ for even degree polynomials as described in (2.22), with derivative odd, we initialise as follows:

1: Fit unconstrained polynomial of the form $y^* = A + Bx^* + Cx^{*2}$ where (x^*, y^*) are the rescaled (x, y) pairs.
2: Initialise $\delta = \hat{A}$
3: Initialise $\beta_{02} = \sqrt{|\hat{B}|}$
4: Initialise $\beta_{01} = -\sqrt{|\hat{B}| + 2|\hat{C}|}$
5: Initialise all other $\beta_{ij} = 0$.

Combining these values into (2.22) with corresponding rescaling of a and b gives:

$$\begin{aligned} p(x^*) &= \hat{A} + \alpha \int_0^{x^*} u p_1(u)^2 + (1-u) p_2(u)^2 \, du \\ &= \hat{A} + \alpha \int_0^{x^*} u\beta_{01}^2 + (1-u)\beta_{02}^2 \, du \\ &= \hat{A} + \alpha \int_0^{x^*} u(|\hat{B}| + 2|\hat{C}|) + (1-u)|\hat{B}| \, du \\ &= \hat{A} + \alpha|\hat{B}|x^* + \alpha|\hat{C}|x^{*2} \end{aligned}$$

Again, once α is determined we see this is then the fitted unconstrained polynomial described in the algorithm above (up to the sign of $|\hat{B}|$).

If we are fitting monotone polynomials over $\mathcal{R} = [a, b]$ for odd degree polynomials as described in (2.21), with derivative even, we initialise as follows:

1: Fit unconstrained polynomial of the form $y^* = A + Bx^* + Cx^{*3}$ where (x^*, y^*) are the rescaled (x, y) pairs.
2: Initialise $\delta = \hat{A}$
3: Initialise $\beta_{01} = \sqrt{|\hat{B}|}$
4: Initialise $\beta_{11} = -\beta_{01} - \sqrt{\beta_{01}^2 + 3|\hat{C}|}$
5: Initialise $\beta_{02} = \sqrt{\beta_{11}^2 - 3|\hat{C}|}$
6: Initialise all other $\beta_{ij} = 0$.

We note here that

$$\begin{aligned}
\beta_{11} &= -\left(\beta_{01} + \sqrt{\beta_{01}^2 + 3|\hat{C}|}\right) \\
&= -\left(\sqrt{|\hat{B}|} + \sqrt{|\hat{B}| + 3|\hat{C}|}\right) \\
&< 0
\end{aligned}$$

Hence combining these values into (2.21) with corresponding rescaling of a and b gives:

$$\begin{aligned}
p(x^*) &= \hat{A} + \alpha \int_0^{x^*} p_1(u)^2 + u(1-u)p_2(u)^2 \, du \\
&= \hat{A} + \alpha \int_0^{x^*} \beta_{01}^2 + 2\beta_{01}\beta_{11}u + \beta_{11}^2 u^2 + (u - u^2)\beta_{02}^2 \, du \\
&= \hat{A} + \alpha \int_0^{x^*} \beta_{01}^2 + (2\beta_{01}\beta_{11} + \beta_{02}^2)u + (\beta_{11}^2 - \beta_{02}^2)u^2 \, du
\end{aligned}$$

Note that $\beta_{11}^2 - 3|\hat{C}| > 0$ by construction, hence by definition of β_{02}^2 we have $\beta_{11}^2 - \beta_{02}^2 = \beta_{11}^2 - (\beta_{11}^2 - 3|\hat{C}|)$. Furthermore,

$$\begin{aligned}
2\beta_{01}\beta_{11} + \beta_{02}^2 &= 2\beta_{01}\beta_{11} + \beta_{11}^2 - 3|\hat{C}| \\
&= (\beta_{11} + \beta_{01})^2 - \beta_{01}^2 - 3|\hat{C}| \\
&= \left(-\sqrt{\beta_{01}^2 + 3|\hat{C}|}\right)^2 - \beta_{01}^2 - 3|\hat{C}|
\end{aligned}$$

Hence we get

$$p(x^*) = \hat{A} + \alpha|\hat{B}|x^* + \alpha|\hat{C}|x^{*3}$$

To reiterate, the final step of all the initialisations described is to set the α to be either -1 or $+1$ dependent on the correlation of (x_i, y_i) observations. However, the implementation in our R package `MonoPoly` allows the user to override this choice and specify whether α should be -1 or 1.

2.5.3 *Stopping criteria*

2.5.3.1 *Elphinstone*

We considered several criteria for stopping the algorithm. Namely monitoring the change in $\boldsymbol{\beta}$, monitoring the change in RSS, and monitoring the entries of the gradient vector ∇RSS. Our numerical work showed that RSS as a function of $\boldsymbol{\theta}$ is typically relatively flat in a neighbourhood of the optimal solution.

Hence monitoring changes in $\boldsymbol{\beta}$ or RSS, and basing a stopping criteria on either absolute or relative changes in either of these quantities being small, leads in many situations to a premature termination of the optimisation algorithm. Based on our numerical experiments, the only reliable way to determine convergence is to monitor the gradient vector ∇RSS, with a suitable default stopping criteria being that the absolute value of each entry in ∇RSS is smaller than 10^{-5}. Of course, in the parameterisations that employ c_j instead of c_j^2 and optimise RSS under the constraints that $c_j \geq 0$, $j = 1, \ldots, K$, the entries in ∇RSS that corresponds to c_js that are zero are not required to be smaller than 10^{-5}.

2.5.3.2 *Sum of squared polynomials*

We determine convergence by monitoring the gradient vector ∇RSS, with a suitable default stopping criteria being that the absolute value of each entry in ∇RSS is smaller than 10^{-5}.

2.6 MISCELLANEOUS COMMENTS

Previous applications of fitting monotone polynomials to data (Elphinstone, 1983; Heinzmann, 2008) involved fitting monotone polynomials of increasing degree. The suggestion by Elphinstone (1983) was to use the fitted parameters from a monotone polynomial of degree $q = 2K - 1$ as starting values for fitting a monotone polynomial of degree $q = 2K + 1$, initialising the two new additional parameters to zero. In Section 2.4.1.2 we demonstrated that, with the isotonic parameterisations described by Elphinstone (1983) and Heinzmann (2008), this strategy is problematic if a gradient based optimisation algorithm is used for minimising the objective function, as one of the new parameters would remain fixed at zero.

However, we note that using the new sum of squares formulation, implementing this method of choosing starting values when iteratively fitting monotone polynomials of increasing degree does not suffer from such problems. Moreover, for monotone polynomials over compact or semi-compact intervals, it is also possible to use such a strategy in instances when the degree is incremented by one. As monotone polynomials over semi-compact intervals have a single parameterisation, given by (2.20), it would be easy to implement such a strategy. By way of contrast, taking the same incremental approach for fitting monotone polynomials over compact intervals, would necessitate alternating between parameterisations (2.21) and (2.22).

As noted earlier in this chapter, the RSS as a function of $\boldsymbol{\beta}$ is a convex function, and strictly convex if the number of unique design points is larger than the degree of the polynomial fitted to the data, with no local extrema. Conversely, the RSS as a function of $\boldsymbol{\theta}$ is not convex, and consequently may have local extrema. Later we will show that, for some of the Elphinstone derived isotonic parameterisations used, the iterative optimisation algorithm has convergence problems or converged to a local minima. In our extensive numerical experience, when using the sums of squares parameterisation to fit polynomials that are monotone over the whole real line, and the starting values described in the previous section, there is no evidence of such problems arising, that is we have not observed any convergence issues or any convergence to local minima. However, when fitting polynomials that are constrained to be monotone over a (semi-)

compact region the potential of the algorithm converging to a local minima seems to be greater. In our extensive simulations, we have experienced instances of convergence to a local minima. Specifically we observed, for polynomials constrained to be monotone over a semi-compact region, upon convergence a RSS larger than that of a polynomial of the same degree which was constrained to be monotone over the whole real line fitted to the same data. This clearly indicates convergence to a local minima in the former instance. Thus, as in all optimisation problems in which local minima could be a potential issue, we recommend the use of random starting values. Additionally, when fitting polynomials (of odd degree) constrained to be monotone over a (semi-)compact interval, we recommend comparing the fit with the polynomial that is constrained to be monotone over the whole real line to ensure that one has obtained at least as good a fit.

2.7 NUMERICAL EXPERIMENTS AND RESULTS

2.7.1 *Simulated data examples*

To assess the effectiveness of our methodology, and to determine which combinations of parameterisations and formulations provide stable results, we carried out numerical experiments using four simulated datasets.

We used the simulated data ($n = 50$) published in Hawkins (1994), growth data with $n = 31$ measurements for the first male individual from the Berkeley Growth Study described in Ramsay & Silverman (2002), and two simulated data sets from the regression models (2.25) and (2.26), respectively:

$$W0: \quad Y_i = p(x_i) + \epsilon_i = 0.1x_i^3 + \epsilon_i, \tag{2.25}$$

where $x_i = -1 + \frac{i-1}{10}$ and $\epsilon_i \overset{i.i.d}{\sim} N(0, 0.01^2)$, $i = 1, \ldots, 21$; and

$$W2: \quad Y_{ij} = 4\pi - x_i + \cos(x_i - \pi/2) + \epsilon_{ij}, \tag{2.26}$$

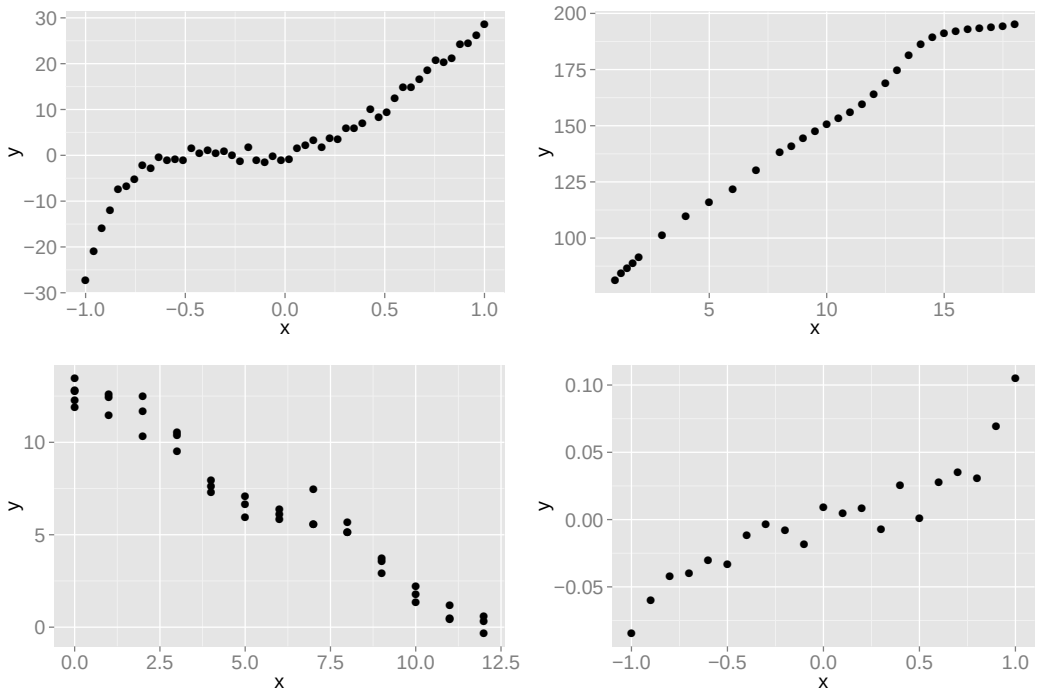

Figure 2.2: Data sets used for numerical experiments. Clockwise from top left: *Hawkins*, *RS*, *W*0 and *W*2.

where $x_i = i - 1$, $i = 1, \ldots, 13$ (restricting the maximum degree polynomial that can be fitted to be 11), and $\epsilon_{ij} \overset{i.i.d.}{\sim} N(0, 0.5^2)$ with $j = 1, \ldots, n_i$ and all $n_i = 3$ except for $n_1 = 5$, for a sample size of $n = 41$. We note that the *W*2 data is also not simulated from a polynomial model, further testing the ability of monotone polynomials to effectively solve problems where the underlying phenomenon is not polynomial in nature.

We denote these four data sets for the remainder of this chapter as *Hawkins*, *RS*, *W*0 and *W*2, respectively, and show scatterplots in Figure 2.2. The latter of these was specifically generated to mimic our motivating problem described in the introduction of this chapter, and is typical of any of the datasets in Firmin, Müller & Rösler (2011, 2012).

Figure 2.2 shows that the ranges for the x and y values are quite different for the various examples. Our numerical experiments indicate that it is beneficial, for numerical stability, to re-scale all x and y values to have minimum and maximum of -1 and 1, respectively, before starting the iterative optimisation process described in Section 2.5.1.

Using the methodology described in the previous section, we fitted monotone polynomials to each data set, for the three different Elphinstone type parameterisations,

Table 2.1: Objective function value RSS/n $\times 10,000$, by parameterisation for the four data sets (* indictates algorithm did not converge).

		Using (2.6)		Using (2.7)		Using (2.8)		Hawkins'	SOS
		c^2	$c \geq 0$	c^2	$c \geq 0$	c^2	$c \geq 0$	approach	Approach
Data 1	K=1	83.16	83.16	83.16	83.16	83.16	83.16	83.16	83.16
(*Hawkins*)	K=2	16.34	16.34	17.44	16.34	16.34	16.34	16.34	16.34
	K=3	11.55	10.79	11.55	10.79	11.55	10.79	10.79	10.79
	K=4	10.77	10.77	11.15	10.77	10.77	10.77	10.77	10.77
Data 2	K=1	34.75	34.75	34.75	34.75	34.75	34.75	34.75	34.75
(*RS*)	K=2	13.83	13.83	34.98	13.83	34.98	13.83	13.83	13.83
	K=3	5.86	4.05	35.06	4.05	4.05	4.05	4.05	4.05
	K=4	5.88*	4.04	26.18	4.04	4.04	4.04*	4.04	4.04
Data 3	K=1	105.52	105.52	105.52	105.52	105.52	105.52	105.52	105.52
(*W0*)	K=2	77.52	77.52	77.52	77.52	77.52	77.52	77.52	77.52
	K=3	73.82	73.82	77.14	73.82	75.05*	73.82	73.82	73.82
	K=4	73.76	73.75	73.75	73.75	73.83*	73.75	73.75	73.75
Data 4	K=1	130.93	130.93	130.93	130.93	130.93	130.93	130.93	130.93
(*W2*)	K=2	129.57	129.57	129.57	129.57	129.57	129.57	129.57	129.57
	K=3	58.37	57.35	58.37	57.35	58.37	57.35	57.35	57.35
	K=4	56.51	56.51	56.53	56.51	58.41	57.35*	56.51	56.51

using both the constrained and unconstrained formulation (for the c_j parameters) up to degree 9. We chose 9 to mirror the approach described by Hawkins (1994) where he indicated that polynomials of higher degree are of little practical use. However, we do revisit this assertion later, where we describe the fitting of monotone polynomials to much higher degrees. These results are described in Table 2.1, which also includes results using Hawkins' (1994) approach and the sum of squared polynomial approach.

During our numerical experiments we noticed that when using the c^2 formulations, the RSS indeed appears to have local extrema. This is consistent with our comments earlier regarding second derivative based algorithms. For example, when fitting a degree 9 polynomial to the *RS* data, using (2.7) with c_j^2, we note that all c_j estimates become zero early in the iterative process and are subsequently trapped at that value. For the same data and degree polynomial with the constrained formulation, the same phenomenon again occurs early in the iterative process, and all c_js become zero. However, subsequently they depart from zero and the process continues giving a much lower objective function value on termination of the algorithm. This is described in Figure 2.3 where in the top panel, representing the c_j^2 formulation we see that after only 15 iterations we achieve convergence at a local minima, and note that all c_js have

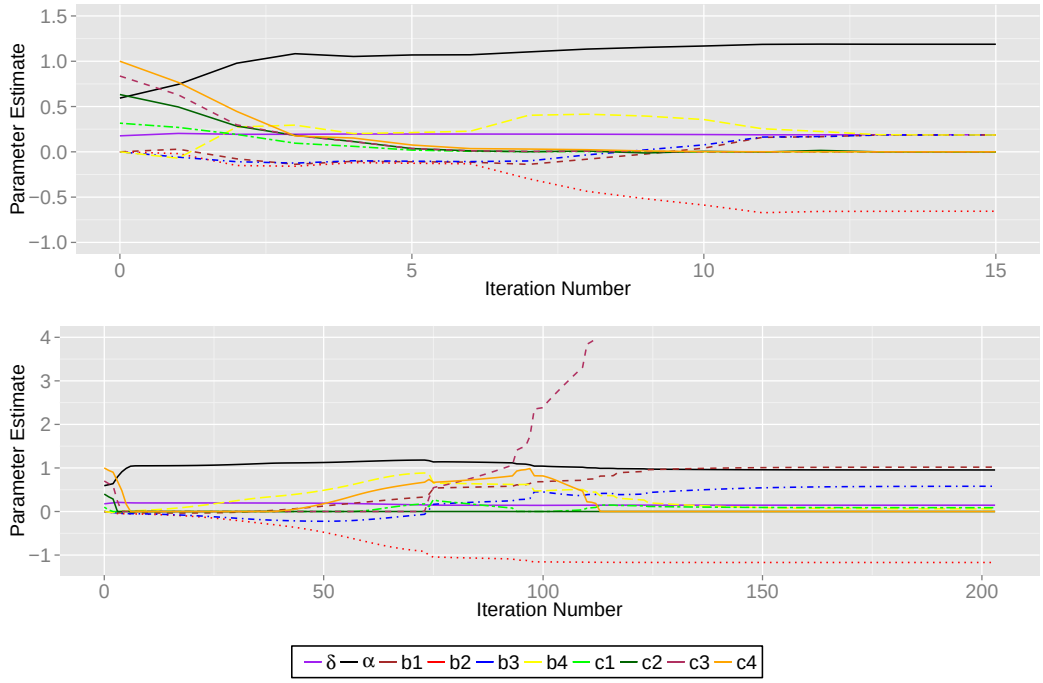

Figure 2.3: How the parameter estimates change through the optimisation iterative stages for the *RS* data when fitting a degree 9 polynomial using (2.7) parameterisation. Top panel c_j^2 formulation; bottom panel $c \geq 0$ formulation.

settled early at zero. However, the bottom panel of Figure 2.3, representing the corresponding $c_j \geq 0$ formulation shows that only two of the four c_j parameter estimates end up at zero even though all four estimates were zero at some point during the iterative process. Subsequently the iterative process was allowed to continue updating c_j values and ultimately achieving a lower objective function value.

For all Elphinstone type parameterisations we have $\delta = \beta_0$ and for (2.6) and (2.7), $\alpha \propto \beta_q$ and $\alpha \propto \beta_1$ respectively. Thus, for these two parameterisations α is identifiable, and only the remaining β_js are functions of α, the b_js and the c_js. By way of contrast, for the parameterisation (2.8) all β_js ($j \neq 0$) are functions of α, the b_js and the c_js which makes none of the latter parameters identifiable. This can lead to additional problems when (2.8) is used as we observed in our numerical experiments. For this parameterisation we observed instances in which $\alpha \to 0$ and at least one of the $(b_j, c_j) \to \infty$, hence at least partially cancelling each other's effect on $\boldsymbol{\beta}$. We display this phenomenon in Figure 2.4 and note in the left panel that most of the parameters have achieved a stable estimate after approximately 150 iterations. However, in the right panel we see that the

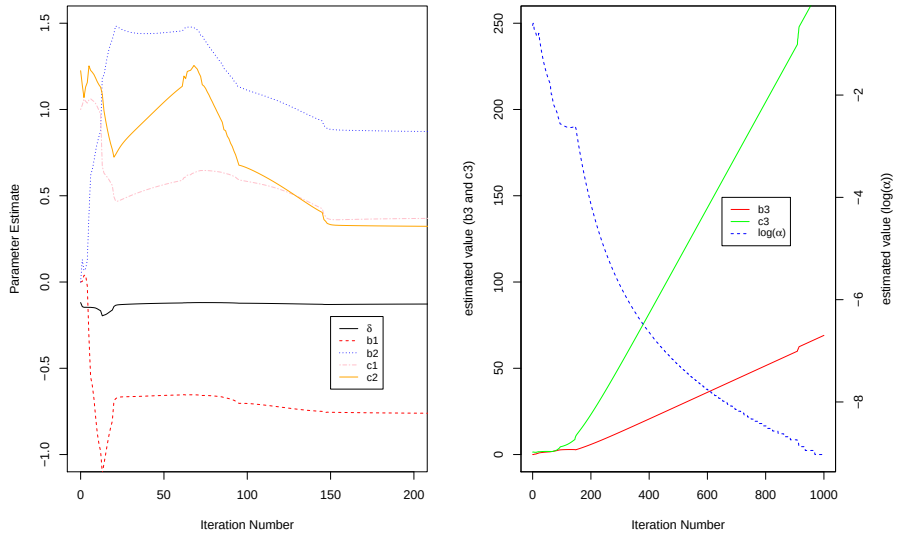

Figure 2.4: Parameter estimates changing through the optimisation iterative stages using (2.8) parameterisation. Left panel c_j^2 stable parameter that achieve stability; right panel cancelling effects of other parameter estimates.

(b_3, c_3) pair of parameters are still increasing rapidly after 1,000 iterations and the α parameter is continuing to decrease, hence the cancelling of effects on the $\boldsymbol{\beta}$.

Specifically, from Table 2.1, we also note the following: The performance (that is lower objective function values in smaller number of iterations) when using (2.8) is inferior to that of (2.6) and (2.7). When comparing the performance of the latter two parameterisations there is little difference when using the constrained formulation ($c \geq 0$). However, (2.6) seems to perform better when the unconstrained formulation is used. Given the demonstrated problems with local minima with the c^2 parameterisation, we would ordinarily not recommend its use. However, if one does use it, we have found it beneficial to perturb any c_j that was zero at the solution and restart the algorithm with these perturbed estimates as starting values. This usually ensured that the objective function value after the restart was comparable with other parameterisations' performances.

We also make some comments on the sums of squares parameterisation. First, we note that problems of local minima were not experienced at all using this parameterisation for the data sets considered. Second, we note that the speed of convergence using this parameterisation was relatively quick compared to all three Elphinstone derived

isotonic parameterisations. We also note that the Hawkins algorithm, with our implementation is efficient and speedy. Further comparisons of these algorithms are made in Chapter 4.

2.7.2 *Optimisation speed*

As mentioned previously, the sums of squares parameterisations and Hawkins algorithm appear to be the most efficient in terms of speed. We extended the experiments described in Section 2.7.1 to provide an insight into the convergence at higher degrees. For each of the four data sets we fit odd degree polynomials of degree 1 through to 23. Figure 2.5 depicts the processing times for each of these datasets and each algorithm/parameterisation, using a standard installation of R running on a relatively modest machine (MacBook Air, Processor 2GHz Intel Core i7, RAM 8GB). We observe from our experiments that higher degree polynomials are somewhat cumbersome for the three isotonic parameterisations (2.6), (2.7) and (2.8). Specifically, monotone polynomials of degree 9 or higher appear to have much longer processing times to achieve convergence. However, parameterisation (2.7) appears to perform moderately better for polynomials up to degree 15. When examining the sums of squares approach and our implementation of the Hawkins' algorithm, we observe very quick processing times for both indicated by the overlaying lines very close to zero in Figure 2.5. In fact, in terms of speed, both algorithms are comparable for all degree monotone polynomials. However, using our implementation of the Hawkins' algorithm would only allow a limited degree polynomial to be fit before encountering problems mainly related to the QR factorisation. Specifically in our examples using the $W0$ data and $W2$ data polynomials above degree 19 and degree 11 respectively can not be fit using this algorithm. For completeness we show in Figure 2.6 the values of the RSS after convergence. We note that up to degree 9 the converged RSS values are essentially identical for all methods except degree 9 polynomials fit using (2.8) for the $W2$ data. Beyond degree 9 there are some slight discrepancies, again particularly relating to the use of (2.8) but also in some instances using (2.6). The sums of squares approach, (2.7), and Hawkins ap-

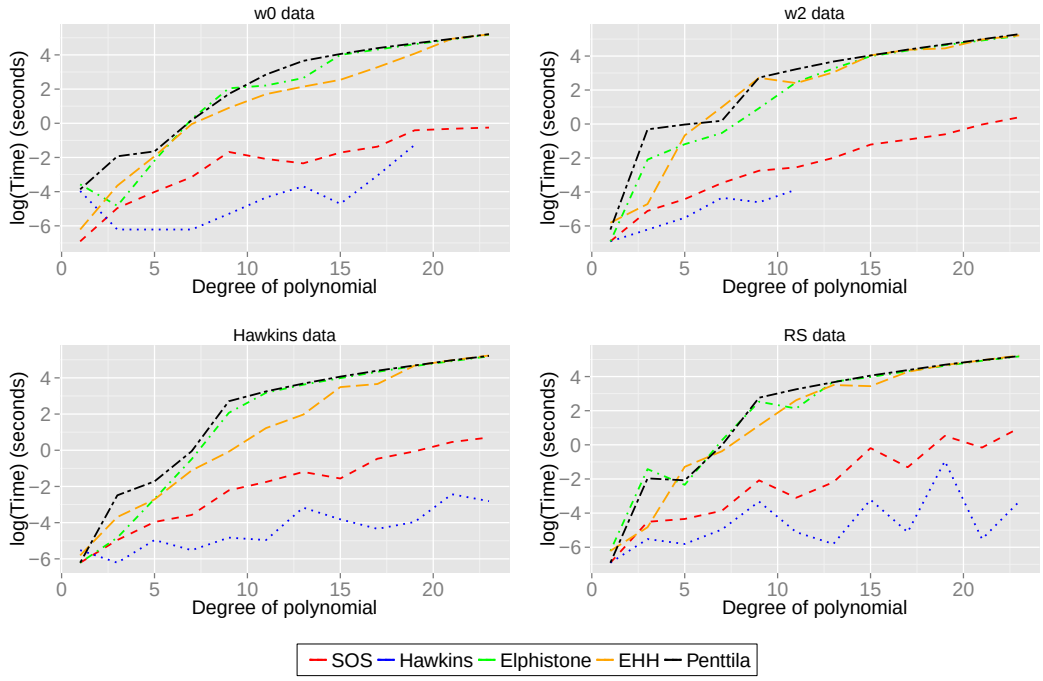

Figure 2.5: Processing time for algorithms to converge by degree of polynomial and different data sets

proach (in situations where problems with the QR factorisation are not encountered) all seem comparable otherwise.

We propose at this point that Figure 2.6 can be useful in addressing the model selection problem, that is which degree polynomial should we use? This is something we consider in more detail in subsequent chapters. However, for now we note that the use of monotone polynomials dramatically reduces the model selection problem when compared to alternative approaches (for example smoothing splines), in that one only has to select a model from d distinct models where d is the largest polynomial to be considered (assuming we follow a sums of squares approach and allow even degree polynomials that is). We will show later, that if we look at a loss function such as RSS as we have here or more generally the log-likelihood, we can immediately rule out several models based on the convex nature of plotting maximised likelihood against model dimension in a similar fashion to Figure 2.6. In particular we will develop what we describe as the maximum lower enveloping convex curve (MELCC) in Chapter 6. Using this and an information criteria approach to the model selection problem, we show that specific models lying inside this curve should not be considered.

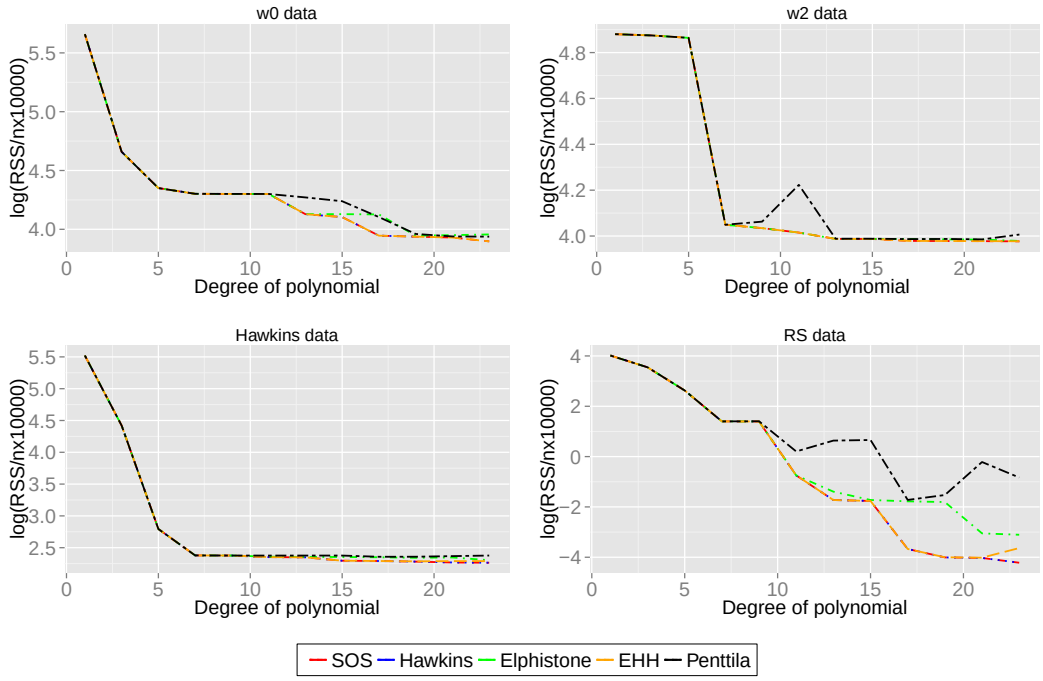

Figure 2.6: RSS for final converged iteration by degree of polynomial for different algorithms and data sets

2.7.3 *Random starting values*

As in other situations when an objective function with local extrema is optimised using a derivative based algorithm, we recommend starting this optimisation process from several randomly chosen starting values. In our numerical experiments, for constrained parameterisations (2.6) and (2.7) we took 100 simulations using random starting values for the b_js and c_js by simulating from a standard Normal distribution, taking the absolute value for the constrained c_js. In all simulations the algorithm converged to the value shown in Table 2.1. However, the same amount of success was not seen when the y values were not rescaled initially.

2.7.4 *Rank deficient designs*

To further illustrate our methods, and to illustrate the point mentioned in Section 2.3 regarding situations with a sparse design but multiple observations per x-value, we subset our W2 data to retain only the even values of x. Using Hawkins' (1994) ap-

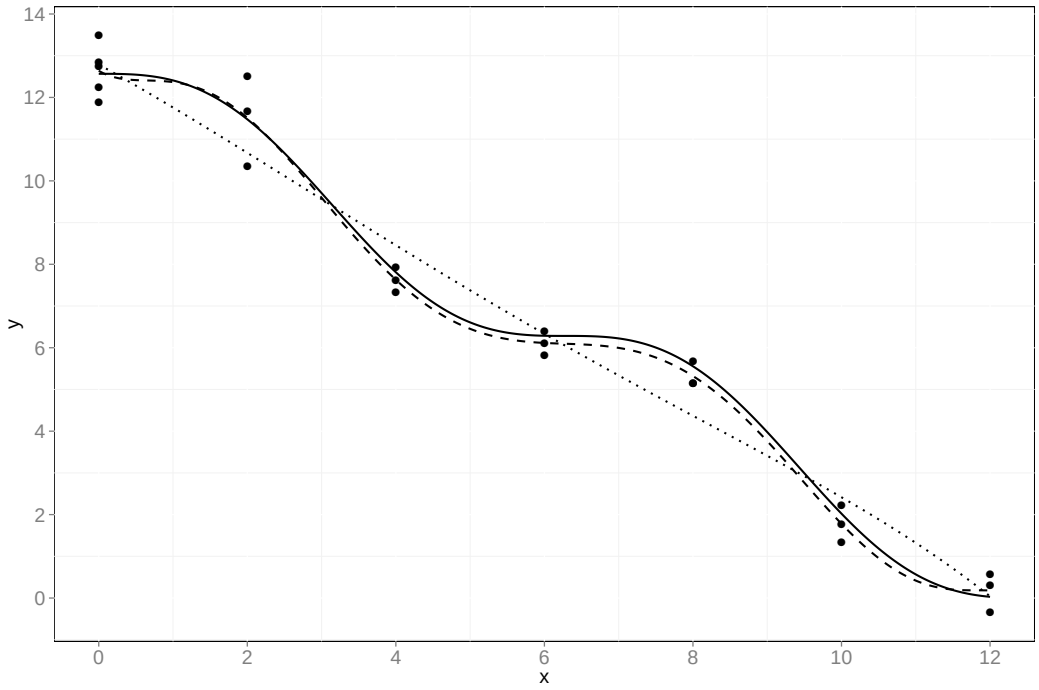

Figure 2.7: Monotone fits to subset of *W*2 data. Solid line is the underlying regression function while the dotted and dashed lines are the best fitting monotone polynomials of degree 5 and 7, respectively.

proach and the isotonic parameterisations described previously, we consider fitting monotone polynomials of degrees 5 and 7. The results are shown in Figure 2.7. Here, the solid line shows the underlying regression function and the dotted line is the monotone polynomial of degree 5, which all isotonic parameterisations and Hawkins' method agree upon. The dashed line is the monotone polynomial of degree 7, which is consistently produced using our three Elphinstone type isotonic parameterisations and the sums of squares approach. However, due to the Hawkins' method starting at the unconstrained solution, fitting a degree 7 polynomial to this data is not achievable using his methodology.

We observe that Hawkins' approach starts at the unconstrained solution as he proposes to use the Goldfarb-Idnani (Goldfarb & Idnani, 1982, 1983) quadratic programming algorithm iteratively to solve the semi-infinite quadratic programming problem

$$\text{minimise}_{\boldsymbol{\beta}}\, \text{RSS}(\boldsymbol{\beta}) \quad \text{subject to} \quad p'(x;\boldsymbol{\beta}) \geq 0, \forall x.$$

If Hawkins' approach were substantially modified to use another quadratic programming problem solver it could possibly work for rank deficient designs.

2.8 CONCLUSIONS

In this chapter we have revisited Hawkins' (1994) algorithm for fitting monotone polynomials and discussed its implementation. We have identified two situations in which this algorithm becomes problematic, namely when q is large causing problems with the QR factorisation of the design matrix, or when q is large relative to the number of unique points in the design matrix. Both these situations are important for using monotone polynomials to detect the location and number of inflection points, our major motivation for this work. Consequently we have provided a new algorithm for fitting monotone polynomials to data using various formulations for isotonic polynomials, which overcomes these issues.

We have also identified some of the drawbacks of previous methodologies, that are not based on monotone polynomials, and conclude that our method, which has the advantage of being parametric, is an alternative to many of the existing smoothing spline based techniques.

Four numerical examples for fitting monotone polynomials up to degree nine with our algorithm were presented. However, in other investigations—shown in Chapter 4—comparing our methodology to smoothing spline based approaches, we have fitted models up to degree 25, which indicates that in principle our methods are not degree constrained.

In comparing the isotonic parameterisations (2.6), (2.7) and (2.8) and formulations, we have shown that using the c^2 formulation as originally described in Elphinstone (1983), and subsequently used by other authors, can produce converged objective function values at a local extrema when a derivative based optimisation algorithm such as Newton–Raphson is used. This problem is observed in particular when the 'step up' approach is used but is not limited to this. It can also be problematic when one (or more) c_j parameter estimate becomes zero.

We have provided recommendations on the choice of sensible starting values and which of the formulations provide more consistent results.

Fitting unconstrained polynomials to data has historically been an easy task and is popular because of the simplicity of the functions and ease of interpretability of corresponding estimates. The constrained version of such polynomials has not received as much publicity, mainly due to the lack of available software to implement such methods and the difficulty in providing effective optimisation algorithms. We have shown through our simulations, that there is not only a need for such methodology but a necessity in some instances where standard polynomials do not provide intuitively correct estimated functions, for example ensuring monotonicity. Initially using parameterisations (2.6), (2.7) and (2.8), ideas were postulated on how such models can be fitted with various different formulations. These methods were made available through the `MonoPoly` package in R. We have identified some of the short-comings of these parameterisations and extended the work, now including the ability to fit monotone polynomials using the *sum of squared polynomial* formulation. This new formulation itself has led to a dramatic reduction of time to fit such models compared to the other methods. In addition, and as a consequence of the sums of squares formulation, we presented methodology in which we can constrain the polynomial to be monotone over the full region of x values, over a semi-compact region, or over compact regions. These latter two methodologies result in allowing even degree polynomials to be considered, something previously unattainable in the fitting of monotone polynomials to data. Our numerical examples have demonstrated that using this methodology can describe a relationship in a more efficient way, and, allows even degree polynomials that are monotone over a compact or a semi-compact region to be fitted.

All code to fit the models in this article are available in the most recent version of the R (R Core Team, 2015) package `MonoPoly` (version 0.3-6 or later) currently available on CRAN.

3

INFERENCE FOR MONOTONE POLYNOMIALS

SUMMARY

Estimates of variation in monotone polynomials has received no attention in the literature to date with the main focus being on trying to find effective algorithms to fit monotone polynomials to data. We provide extensive investigations into estimates of variance for the fitted values and to estimates of variance around the parameter estimates. We use Monte-Carlo simulations and compare several bootstrapping algorithms to not only demonstrate the need for such techniques but to identify situations in which these may need adjusting. We initially start by demonstrating consistency of monotone polynomials and provide, through Monte-Carlo sampling, empirical results which proffer justifications for considering alternative approaches to standard least squares unconstrained model fitting for monotone polynomials. We describe situations where the use of standard bootstrap methodology for polynomials under the monotonicity constraints produces particularly low coverage probabilities in certain areas of the fitted curve, dependent on whether the true function is on the boundary of the cone of monotone polynomials. Consequently we describe alternative methodologies to address this issue and show that an adjustment by using either the m out of n bootstrap or a post hoc symmetrisation of the confidence bands is necessary to achieve more uniform coverage probabilities over the whole range of the curve. We show that using standard bootstrap techniques to generate point wise *prediction intervals* does not appear to suffer from the same under coverage problems as the corresponding confidence intervals.

3.1 INTRODUCTION

The fitting of monotone polynomials to data has historically been complex due to both computational issues and problems providing adequate parameterisations that ensure monotonicity in the fitted function. In the previous chapter, and in Murray, Müller & Turlach (2013, 2015), we have shown, using our new techniques for fitting monotone polynomials to data, that this process is now more tenable and relatively simple solutions can be achieved. However, as always it is important to provide estimates of variability of parameter estimates and in particular of fitted regression curves, that are statistically sound. Furthermore, in some situations where prediction is the most important aspect of modelling we may wish to provide prediction intervals. To date it would appear there is little work on variance estimation or confidence intervals for monotone polynomials. Whilst there is a large body of literature on bootstrap methodology for estimating confidence bands or point wise confidence intervals in a non-parametric, or semi-parametric framework (see for example the summary provided in Hall & Horowitz, 2013, and references therein), to our knowledge there is a lack of available bootstrap methodology for our specific problems. When the estimation of confidence bands or point wise confidence intervals is of primary interest, we note there are a couple of exceptions that address some of the more general issues (not specific to monotone polynomials) arising when calculating such bands under shape constraints described in the literature. In particular Hall & Horowitz (2013) describe the construction of bootstrap confidence bands, allowing for the fact that the standard bootstrap bias estimators suffers from relatively high frequency stochastic error. They combine this with a technique based on quantiles, leading to simple to construct confidence bands. Another exception is Dümbgen (2003) who constructed confidence bands based on a kernel smoothing approach for a given function with guaranteed minimum coverage probability, under the assumption that the function is isotonic or convex.

However, given the lack of direct literature for confidence bands and variance estimation in monotone polynomials, in this chapter we consider through numerical examples the impact of using various bootstrapping techniques, including the stand-

ard non-parametric (paired) bootstrap, the residual bootstrap, the weighted bootstrap and m out of n bootstrap, in situations where the function is constrained to be monotone. We examine their performance under such constraints and provide some general advice when considering the calculation of such confidence and prediction bands for monotone polynomials.

Specifically in Section 3.2 we provide a description of the data used throughout this chapter in our simulations, and provide motivation through a simple Monte-Carlo simulation. In Section 3.3 we examine the theoretical properties of monotone polynomials and provide consistency arguments. In Section 3.4 we look at empirical properties of monotone polynomials, providing further supporting for our consistency argument, and we motivate the need for alternative methodology for confidence interval estimation in monotone polynomial fitting through Monte-Carlo simulations. In Section 3.5 we investigate the use of bootstrapping and describe and compare various bootstrap methods with monotone polynomials. We also provide a comparison to the standard unconstrained model fitting when using bootstrapping techniques and demonstrate in which situations the solutions differ. In Section 3.7 we extend these bootstrapping ideas to enable the estimation of point wise confidence and prediction bands, and for our simulated data look at estimates of coverage probabilities for such bands. We also make two suggestions for the tuning of the sometimes asymmetric bands produced from these techniques to enable desired and/or uniform coverage probabilities. Finally, in Section 3.8, we provide some conclusions and a brief discussion.

3.2 SIMULATED DATA AND MOTIVATION

In this section we describe results from simulations that provide motivation for the necessity, and careful consideration, of variance estimation and confidence interval calculations for polynomials under the monotonicity constraint. For the remainder of this chapter we consider the following three monotone polynomial functions, and generate our data based on these:

$$p_1(x) = x^3, \quad p_2(x) = 3x^3 + x + 1, \quad p_3(x) = x^5.$$

We simulate responses for n equidistant design points over $[-1,1]$ from the linear regression models

$$Y_i \quad = \quad p_k(x_i) + \epsilon_i, \quad k = 1,2,3, \quad x_i = -1 + 2\frac{i-1}{n-1}, i = 1,\ldots,n.$$

For the two cubic regression models the errors are independent $N(0,2^2)$ and for the quintic the errors are independent $N(0,0.3^2)$. These three monotone polynomials are specifically chosen to demonstrate how the different variance and confidence interval techniques work under monotonicity constraints. In the case of $p_1(x)$ and $p_3(x)$ we have functions which lie on the boundary of the cone of monotone polynomials; conversely $p_2(x)$ lies well within this boundary. Furthermore, through $p_3(x)$ we demonstrate the impact of having a long virtually horizontal region in the regression function, a feature that many smoothing techniques find difficulty dealing with.

Standard least squares regression theory states that when fitting unconstrained polynomials of degree q, the distribution of the $q+1$ vector of parameter estimators, $\hat{\boldsymbol{\beta}}$, has a multivariate normal distribution when the errors are independent and identically normal. Specifically $\hat{\boldsymbol{\beta}} \sim MVN(\boldsymbol{\beta}, \boldsymbol{\Sigma})$, where $\boldsymbol{\Sigma}$ is the variance covariance matrix given by $\sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}$ and $\boldsymbol{X}$ is the design matrix with i^{th} row $\mathbf{x}_i = (1, x_i, x_i^2, \ldots, x_i^q)$. We also denote by $\mathcal{C}_{\mathcal{R}}^q$ the set of all $\boldsymbol{\beta} \in \mathbb{R}^{q+1}$ for which $p(x;\boldsymbol{\beta})$, a polynomial of degree at most q, is monotone for all $x \in \mathcal{R} \subseteq \mathbb{R}$. Note that $\mathcal{C}_{\mathcal{R}}^q$ is a (non-pointed) closed convex cone in $\mathbb{R}^{q+1}$. For the sake of brevity, we will refer below to polynomials lying within $\mathcal{C}_{\mathcal{R}}^q$ or on its boundary instead of the vector of coefficients of a polynomial lying within $\mathcal{C}_{\mathcal{R}}^q$ or on its boundary.

To provide an idea of the necessity of constraining fitted polynomials to be monotone, we carry out Monte-Carlo integration of the density of $\hat{\boldsymbol{\beta}}$, over the region of parameter vectors corresponding to monotone polynomials. The observed probabilities of monotone fits, using standard unconstrained polynomials and least squares estimation, are shown in Table 3.1.

As one would expect, for $p_2(x)$ the probability of a fit being monotone quickly converges to one. By way of contrast, for $p_1(x)$ and $p_2(x)$, which lie on the boundary of $\mathcal{C}_{\mathbb{R}}^q$, the probability of a fit being monotone does not converge to one as $n \to \infty$.

Table 3.1: Expected proportion of monotone fits for a range of sample sizes, when the data generating polynomial is $p_k(x)$, $k = 1, 2, 3$, and the standard least squares estimator is used to fit unconstrained polynomials.

n	$p_1(x) = x^3$	$p_2(x) = 3x^3 + x + 1$	$p_3(x) = x^5$
50	0.131	0.727	0.086
100	0.187	0.858	0.111
200	0.275	0.946	0.141
500	0.405	0.995	0.194
1000	0.465	1.000	0.240
10000	0.493	1.000	0.356

3.3 THEORETICAL PROPERTIES

In this section we provide some theoretical properties of monotone polynomial estimators. Specifically we describe a consistency argument for monotone polynomials and make some comments on bias and variance.

3.3.1 *Consistency*

Following on from Section 3.2, we note that a monotone polynomial estimator of a polynomial lying within $\mathcal{C}^q_{\mathcal{R}}$ (for some q and some $\mathcal{R}$) is asymptotically equivalent to the unconstrained polynomial estimator, and therefore inherits all the properties of the latter, such as consistency and multivariate normality of the estimated vector of coefficients. Conversely, for a polynomial lying on the boundary, the monotone polynomial estimator is, by the triangle inequality, closer to the "true" polynomial than the unconstrained polynomial estimator, as the former is the orthogonal projection (with respect to some norm) of the latter onto $\mathcal{C}^q_{\mathcal{R}}$. Consistency of the constrained polynomial follows from this observation, although in this case the asymptotic distribution of the constrained polynomial estimator is more complicated; see also Barlow, Bartholomew, Bremner *et al.* (1972) or Silvapulle & Sen (2005).

A slightly more careful argument for consistency of the monotone polynomial estimator would be necessary if q is also estimated, for example via m out of n bootstrap, as

we propose and discuss in Chapter 4. However, the theory of the m out of n bootstrap (Shao, 1996) indicates that it provides a consistent estimator $\hat{q}$ of the degree of an underlying monotone polynomial thus, the monotone polynomial fit from $\mathcal{C}_{\mathcal{R}}^{\hat{q}}$ would be consistent for the underlying monotone function.

While our numerical examples concentrate on underlying response functions that are monotone polynomials, one might wonder what can be said if the true underlying response function $m(x)$ is monotone over $\mathcal{R}$ but not a polynomial. The results in Shevchuk (1993) would allow us to establish point-wise consistency for any $x_0 \in \mathcal{R}$ if $\mathcal{R}$ is a compact interval. Essentially, Shevchuk (1993) proved that monotone functions over compact intervals can be approximated uniformly to an arbitrary precision by monotone polynomials of sufficiently high degree q . This implies that for the monotone polynomial $p_{m,q}(x)$ in $\mathcal{C}_{\mathcal{R}}^{q}$ onto which $m(x)$ is projected, $|p_{m,q}(x_0) - m(x_0)| \leq \epsilon/2$, for any pre-specified ϵ, if q is chosen sufficiently large. Since the monotone polynomial estimator $\hat{p}_{m,q}(x_0)$ will be consistent for $p_{m,q}(x_0)$, for sufficiently large sample sizes $\hat{p}_{m,q}(x_0)$ will be with high probability less than $\epsilon/2$ away from $p_{m,q}(x_0)$ and, thus, by the triangle inequality, with high probability less than ϵ away from $m(x_0)$, establishing that the monotone polynomial estimator is also consistent in such settings. Establishing the rate of consistency would require a more careful argument and that $q = q(n) \to \infty$ as $n \to \infty$. While establishing the rate of consistency might be of some theoretical interest, it is not pursued further in this thesis.

3.3.2 *Bias and variance*

Whilst consistency properties are straightforward to provide evidence for, the bias in the estimators and associated variance are still open research problems. This is mainly due to the mathematical complexity of these when the estimation is constrained. Furthermore, in instances where the underlying regression function is inside the boundary of the cone of monotone polynomials, bias and variance may be more simple to describe due to the asymptotic nature of these estimates being identical to the that of the unconstrained estimates. When the underlying regression function is on the boundary of the cone of monotone polynomials, the same properties of estimators that hold in

least squares estimation do not hold due to there being no closed form solution, which can be a major problem.

Consequently we do not pursue further theoretical properties of monotone polynomials, mainly due to the mathematical complexity, which itself is not the major focus of this thesis. However, we will show empirical properties of such estimators in the next sections.

3.4 EMPIRICAL PROPERTIES

3.4.1 *Consistency*

We demonstrate through simple simulations, using the polynomials $p_i(x)$, for $1 = 1, 2, 3$, consistency arguments. Figure 3.1 shows simulated data for sample sizes 50 and $1,000$. In these instances, we have in all but one case selected an example where the standard unconstrained polynomial fit is not monotone. The exception is when using data generated from p_2 with $n = 1,000$, which, due to the location of the function inside the cone of monotone polynomials, would almost always provide a monotone increasing fit, as indicated in Table 3.1. We see from the top row of Figure 3.1, that even with a relatively small sample size ($n = 50$), the estimated curves using monotone polynomials (green dashed lines) are relatively close to the underlying population curve (black line). The bottom row demonstrates that as we increase the sample size the fit becomes closer, even in the case of p_2 where the underlying signal is extremely weak. This further supports the consistency argument in cases where the underlying function is a polynomial.

However, we argue that these consistency arguments would also hold in other instances. In Figure 3.2, we demonstrate the impact of fitting varying degrees of monotone polynomials to data generated using a non-polynomial function. We again simu-

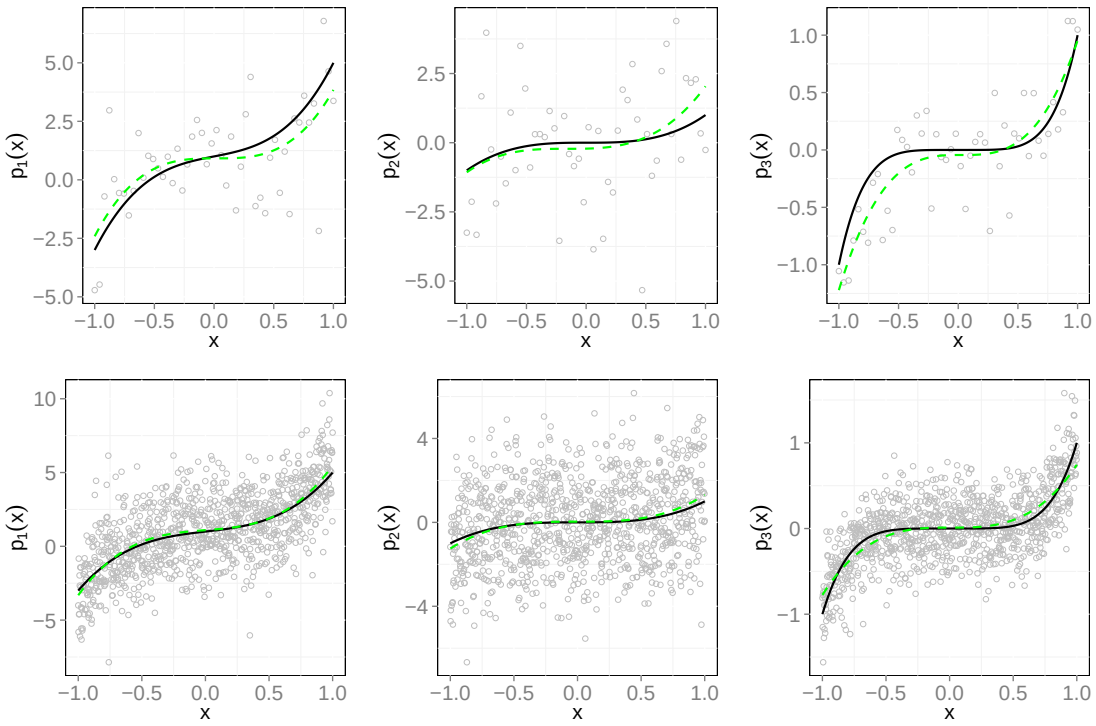

Figure 3.1: Monotone polynomial fitted curves for data generated from $p_1(x) = x^3$, $p_2(x) = 3x^3 + x + 1$, and $p_3(x) = x^5$ for sample size $n = 50$ (top row) and $n = 1,000$ (bottom row). Green dashed line - fitted curve; black solid line - underlying population polynomial function.

late responses for n equidistant design points over $[-1, 1]$, this time from the sigmoidal model, using the following model:

$$y = \frac{10}{1 + \exp(-8x)} + \epsilon \tag{3.1}$$

where the errors are independent $N(0, 1)$. We see that with a relatively low sample size, using a monotone polynomial of a sufficiently high degree, one can obtain a reasonable fit. Furthermore increasing the sample size dramatically improves the fit. Note in this instance we have arbitrarily chosen the degrees of polynomial to demonstrate the effectiveness when the underlying function is not a polynomial. We consider further methods for determining the 'most appropriate' degree of monotone polynomial in Chapters 5 and 6.

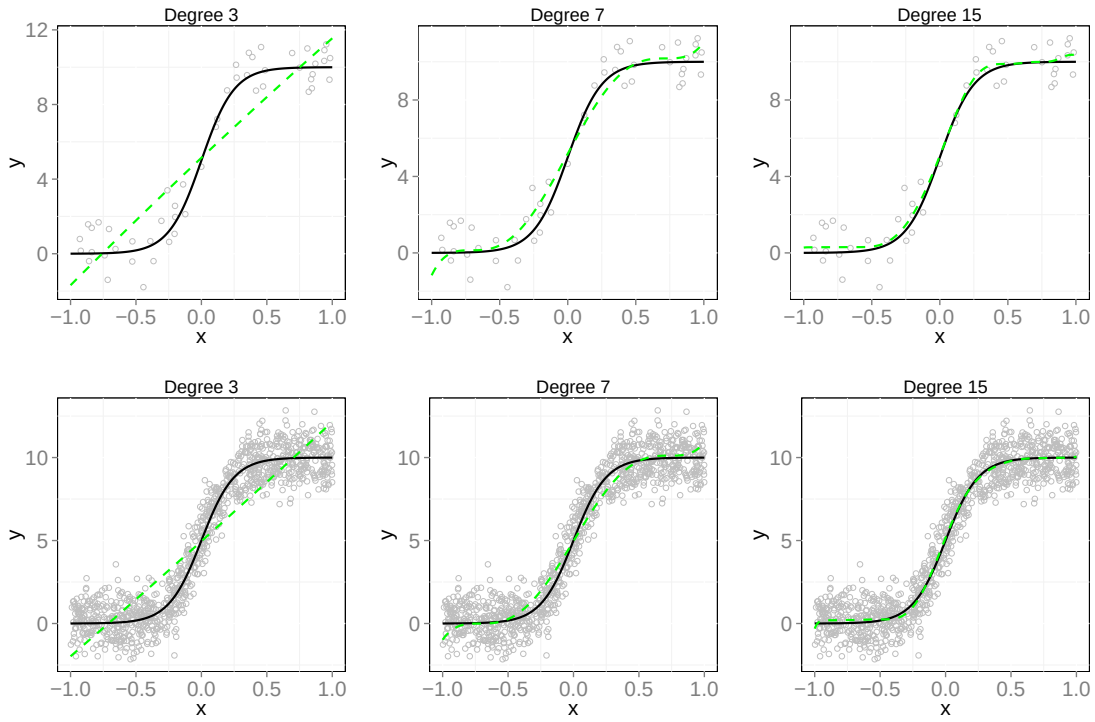

Figure 3.2: Monotone polynomial fitted curves for data generated from sigmoidal functions for sample sizes $n = 50$ (top row) and $n = 1,000$ (bottom row). Green dashed line - fitted curve; black solid line - underlying population polynomial function.

3.4.2 *Bias and variance*

3.4.2.1 *Standard polynomial regression*

In standard polynomial regression situations the estimation of variances, confidence intervals, and prediction intervals is routine under certain assumptions. Mainly the distribution of the estimators are known and hence variability estimates and upper and lower confidence and prediction limits can be easily evaluated. Consider an unconstrained polynomial regression $p(x)$ model

$$p(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_q x^q + \epsilon, \tag{3.2}$$

where q is the degree of the polynomial, the βs are standard regression coefficients and ϵ is an error term which is assumed to be distributed $N(0, \sigma^2)$. We can estimate confidence intervals for $\boldsymbol{\beta}$, and $\mu(x)$ the mean of the regression function, along with

prediction intervals for $p(x)$, using standard least squares estimation and standard linear model theory described earlier in Section 3.2 and in more detail in standard linear regression texts (see, for example, Draper & Smith (1998)).

Whilst in standard unconstrained polynomial regression we can use this theory to examine the estimates and variability around our estimated response and parameter estimates, in other situations, where the standard assumptions do not hold, alternative estimation approaches for the variance and those functions derived from it, need to be investigated. Examples of such situations could be when the errors are not independent and identically normally distributed, when sample sizes are small, or there is some constraint(s) on the parameter estimates (as with monotone polynomials), effectively removing our ability to determine their distribution. The bootstrap is one such approach which we introduce and describe in subsequent sections with specific focus on (constrained) regression scenarios.

3.4.2.2 *Monte-Carlo simulations for variance estimates*

As described previously we motivate the following sections with some Monte-Carlo simulations. We generate 1,000 simulated data sets from the polynomial functions $p_1(x)$, $p_2(x)$ and $p_3(x)$ and examine the Monte-Carlo estimates of the polynomial regression parameters and their standard errors shown in Table 3.2 and Figure 3.3. These estimates are calculated using:

$$\overline{\hat{\beta}}_i = m^{-1} \sum_{j=1}^{m} \hat{\beta}_i^{[j]}, \tag{3.3}$$

where $m = 1{,}000$ is the number of Monte-Carlo simulations, β_is are the polynomial regression parameters and the $\hat{\beta}_i$s their corresponding estimates, and

$$SE(\hat{\beta}_i) = \sqrt{\frac{1}{m-1} \sum_{j=1}^{m} (\hat{\beta}_i^{[j]} - \overline{\hat{\beta}}_i)^2}. \tag{3.4}$$

The top row in Figure 3.3 shows the distribution of parameter estimates, when the underlying data generating function is $p_1(x) = x^3$, using a standard unconstrained

Table 3.2: Monte-Carlo estimates and standard errors using standard unconstrained polynomial least squares fitting and monotone polynomial fitting from our simulated data examples, based on 10,000 simulated data sets with sample size 1,000

Sample Size	Estimation	Parameter	$p_1(x) = x^3$		$p_2(x) = 3x^3 + x + 1$		$p_3(x) = x^5$	
			estimate	SE	estimate	SE	estimate	SE
50	Unconstrained polynomial	β_0	-0.002	0.422	0.980	0.416	0.001	0.081
		β_1	-0.008	1.188	1.015	1.187	0.005	0.322
		β_2	0.012	0.908	0.011	0.895	-0.010	0.501
		β_3	1.062	1.728	2.985	1.753	0.002	1.223
		β_4	-	-	-	-	0.013	0.540
		β_5	-	-	-	-	0.987	1.039
	Monotone polynomial	β_0	0.000	0.347	0.980	0.410	0.000	0.057
		β_1	0.312	0.386	1.142	0.920	0.114	0.119
		β_2	0.007	0.588	0.010	0.873	-0.003	0.192
		β_3	0.701	0.646	2.823	1.409	-0.183	0.516
		β_4	-	-	-	-	0.006	0.266
		β_5	-	-	-	-	1.048	0.528
1000	Unconstrained polynomial	β_0	0.001	0.093	0.999	0.092	0.000	0.018
		β_1	-0.005	0.277	1.017	0.283	-0.002	0.072
		β_2	0.002	0.205	0.011	0.206	-0.001	0.108
		β_3	1.005	0.417	2.975	0.424	0.002	0.280
		β_4	-	-	-	-	0.000	0.119
		β_5	-	-	-	-	1.001	0.244
	Monotone polynomial	β_0	0.001	0.090	0.999	0.092	0.000	0.014
		β_1	0.113	0.153	1.017	0.283	0.037	0.039
		β_2	0.002	0.195	0.011	0.206	0.000	0.060
		β_3	0.841	0.267	2.975	0.424	-0.120	0.178
		β_4	-	-	-	-	-0.001	0.073
		β_5	-	-	-	-	1.089	0.174

least squares fit with sample size $n = 50$. As expected these estimators appear to be normally distributed and centred around the true parameter estimates, which in this instance are $[0, 0, 0, 1]$. The third row in the same figure shows the difference when monotonicity is imposed and monotone polynomial fitting is carried out using the methodology described in the previous chapter. Whilst there does not appear to be an impact on all of the parameter estimates there is a marked impact on $\hat{\beta}_1$ and $\hat{\beta}_3$. Both of these show asymmetric distributions with smaller variability than that obtained from the standard unconstrained least squares fit. In a similar fashion for $p_2(x)$ we show the results from standard unconstrained least squares estimation with $n = 50$ in the top row of Figure 3.4, and the results from monotone polynomial estimation in row three. Again, the standard least squares estimation provides what would be expected. However, the monotone polynomial estimation, in this instance, shows that $\hat{\beta}_1$ displays the largest discrepancy to standard least squares estimation, with the estimates truncated at zero. Again smaller variance is associated with the monotone polynomial parameter estimates. The estimated means and variances for all the parameters are shown in Table 3.2 which appears to confirm smaller variances for monotone estimates in general for the simulated data and functions we have considered. We note also that this apparent asymmetry and decreased variability in the estimates are seen for $p_1(x)$ even when the sample size is increased to $1,000$ (rows two and four of Figure 3.3) but not for $p_2(x)$ (rows two and four of Figure 3.4) at the same sample size. We believe this is due to $p_1(x)$ lying on the boundary of $\mathcal{C}^q_{\mathcal{R}}$ whilst $p_2(x)$ lies inside the boundary of $\mathcal{C}^q_{\mathcal{R}}$.

The information from this section suggests that alternative variance estimates are indeed needed for polynomials constrained to be monotone, in particular in situations where the sample size is smaller, or when the underlying monotone function is close to, or on the boundary of $\mathcal{C}^q_{\mathcal{R}}$. Unfortunately it is seldom know in practice whether the latter is true.

3.5 BOOTSTRAPPING

Bootstrap techniques first became prominent with the publication of the 1977 Rietz Lecture by Efron (1979). He succinctly described the problem to be

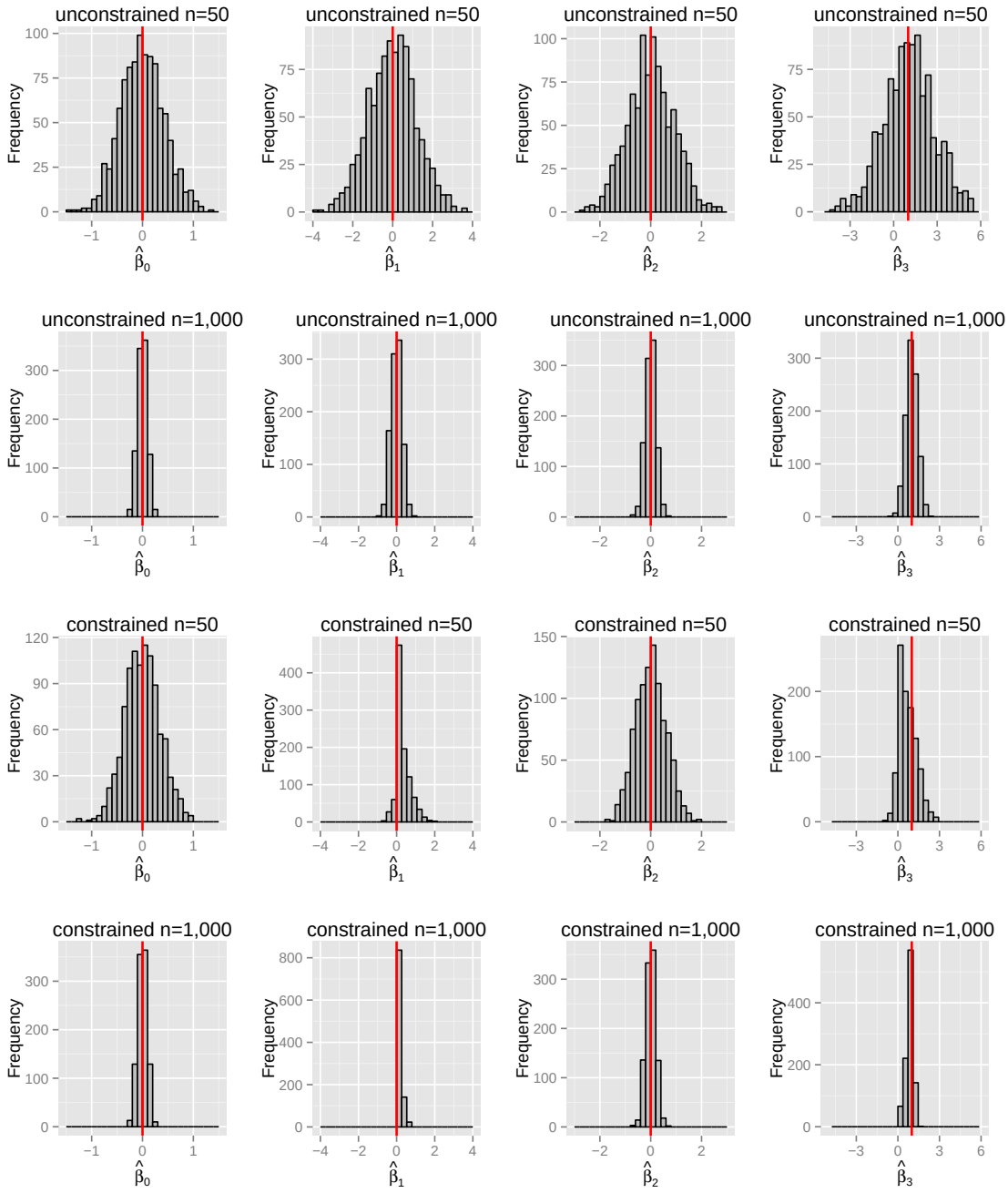

Figure 3.3: Standard unconstrained linear model fits and monotone polynomial fits to data generated from $p_1(x) = x^3$: Top two rows standard least squares estimates for $n = 50$ and $n = 1,000$ respectively; bottom two rows monotone polynomial estimates for $n = 50$ and $n = 1,000$ respectively. Red vertical lines indicate the location of the true parameter value.

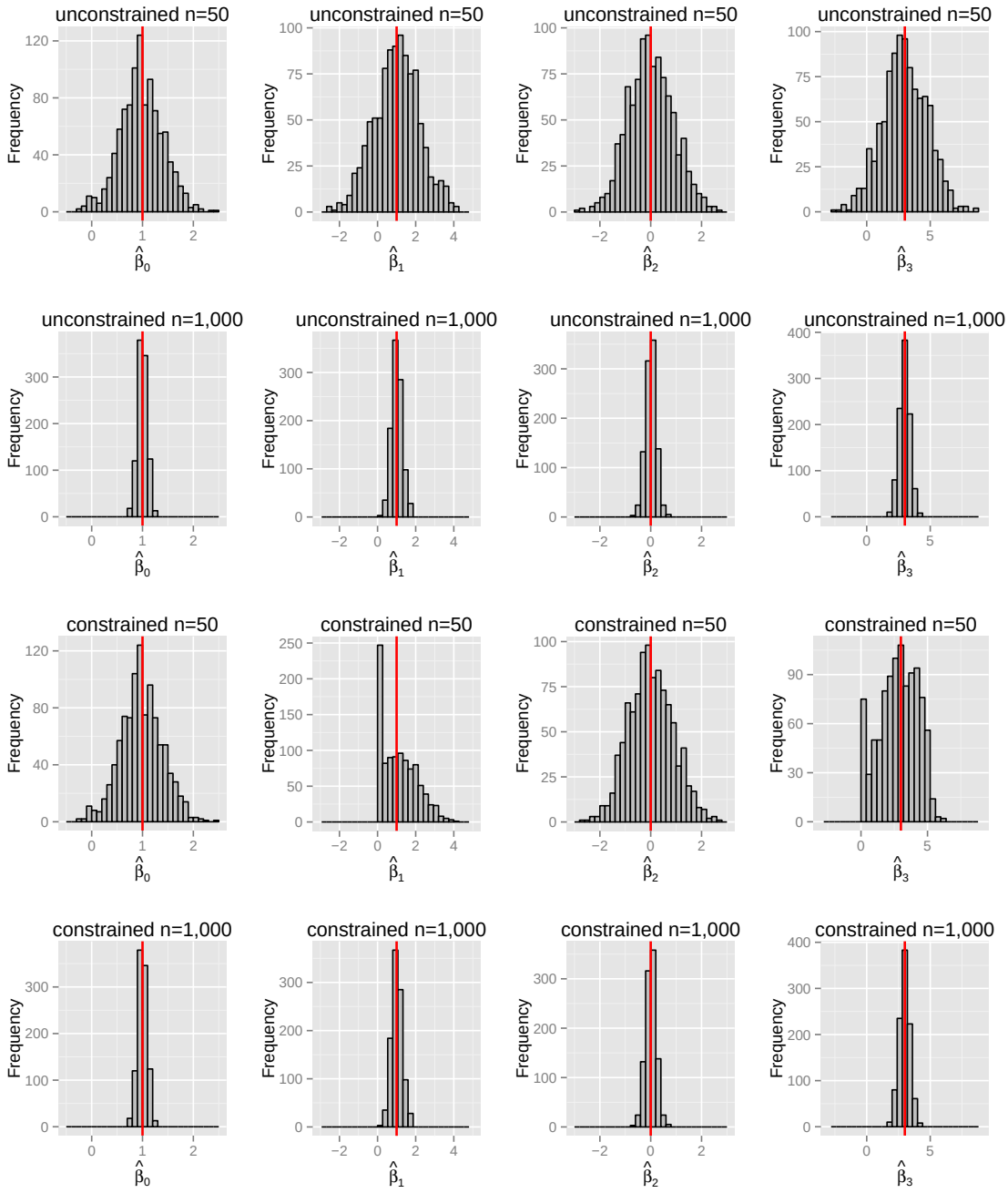

Figure 3.4: Standard unconstrained linear model fits and monotone polynomial fits to data generated from $p_2(x) = 3x^3 + x + 1$: Top two rows standard least squares estimates for $n = 50$ and $n = 1,000$ respectively; bottom two rows monotone polynomial estimates for $n = 50$ and $n = 1,000$ respectively. Red vertical lines indicate the location of the true parameter value.

> Given a random sample $X = (X_1, X_2, \ldots, X_n)$ with unknown probability distribution F, estimate the sampling distribution of some pre-specified random variable $R(X, F)$, on the basis of the observed data x.

In general a standard bootstrap procedure would involve attempting to say something about the parameter θ say, by resampling from the original data source used to estimate θ. The procedures themselves are wide and varied and have grown in popularity as computing power has become increasingly larger. For a more detailed discussion on general bootstrap methodology see for example either Efron & Tibshirani (1993) or Davison & Hinkley (1997).

3.5.1 *Bootstrap in regression*

Bootstrap techniques in linear regression situations have also been well developed over the years with most applications of the bootstrap to regression modelling scenarios utilising one of two approaches; either case resampling (non-parametric paired bootstrap) or model based resampling (residuals bootstrap). However, more recently attention has been given also to the weighted bootstrap (Barbe & Bertail, 1995). The selection of the specific bootstrap technique is usually made by consideration of the design matrix of predictors and how much faith we have in the underlying regression model (Efron & Tibshirani, 1993). However, regardless of the assumptions we make on the predictors, if we can reasonably assume that the residuals are identically distributed then the residuals bootstrap is generally more preferable. In situations like ours where we have monotonicity constraints imposed, or in more general situations, where there is some knowledge or evidence of heterogeneity, a more prudent approach may be to adopt the non-parametric bootstrap. We consider these three bootstrap techniques in the context of monotone polynomial fitting starting in the next section where we describe simple versions of the bootstrap algorithms we implement. We further investigate these and their appropriateness for constrained regression.

3.5.2 *Notation and bootstrap algorithms*

Following standard notation for Bootstrapping, for example, see Davison & Hinkley (1997), we define the following:

Let our observed data be denoted by (x_j, y_j), and our bootstrap resamples denoted by (x_j^*, y_j^*). The calculation of our bootstrap data is described in the following sections.

3.5.2.1 *Case resampling (paired bootstrap)*

The non-parametric (or paired) bootstrap, is a simple technique that is without parametric assumptions regarding the resampling distribution. In its simplest form we consider the set of paired observations (x, y) where x is an n by p matrix.

For polynomial regression models we sample pairs with replacement from the (x_j, y_j), which is achieved by taking a sample of size n, with replacement from $I = 1, 2, 3, \ldots, n$. We let (x_j^*, y_j^*) be a sample of the (x, y) pairs. For each of our bootstrap samples we estimate regression coefficients $\hat{\beta}_i^*$ using the monotone polynomial algorithms described in the previous chapter and Murray, Müller & Turlach (2013). Our bootstrap algorithm is adapted from Davison & Hinkley (1997):

1: Set $b = 1$,
2: **loop**
3: sample $i_1^*, \ldots, i_n^*$ with replacement from $1, 2, \ldots, n$
4: for $j = 1, \ldots, n$, set $x_j^* = x_{i_j^*}$ and $y_j^* = y_{i_j^*}$ and,
5: Fit polynomial regressions to $(x_1^*, y_1^*), \ldots, (x_n^*, y_n^*)$ giving regression parameter estimates $\hat{\beta}_{0,b}^*, \ldots, \hat{\beta}_{p,b}^*$
6: **if** $b = B$ **then**
7: terminate algorithm
8: **else**
9: Set $b = b + 1$
10: **end if**
11: **end loop**

We use this algorithm and many of the proposed variations of this throughout the remainder of this thesis. These include the m out of n bootstrap (Shao, 1994), the weighted bootstrap (see Barbe & Bertail (1995), Müller & Welsh (2005, 2009)), and the stratified bootstrap (see Müller & Welsh (2005, 2009)). The latter of these is seen as an important tool for situations where there is a small number of fixed design points to resample from, something we consider in more detail in our real world examples later in this thesis.

3.5.2.2 *Model based resampling (residuals bootstrap)*

In regression situations, the natural alternative to the non-parametric bootstrap, is the model based resampling technique or the residuals bootstrap. This method effectively fits the regression model to the original (x_j, y_j) pairs of observations, calculates the residuals from this model, then resamples from the residuals. The bootstrap algorithm we describe below is again adapted from Davison & Hinkley (1997):

1: Fit appropriate polynomial regression model to original $(x_1, y_1), \ldots, (x_n, y_n)$ data giving regression parameter estimates $\hat{\beta}_0, \ldots, \hat{\beta}_q$
2: Calculate residuals (e_j) for each set of n paired observations
3: Set $r = 1$,
4: **loop**
5: set $x_j^* = x_j$
6: randomly sample ϵ_j^* from $e_1, \ldots, e_n$
7: set $y_j^* = \hat{\beta} x_j + \epsilon_j^*$,
8: Fit polynomial regressions to $(x_1^*, y_1^*), \ldots, (x_n^*, y_n^*)$ giving regression parameter estimates $\hat{\beta}_{0,b}^*, \ldots, \hat{\beta}_{p,b}^*$
9: **if** $b = B$ **then**
10: terminate algorithm
11: **else**
12: Set $b = b + 1$
13: **end if**
14: **end loop**

A drawback of this technique is that it assumes a known distribution for the errors. For example, in standard polynomial regression models the errors are assumed to have a normal probability distribution. However, in some instances one is not able to specify a form on the error distribution and hence we either assume we know the distribution of the errors (not ideal for monotone polynomials) or we have to modify our approach.

One such modification proposed for a standard linear regression situation is based on the variance of the residuals, that is:

$$\text{Var}\,[e_j] = (1 - h_j)\sigma^2 \tag{3.5}$$

where h_j is the j^{th} element of the diagonal of the hat matrix (see Draper & Smith (1998) for standard ordinary least squares theory). In this instance it is obvious that as the leverage increases the variance of the residuals tends to zero. In light of this one needs to be careful when considering using the residuals bootstrap in a linear regression situation, or more importantly in our constrained polynomial regression situation. However, it is recommended, especially for small sample sizes that instead of resampling the raw residuals, one should resample the modified residuals (see Weber (1984); Davison & Hinkley (1997); Fox & Weisberg (2011) for more details). In summary, in order to take into account the leverages of the observations, which can be problematic especially in situations where there are one or two points of extremely high leverage and as a consequence have residuals with much lower variability, we can simply resample modified residuals $e_j/(1 - h_j)$ where e_j and h_j represent the residual and its corresponding 'hat matrix' value for the j^{th} observation.

However, we should note at this point that whilst this adjustment has been shown to be good for small sample regression situations or situations where there are instances of high leverage, it is dependent on being able to calculate the hat-matrix. In many other situations, for example non-parametric regression, or many shape constrained regression examples (including the fitting of monotone polynomials) such a quantity is not readily defined or available. Consequently, when considering the adjustment of residuals, the quantity by which to make the adjustment is itself ill defined.

3.5.2.3 *The weighted bootstrap*

Finally in this section we provide an algorithm for the weighted bootstrap:

1: Set $b = 1$,
2: **loop**
3: sample $w_1^*, \ldots, w_n^*$ with replacement from a known distribution (we will assume a standard exponential distribution)
4: for $j = 1, \ldots, n$, set $x_j^* = x_j$ and $y_j^* = y_j$ and,
5: Fit weighted polynomial regressions to $(x_1^*, y_1^*), \ldots, (x_n^*, y_n^*)$ giving regression parameter estimates $\hat{\beta}_{0,b}^*, \ldots, \hat{\beta}_{p,b}^*$
6: **if** $b = B$ **then**
7: terminate algorithm
8: **else**
9: Set $b = b + 1$
10: **end if**
11: **end loop**

Simply put, this algorithm uses the same (x_i, y_i) pairs in all iterations of the algorithm, which itself is the original data. However, each time a randomly selected weight is attached to each of our n pairs.

3.6 BOOTSTRAPPING - NUMERICAL EXAMPLES

In this section we discuss, via numerical examples, bootstrap methods for calculating variance estimates and point wise confidence and prediction intervals for monotone polynomials. We will demonstrate for confidence intervals, that standard bootstrap methodology typically leads to under coverage in certain regions and show that these areas of under coverage appear to be related to the curvature of the underlying regression function. We also demonstrate the same problem is not observed when considering prediction intervals. To address this problem of under coverage for confidence intervals we consider two solutions: first, we investigate an m out of n bootstrap which results in similar patterns of coverage probabilities to standard bootstrap methodology,

but yields more conservative confidence bands; second, by making confidence bands symmetric, we show we can ensure the coverage probabilities are more uniform over the range of the regressor variable. We summarise our findings from our bootstrap simulation exercises as follows:

1. The method of bootstrap (paired, residual, weighted) has minimal impact on the parameter estimates and associated variability.

2. If the underlying data generating function is on the boundary of $\mathcal{C}^q_{\mathcal{R}}$ then:
   - At both small and large sample sizes there are significant differences between unconstrained least squares fitting and monotone polynomial fitting with respect to the estimates and standard errors from the bootstrapping.
   - These differences are marginally more pronounced with data that would provide a non-monotone fit when standard unconstrained least squares estimation is used.

3. If the underlying data generating function is *not* on the boundary of $\mathcal{C}^q_{\mathcal{R}}$ then:
   - At large sample sizes the underlying fit of an unconstrained polynomial is almost always going to be monotone and consequently there is little (if any) differences between the bootstrap estimates from standard unconstrained least squares fitting and monotone polynomial fitting.
   - At small sample sizes differences are observed between standard least squares unconstrained fitting and monotone polynomial fitting, and are more pronounced when the data sampled provides an underlying standard least squares unconstrained polynomial fit which is *not* monotone.

Given we very rarely know the nature of the underlying phenomenon in practice, one can argue that at small sample sizes there is sufficient difference between the constrained and unconstrained fits to consider further alternative methodologies for bootstrapping monotone polynomials, and that in practice a good check would be whether or not the underlying fit is on the boundary. Furthermore, we note that as a consequence, these differences will carry through to confidence intervals and prediction intervals. The extent to which this is apparent is discussed in Section 3.7. However,

first we provide a more detailed commentary on what led us to the aforementioned summary of bootstrapping in regression in Section 3.6.1.

3.6.1 *Estimating the variance of regression parameters*

We start by providing bootstrap estimates for the regression parameters and standard errors for these under various scenarios. For the three polynomial regression models $(p_1(x), p_2(x), p_3(x))$ described previously, at two sample sizes $n = 50$ and $n = 1,000$, for two selected different example data sets from our generated data in each case (one that gives a standard unconstrained least squares fit that is monotone and one that gives a standard unconstrained least squares fit that is not monotone) we estimate parameters and variances of these estimates using the residuals bootstrap, the adjusted residual bootstrap, the non-parametric (paired) bootstrap and the weighted bootstrap, for both standard unconstrained linear model fitting and monotone polynomial model fitting. We use the standard definition of the bootstrap parameter estimates from the polynomial regression models as:

$$\hat{\beta}_{bs,i} = B^{-1} \sum_{j=1}^{B} \hat{\beta}_i^{*[j]}, \tag{3.6}$$

where $B = 1,000$ is the number of Monte-Carlo simulations, β_is are the polynomial regression parameters and the $\hat{\beta}_{bs,i}$s their corresponding bootstrap estimates. The associated standard errors are given by:

$$SE(\hat{\beta}_{bs,i}) = \sqrt{\frac{1}{B-1} \sum_{j=1}^{B} (\hat{\beta}_i^{*[j]} - \hat{\beta}_{bs,i}^{*})^2} \tag{3.7}$$

3.6.1.1 *Results comparing different bootstrap methodologies*

The four different bootstrap approaches considered provide very similar estimates of the polynomial regression parameters and their corresponding standard errors. We

can succinctly summarise these results by noting that regardless of the underlying data generating function $(p_1(x), p_2(x), p_3(x))$, the sample size, the method of fitting the polynomial (constrained or unconstrained) or whether the underlying data would provide a monotone fit should a standard linear model be used to the raw data, the bootstrap methodologies provide similar results. All results comparing the various bootstrap methodologies are shown in Appendix B.

3.6.1.2 *Comparing the results from standard bootstrap methods*

Given that the bootstrap estimates from the different techniques are similar in nature, for the remainder of this discussion we will only consider the paired bootstrap estimates and standard errors for comparing further scenarios. When examining these estimates and standard errors for $p_1(x)$ we note the following: For the relatively small sample size ($n = 50$), the difference in the bootstrap parameter estimates and variability associated with these, appears to be dependent on two issues. First, the method of fit and second, whether the data would provide a monotone fit with unconstrained least squares regression. Consider the top two rows in Figure 3.5 showing the kernel density plots for the different estimation methods of the regression parameters. The top row are bootstrap estimates when the data set used provides a monotone fit using standard unconstrained least squares estimation. The second row are comparable estimates for a second alternative data set which yields a non-montone fit to the data with unconstrained least squares estimation. We see that there are substantial differences in the distributions of $\hat{\beta}_1$ and $\hat{\beta}_3$ between constrained and unconstrained fits regardless of the data set. However, there are also subtle differences in the other two parameter estimates when the unconstrained fit to the data is non-monotone suggesting these estimates are more affected in such instances. Furthermore, when we consider increasing the sample size to $1,000$ (see Figure 3.5 row three and row four for unconstrained monotone fit and unconstrained non-montone fit respectively), we note there are still similar differences in the empirical distributions of these parameter estimates but these are now much more pronounced in the data where the unconstrained fit is not monotone (row four) as opposed to the unconstrained fit being monotone (row three). We note that these findings seem to suggest that if the data generating function is on

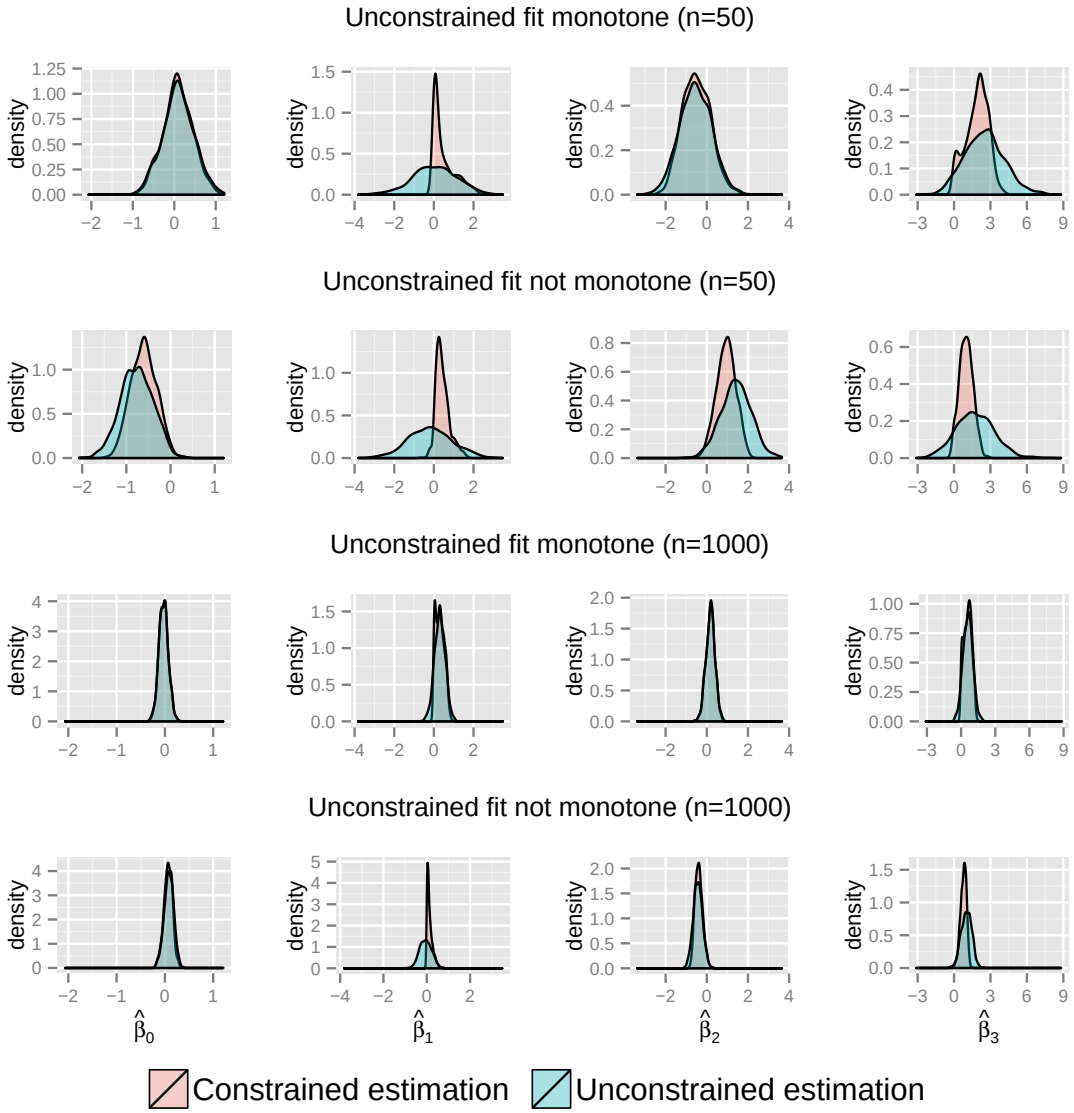

Figure 3.5: Results of non-parametric bootstrap using two separate data sets (unconstrained least squares fit monotone rows; unconstrained least squares fit not monotone) at two different sample sizes (n=50 and n=1,000) generated from the underlying polynomial function $p_1(x) = x^3$. Overlaid kernel density estimates in each plots are for bootstrap estimates of regression parameters using standard unconstrained least squares estimation (blue) and using monotone polynomial estimation (pink).

the boundary of $\mathcal{C}^q_{\mathcal{R}}$ (as $p_1(x)$ is), then increasing the sample size does not provide comparable results between standard least squares bootstrap parameter estimates and standard errors and those produced from monotone polynomial bootstrap estimation, in particular when the data under consideration was variable enough to yield a fitted polynomial that is not monotone.

Conversely when we focus on $p_2(x)$ (see Figure 3.6), which itself does not lie on the boundary of $\mathcal{C}^q_{\mathcal{R}}$, an increase in sample size has two consequences. First, with supporting evidence from our simulations, we note that a non-monotone fit using standard least squares constrained estimation is highly unlikely. In our $1,000$ simulated data sets with $n = 1,000$, not one of them provided a non-monotone fit. As a consequence we conclude that when the underlying function is not on the boundary then, with a sufficiently large sample size, the standard unconstrained least squares estimates will provide the same results as the constrained monotone polynomial estimates. However, if the sample size is small enough then there are situations in where even though the underlying data generating function is not on the boundary of $\mathcal{C}^q_{\mathcal{R}}$, the data generated will provide fits which are not monotone using unconstrained least squares estimation. In these instances there are substantial differences between the bootstrap parameter estimates using monotone polynomial estimation and using standard least squares unconstrained estimation, as can be observed by row two of Figure 3.6. These can also be seen in more details in Appendix B.

We should note at this point that whilst we have presented results from only two data sets (monotone unconstrained fit versus non-monotone unconstrained fit) in each situation, further simulations have served to reinforce the results of the simulations presented.

Finally, given the nature of $p_3(x)$, in that this function is itself on the boundary of the cone of monotone polynomials, it is not surprising that the results and conclusions reached when we consider simulations involving $p_3(x)$ are similar to those seen for $p_1(x)$.

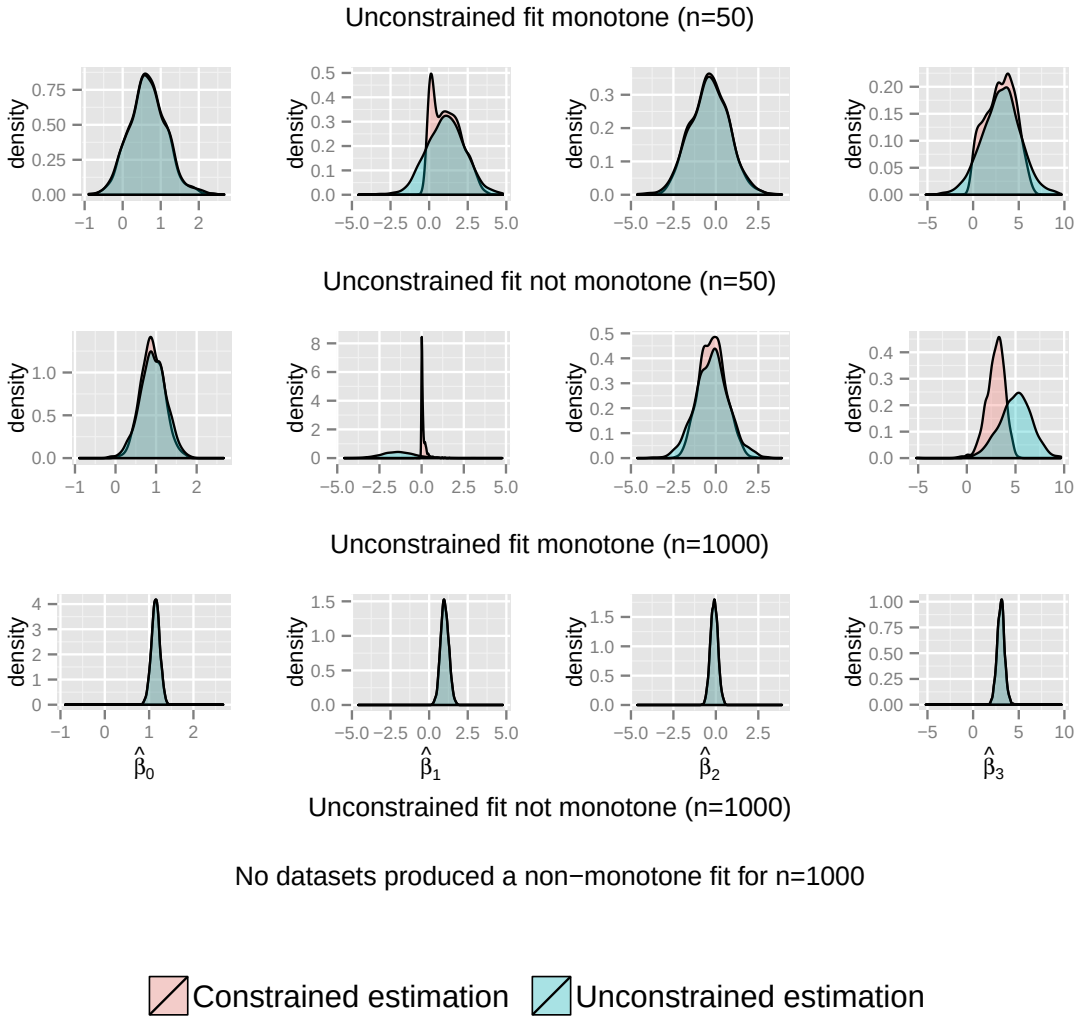

Figure 3.6: Results of non-parametric bootstrap using two separate data sets (unconstrained least squares fit monotone rows; unconstrained least squares fit not monotone) at two different sample sizes (n=50 and n=1,000) generated from the underlying polynomial function $p_2(x) = 3x^3 + x + 1$. Overlaid kernel density estimates in each plots are for bootstrap estimates of regression parameters using standard unconstrained least squares estimation (blue) and using monotone polynomial estimation (pink).

3.7 BOOTSTRAP CONFIDENCE INTERVALS AND PREDICTION INTERVALS

In standard least squares regression we can calculate bootstrap confidence intervals for the mean value of the regression function by applying any of the bootstrap methods outlined in the previous sections. We estimate the mean value for each bootstrap sample evaluated at a specific design point, for instance x_0, and can then estimate bootstrap confidence intervals by extracting the appropriate quantiles of these bootstrap estimates.

For a given future observation or set of observations it is often useful to be able to predict the value of the response for a given predictor. In this instance we often find it useful to present a *prediction interval* for the predicted value of the response. Existing methodology makes use of the residuals bootstrap. We describe here the method proposed by Stine (1985) for prediction intervals in a regression model, acknowledging the large sample theory for prediction intervals for the general regression set up described in Olive (2007).

We take the same approach as that of calculating confidence intervals using the residuals bootstrap in that we define the (x_i, y_i) pairs of observations and generate $\hat{e}_i$ by fitting a standard regression model. We then resample with replacement from the residuals to create our bootstrap value y_i^* and at the same time we create y_f^*, the future predicted values of the response for a given set of f further values. We then regress y_i^* on x_i and generate our $\hat{\beta}^*$ the bootstrap sample regression coefficients. We then generate $D_{f,b}^* = y_f^* - x_f^T \hat{\beta}^*$. Repeating this B times we then use the empirical distribution of the Ds to give our bootstrap prediction interval $\left(\hat{y} + D_f^*[\frac{1-\alpha}{2}], \hat{y} + D_f^*[\frac{1+\alpha}{2}]\right)$.

3.7.1 *Confidence bands for polynomials using standard least squares estimation*

In fitting monotone polynomials to data, our major focus mainly has been on finding the fitted value of the response for a given x value. We used simulated data from the three data generating functions ($p_1(x)$, $p_2(x)$, $p_3(x)$) described earlier to investigate the effectiveness of monotone polynomials for this purpose. For least squares estimation

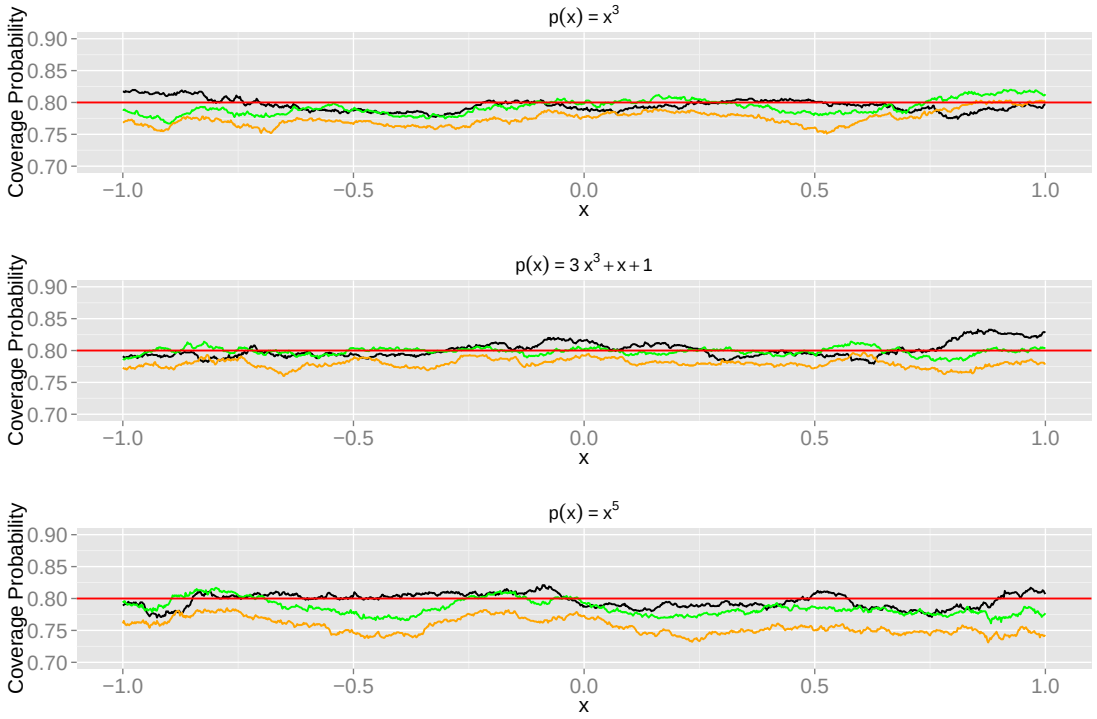

Figure 3.7: Coverage probabilities for 80% confidence intervals for three different polynomial functions using standard unconstrained least squares estimation. Orange lines residuals bootstrap n=50; green lines adjusted residuals bootstrap n=50; black lines residuals bootstrap n=1000

without monotonicity constraints, the (adjusted) residuals bootstrap should provide consistent estimates for confidence intervals for our mean values. Figure 3.7 describes, for the three different models considered, the coverage probabilities when 80% confidence intervals are estimated in this fashion.

As expected the coverage probabilities are reasonably flat for all three functions using the two different sample sizes, regardless of whether the adjusted residual bootstrap or the straight residual bootstrap is chosen. Table 3.3 describes the mean coverage probabilities over the range of x values for each of the approaches.

We see that for each of these, two observations are apparent. First, with small samples size ($n = 50$) there is under coverage for each function, with the mean percent coverage of 77.50, 77.90 and 75.67, for $p_1(x)$, $p_2(x)$ and $p_3(x)$ respectively. This apparent under coverage can be rectified in one of two ways. First, we could increase the sample size (for example to $n = 1,000$ as we have done), which gives mean percent coverage of 79.53, 80.10 and 79.67 for the three functions, much closer to the 80%

Table 3.3: Mean coverage probabilities over the range of x for 80% confidence intervals, when the data generating polynomial is $p_k(x)$, $k = 1, 2, 3$, and the standard least squares estimator is used to fit unconstrained polynomials.

n and method	$p_1(x) = x^3$	$p_2(x) = 3x^3 + x + 1$	$p_3(x) = x^5$
50 residuals	77.50	77.98	75.67
50 residuals adjusted	79.28	79.88	78.57
1000 residuals	79.53	80.10	79.67

nominal value. Second, a more suitable approach when large sample sizes cannot be attained in practice, is to adjust the residuals by their corresponding diagonal entry of the hat matrix (or the leverage), which itself has the impact of increasing the mean percent coverage in a similar fashion to increasing the sample size, giving mean percent coverage of 79.28, 79.88 and 78.57 for the three functions, again in close proximity to the nominal 80%.

These results are in line with bootstrap theory for confidence interval estimation in standard linear regression; for a discussion we refer to Hall (1989). The impact on the confidence bands produced when the fit is constrained to be monotone for these three functions is examined later in this chapter, however we note that the simple simulation results in this section suggests these approaches are applicable for unconstrained polynomial regression modelling of our three functions $p_1(x)$, $p_2(x)$ and $p_3(x)$.

3.7.2 *Confidence bands and prediction intervals for monotone polynomials*

Due to the complex nature of the isotonic parameterisations, we estimate (point wise) confidence bands using bootstrap methodology. We present the results for the model $p_3(x)$, with $n = 50$ and $n = 1,000$, noting that similar results are observed for the other functions that lie on the boundary of the cone of monotone polynomials (for example $p_1(x)$). We use a non-parametric (paired) bootstrap approach, with $1,000$ bootstrap replications, recalling that the different bootstrap approaches appear to have minimal impact on the estimates or standard errors. An example of the resulting confidence bands is shown in the bottom left panel of Figure 3.9. We note the asymmetric nature of these bands and observe that this asymmetry is more pronounced in certain regions

as demonstrated in the bottom right panel of Figure 3.9. Using either of these bootstrap methods the confidence bands are monotonic increasing and, as the top left panel of Figure 3.9 demonstrates, exhibit poor coverage probabilities in some regions. In the next section we shall argue that estimates of these bands should be based on a methodology that allows these bands to be non-monotone.

In addition to the investigation of confidence bands, we also studied and adapted the approach of Stine (1985) to use the bootstrap to calculate prediction intervals for monotone polynomials. In our extensive numerical experiments we found that, in contrast to the confidence bands described earlier, these prediction bands work equally well for polynomials lying inside the cone of monotone polynomials of order at most q as for polynomials on the boundary of the cone. The top two panels of Figure 3.10 show, that in our simulated data examples $p_1(x)$ and $p_3(x)$, the prediction intervals are much more symmetric than the confidence intervals. We demonstrate in the next section the impact this has on the coverage probabilities.

3.7.3 *Coverage probabilities*

To examine the effectiveness of the bootstrapping, we estimate coverage probabilities extracting the 10^{th} and 90^{th} percentiles of the $1,000$ bootstrap replications for each of $1,000$ simulation runs, for both confidence intervals and prediction intervals. We describe coverage probability plots where in general our aim is to produce upper and lower limits that contain within them the true model $100(1-\alpha)\%$ of the time, where $(1-\alpha)$ is the nominal confidence level. Generally speaking for some function, f, say, we require:

$$P(L_l < f < U_l) = 1 - \alpha, \tag{3.8}$$

where L_l and U_l are our upper and lower limits respectively.

Figure 3.8 shows the coverage probabilities for $p_2(x) = 3x^3 + x + 1$ (top panel) and $p_3(x) = x^5$ (bottom panel), using both standard least squares estimation (left panel) and monotone polynomial fitting (right panel). We note that with standard least

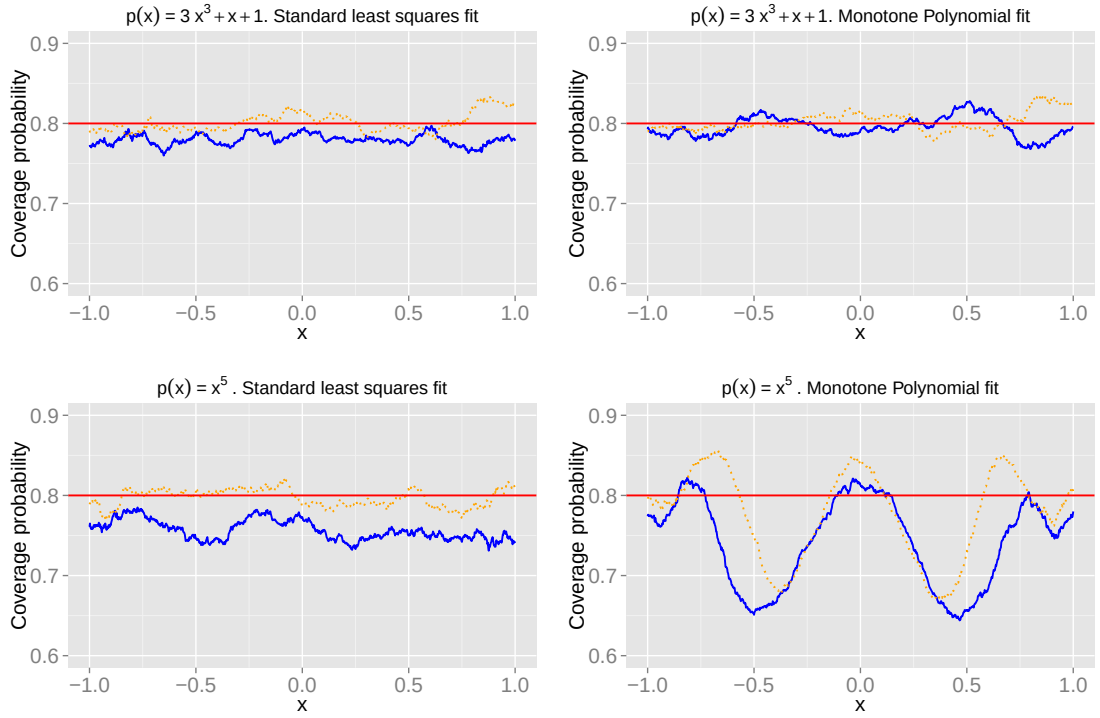

Figure 3.8: Coverage probabilities for 80% confidence intervals for the cubic $p_1(x) = 3x^3 + x + 1$ and the quintic $p_3(x) = x^5$, for sample sizes $n = 50$ (blue solid line) and $n = 1,000$ (orange dashed line). Left panel – Standard least squares fits using paired bootstrap. Right panel – Monotone polynomial fits using paired bootstrap.

squares fitting the coverage probabilities are relatively uniform over the range of the regressor variable and there is under-coverage in the smaller sample size ($n = 50$, solid blue line) as opposed to the near nominal coverage for $n = 1,000$ (orange dashed line), as expected. However, when these models are fit using monotone polynomials we make two observations. First, for $p_2(x)$, the coverage probabilities are near the nominal level for both sample sizes. However, when we look at the coverage probabilities for $p_3(x)$ we see a distinct pattern with undulations around $x = -0.5$ and $x = 0.5$ depicting areas of extreme under-coverage (sometimes as low as 0.65). Second, we note that increasing the sample size does not impact on this to any great extent. From this it is evident that straight forward off the shelf bootstrap methodology alone is not suitable for monotone polynomials in all situations, and is particularly poor when the underlying regression function is on the boundary of $\mathcal{C}^q_{\mathcal{R}}$. Furthermore, and upon examination of alternative regression functions, we suggest that this observed under coverage, us-

ing bootstrap methodologies, is not restricted to the specific degree of polynomial nor the sample size.

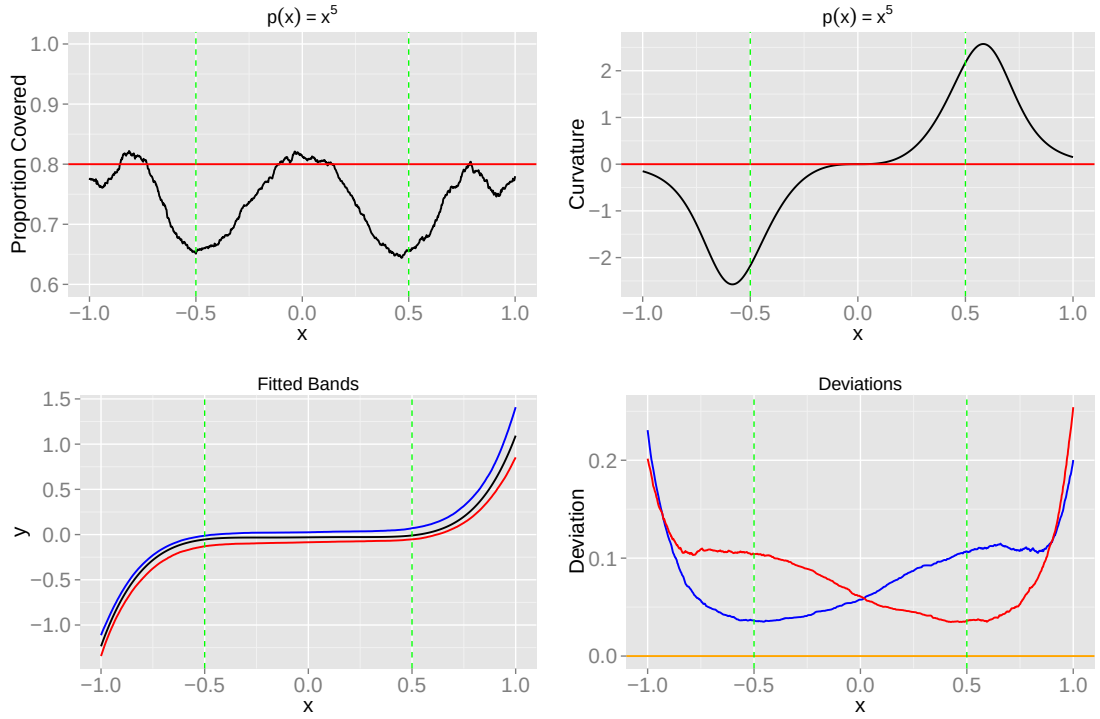

Figure 3.9: Non-parametric bootstrap results using the underlying polynomial function $p_3(x) = x^5$. Top left: Coverage probabilities for 80% confidence interval; Top right curvature for underlying function; Bottom left: example bootstrap confidence upper and lower limits (blue and red respectively); Bottom right: deviations from fitted curve for upper and lower limits (blue and red respectively)

However, when considering prediction intervals, it would appear that using the bootstrap methodology described by Stine (1985) we can achieve nominated coverage (or close to) with all of our simulated examples. Presumably the extra variance term that appears in the calculation of prediction intervals dominates the width of these intervals and, consequently, close to nominal coverage is also obtained for polynomials on the boundaries, for which the usual asymptotics with respect to asymptotic distribution and/or confidence intervals do not apply. This is demonstrated in Figure 3.10 where we note for $p_1(x)$ and $p_3(x)$ the relatively uniform coverage probabilities from prediction intervals over the whole range of x values regardless of the sample size (in contrast to what was previously observed for the confidence intervals). We also note that as the sample size increases these coverage probabilities become more similar to the nominal coverage (red line) with the green lines in both plots denoting coverage

probabilities for sample size $n = 50$ being slightly lower than the blue line $n = 1,000$. We also note that the asymmetric nature of the confidence bands described previously is not as apparent in the prediction bands with these much more balanced for all x.

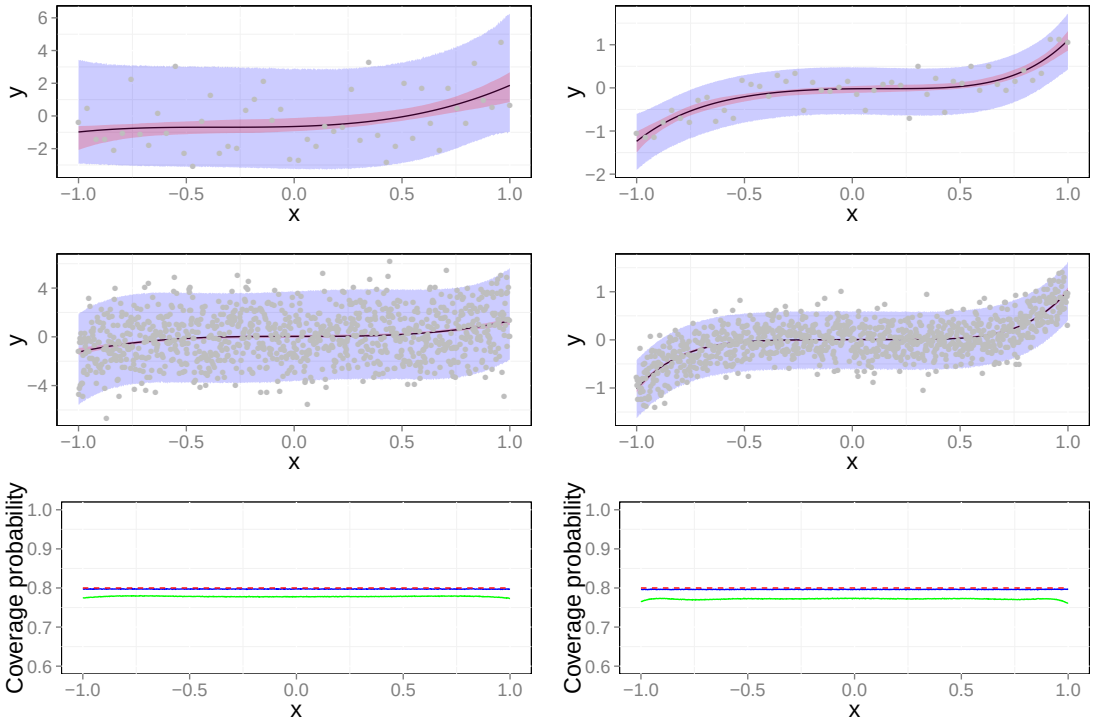

Figure 3.10: Prediction interval plots for: left panel $p_1(x) = x^3$, and right panel $p_3(x) = x^5$. Top row fitted curves and 80% confidence and prediction intervals for $n = 50$. Middle row fitted curves and 80% confidence and prediction intervals for $n = 1,000$. Bottom row converge probabilities for $n = 50$ (green line) and $n = 1,000$ (orange line) with 80% nominal coverage (red line). Note the scale on the coverage probability plots is deliberately set to be compared to the previously described coverage probability confidence interval plots in Figure 3.8.

3.7.3.1 *Using the m out of n bootstrap for confidence bands*

In our numerical examples, adopting an m out of n bootstrap, as described by Shao (1994), rectifies this under coverage observed in the point wise confidence intervals. Figure 3.11 shows coverage probabilities for 80% confidence intervals based on the m out of n bootstrap, varying m over 10, 15, 20, 25, 30, 35, 40 and 50 for $n = 50$, as well as for the n out of n bootstrap for $n = 1,000$.

The right panel of Figure 3.11 shows coverage probabilities for 80% confidence intervals based on the 'm out of n' bootstrap, varying m over 10, 15, 20, 25, 30, 35, 40 and 50 for $n = 50$, whilst the left panel shows the coverage probabilities for the n out of

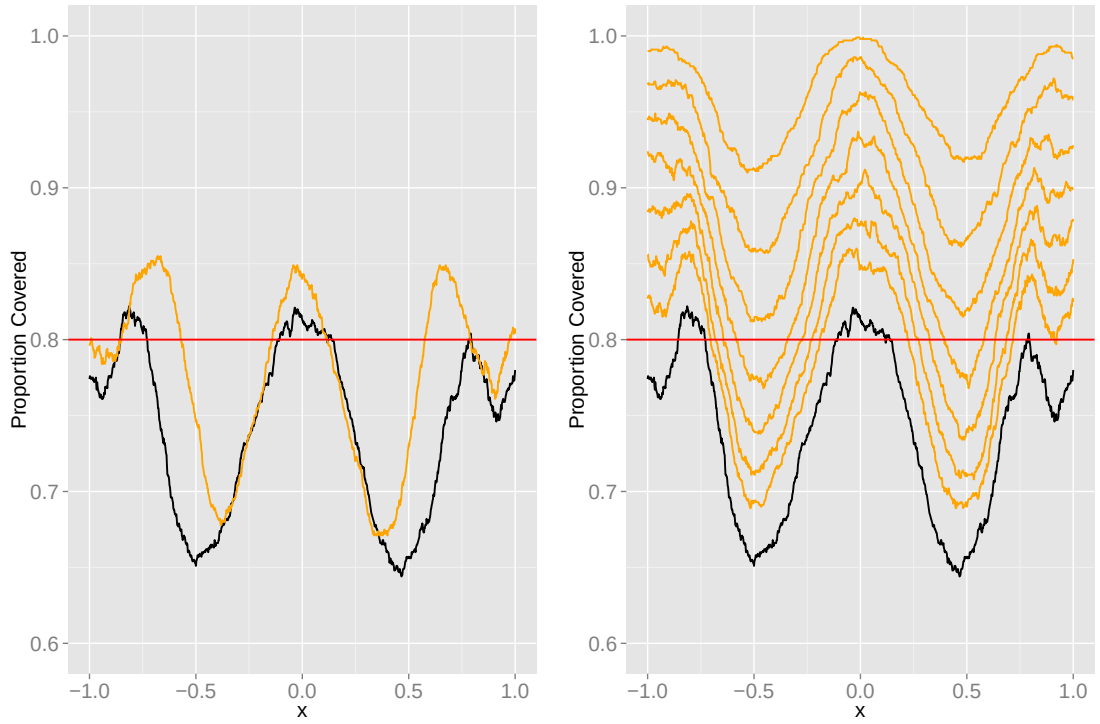

Figure 3.11: Coverage probabilities for 80% confidence intervals for the quintic monotone polynomial $p_3(x) = x^5$. Left panel – n out of n residuals bootstrap: Black line $n = 50$; orange line $n = 1,000$. Right panel – m out of n bootstrap: decreasing values of m from bottom (black) line $m = n$ through $m = 40, 35, 30, 25, 20, 15, 10$ for $n = 50$ design points.

n bootstrap for $n = 50$ and $n = 1,000$. We note that using an 'm out of n' bootstrap gives more acceptable results in terms of coverage probabilities for an astute choice of m. In this example we observe that $m = 35$, which is relatively close to n, achieves an average coverage probability that is close to the nominal level, whereas for small choices of m the resulting confidence bands are very conservative. We found this to be true for a range of other examples and depending on m there are three possibilities in general: we either choose m too large and get under coverage; we can choose m too small so that the coverage probabilities on the whole exceed the nominated confidence level, hence we get substantial over coverage in some areas; or we select m approximately optimal to obtain average coverage over the range to be at the nominated confidence level, which results in some areas with under coverage and some areas with over coverage. However, we note that a data driven choice of m remains an open research question. Table 3.4 summarises coverage probabilities for varying size m. In particular we note that using $m = 35$ gives the best overall mean coverage probabilities.

Table 3.4: Summaries of coverage probabilities from different m out of n bootstrapping algorithms and post hoc adjustments. Signed area between curves (SABC) denotes the signed area of the difference of the estimated coverage probability curve and the nominal coverage line.

Algorithm	Mean	Median	Min	Max	Mean deviation	Median deviation	SABC	% above 0.8
$m = 10$	0.961	0.966	0.910	0.999	0.161	0.166	0.321	1.000
$m = 15$	0.924	0.929	0.857	0.986	0.124	0.129	0.249	1.000
$m = 20$	0.890	0.900	0.812	0.963	0.090	0.100	0.180	1.000
$m = 25$	0.859	0.872	0.768	0.937	0.066	0.072	0.118	0.784
$m = 30$	0.831	0.849	0.734	0.912	0.057	0.060	0.061	0.660
$m = 35$	0.804	0.820	0.708	0.880	0.052	0.055	0.008	0.564
$m = 40$	0.782	0.798	0.689	0.860	0.049	0.045	-0.037	0.490
$m = n$ (No Adj.)	0.741	0.757	0.644	0.822	0.063	0.043	-0.118	0.185
Curv Adj.	0.840	0.840	0.759	0.887	0.043	0.040	0.081	0.909

However, using a criteria such as minimising mean or median absolute deviation may suggest an alternative choice for m.

We further observe from the right panel of Figure 3.11 that regardless of the magnitude of m the coverage probabilities are not consistent for all x. To examine this in more detail we propose the idea of a post hoc adjustment. Our conjecture is that this under coverage results from the asymmetric nature of the confidence bands arising from the monotonicity constraint, and that the resulting under coverage is particularly pronounced in areas of high curvature.

3.7.3.2 *Post hoc adjustments to confidence bands*

In order to rectify the problem of non-uniform coverage, we consider a post hoc adjustment to our estimated confidence bands and illustrate it for $p_3(x)$. We propose a simple adjustment using an n out of n bootstrap; specifically we make these bands symmetric around $\hat{p}(x)$. Let $\delta_{u,x}$ and $\delta_{l,x}$ be the upper and lower deviations from $\hat{p}(x)$ for a given x. We define an adjusted confidence band for a given x to be $[\hat{p}(x) - \max\{\delta_{u,x}, \delta_{l,x}\}, \hat{p}(x) + \max\{\delta_{u,x}, \delta_{l,x}\}]$. Such an adjustment is illustrated in Figure 3.12 with the left panel showing the fitted confidence band for an example simulation realisation, and the right panel showing the new coverage probabilities after this symmetrisation of the confidence bands over the 1,000 simulation runs. We ob-

serve that the apparent under coverage initially observed has now been removed by this adjustment and the coverage is much more uniform over the whole range of x. Furthermore, our numerical experiments using other functions, which would initially suffer from under-coverage issues in specific areas, with varying degree polynomials, show similar positive results, suggesting such a post hoc adjustment could be beneficial for monotone polynomial bootstrap confidence bands.

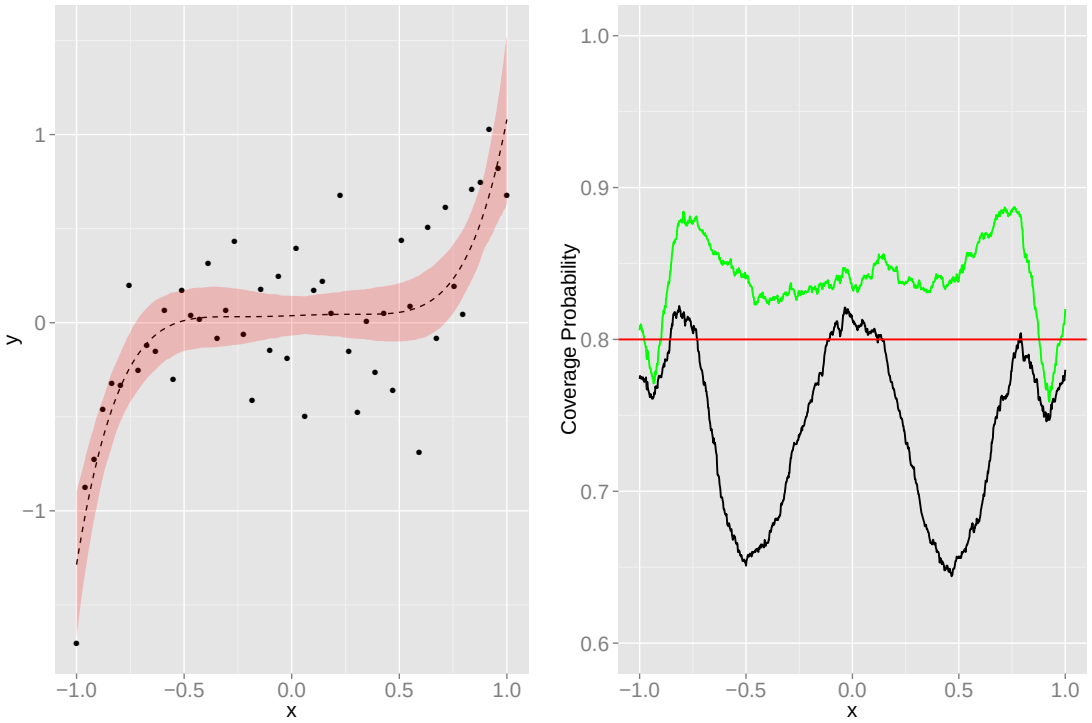

Figure 3.12: Left panel: Fitted monotone polynomial based on simulated data from underlying function $p_2(x) = x^5$ with adjusted 80% and 95% bootstrap confidence bands. Right panel: Coverage probabilities for adjusted 80% bootstrap confidence bands on monotone polynomials based on the underlying function $p_3(x) = x^5$; adjusted probabilities (green), unadjusted (black).

Finally, we observe from Figure 3.9 that, there appears to be some correlation between the areas of under coverage and the underlying curvature of the data generating function, which we defined here by:

$$\kappa(x) = \frac{p''(x)}{(1 + p'(x)^2)^{\frac{3}{2}}} \tag{3.9}$$

This relationship suggests areas of high curvature are reflected in the areas of under coverage. This phenomenon exists in other functions, as can be seen in Figure C.1 in Appendix C, which shows the coverage probabilities and corresponding curvature for $p_4(x) = (x - 0.5)^5$, where $Y_i = p_4(x_i) + \epsilon_i$ with 50 equidistant design points and the errors independent $N(0, 0.3^2)$ as described previously, and $m = n$. Furthermore, additional numerical experiments using other functions, with varying degree polynomials, show similar positive results, to suggest such a curvature adjustment could be beneficial for monotone polynomial bootstrap confidence bands. We speculate that such curvature adjustments would also have similar potential in non-parametric and semi-parametric regression settings. However, we also note that whilst in simulation exercises we can demonstrate success by making such curvature adjustments, in practice the underlying curvature is unknown and would have to be estimated making such an adjustment problematic.

3.8 CONCLUSIONS

The work described in Chapter 2 and in Murray, Müller & Turlach (2013, 2015) has made fitting monotone polynomials to data much more accessible, with various different formulations and parameterisations now available through the R package `MonoPoly`. We have further demonstrated, through our simulations in this chapter, that there is not only a need for such methodology but a necessity in some instances where standard polynomials do not provide intuitively correct estimated functions, for example ensuring monotonicity. We have also provided theoretical and empirical consistency results for monotone polynomials and have argued that even if the underlying function is not a polynomial then our methods should provide results that are consistent.

To date, and to our knowledge, there has been little work carried out on variance estimation for monotone polynomials. We have shown through Monte-Carlo simulations that estimates and confidence intervals for monotone polynomials may require different methodologies depending on the location of the function in reference to the cone of monotone polynomials (on the boundary or not), the sample size and whether the initial unconstrained least squares solution is on the boundary. Furthermore, we have

demonstrated that, in some instances, standard unconstrained least squares bootstrap methodology may be satisfactory when estimating variances and functions thereof for monotone polynomials. However, in other instances these methods will not be suitable and it is extremely difficult to define instances in which this would be the case.

Consequently, we considered various different methods for confidence and prediction bands and have further proposed methodology for estimating these advocating one of two approaches; either using an m out of n bootstrap, or by employing a post hoc curvature adjustment. Initial testing of such approaches has yielded some encouraging results with the m out of n bootstrap providing an average coverage probability over the range of x at the nominated level, if one selects the right choice of tuning parameter. However, the post-hoc adjustment appears to have practical promise, given its ability to ensure the coverage probabilities are more uniform, and it is apparent such adjustments could be extended to techniques in both non-parametric and semi-parametric regression. Details of the mathematical properties of such adjustment remain to the best of our knowledge an open research question.

4

A COMPARISON OF MONOTONE REGRESSION TECHNIQUES

SUMMARY

We examine the effectiveness of monotone polynomials when compared to two of the more commonly used non-parametric smoothing techniques. We show through Monte-Carlo simulations that the bias and variance produced from monotone polynomial curve fitting are at least comparable to those from the non-parametric smoothing techniques, and for certain functions monotone polynomials provide better results. This is demonstrated not only for data which are generated from polynomial functions but also for those that originate from a sigmoidal or trigonometric function.

Speed comparisons of existing methods available to fit monotone polynomials to data demonstrate that the sums of squares formulation is much faster than previous monotone formulations and comparable to the semi-infinite programming approach of Hawkins (1994), with the sums of squares formulation proving much more flexibility. When compared to smoothing spline approaches we demonstrate that monotone polynomial model fitting performs comparatively well for small sample sizes, and is more effective with smaller run times when the sample sizes are larger. We argue that the efficiency of our methodology is pertinent to the ability to use computationally intensive techniques such as the bootstrap.

4.1 INTRODUCTION

The fitting of a monotone curve to data is increasingly important in modern statistics. Whilst we have described to date the impact and flexibility of using monotone polynomials, we recognise that there is a huge body of literature that would recommend smoothing splines, kernel smoothing, non-parametric or semi-parametric techniques.

As indicated in the introduction of this thesis it is not our desire to demonstrate that such techniques are not useful or necessary. However, given the attractiveness and simplicity of monotone polynomials we aim to describe and demonstrate these models as an effective and viable alternative. In doing this we do recognise there is a desire for people to want to make such comparisons between monotone polynomials and some of the other monotone regression techniques. Hence, in this chapter, we demonstrate with both polynomial functions and with non-polynomial functions, the effectiveness of using monotone polynomials in comparison to some of the other more commonly used techniques.

In Chapter 2 (and in Murray, Müller & Turlach (2013, 2015)) we revisited the idea of using monotone polynomials and provided algorithms for the fitting of monotone polynomials to data, initially based on the isotonic parameterisations of Elphinstone (1983) and the semi-infinite programming algorithm by Hawkins (1994), and subsequently using a sums of squares parameterisation.

In this chapter we compare the fitting of monotone polynomials to data with some of the more frequently used practical approaches to curve fitting. We further discuss these comparisons in more detail in Chapter 6 where we examine comparisons of monotone polynomials to non-parametric approaches with respect to the specific task of identifying the number and location of inflection points. For the remainder of this section we compare monotone polynomials to the smoothing spline approach using B-spline basis functions as described in Ramsay (1998) and Ramsay & Silverman (2002, 2006) in addition to the approach described in Meyer (2008, 2012), using the R packages created to implement these methodologies, `fda` (Ramsay, Wickham, Graves *et al.*, 2013) and `cgam` (Meyer & Liao, 2014) respectively. The approach of Meyer uses constrained generalized additive model (CGAM) fitting using maximum likelihood estimation, where shape or order restrictions can be imposed on the non-parametrically modelled predictors with optional smoothing (Meyer & Liao, 2014).

We start initially in Section 4.2 by providing some results of timing experiments that compare the three different methodologies for fitting monotone polynomials to data. That is the semi-indefinite programming approach of Hawkins (1994), approaches based on Elphinstone (1983) parameterisations, implemented and described in Chapter 2

and in Murray, Müller & Turlach (2013), and the sums-of-squared polynomials approach also described in Chapter 2 and Murray, Müller & Turlach (2015). In Section 4.3 we describe the results from Monte-Carlo simulations comparing the three different approaches in terms of bias, variance and mean squared error (MSE). Finally, in Section 4.4 we provide the results from the timing experiments comparing the three different approaches (monotone polynomials, CGAM and B-splines) to fitting monotone regression curves to data.

4.2 COMPARISONS WITH PREVIOUS PARAMETERISATIONS

In our experience, fitting a monotone polynomial to data using the sum-of-squares parameterisation is markedly faster than using any of the isotonic parameterisations considered previously and is comparable speed wise with the semi-infinite programming approach of Hawkins (1994). While Hawkins' (1994) approach could be extended to allow fitting of polynomials that are monotone only over a specified compact or semi-compact region we do not investigate this further here since in Chapter 2 and in Murray, Müller & Turlach (2013) we noted that isotonic parameterisations of the Elphinstone (1983) type allow the fitting of monotone polynomials in some situations in which the semi-infinite programming approach fails; this also being true for the sum-of-squares parameterisation. Furthermore, we noted that the semi-infinite programming approach might require the careful choice of numerical precision parameters to allow the various numerical algorithms used in its implementation to successfully work together. We would expect these numerical precision problems to increase when a polynomial is restricted to be monotone over a compact or semi-compact interval and, in the terminology of Hawkins (1994), a decision would have to be made on which side of the boundary a 'horizontal inflection point' falls.

We report results from one of our timing experiments that we performed using a sigmoidal function as the true underlying regression function. This target function is selected as it provides a suitable target for monotone polynomials of increasing degree. In fact, Shevchuk (1993) shows that monotone functions over compact intervals can be approximated uniformly to an arbitrary precision by monotone polynomials of suffi-

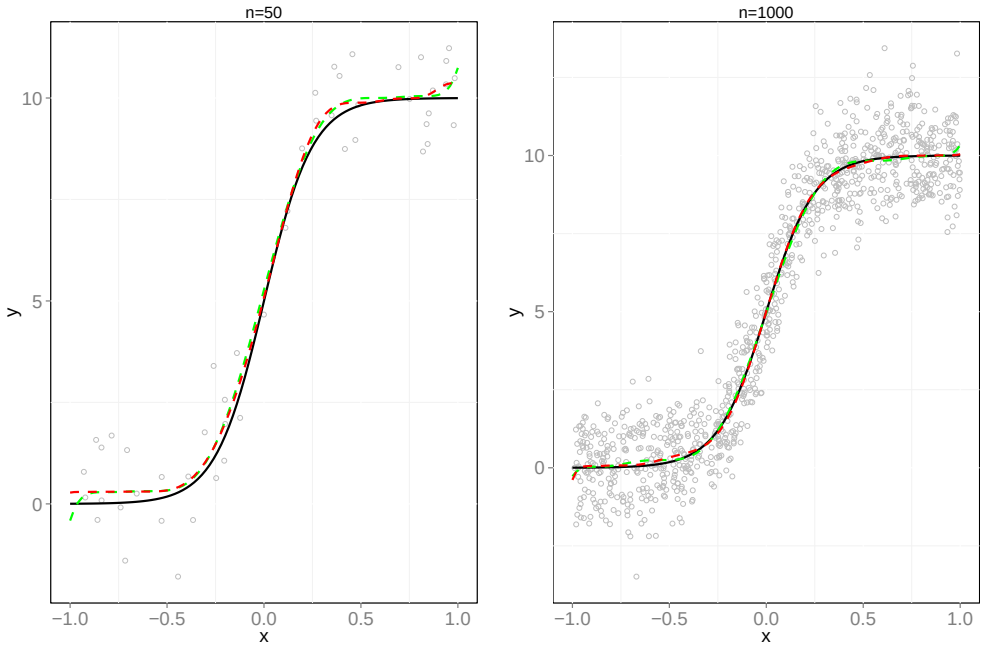

Figure 4.1: Sigmoidal curve (black) with $n = 50$ (left panel) and $n = 1,000$ (right panel) observations generated from this regression curve. Monotone polynomial fit of polynomial of degree 9 (green dashed lines) and degree 15 (red dashed lines) are shown.

ciently high degree q. Figure 4.1 depicts the sigmoidal function used as a solid black line together with a typical data set of size $n = 50$ (left panel) and $n = 1000$ (right panel). Fitted monotone polynomials of degrees $q = 9$ and $q = 15$, respectively, are overlaid in both panels and illustrate that monotone polynomials of degree 9 approximate the underlying regression curve fairly well.

In our timing experiment, we used sample sizes of $n = 50$, 100, 200, 250, 400, 500, 800, 1000, 1600 and 2500, respectively, and monotone polynomial of degrees $5, 7, \ldots, 19$ and 21. For each sample size, 10 replicates were generated, that is we sampled x_i, $i = 1, \ldots, n$, from a uniform distribution, and generated corresponding y_is by evaluating the sigmoidal regression function at the x_is and adding standard normally distributed noise. For each simulated data set the time needed to fit a monotone polynomial using the sum-of-squares parameterisation, Hawkins' (1994) semi-indefinite programming approach and the Elphinstone (1983) type isotonic parameterisation recommended by Murray, Müller & Turlach (2013) were recorded and, for each method, averaged over the 10 replications. The resulting average times are shown in Figures 4.2 and 4.3.

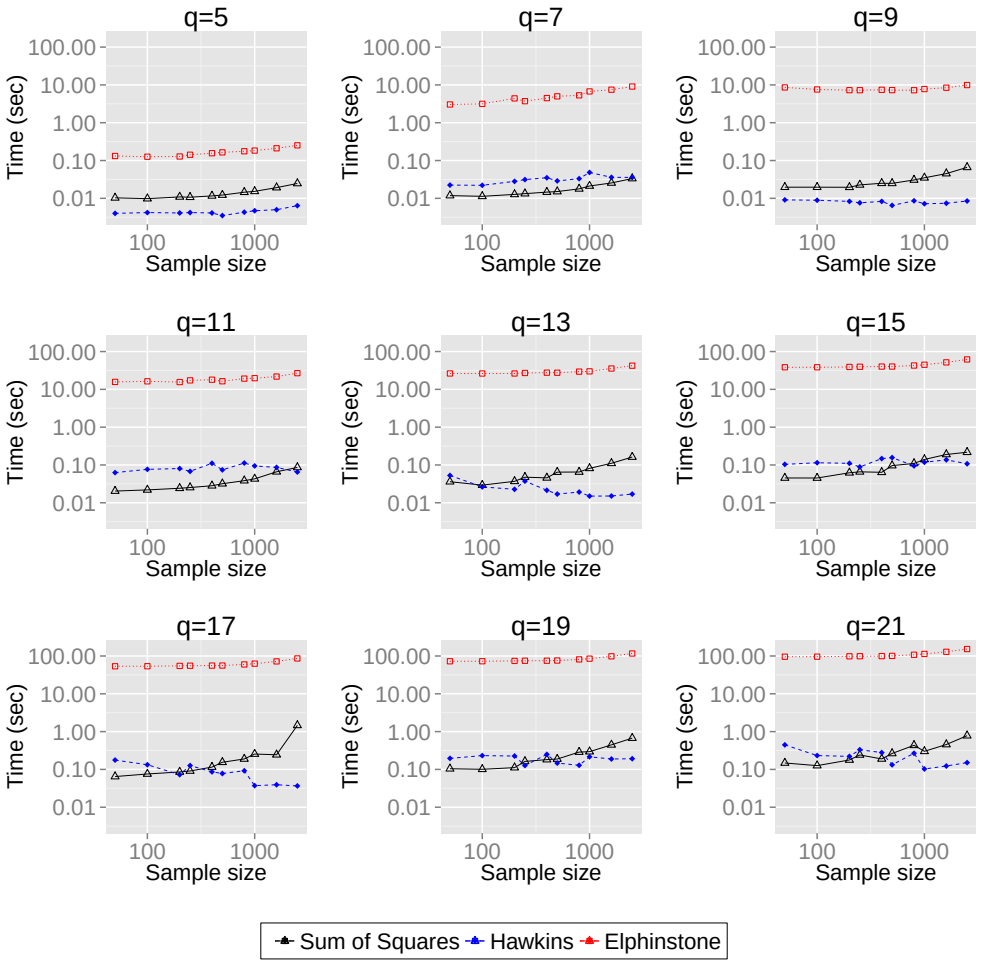

Figure 4.2: Monotone polynomials of degree $q = 5, 7, \ldots, 21$ fitted to simulated data with various sample sizes from a sigmoidal curve (Figure 4.1). The plot shows for three approaches to fitting monotone polynomials the average time, over 10 replications, needed to fit the model against the sample size. Both axes are on a log scale.

These figures show that the sum-of-squares parameterisation is markedly faster than using the preferred Elphinstone (1983) type isotonic parameterisation and is comparable with the semi-infinite programming approach of Hawkins (1994).

During our timing experiments, we also noticed that the sum-of-squares isotonic parameterisation approach and Hawkins' (1994) semi-infinite programming approach always resulted in the same fitted monotone polynomial, while the fitted monotone polynomial using the Elphinstone (1983) type isotonic parameterisation occasionally differed if the true regression function was a polynomial and somewhat more often with the sigmoidal target function. This suggests that fitting monotone polynomials using the sum-of-squares parameterisation, instead of an Elphinstone (1983) type iso-

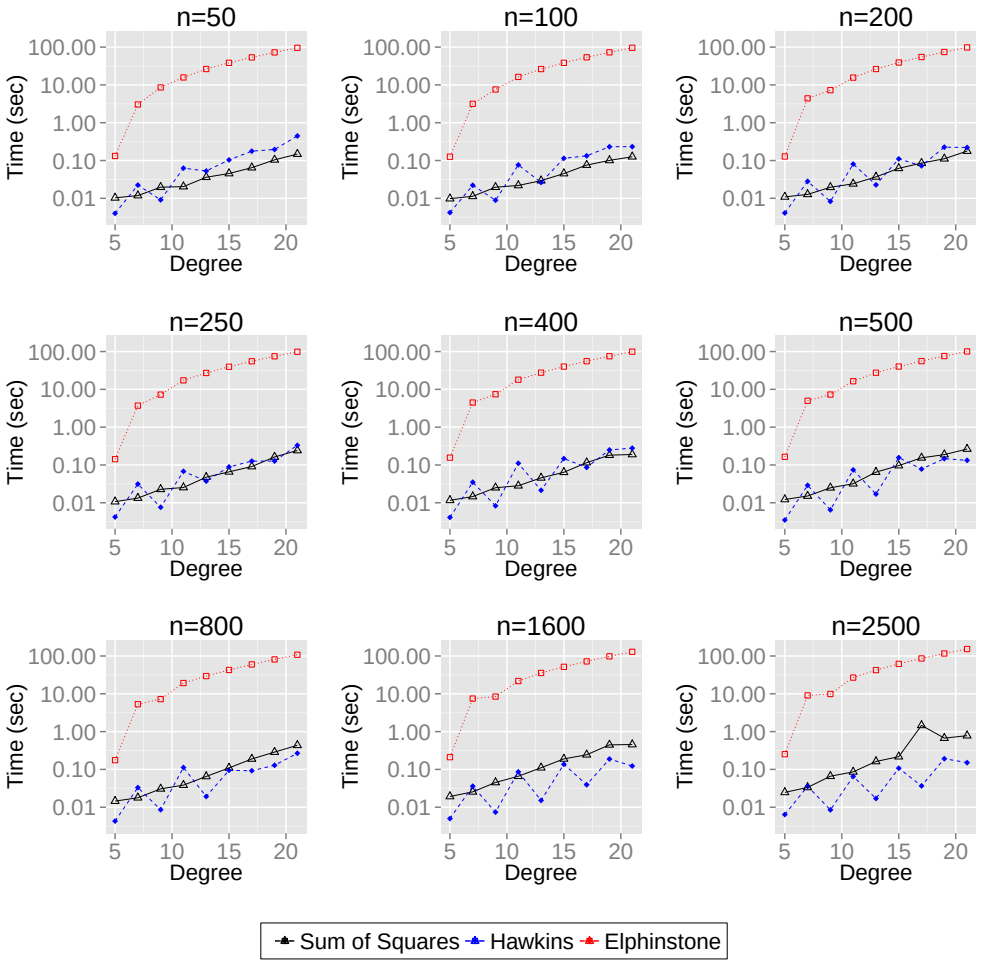

Figure 4.3: Monotone polynomials of various degrees were fitted to simulated data from a sigmoidal curve (Figure 4.1). Sample sizes used in this timing study are indicated in the strip of each panel. The plot shows for three approaches to fitting monotone polynomials the average time, over 10 replications, needed to fit the model against the degree of the polynomial.

tonic parameterisation, is not only beneficial in terms of run-time but also reduces the potential risk of the algorithm converging to a local minima.

4.3 MONTE-CARLO SIMULATIONS TO COMPARE CONSTRAINED REGRESSION TECHNIQUES

For each of $p_2(x)$, $p_3(x)$ and the sigmoid function, all described in Section 3.2, we generate 1,000 simulated responses to examine and compare bias, variance and mean

squared error for the monotone polynomial fits, the CGAM fits and the B-spline fits. Furthermore, we simulate from the following trigonometric function:

$$y = 5\pi - x + \cos(x - \pi/2 - 2) + \epsilon \tag{4.1}$$

where x are at $0, 1, \ldots, 12$ in triplicate and $\epsilon \sim N(0, 1)$. This final simulation reflects the data described in Firmin, Müller & Rösler (2011, 2012) which formed the motivation for further development of monotone polynomial model fitting tools and ultimately this thesis.

Figures 4.4 and 4.5 show the bias, variance and mean squared error (MSE) for data generated from the cubic and quintic polynomial functions $p_2(x)$ and $p_3(x)$, respectively. In each figure we compare the three methods of curve estimation for these data for sample sizes $n = 50$ and $n = 1,000$. We observe for the cubic, that is the function which sits inside the boundary of $\mathcal{C}^q_{\mathcal{R}}$, comparatively good performance from the monotone polynomial fits when compared to the `cgam` and `fda` smoothing spline approaches. This performance is apparent for both sample size 50 and sample size $1,000$ but is more pronounced in the latter. This is to be expected given the fits derived using monotone polynomials are generally the same as that using a standard unconstrained polynomial when the underlying function lies inside the boundary of $\mathcal{C}^q_{\mathcal{R}}$, and the sample size is sufficiently large. Consequently by standard linear model theory this should become an arbitrary accurate estimate of the underlying polynomial function as $n \to \infty$. Comparatively, the results from using the three different methodologies with $p_3(x)$, a quintic which lies on the boundary of $\mathcal{C}^q_{\mathcal{R}}$, are not as clear. For both sample sizes the fits from the `cgam` and the fits from the monotone polynomial are relatively comparable in terms of bias, variance and MSE, with the monotone polynomial providing marginally better results. However, it is notable that the `fda` smoothing spline approach does not appear to perform as well. Whilst this may be a reflection on the methodology itself, we suspect that it is more a reflection on the judicious choices needed to be made in terms of the number and location of knots, smoothing parameters and basis functions. In this instance we used the same settings as for the cubic, and believe this demonstrates, to some extent, some issues with repeatability and robustness of this methodology.

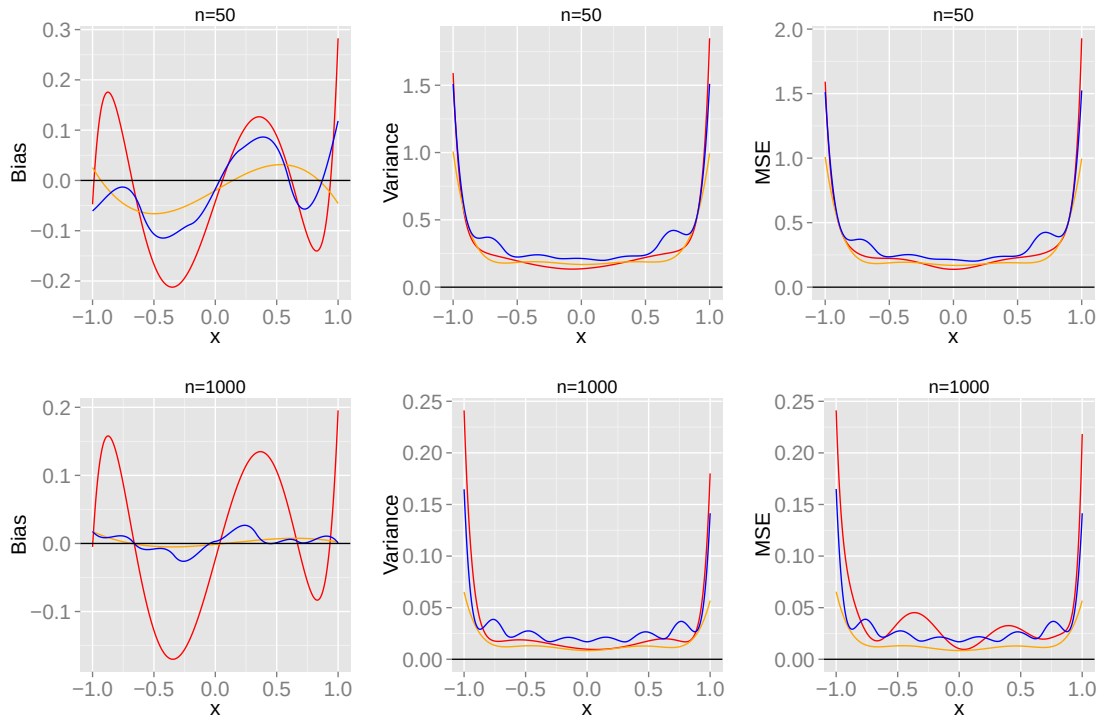

Figure 4.4: Bias, variance and mean squared error based on 1,000 monte-carlo simulations of size $n = 50$ (top row) and $n = 1{,}000$ (bottom row), comparing model fits using monotone polynomials (orange line), `fda` smoothing spline (red line) and CGAM (blue line) for underlying data generating function $p_2(x) = 3x^3 + x + 1$.

Figure 4.6 shows the bias, variance and mean squared error for $n = 50$ and $n = 1{,}000$ for each of the three methods to be compared for data generated from the sigmoidal function. It is clear from these plots that the choice of parameters used to find fits with the three methods, whether it be smoothing parameters, knots, or the choice of degree of polynomial for the monotone polynomial fits, has a significant impact on the performance. It is evident that, at the smaller sample size the smoothing spline approach is comparable to the monotone polynomial approach and whilst the CGAM approach provides lower variance overall it would appear not as effective as the others. At the larger sample size the bias would be somewhat smaller for the monotone polynomial fit, with larger variance but in terms of MSE it could be argued to perform 'better'.

Finally, results using the data generated from the trigonometric function (described by (4.1)), which has limited design points and multiple inflection points, are shown in Figure 4.7. Immediately apparent from this simulation exercise, and numerous comparable other exercises we carried out, is the impact of the smoothing parameters and

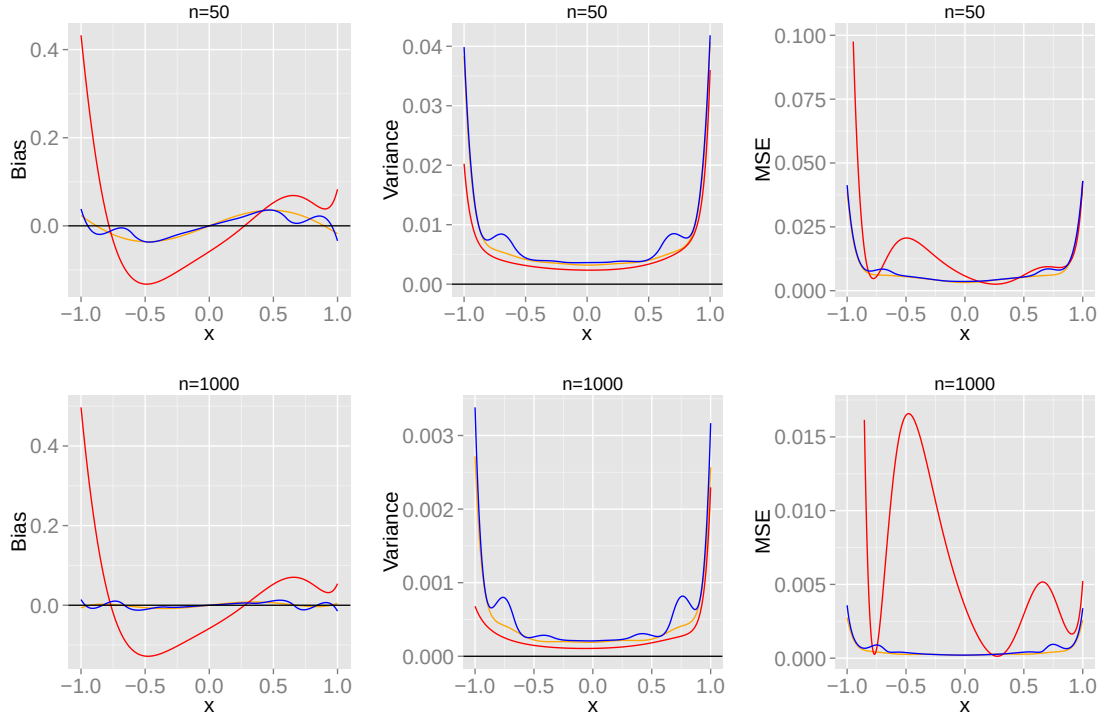

Figure 4.5: Bias, variance and mean squared error based on 1,000 monte-carlo simulations of size $n = 50$ (top row) and $n = 1{,}000$ (bottom row), comparing model fits using monotone polynomials (orange line), `fda` smoothing spline (red line) and CGAM (blue line) for underlying data generating function $p_3(x) = x^5$.

placement of knots. This is something we consider further in Chapter 6 when we use monotone polynomials to search for inflection points. However, in general, we observe that the monotone polynomials outperform the other two methods.

4.4 SPEED COMPARISONS WITH DIFFERENT APPROACHES TO FITTING MONOTONE CURVES

We provide in this section a very brief demonstration of speed comparisons for each of the methods considered. We recognise that there are many elements which will contribute to speed, for example processor speed and memory, however for information we have given a comparison again using a relatively modest machine (MacBook Air, Processor 2GHz Intel Core i7, RAM 8GB). Specifically we use the sigmoidal function described in (3.1) and simulate increasing samples sizes from 50 through to 5,000 from this function. Again, we consider x to be randomly generated from a uniform distri-

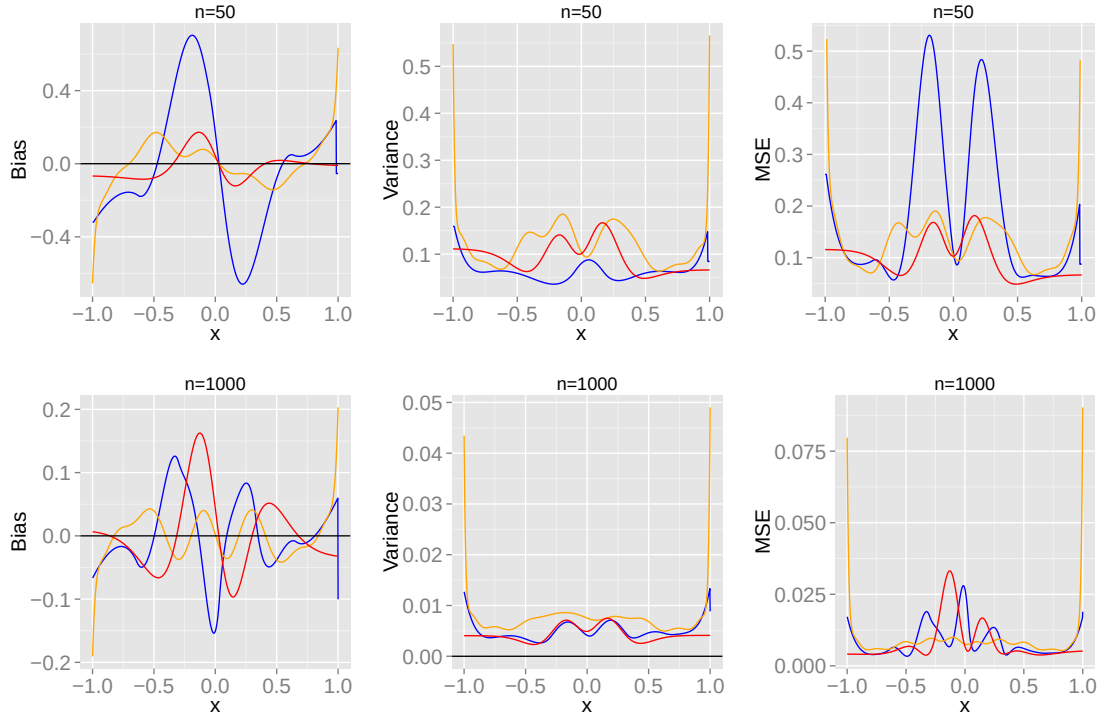

Figure 4.6: Bias, variance and mean squared error based on 1,000 monte-carlo simulations of size $n = 50$ (top row) and $n = 1000$ (bottom row), comparing model fits using monotone polynomials (orange line), `fda` smoothing spline (red line) and CGAM (blue line) for underlying data generating function $f_4(x) = 10/\left[1 + \exp(-8x)\right]$.

bution over the range $[-1, 1]$. The results are shown in Figure 4.8 for each of the three methods under consideration. The monotone polynomial fit is consistently lower in run time than the smoothing spline approach and outperforms the `cgam` approach for all but relatively low sample sizes. The standard `cgam` implementation has dramatically increased run time as sample sizes get relatively large and under-performs the smoothing spline approach. This is probably due to the default settings in the `cgam` package which would appear to produce a large amount of knot placements. Whilst this increased run time may not be important for the fitting of one curve to the data, it will have a dramatic impact on the time to carry out model selection searches for the 'best' combination of parameters needed to fit these models, and in the estimation of confidence and prediction intervals using, for example, computationally intensive techniques such as the bootstrap.

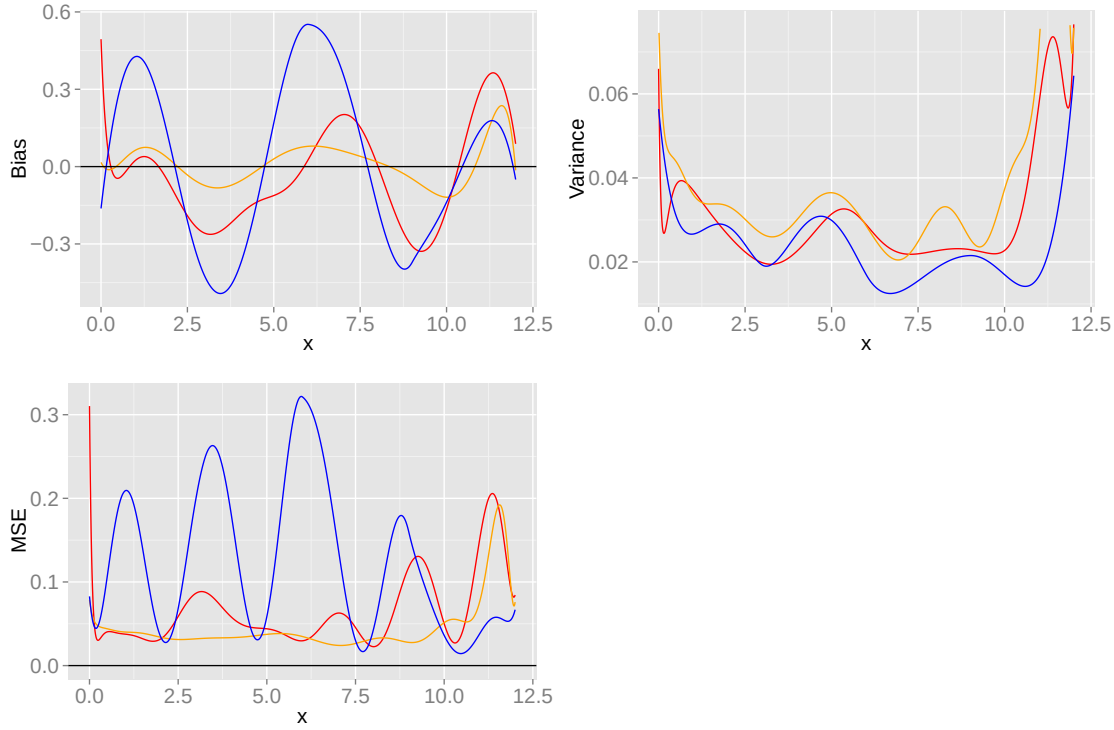

Figure 4.7: Bias, variance and mean squared error based on 1,000 monte-carlo simulations, comparing model fits using monotone polynomials (orange line), `fda` smoothing spline (red line) and CGAM (blue line) for underlying trigonometric data generating function $f_5(x) = 5\pi - x + \cos(x - \pi/2 - 2)$.

Figure 4.8: Comparison of time to fit models for different sample sizes using monotone polynomials (orange line), `fda` smoothing spline (red line) and CGAM (blue line) for underlying data generating function $f_4(x) = 10/\left[1 + \exp(-8x)\right]$.

4.5 DISCUSSION

We have demonstrated in this chapter that for certain functions fitting monotone polynomials to data can provide at least a comparative performance when compared to the current alternative shape constrained regression techniques. However, whilst we recognise that for every regression model in which we can demonstrate monotone polynomials are able to perform comparatively well, or even outperform other techniques, there will be other regression models in which smoothing splines, or similar approaches, would be much more suitable. Furthermore, it certainly is *not* our intention to suggest that monotone polynomials are in any way superior to non-parametric or semi-parametric shape constrained regression — which we certainly do not believe to be the case in all situations. However, we do believe we have demonstrated, at least in some situations, that there is place in the literature and applications for monotone polynomial data fitting. We have also described how the choice of model parameters can have a huge impact on any of the three modelling techniques discussed. However, in the case of the `cgam` and `fda` smoothing spline approach many decisions need to be made, some of which are the subject of current research. Determining this optimal choice of model parameters is in some instances not simple, which has the potential to ensure these approaches could be less palatable, even for relatively straight forward functions. Conversely, acknowledging that the selection of the degree of monotone polynomial can have an impact on the model performance, we would argue that this is a more simple problem to resolve given the choice of degree is limited to a number of models defined by the highest degree monotone polynomial considered. Our timing exercises have also demonstrated that monotone polynomials are at least as efficient as any of the other two shape constrained regression techniques considered and perform more efficiently at larger sample sizes.

We have also demonstrated that the new sums of squares methodology proposed in this thesis for fitting monotone polynomials to data has not only outperformed the previous methodologies in terms of speed, but has also increased flexibility allowing, for example, the fitting of monotone polynomials over (semi-)compact regions. The speed with which monotone polynomials can be fitted using the sum-of-squares paramet-

erisation now makes it feasible to explore the use of bootstrap techniques for problems such as model selection (see Chapter 5) and estimation of standard errors and confidence intervals (see Chapter 3). Furthermore, we believe, in some instances, monotone polynomials may be more desirable mainly due to the simplicity of fitting monotone polynomials to data through our newly described methodology in this thesis, and our R (R Core Team, 2015) package `MonoPoly` (version 0.3-6 or later) currently available on CRAN.

5

GRAPHICAL TOOLS FOR MODEL SELECTION

SUMMARY

Model selection techniques have existed for many years however to date simple, clear and effective methods of visualising the model building process are sparse. This chapter describes graphical methods that assist in the selection of models and comparison of many different selection criteria. Specifically, we describe for logistic regression, how to visualize measures of description loss and of model complexity to facilitate the model selection dilemma. We advocate the use of the bootstrap to assess the stability of selected models and to enhance our graphical tools. We demonstrate which variables are important using *variable inclusion plots* and show that these can be invaluable plots for the model building process. We show with two general case studies how these proposed tools are useful to learn more about important variables in the data and how these tools can assist the understanding of the model building process. Furthermore, we extend these techniques to model selection using monotone polynomials, identify some of the issues with model selection in such constrained scenarios, and provide a further two real world examples to illustrate this process.

5.1 INTRODUCTION

Many applied problems involve the collection of data with multiple potential predictor variables. In analysing the data one usually engages in a process of model building, of which a crucial part is to determine one or more appropriate models. For a general introduction and overview into the topic of model building we refer to Miller (2002). One of the most commonly used techniques for model selection, which is probably the least advocated by statisticians, is a "hypothesis test/P-value" stepwise approach,

using either forward selection, backward selection or a combination of the two. These approaches have been shown to be inefficient in many situations, and have particular issues such as multiple testing and localization of solutions. For many models, including the vast array of generalized linear models, the information theoretic approach and the use of the log-likelihood to compare models is widespread in general for model selection purposes. For this reason, the majority of this chapter mainly focuses on measuring the descriptive ability of a model via the log-likelihood, but our ideas extend directly to using other loss functions.

To date, in medical research in particular, data analysts have used many different techniques to select models for prediction purposes. In most instances only one final model is presented, and how such a model is reached is typically not sufficiently described in the statistical methods section of research articles. Reasons for this include space restrictions and a shortage of simple graphical tools that can be shown to explain the reasoning behind the final model. Consequently future researchers have difficulty obtaining a clear understanding of available model selection techniques in current medical research, what these techniques are really doing, and how to replicate and adapt such techniques.

When using an information theoretic approach to model selection, a somewhat controversial and much debated question is whether to model select using the Akaike information criterion (AIC) (Akaike, 1973) or the Bayesian information criterion (BIC) (Schwarz, 1978). The purpose of the analysis drives the model selection. Often a separation is made between the purpose to describe the data well or to obtain a model which has good predictive qualities. A major difference between AIC and BIC is that AIC attains the minimax rate, whilst BIC is consistent (that is as n tends to infinity the probability of selecting the true model tends to one) and aims to model the dimension of the true model but fails to attain the minimax rate. The concept of a true model is debatable (Shao, 1997) and there are strong arguments in opposition of such assumptions (Burnham & Anderson, 2002). Put simply, the argument is that all of the predictor variables proposed for a model should have some effect on the response, even if this effect is extremely small. In spite of this, many questions still arise: which variables are important to describe the data well; which components remain when a

smaller model is preferred to the model selected using AIC; what happens if a slightly different penalty is chosen to the one which led to the calculation of the AIC value.

Graphs are powerful tools in statistics. We quote Tukey who said: "There is nothing better than a picture for making you think of questions you had forgotten to ask (even mentally)" (Tukey & Tukey, 1985). When fitting models it is standard practice to examine model diagnostics using visual aids (residuals plots, QQ-plots, etc) and we are encouraged to visualise our data throughout the majority of any statistical analysis. In light of this, it seems somewhat perplexing that in carrying out model (or variable) selection, little encouragement is given to employ visualisation approaches. In a general setting the works of Loftus (1983) or more recently with a clinical flavour Krause & O'Connell (2012) provide some examples of the benefits of using graphical techniques in data analyses. However, to date, there are relatively few publications that show graphical tools as aids in model selection. The best known is the Mallow's Cp plot (Mallows, 1973), which has been proposed for linear regression situations and some further variants exist (Spjotvoll, 1977; Siniksaran, 2008). Recently, Müller & Welsh (2010) introduced for a $p < n$ setting a variable detection plot, which is similar to the stability paths in Meinshausen & Bühlmann (2010) for the $p >> n$ context, where n denotes the sample size and p the total number of variables that are subject to selection. Both graphs are based on slightly different resampling procedures and among others aim to facilitate the choice of a tuning parameter.

Whilst on the face of it model selection for polynomial regression is a much more simple task, there are still many facets that have to be considered. The general question of 'what degree is the best to describe my data' still needs to be reconciled with the aim of the analysis. However, in standard polynomial regression one can still make use of typical approaches including the information criteria approaches described previously or more sophisticated approaches, for example the bootstrap approach (see Chapter 4) to model selection. In constrained regression, for example, determining the appropriate degree for a monotone polynomial model, little work exists. Hawkins (1994) made a suggestion to determine the 'best' degree polynomial by using 'conventional F-tests' of the sequentially added higher order terms. However, he noted that the hypotheses in

such a model selection do not satisfy the needed regularity conditions for conventional F distributions.

In this chapter, we take a liberal view of model selection, mainly in a medical context, and instead of trying to argue one way or another on issues like AIC versus BIC, or on the advantages of techniques such as model averaging, we aim to provide simple graphical tools for the visualisation of different model selection criteria that facilitate the descriptive process of the data and assist with the model selection problem. It should be noted that we are not advocating the use of our plots for one specific purpose; rather we aim to demonstrate how they can aid in many different aspects of model building.

The outline of the chapter is as follows: In Section 5.2 we describe the terminology and methodology and introduce our graphical concepts by a first case study with simulated data. Additionally, we show how the methods assist in the model selection process. In Section 5.3 we present the results of two further case studies, which have more variables and more features than the simulated data. The first is from a retrospective cross-sectional palliative care study and the second from an ongoing health study analysing the effect of smoking in patients with Crohns disease. In Section 5.5 we demonstrate graphical techniques for model selection in the specific constrained polynomial estimation described in Chapter 2. We further develop our techniques to look at this specific case and demonstrate model selection for monotone polynomials through two real world examples. In Section 5.6 we demonstrate other uses of our model selection plots, in particular in the context of highly correlated regressors and in the presence of outliers. We conclude with a discussion and comments in Section 5.4. R code for reproducing all results and figures shown in this chapter can be found in Murray, Heritier & Müller (2013).

5.2 TERMINOLOGY AND GRAPHICAL METHODS

5.2.1 *Generalised linear model framework*

Assume that n independent observations $y = (y_1, \ldots, y_n)^T$ and a $n \times p$ design matrix X is available, whose columns are indexed by $1, \ldots, p$. Let α denote any subset of p_α distinct elements from $1, \ldots, p$. Let X_α be the corresponding $n \times p_\alpha$ design matrix and $x_{\alpha_i}^T$ denotes the ith row of X_α.

Then a generalised linear regression model (GLM) α for the relationship between the response y and the design matrix X_α is specified by

$$\mathrm{E}(y_i) = h(\beta_{0,\alpha} + \eta_i), \text{ and } \mathrm{Var}(y_i) = \sigma^2 v^2(\eta_i) \text{ with } \eta_i = x_{\alpha_i}^T \beta_\alpha, \quad i = 1, \ldots, n, \quad (5.1)$$

where $(\beta_{0,\alpha}, \beta_\alpha^T)^T$ is an unknown $(p_\alpha + 1)$-vector of regression parameters and σ is an unknown scale parameter, or 1 in the binomial and Poisson models. Here h is the inverse of the usual link function and, for simplicity, we have absorbed h into the variance function v. Both h and v are assumed known. In this chapter we initially focus on the logistic regression model, which has binomial response and logit link function, however our methods can be extended to any GLM, and we demonstrate later how transferable they are to monotone polynomial regression models.

5.2.2 *Measuring description loss and complexity in GLM*

The purpose of model selection is to choose one or more models α, from all candidate models $\mathcal{A}$, with specified desirable properties. Many model selection procedures involve the minimisation of an expression which can take the simple form:

$$\textit{Description Loss} + \lambda \times \textit{Model Complexity}. \quad (5.2)$$

Most prominently, the 'description loss' of a model can be measured by $-2\times$ log-likelihood. Here, we use the term 'description loss', which we regard as any function L_n that measures how well a model fits the data. Model complexity, in its simplest form, is the number of independent regression model parameters. We refer to λ as the penalty multiplier, which often drives the properties of the selection criteria (Müller & Welsh, 2010). We will focus on procedures that use the information theoretic choice of description loss, that is minus twice the log likelihood. The AIC is of form (5.2) and has penalty multiplier $\lambda = 2$; similarly, the BIC has $\lambda = \log(n)$ or more generally the generalised information criterion (GIC) (Konishi & Kitagawa, 1996) has penalty multiplier $\lambda \in \mathbb{R}$.

5.3 CASE STUDY I – SIMULATED LOGISTIC REGRESSION EXAMPLE

For ease of presentation we illustrate our techniques through logistic regression examples. However, these are easily extendable to any GLM. Initially we describe a simulated logistic regression example in which $n = 250$ responses ($y_i's$) are modelled with seven potential predictors $x_1 - x_7$. The predictors themselves are simulated from a multivariate normal distribution, with correlations of 0.7 introduced between predictors x_1 and x_2 and between x_5 and x_6. The data generating model samples responses from Bernoulli random variables with probability of success π_i, such that

$$\log\left(\frac{\pi_i}{1-\pi_i}\right) = \beta_0 + \beta_1 x_{i1} + \ldots \beta_7 x_{i7}, i = 1, \ldots, n, \tag{5.3}$$

and the parameter values for the β's (excluding the intercept) are set to be (2.1, 0, 1.5, 0, 0, 0.9, 1.1).

The full model including all 7 regressors and intercept, that is $\alpha_f = \{1, 2, 3, 4, 5, 6, 7, 8\}$, has $-2 \times \text{LogLik}(\alpha_f) = 135.9$, whilst the model including only an intercept has $-2 \times \text{LogLik}(\{1\}) = 207.7$. Because there are seven explanatory variables there are a total of $2^7 = 128$ possible logistic GLM's, that is $\mathcal{A} = \{\{1\}, \ldots, \alpha_f\}$. The description loss and dimension for all $\alpha \in \mathcal{A}$ are visualised next.

5.3.1 *Visualising L_n and p_α*

Basic scatter plot

The left panel in Figure 5.1 shows for all possible $\alpha \in \mathcal{A}$ a simple scatter plot of the number of regression parameters (horizontal axis) and $L_n = -2 \times \text{LogLik}$. As the dimensionality increases the general pattern is for the description loss (L_n) to decrease, that is become better. We can use these simple plots in many different ways. First, we observe for any given model dimensionality that we have a model of rank 1. This is the model which provides minimum L_n value among those models having the same dimension (number of explanatory variables). For example in Figure 5.1, with two regression parameters there is a model with $L_n = 173$, which is the lowest of all seven models at this dimension and hence is defined as the rank 1 model for dimension 2. We have added also to this plot two dashed lines which represent a line of slope -2 which intersects the AIC minimum solution (dimension 6), and a line of slope $-log(n)$ which intersects the BIC minimum solution (dimension 5). From this one can determine that no other solutions using either a minimum AIC or BIC as model selection criteria exist for this particular data. We consider this in more detail in 5.3.3.

Enriching the scatter plot

The right panel of Figure 5.1 is an enriched scatter plot. In this instance we have labelled models that include the variable x_1 (red triangle) differently to those that do not (black circle). In this example we observe separation of L_n values over all dimensions, in that all models that include this important variable (x_1) give a much lower L_n than all models that do not include this variable. We can generalise this approach to examine all variables and those interactions that are of interest.

By visual inspection of such graphs, we can determine that this particular variable is “a must" for the inclusion in the final model (or subset of models), without having to make a choice on the size of the penalty multiplier, or delving into arguments about AIC versus BIC. Mathematically, it is clear that complete separation can only occur when the grouping of models is according to variables present in the best model of a

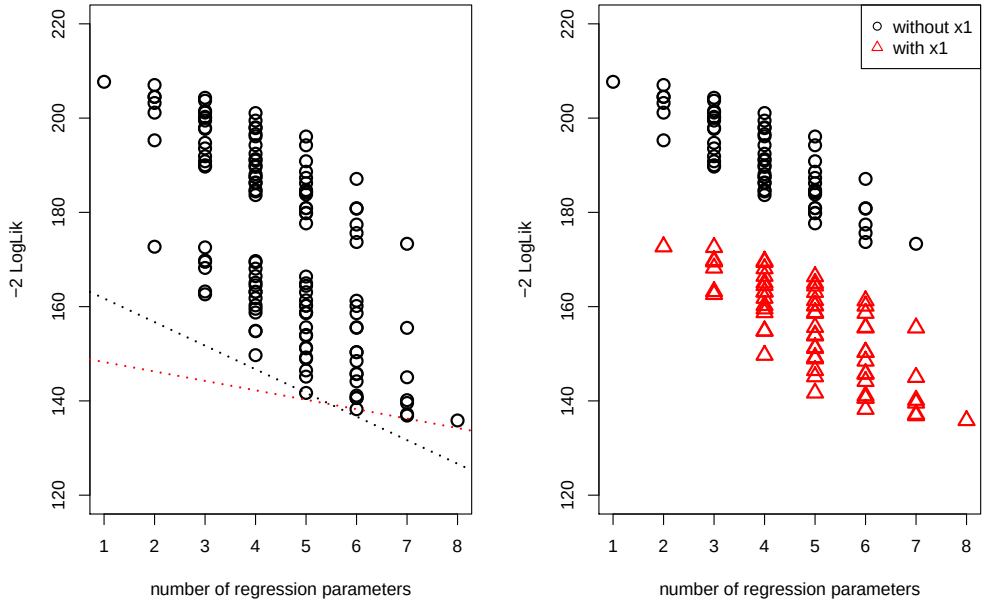

Figure 5.1: Left panel - L_n vs dimension. Right panel - L_n versus dimension by model type, that is presence or absence of an highly influential variable (triangle = included, circle = not included)

given dimension. For example by construction of the maximum likelihood approach adding any variable $k \in 1, \ldots, p$ to a given model $\alpha \in \mathcal{A}$ satisfies $-2 \times \text{LogLik}_{\alpha^*} \leq -2 \times \text{LogLik}_{\alpha}$, where $\alpha^* = \alpha \cup \{k\}$. Then, we can use these plots to list and see all models of a given dimension that represent a local minimum in terms of L_n.

5.3.2 *Assessing model stability through bootstrapping*

A slight modification to these scatter plots is choosing the plot symbol size to be proportional to a measure of model stability, a concept that was independently introduced by Meinshausen & Bühlmann (2010) and Müller & Welsh (2010) for different linear regression situations. Model stability can be estimated by bootstrapping, which was successfully used for model selection in GLMs by Müller & Welsh (2009). There exist different bootstrapping techniques including the residual bootstrap, the paired bootstrap and the weighted bootstrap as described by Barbe & Bertail (1995) and used recently by Minnier, Tian & Cai (2011). We demonstrate how our model selection plots can be

modified to give an idea of model stability at each dimension using the weighted bootstrap, which is simple to implement because it is based on repeatedly re-weighting observations. Conversely, the residual bootstrap can be problematic for generalised linear models since adding bootstrapped residuals to fitted values can result in unobservable responses, whilst the non-parametric bootstrap is known to lead to potential separation problems for logistic regression models.

To obtain our measure of model stability, let w_i be the weight for observation i in our original data, and sample the w_i's from an exponential distribution with expectation 1 as described in Janssen & Pauls (2003) and Minnier, Tian & Cai (2011). We refit our original models using these weights, and resample B times, where B is the number of bootstrap samples which we set to 1,000. For each bootstrap sample we rank models within each dimension and count the number of times each model is of rank 1. We plot L_n versus dimension with symbol size proportional to the number of times model α has rank 1. This incorporates information on model stability into the scatter plot. Consequently those models selected a large proportion of times in the weighted bootstrapping have large symbols and those selected very few times have small symbols. We also highlight the underlying model we simulated from (the data generating model) with the full red circle. Figure 5.2 shows an example and we note that the size of the plot symbol for dimension 1 (intercept only) and dimension 8 (α_f) is at its largest since there is only one possible model choice within these dimensions. However, at dimensions 2 through to 7 we can assess the stability of models with lowest description loss visually, and determine how reliable any one chosen model may be. For example, at dimension 2 there is one model that stands out as the single best model, whilst at dimensions 3 and 7 there are at least two models that have very similar size symbols at the optimum end of the scale. For information we include below the figure those variables included in the rank 1 model for each dimension. For example, at dimension 4, the model with lowest L_n includes the regressor variables x_1, x_3 and x_6.

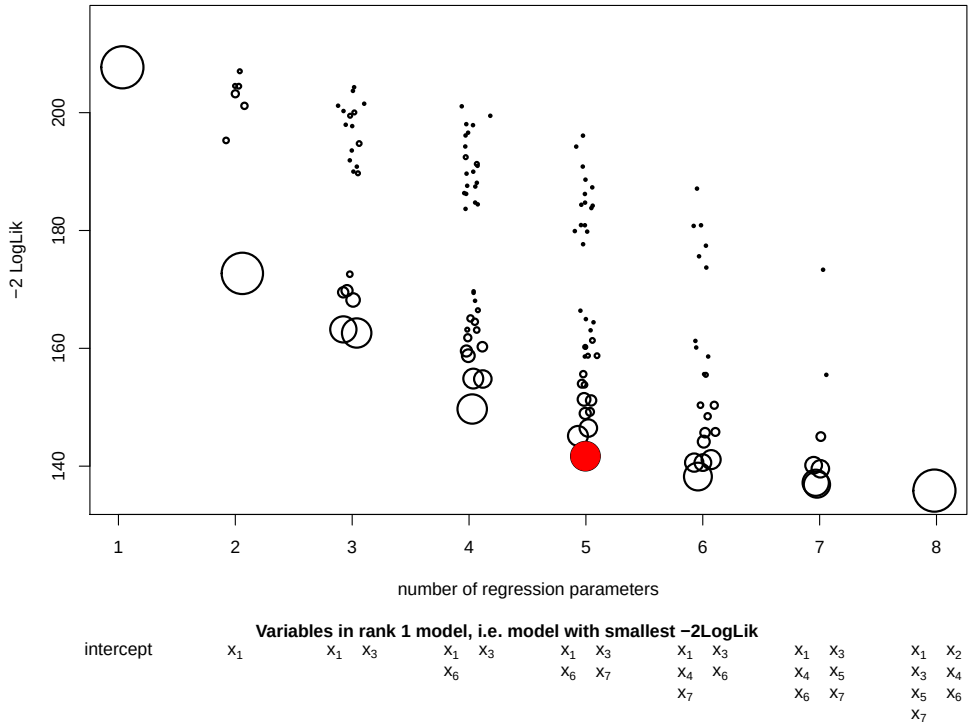

Figure 5.2: Assessing model stability using weighted bootstrapped probabilities at each dimension with probability proportional to symbol area. Note: number of regression parameters is jittered to avoid symbols completely overlapping. The data generating model is shown in red. The panel below shows the variables included in the rank 1 model for each dimension

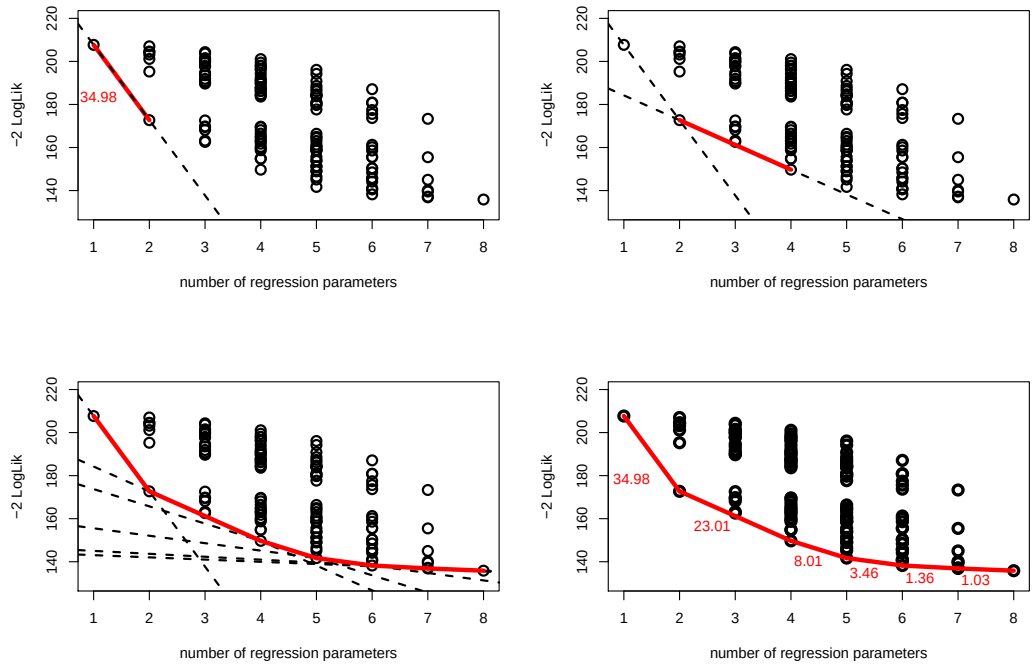

Figure 5.3: Constructing the maximum enveloping lower convex curve and the decrease in L_n for models lying on this curve

5.3.3 *The maximum enveloping lower convex curve*

We define and describe the *maximum enveloping lower convex curve* (MELCC) which is developed by the following algorithm. Starting with dimension 1 (or an equivalent appropriate starting point) draw a straight line from the minimum L_n at this point to either the minimum L_n at the next dimension, or the minimum L_n at the next dimension that envelops all other minimum L_n/dimension combinations inbetween. Repeat this process until the highest dimension to obtain the curve. Figure 5.3 depicts this process, starting in the top left panel where we join the minimum L_n at dimension 1, to the minimum L_n at dimension 2. Continuing we join the minimum L_n's at subsequent dimensions until the curve is complete. We show these steps with the thick red line indicating the part of the MELCC added. The final curve is shown in the bottom left and right panel of Figure 5.3 (with all lines and the partial parts of the curve displayed in the bottom left panel). Note, in some instances the minimum L_n at a given dimension will be omitted from the curve if it is enveloped by a line joining minimum L_n's at dimensions above and below. This is seen in the top right panel of Figure 5.3 where the minimum L_n at dimension 3 is fully enveloped by the minimum L_n's at dimensions 2 and 4. Intuitively these model dimensions that do not have an L_n lying on the MELCC indicate that a model of that dimension would never be selected according to a GIC model selection strategy, irrespective of the penalty. The reasoning is simple: We note that (i) the MELCC has at most p different slopes, a property that follows from the convexity and (ii) that the GIC solution is found by having a straight line passing through a point such that all other points are not below that line. From (i) and (ii) it follows that only points on the MELCC are GIC solutions.

Visual inspection of this curve shows the lowest L_n for all dimensions effortlessly and identifies the model dimensions that do not appear on the MELCC. For completeness Figure 5.3 also shows the finalised MELCC with the value of the decrease for each move along the curve. The first decrease is large (34.98) and subsequent decreases much lower. These decreases determine whether a model with higher dimension would be selected for a given penalty multiplier. For example, with AIC, the penalty multiplier is $\lambda = 2$. Hence we would select models on the curve that have a decrease in L_n of 2

or greater for each unit increase in dimension. Alternatively we could use BIC, that is $\lambda = \log(n)$, which in this example is 5.52, and to accept a larger model the decrease in L_n would have to be in excess of 5.52 units per unit increase in dimension. Using AIC we arrive at the solution of the point at minimum L_n on the curve with dimension 6, and in this instance using BIC gives a solution at dimension 5.

5.3.4 *The variable inclusion plot*

In addition to being able to simply visualise the loss and dimensionality we show here how a "variable inclusion plot" can provide insightful information. The variable inclusion plot (VIP) was introduced by Müller & Welsh (2010) for linear regression models. Here the VIP is introduced for GLMs using the weighted bootstrap. The VIP visualises 'inclusion probabilities' as a function of the penalty multiplier λ. For each variable x_j subject to selection, the proportion of times this variable is retained in the B final selected bootstrapped model is plotted for a range of λ values, for example $\lambda \in [0, 2\log(n)]$. More specifically, we calculate for bootstrap sample $b = 1, \ldots, B$ and for each considered λ multiplier value that model $\hat{\alpha}_\lambda^{(b)} \in \mathcal{A}$ which has smallest $\text{GIC}(\alpha; \lambda) = -2 \times \text{LogLik}(\alpha) + \lambda p_\alpha$ value. Thus the inclusion probability for variable x_j is estimated by

$$\frac{1}{B} \sum_{b=1}^{B} 1\{j \in \hat{\alpha}_\lambda^b\},$$

where the indicator function $1\{j \in \hat{\alpha}_\lambda^{(b)}\}$ is one if variable x_j is in the final model and zero otherwise.

Figure 5.4 shows the VIP for all seven variables in our simulated data and to help identify what happens when using AIC and BIC as a model selection criteria we show vertical lines at the AIC penalty multiplier value $\lambda = 2$ and for BIC at $\lambda = \log(n)$, respectively.

From this chart we can immediately see two things: First, regardless of the value of the penalty multiplier, variable x_1 is always going to be included in a final model with a bootstrapped probability of 1 for penalty parameter values up to 10. Similarly for relatively large λ values variable x_3 seems to maintain a high inclusion probability.

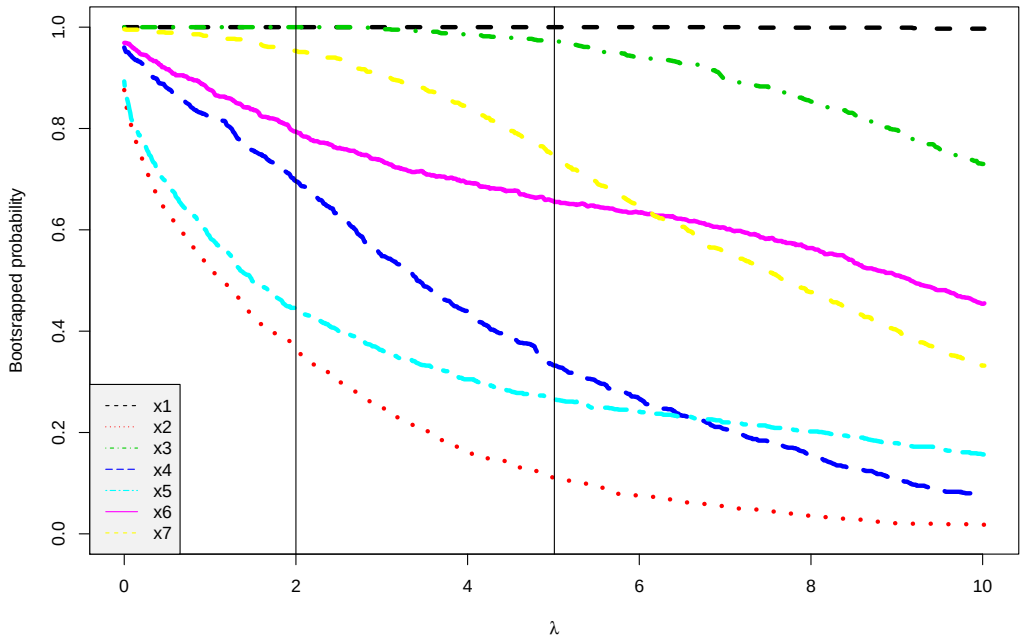

Figure 5.4: Variable inclusion plot for simulated data showing the bootstrapped inclusion probability against the penalty parameter for each variable considered. The horizontal lines indicate the penalties associated with AIC and BIC

Beyond this we can examine each curve individually. Variable x_6 appears to have a reasonably high inclusion probability regardless of the penalty as does variable x_7 but the latter tails off more steeply as the penalty is increased. On the other hand, since we know the data generating model, redundant variable x_4 has a relatively high chance of being included with the AIC, only a fair chance with BIC and beyond that is not likely to be selected. Redundant variables x_2 and x_5 both have a fair chance of inclusion with AIC but beyond that the bootstrapped probabilities diminish, indicating that a higher penalty would result in neither of these variables being included. One of the big advantages of the VIP is to visualise differences between AIC and BIC and any other generalized information criteria (GIC) as a model selector, which reveals much more information than when 'blindly' choosing a fixed λ value.

5.4 RESULTS AND FURTHER APPLICATIONS

We further investigate the ideas described in the previous section through two additional case studies, and show more ways in which combinations of our graphical displays can be used.

5.4.1 *Case study II – Cross-sectional palliative care study*

We analyse data from a retrospective cross-sectional study on how home based palliative care services affect hospitalisations of people in a cohort identified systematically through death registrations as described in Rosenwax, McNamara, Murray *et al.* (2011) and McNamara & Rosenwax (2010). The cohort comprised 1,071 people who died in Western Australia between 1 August 2005 and 30 June 2006, had an informal primary carer at the time of death, did not reside in a residential aged care facility, and died of one of 10 conditions amenable to palliative care.

Using a logistic GLM with logit link function, and the terminology described previously, we model the dichotomous response variable 'did' or 'did not receive community based palliative care'. It is known that cancer patients are much more likely to receive such care because of the nature of their condition, so we considered the dichotomous variable underlying cause of death (cancer or non-cancer) as a vitally important explanatory variable; we label this x_1 for simplicity. Additionally we considered variables age at death (x_2) and age squared (x_3); gender (x_4); number of days spent in hospital in final year of life (x_5); number of emergency department visits in final year of life (x_6); and usual place of residence (metropolitan or rural, based on postcode) (x_7). Therefore, $p = 7$ potential parameters are included in the full model, resulting in $\#\mathcal{A} = 2^p = 128$ possible sub-models that include an intercept. However, if the marginality principle is obeyed the age squared term is only included in the model together with the linear term. This reduces the number of models to $\#\mathcal{A} = 96$. In Table 5.1 we show parameter estimates and standard errors for the full model, the model selected using both AIC and BIC as selection criteria, along with a p-value based forward

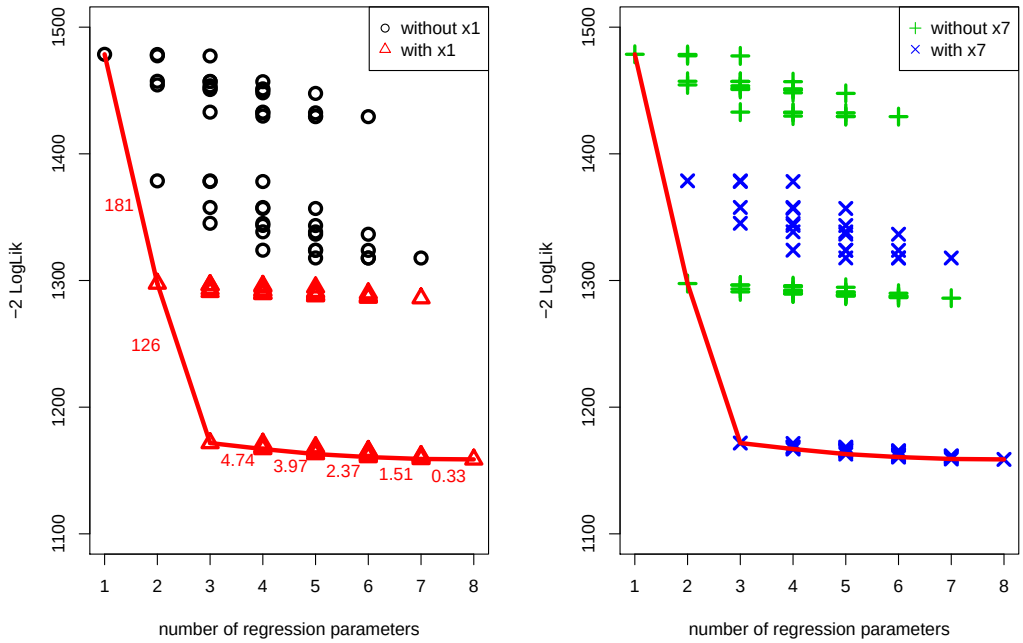

Figure 5.5: Basic and enriched scatter plot with MELCC and reduction in L_n for models on the MELCC for case study II

and backward hypothesis testing approach to select a final model. Note the substantial differences in the models selected between these methods, even with our relatively small p. For example the AIC model includes five explanatory variables whereas both BIC and backward selection only include x_1 and x_7.

The basic scatter plot is shown in Figure 5.5. We show the impact of including or excluding the two most important main effects by enriching this plot by choosing different symbols for models with and without x_1 and with and without x_7 as well as drawing the MELCC. We observe the following: first, in the left hand plot a stark separation of the model L_n for all dimensions is observed, when the variable of interest (in this case cause of death) is either included or not included in the model. Such separation does not exist for any of the other main effects (not shown here) and we conclude that cause of death is clearly the single most important variable irrespective of model dimension or concepts such as p-values. Second, the right hand panel of Figure 5.5 shows the impact of including a second important variable (x_7) in the model, subsequent to fixing either level of cause of death in the model. We observe that the model with both Cause of Death and Usual Place of Residence give us consistently

Table 5.1: Full model, AIC and BIC best model, forward/backward best model, parameter estimates and standard errors

Variable	Full model		AIC best		BIC best		Forward Selection		Backward Selection	
	Estimate	SE	Estimate	SE	Estimate	SE	Estimate	SE	Estimate	SE
intercept	-2.127	1.514	-2.408	1.496	-1.392	0.153	-0.220	0.507	-1.392	0.153
x_1	2.118	0.185	2.086	0.182	2.250	0.176	2.105	0.182	2.250	0.176
x_2	0.057	0.045	0.057	0.045			-0.012	0.006		
x_3	-0.001	0.000	-0.001	0.000						
x_4	-0.087	0.150								
x_5	-0.084	0.069								
x_6	-0.193	0.115	-0.243	0.108			-0.244	0.108		
x_7	-1.918	0.185	-1.924	0.185	-1.884	0.182	-1.902	0.184	-1.884	0.182

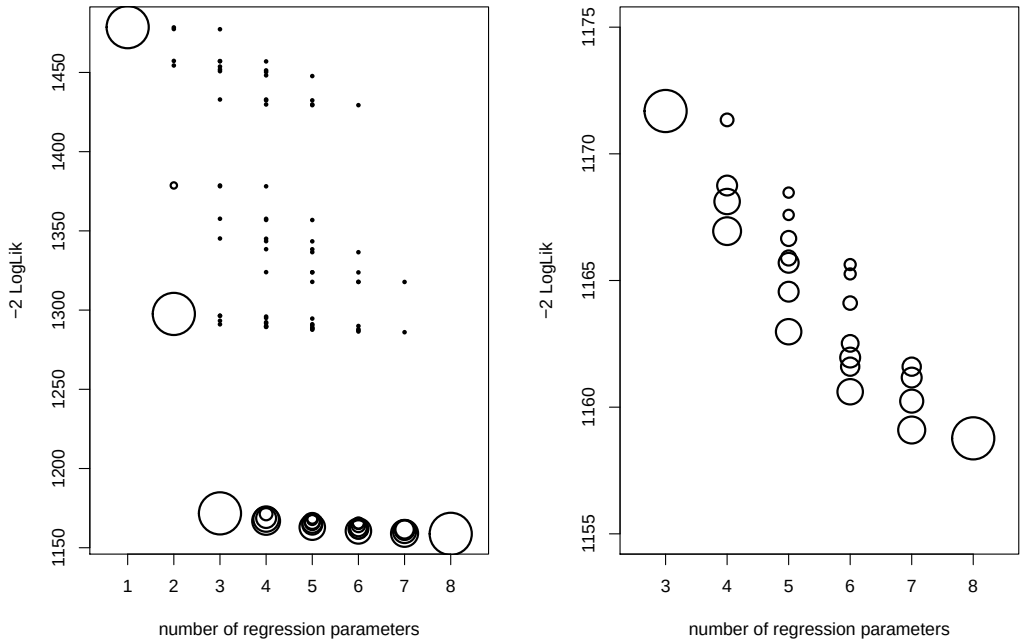

Figure 5.6: Model stability for case study II data with weighted bootstrapped proportions conditional on dimension proportional to symbol area. Left panel full plot, right panel zoomed in plot.

lower L_n regardless of dimension. Again, such a pattern is not apparent for other variables when combined with Cause of Death. We conclude that these two variables are important as a pair. We could continue in this fashion, considering other effects including interactions and higher order terms.

Figure 5.5 also shows the MELCC for this data. We note the dramatically steep decline as we move from the intercept only model, to the models at dimensions 2 and 3, further highlighting the importance of the two variables x_1 and x_7. From the left hand plot we see the reduction from dimension 3 to 4 is 4.74, which is less than $log(n) = 6.98$. Here, the BIC solution has dimension 3. Furthermore when continuing down the curve until a reduction of less than 2 is observed, we obtain the AIC model. This is attained at dimension 6 as also seen in Table 5.1. For this case study the change in the L_n values as we move from dimension to dimension is monotone decreasing, which is not the case in general.

The model stability plot shown in Figure 5.6 indicates again the stability at each of dimension 2 and dimension 3 with large symbols indicating one model is selected the

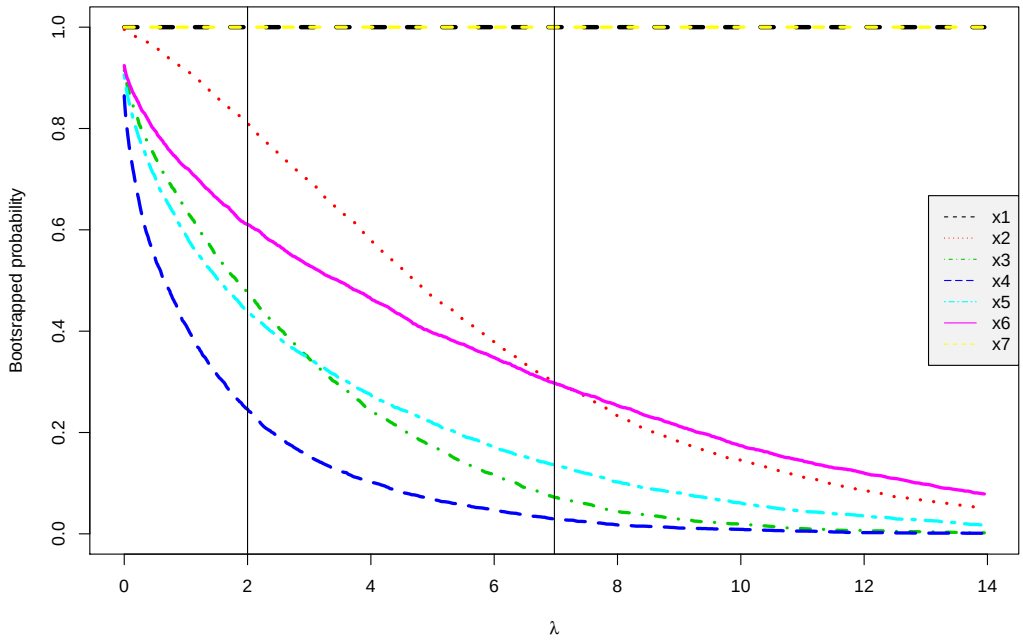

Figure 5.7: Variable inclusion plot for case study II

majority of the time at these dimensions. However, in the right hand panel Figure 5.6, we show the zoomed in version of this plot focussing on dimensions 4 through 7 and identify several models that have similar size symbol to that of the optimal model at each of these dimensions.

The variable inclusion plot shown in Figure 5.7 gives an indication of which variables would and should be included, dependent on the value of the penalty parameter. First, we note again the importance of variables Cause of Death (x_1) and Usual Place of Residence (x_7), which are always included regardless of the penalty multiplier used, as shown by the two overlaying horizontal lines with probability of 1 for all λ values in the range given. Second, we see that with lower values of λ, a large proportion of the time variables x_2 and x_6 would be included in the 'best model', and to some extent variables x_3 and x_5 could also be considered. At higher penalty values it would be difficult to argue for the inclusion of anything more than the two main variables, which is reflected in the BIC model for this data. The VIP clearly shows that variables x_2 and x_6 are important to describe the data but are not informative to predict the response variable.

5.4.2 *Case study III – Smoking in patients with Crohns disease*

The final case study consists of data from an ongoing study described in Lawrance, Murray, Batman *et al.* (2013) that investigates the impact of smoking in patients with Crohns Disease, specifically focussing on those patients who have already had one surgery relating to their condition since their initial diagnosis. We examine the dichotomous response of having a repeat surgery (yes/no) and the impact of potential explanatory variables: age group (young, middle aged and old); sex; whether the patient had ever been a smoker (yes/no); the amount of time they were followed up (measured in years post their first surgery); the location of the condition (broken down into one of three locations); and an additional indicator location variable depicting severity of location. In addition, whether the patient was using steroids (yes/no); had a perianal condition (yes/no); was classified as immuno suppressed (yes/no); or had received Anti TNF α agents (yes/no) were included as potential explanatory variables. In total ten variables were examined (seven binary indicator variables, one continuous variable, and two factors each with 3 levels). Changing the factors to dummy variables(two additional parameters each) we arrive at a total of twelve possible explanatory variables. We retained the factors as a whole in the model, hence the two parameters associated with each of the two factors would be regarded as a grouped variable, with levels that are either simultaneously included or excluded.

We use logistic GLMs with logit link function to model the occurrence of a second surgery (yes/no) response and demonstrate further our ideas. With $p = 12$ potential variables in the final model, we have $\#\mathcal{A} = 2^{12} = 4,096$ possible sub-models that include an intercept. Taking into consideration the two factors and ensuring that the corresponding dummy variables are either simultaneously excluded or included, the number of potential models reduces to $\#\mathcal{A} = 2^{10} = 1,024$.

In Table 5.2 we report the model selection for this data by showing for each dimension what model is optimal (has smallest L_n value). Note both backward and forward variable selection using a p-value approach would give us a model of dimension 5, the minimum AIC model is a model of dimension 6, and the minimum BIC model is

Table 5.2: Best models by dimension showing decrease in L_n for Case Study III

p_α+1	model	$-2 \times$ LogLik	decrease	
1	intercept only	711.73	NA	
2	$+ x_5$	634.26	77.47	
3	$+ x_5 + x_{11}$	614.35	19.91	
4	$+ x_5 + x_7 + x_{11}$	601.34	13.01	BIC
5	$+ x_5 + x_7 + x_{10} + x_{11}$	595.65	5.69	bw and fw
6	$+ x_5 + x_6 + x_7 + x_{10} + x_{11}$	593.35	2.30	AIC
7	$+ x_5 + x_6 + x_7 + x_{10} + x_{11} + x_{12}$	592.28	1.07	
8	$+ x_5 + x_6 + x_7 + x_8 + x_{10} + x_{11} + x_{12}$	591.48	0.80	
9	$+ x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12}$	591.09	0.39	
10	$+ x_1 + x_2 + x_5 + x_6 + x_7 + x_8 + x_{10} + x_{11} + x_{12}$	590.61	0.48	
11	$+ x_1 + x_2 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12}$	590.25	0.36	
12	$+ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_{10} + x_{11} + x_{12}$	590.32	-0.07	
13	$+ x_1 + x_2 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 + x_{10} + x_{11} + x_{12}$	589.90	0.42	

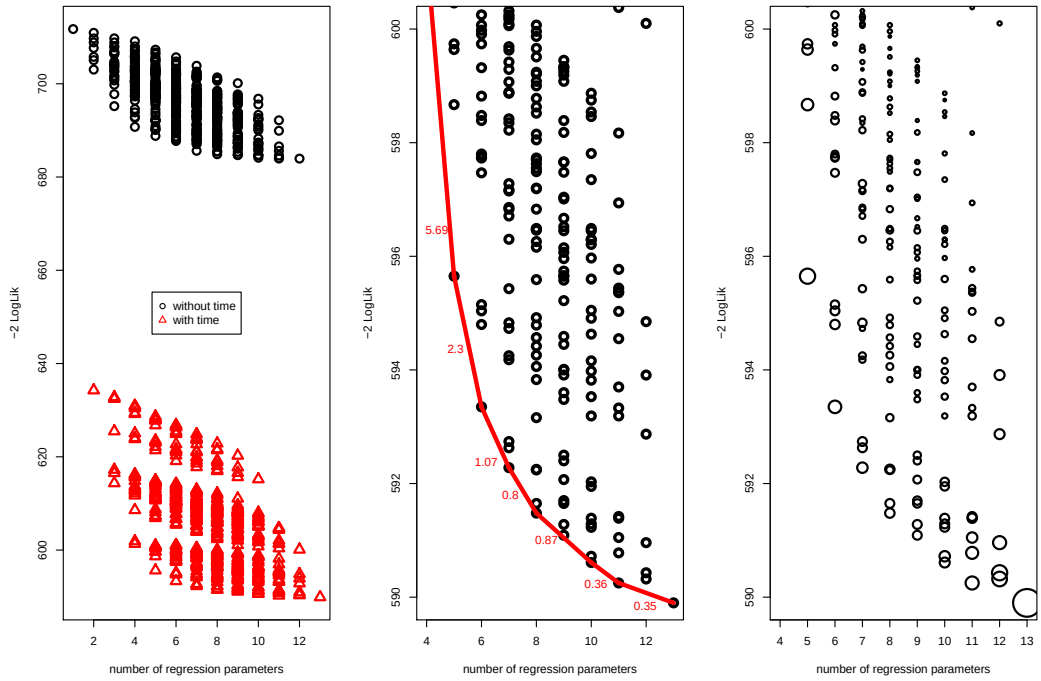

Figure 5.8: Case study III - L_n vs dimension.: Left panel - Enriched scatter plot; Middle panel - Zoomed in MELCC with reduction in L_n; Right panel - Zoomed in model stability plot

a model of dimension 4. Again this confirms that there is often lack of agreement in model selection depending on what the objective is and what methodology is used.

The basic scatter plot is shown in the left panel of Figure 5.8. There is clear separation between those models that include the variable 'time post first surgery' and those that do not, indicating this to be an important variable. Intuitively this makes sense, given patients are more likely to have recorded a second surgery if their followup time is longer.

The middle panel of Figure 5.8 shows a zoomed version of the scatter plot and the MELCC for this data. As before we can make several 'simple' observations from this plot: For example, dimensions 9 and 12 do not lie on the MELCC. Hence, regardless of the choice of penalty multiplier, one can rule these out of the subset of *optimal* models, that is regardless of the choice of λ, no GIC method will select a final model with 8 or 11 explanatory variables.

As expected, examination of model stability indicates that the model including only the variable time is not only the best model at dimension 2, but extremely stable.

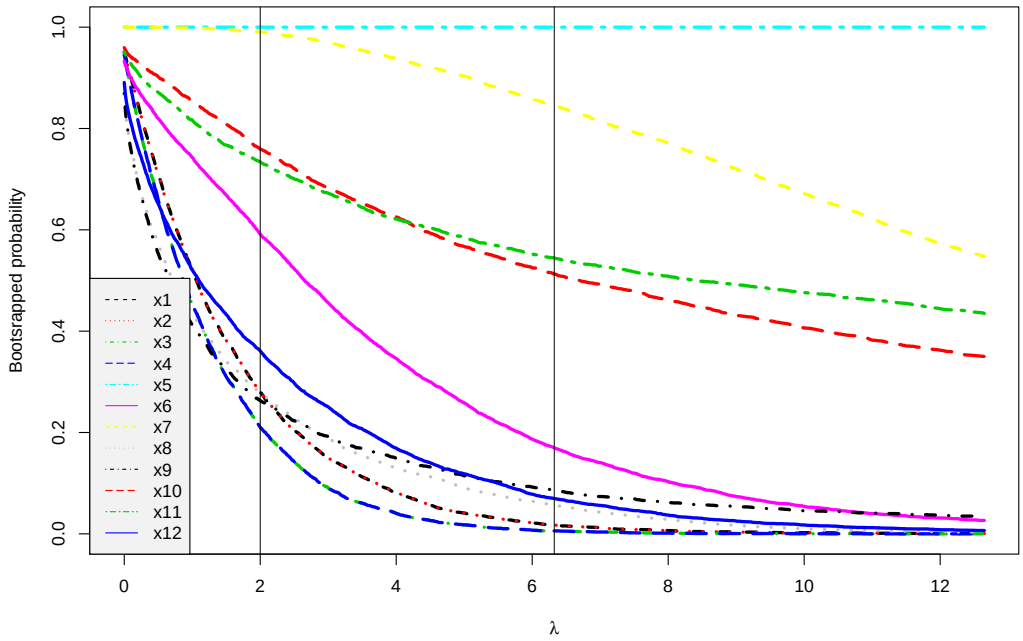

Figure 5.9: Variable inclusion plot for case study III

However, as seen in the right hand panel of Figure 5.8, beyond dimension 3 through to dimension 12 there are many models that are identified almost as frequently as the optimal models lying on the MELCC. This should prompt consideration of models that lie within close proximity of the optimal model at these dimensions.

The variable inclusion plot shown in Figure 5.9 gives an indication of which variables would be included, dependent on the value of the penalty parameter. Notice the variables x_1 and x_2 have trajectories which are exactly the same. This is true of variables x_3 and x_4 and is due to the criteria we set for these variables which represent the dummies for age and location respectively. The horizontal line at probability 1 corresponds to the variable time (x_5) and again demonstrates the importance of this variable which would be included in any final model. We note also the high line corresponding to x_7 which for both AIC and BIC criteria has probability in excess of 0.8. Beyond this one could argue for the inclusion of variables x_{10} and x_{11} using criteria with penalties at or below the BIC penalty. However, only lower penalties would include x_6 and it would be difficult to argue for the inclusion of other variables regardless of penalty parameter.

5.5 MODEL SELECTION PLOTS WITH MONOTONE POLYNOMIALS

We have limited our examples and discussion to date to look at the specific role of these graphical techniques using binary responses and logistic regression. However, we can use any form of GLM as long as we can estimate a loss and measure model complexity. When considering polynomials we note that unconstrained standard polynomial regression belongs to the GLM family with normal errors and identity link function. Hence, model selection for polynomial regression could follow the approaches described earlier in this chapter. The only decision would be on what is the maximum degree polynomial to consider which can usually be determined by the nature of the problem. If we set q_{max} to be highest degree polynomial we are willing to consider then we note that there will be $\#\mathcal{A} = 2^{q_{max}}$ potential models. However, one would pay particular attention to the marginality principle in that lower order terms be retained in the model should a higher degree term be deemed necessary. In light of this we would usually only consider $\#\mathcal{A} = q_{max} + 1$ models in a polynomial regression model selection, dramatically reducing the task. We consider this problem for unconstrained polynomial regression through a simulation in Section 5.5.1 and demonstrate using our model selection plots the models which are generally included.

5.5.1 *Simulated data example*

We use an example data set from the data generated based on the function:

$$p_3(x) = x^5.$$

We simulate responses for n equidistant design points over $[-1, 1]$ from the linear regression models

$$Y_i = p_3(x_i) + \epsilon_i, \quad x_i = -1 + 2\frac{i-1}{n-1}, i = 1, \ldots, n$$

where the errors are independent $N(0, 0.3^2)$.

Given this is a quintic polynomial we would like to consider models of higher degree for the purpose of model selection. In this instance we set q_{max} to be 9 and note there are $\#\mathcal{A} = 2^9 = 512$ models to consider. However, if we obey marginality this number reduces dramatically to 10 models.

We start by looking at the basic scatter plot of L_n by dimension in the left panel of Figure 5.10 where the triangular symbols identify models that obey the marginality principle. Immediately we see a distinct separation between models with L_n between 100 and 200 and the reminder which all have $L_n > 250$. The latter of these is indicative of models only including the intercept and one or two higher order terms not obeying the marginality principle. For example one such model is a model that includes the intercept and the degree 8 term without any of the lower order terms. Another, is the model that includes the intercept and the degrees 4 and 6 terms only. Noticeable is that none of these poor performing models include an odd degree term, something which is not unsurprising given the degree of the underlying polynomial that the data was generated from was odd. We also note that the models that obey the marginality principle do not on the whole provide the 'best' model at each dimension. This is not surprising given the nature of the problem however we note that the benefits of obeying marginality have in general been seen to outweigh the drawbacks. In the right hand panel of Figure 5.10 we demonstrate the model stability plots using the weighted bootstrap. Again the models obeying marginality are highlighted in red and we note there are many models that are seen to be comparable at most dimensions.

5.5.1.1 *Some comments on model selection plots for monotone polynomials*

Whilst these plots are of some use when looking for model selection for polynomial regression we note that there are many issues to be considered. First, as can be seen, the decision to plot all the models, including those that do not obey marginality, is something that needs to be carefully thought through. As mentioned previously if we obey the marginality principle then the model selection problem becomes very simple in theory, with only a small number of competing models to consider. Second, whilst in standard polynomial regression the degrees of freedom (or the model dimension) is easy to define (that is, the dimension is simply the degree of polynomial plus one), we

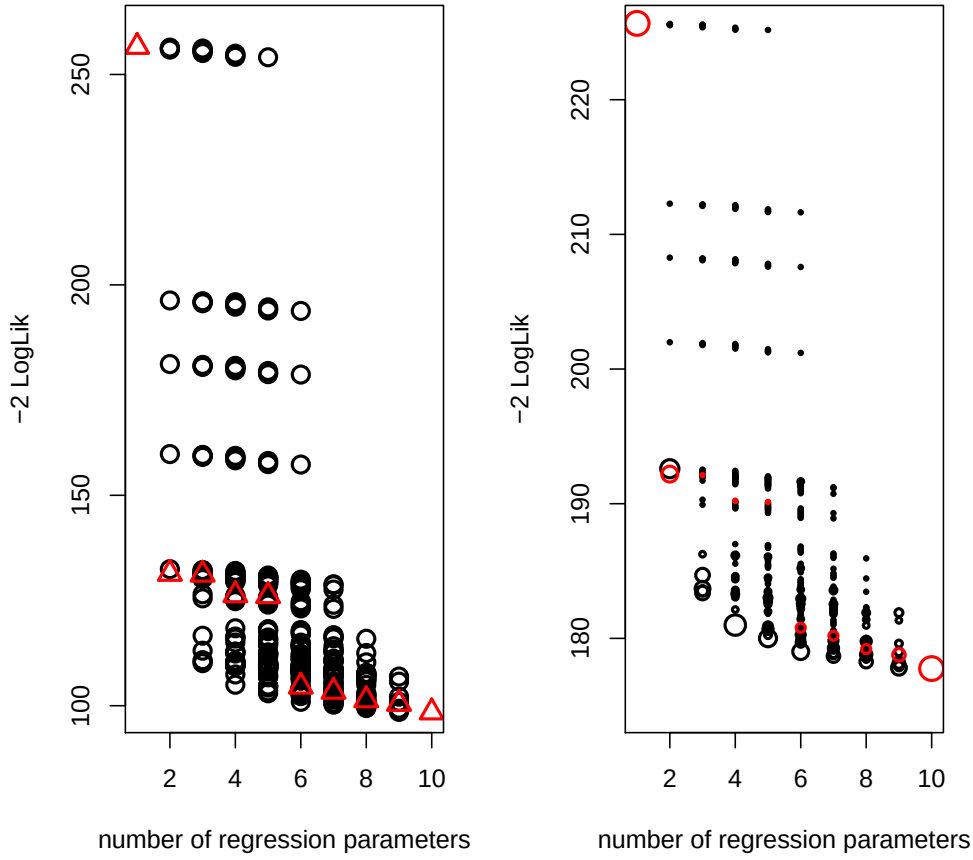

Figure 5.10: Left panel - L_n vs dimension for polynomial model selection using $p_3(x)$, red triangle = marginality obeyed, black circle = marginality not obeyed. Model stability plots using bootstrapping, red circle=marginality obeyed back circle=marginality not obeyed.

observe that introducing monotonicity constraints makes defining the degrees of freedom more problematic. This obviously has a larger impact when using an information criteria approach for model selection, although one may argue that the comparison of AIC values, for example, could be seen to be independent of the estimation of the degrees of freedom. However, in light of these issues, in Section 5.5.2, we make further use of the bootstrapping methodology described in Chapter 3 to demonstrate more suitable plots, and model selection techniques, through our real world examples of constrained polynomial estimation.

5.5.2 *Case study IV - Brain function data*

Adapted from Firmin, Müller & Rösler (2011, 2012) we describe our example data set. The Latency of Motor evoked potentials (MepL) distribution for each of 50 subjects was derived from the triple stimulation technique (TST) amplitude measurements, where the expected TST amplitude $E(Y_{ijk})$, which is a function of δt, is an unscaled survival function of the MepL distribution. The maximal value of the expected TST amplitude served as baseline for 100% survival and proportions thereof resulted in $1 - F_i(x)$, where F denotes the true but unknown MepL distribution, $x = \delta t$, and i is the index for the ith subject. TST amplitude measurements were available at stimulus delays in steps of 1ms and for most stimuli with smaller delay extension, $j < 9$, there were at least $n_{ij} = 3$ repeated measurements per subject.

An initial examination of the 50 individual datasets involved fitting cubic, quintic and septic unconstrained polynomials using standard least squares. Of the 150 unconstrained models fitted we observed only 13 which provided monotone fits, further strengthening the argument of the need for the constrained version of polynomials. We consider for demonstrative purposes one of these patients and describe the differences between the three different polynomial methodologies considered in Chapter 2 with Figure 5.11, where we show the data and three fits of degree 7 polynomials (the unconstrained fit, the monotone over the whole real line fit, and the monotone over $[0, \infty)$ fit). The horizontal axis has been extended outside the range of the observed x data values, for illustrative purposes only, to demonstrate where the curves become non-monotone. We note that, over the range of the data, there are noticeable differences between the unconstrained polynomial fit and the two constrained polynomial fits, with little difference between the latter two.

Using our example patient's data shown in Figure 5.11 we have {5, 3, 3, 3, 3, 3, 3, 3, 3, 3, 2, 1} observations at each of the increasing design points from 0 through 11 in increments of 1. The particular interest in Firmin, Müller & Rösler (2011, 2012) was to determine the inflection points of the estimated survival function, that is, the modes (peaks) of the estimated MepL density which coincide with the inflection points of the estimated mean curves. Firmin, Müller & Rösler (2011, 2012) estimated these

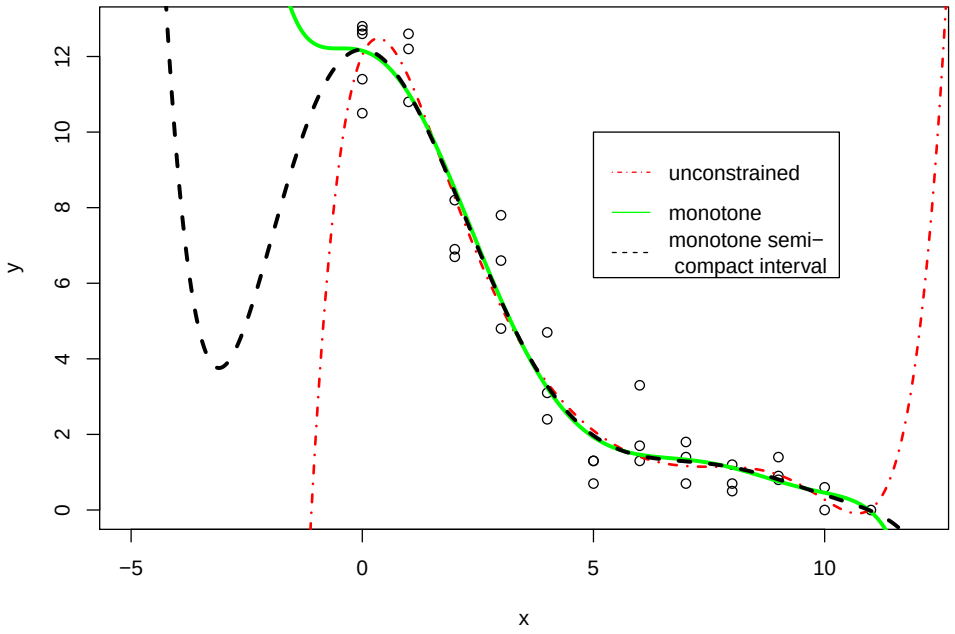

Figure 5.11: Different degree 7 polynomial fits to brain function data of an individual.

using monotone smoothing splines. We will re-examine this problem using monotone polynomials , which have the advantage of well defined inflection points in Chapter 6. However, for now we only consider the associated model selection problem, and for demonstrative purposes restrict our focus to those polynomials that are monotone over $x \in [0, \infty)$.

We consider using the following polynomials:

$$p(x; \boldsymbol{\beta}) = \beta_0 + \beta_1 x + \beta_2 x^2 + \ldots + \beta_q x^q \tag{5.4}$$

where in this instance q is not restricted to be odd, as with previously described monotone polynomial fitting, but $\boldsymbol{\beta}$ such that $p(x)$ is monotone over $x \in [0, \infty)$ reflecting the nature of the brain function data.

For model selection purposes, as suggested by Shao (1996), we consider using the 'm out of n' paired bootstrap, implemented over a range of values for m. For each

bootstrap sample we calculate the prediction errors for fitted monotone polynomials of degrees $q = 1, \ldots, 11$, using:

$$PE(q) = u^{-1} \sum_{j=1}^{u} (y_{i_j} - p(x_{i_j}; \hat{\boldsymbol{\beta}}_q))^2. \tag{5.5}$$

where $\hat{\boldsymbol{\beta}}_q$ is the estimated vector of coefficients for the monotone polynomial of degree q, based on the bootstrap sample; and (x_{i_j}, y_{i_j}), $j = 1, \ldots, u \geq n - m$, are the (unselected) out-of-bag (x, y) pairs. For each bootstrap iteration the polynomial with degree $q = \arg\min PE(q)$, is selected as the 'best' model.

To reflect the nature of the data and the design matrix (that is, a relatively small number of unique design points), we use a stratified version of the paired bootstrap as described in Müller & Welsh (2005, 2009). Stratification facilitates that individual bootstrap samples are not too different to the full data, for example ensuring that samples have sufficient design spread to enable reasonable polynomial fits. We let $D = \{d_1, d_2, \ldots, d_{n_d}\}$ be the set of values at which we have design points, and $\#D = n_d$ be the number of unique design points. We also let $\#d_i$ be the number of values at each design point d_i, $i = 1, \ldots, n_d$. If $n_d \leq m$ we sample one value from each unique design point, and randomly select with replacement and with probability proportional to $\#d_i$ another $m - n_d$ design points to sample additional observations from. If $n_d > m$ we simply sample without replacement m design points and select one observation from each.

Minimising prediction error as the objective function, as described previously, we examine the selection probabilities of the differing degree polynomials. We consider only polynomials up to degree 8 (restricted by the number of design points) that are constrained to be monotone over a semi-compact region and values of m ranging over 10, 15, 20, 25, 30 and 35. Results are described in Table 5.3 and Figure 5.12. One can immediately determine that the best fitting degree of constrained monotone polynomial, irrespective of the size of m, is degree 6, thus highlighting the benefits of our improved algorithm over previous algorithms which were restricted to fitting monotone polynomials of only odd degree.

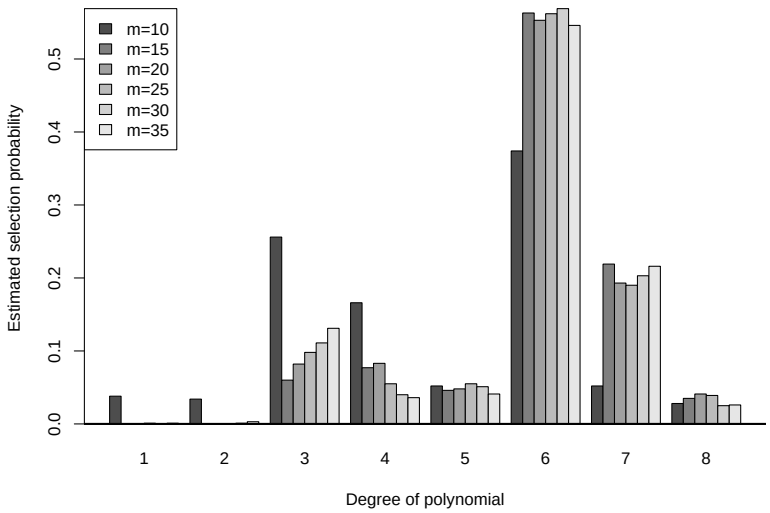

Figure 5.12: Selection probabilities for the m out of n bootstrap by degree of polynomial and m for brain function data

Table 5.3: Brain function data: Proportion of models selected using m out of n bootstrap by degree q and m

	Degree of polynomial p							
m	1	2	3	4	5	6	7	8
10	0.038	0.034	0.256	0.166	0.052	**0.374**	0.052	0.028
15	0.000	0.000	0.060	0.077	0.046	**0.563**	0.219	0.035
20	0.000	0.000	0.082	0.083	0.048	**0.553**	0.193	0.041
25	0.001	0.000	0.098	0.055	0.055	**0.562**	0.190	0.039
30	0.000	0.001	0.111	0.040	0.051	**0.569**	0.203	0.025
35	0.001	0.003	0.131	0.036	0.041	**0.546**	0.216	0.026

For completeness we carried out the bootstrapping without stratification and the results were comparable to the results obtained from the stratified bootstrap. However, in some simulation runs higher degree polynomials could not be fitted or were highly variable, mainly due to attempting to have more parameters estimated than unique design points.

5.5.3 *Case study V - Estimation of dental age*

In forensic science, Demirjian, Goldstein & Tanner (1973) described a method for estimating dental maturity or dental age by reference to a scoring system created from

the radiological appearance of teeth. Using population based data they described a simple plotting method with subsequent extraction of percentiles for age estimation for a given maturity score. Subsequently regression based methods were mainly used for smaller data sets with Chaillet, Nystrom, Kataja *et al.* (2004) comparing the use of polynomials to model the relationship between age and the dental maturity score with the original percentile method in Demirjian, Goldstein & Tanner (1973). Using data from Flood, Mitchell, Oxnard *et al.* (2011) in which $n = 284$ adolescent teenagers were examined, with the objective of fitting models to describe the relationship between a dental maturity score (x) and age of child (y), we consider modelling this relationship using polynomials, and incorporate the underlying widely accepted view in the forensic science literature that the relationship be monotone increasing. Due to the nature of how the score (x variable) is generated, we also constrain the polynomial to be monotone over the compact interval $[0, 100]$. Hence we consider using the polynomials described in (5.4).

For model selection purposes, we again consider using the 'm out of n' paired bootstrap. Here the sample size is $n = 284$ and we consider values of m ranging over 20, 25, 30, 35, 40, 50, 100 and n. For each bootstrap sample we calculate the prediction errors for fitted monotone polynomials of degrees $q = 1, \ldots, 11$, again using (5.5).

Table 5.4 shows the proportion of times the polynomial of degree q is selected. Consistent results are obtained regardless of the magnitude of m, with a distinct mode in the distribution of the proportions at $q = 5$. This is visualised in Figure 5.13. These results indicate a higher degree polynomial than that usually used when polynomials are fit to this type of data in forensic science, which is usually $q = 3$. However, we suspect that the main reason for this is that with increasing degree polynomials the chance of obtaining a monotone fit, as needed for this type of data, is smaller and cubics are generally the only polynomials which will give a monotone fitted curve. Our methodology now provides more flexibility in the degree of polynomials that can be used for these types of examples.

Table 5.4: Estimation of dental age data: Proportion of models selected using m out of n bootstrap by degree q and m

	Degree of polynomial q										
m	1	2	3	4	5	6	7	8	9	10	11
20	0.005	0.176	0.080	0.136	**0.219**	0.098	0.085	0.051	0.067	0.043	0.040
25	0.000	0.128	0.070	0.143	**0.236**	0.104	0.078	0.066	0.074	0.047	0.054
30	0.000	0.106	0.052	0.131	**0.249**	0.107	0.126	0.075	0.076	0.022	0.056
35	0.000	0.078	0.037	0.146	**0.308**	0.108	0.089	0.050	0.084	0.038	0.062
40	0.000	0.050	0.025	0.123	**0.318**	0.115	0.109	0.071	0.084	0.042	0.063
50	0.000	0.043	0.022	0.123	**0.326**	0.105	0.088	0.075	0.113	0.045	0.060
100	0.000	0.007	0.003	0.121	**0.487**	0.085	0.047	0.048	0.130	0.030	0.042
n	0.000	0.000	0.016	0.106	**0.474**	0.088	0.032	0.032	0.141	0.056	0.055

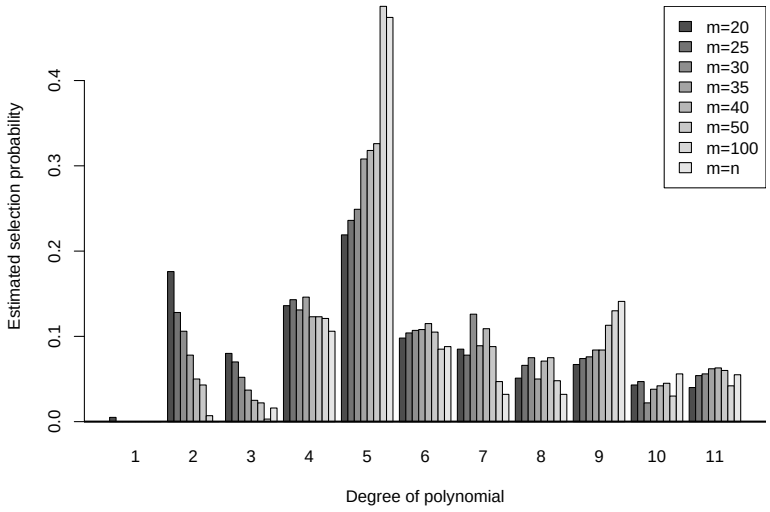

Figure 5.13: Selection probabilities for the m out of n bootstrap for dental maturity example by degree q and a range of m values.

5.6 ADDITIONAL USES FOR MODEL SELECTION PLOTS

In this section we consider two additional challenges that are (potentially) problematic in any model selection problem. First, we examine how our plots perform in the presence of (highly) correlated predictors. Second we consider extensions of our plots when outliers are present in our data, and propose some robust extensions to our methodology.

5.6.1 *Correlated predictors*

We investigate the effect of correlated predictors through a simple example. Let us again consider a similar logistic regression example to that described in Section 5.3. For this example we chose $n = 150$, have 4 potential predictors with the parameter values for the β's (excluding the intercept) set to be $(1.5, 0, 0, 2.5)$. We set the correlation between x_1 and x_2 to be either $0.95, 0.85, 0.7$ or 0.5 in four separate simulations.

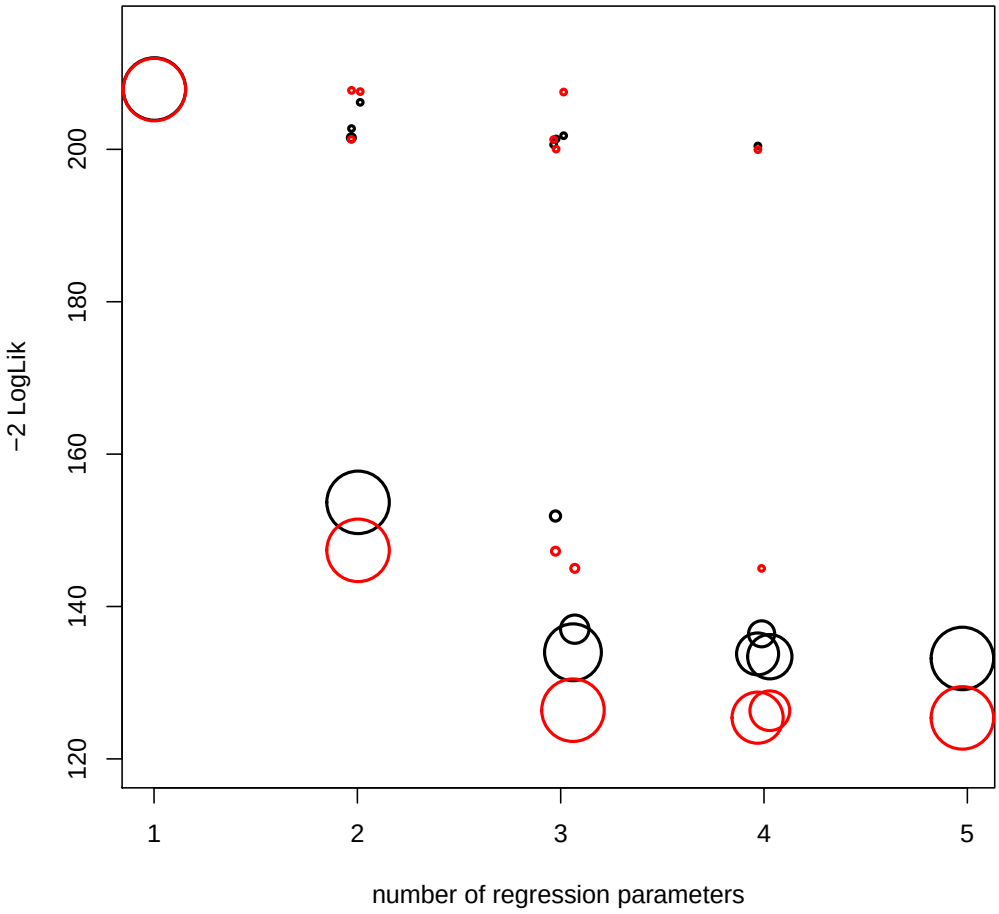

Figure 5.14: Model stability plot for simulated correlated data. Correlations between x_1 and x_2 are 0.95 (black circles) and 0.5 (red circles)

Figure 5.14 shows the impact of the correlated variables on the model stability plots. Superimposed are the results from the simulation with correlation of 0.5 (red circles) and the correlation 0.95 (black circles). Using the smaller correlation (of 0.5) the dis-

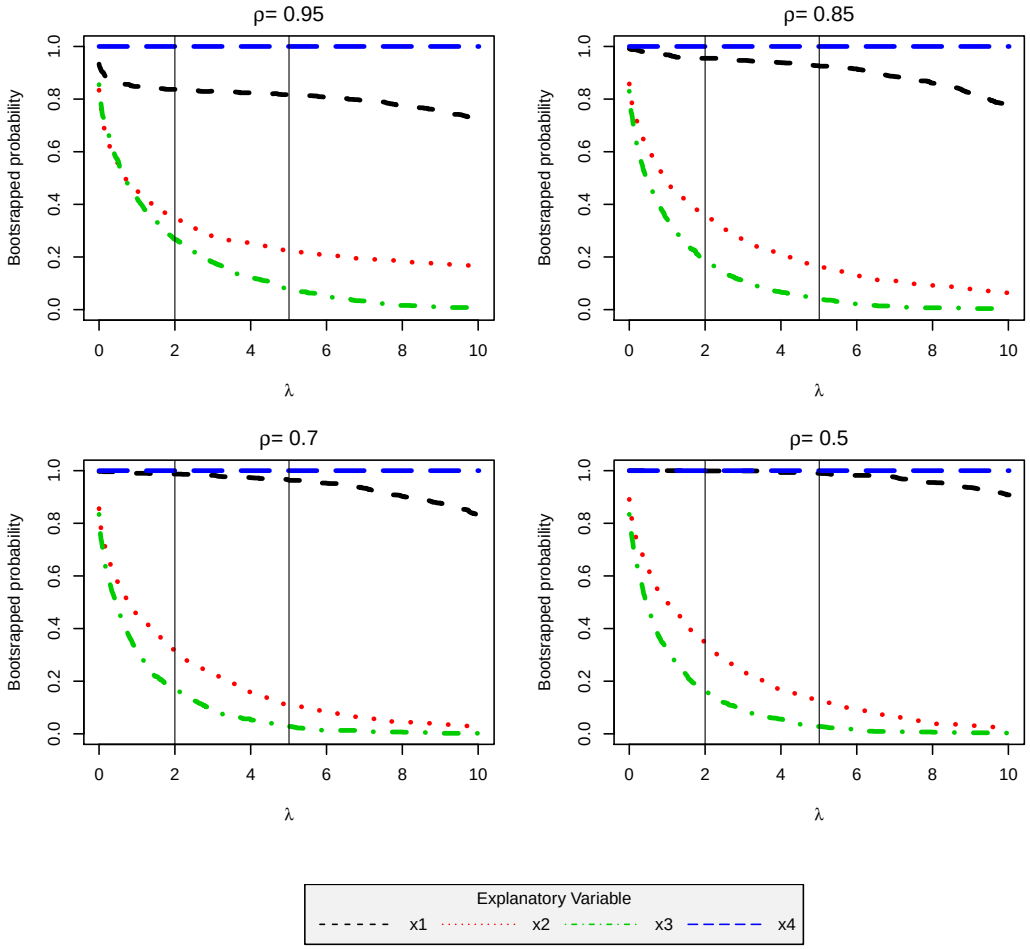

Figure 5.15: Variable inclusion plot for simulated correlated data. Correlations between x_1 and x_2 are 0.95 and 0.85, (top row) and 0.7 and 0.5 (bottom row)

tinction between competing models at dimension 3 (the underlying true dimension) is more pronounced, with only one prominent large red circle. However, even though the results are not as clear with the high correlation, there is still a subtle distinction at dimension 3 between the model that includes x_1 and not x_2 (largest black circle), and the model that includes x_2 and not x_1 (smaller black circle). This indicates that even in the presence of a high correlation we are still able in this instance to get some reasonable results, implying that our graphs are useful in this context.

Figure 5.15, which shows the variable inclusion plot for the four different correlations, is more informative. Clearly the bootstrapped probability of selection of the true variable x_1 from which the data was simulated is much higher than x_2 for a large

range of λ, even for high correlation values like 0.95. As expected the trade-off between selecting x_1 and x_2 becomes more apparent when the correlation gets very close to one.

5.6.2 *Outliers in data*

As with any applications in statistics outliers can cause problems. Given our methods depend on the likelihood as a loss function our plots are influenced by one or more outliers in the data whenever the likelihood is influenced as well. A simple solution to this is pre-processing the data and exclusion of outliers before estimation and selection. However, in the robustness literature deletion of outliers is often not the preferred way of dealing with this challenge. Therefore we prefer to extend our ideas by replacing the likelihood by robust alternatives of the loss function. These, as well as alternative choices of the penalty term, were recently reviewed in Müller & Welsh (2010). In particular one could consider log-quasilikelihoods, L1 and other loss functions for the robust estimation and selection of parameters (Ronchetti & Staudte, 1994; Konishi & Kitagawa, 1996; Ronchetti, 1997; Müller & Welsh, 2005, 2009). Generally speaking, as long as one can quantify the loss function and penalty in some sensible and robust fashion, then slightly modified versions of our plots can be produced for robust model selection. A step in this direction was suggested for GLMs in (Heritier, Cantoni, Copt *et al.*, 2009, p 159).

5.6.3 *Other uses for model selection plots*

We can use our model selection plots, or modifications thereof, in many additional ways to those described in this chapter. To date we have examined the logistic regression model, however these techniques can be easily extended to any generalised linear model or even all models in which an information theoretic approach can be taken for model selection. Adapting the model stability plots to show symbol size proportional to bootstrap probabilities across all dimensions, rather than dimension specific, is one alternative to what we have proposed as is the inclusion of plots to focus on models

included in model averaging. Given the simplicity of the graphs and the ease at which they can be expanded upon they can be easily applied and/or adapted with the sample code as made available in Murray, Heritier & Müller (2013).

5.7 DISCUSSION

We have shown, in the context of information criteria, that by using simple graphical techniques there are many ways to assist in the process of model selection and model building. The presented charts visualise how the choice of penalty multiplier can impact on the models selected and how volatile selected models are. The techniques show for each possible penalty multiplier the corresponding minimum description loss, separately for each possible model dimension. This demonstrates that any model selection procedure, having the simple form of Description Loss + $\lambda \times$ Model Complexity can be visualised. A particular strength of the graphs are that they help to address the questions of which variables to include, model stability, and potentially model averaging. We propose that this prevails regardless of whether the aim is inference or prediction, since varying the penalty multiplier allows examination of both.

Our techniques imply that L_n, a measure of description loss, can be calculated for all possible models. In situations where the number of variables is between four and twenty, that is the maximum number of models to be considered is between 16 and about $1,000,000$, our techniques will provide invaluable information. With only a few models hypothesis testing seems more appropriate than automated all subset procedures. On the other hand, with a growing number of variables the number of models becomes eventually infeasible to fully evaluate. In situations where p is too large, say $p > 20$, even with huge graphical displays it will become increasingly difficult to ascertain exactly what advantage can be gained when visualising all possible models. However, in these situations our techniques are still useful when the number of models is substantially reduced by pre-processing of the data and initial screening of models. How to best pre-process goes beyond the scope of this thesis. A simple and fast way for logistic regression models would be to use the logistic LASSO (Lokhorst, 1999; Roth, 2004) repeatedly (through bootstrapping), and to focus only on those models that ap-

pear on one of the repeated LASSO paths (see Müller, Scealy & Welsh, 2013, Section 4, for a more detailed explanation). Investigating interactions is more challenging and future work is needed. The grouped LASSO (Meier, van de Geer & Bühlmann, 2008) is one possible way to investigate interactions.

Our aim was not to suggest that using the shown techniques is the only way to go about model selection in future. Rather, we suggest that our charts together with their nice geometric properties assist with model selection and help the understanding of what is actually achieved when using some of the well documented techniques, such as AIC or BIC. In this respect we regard our diagnostic tools as very helpful, especially considering it is still widespread to use model selection techniques as "black boxes", or at best a recipe that has to be followed despite the consequences. Graphically displaying loss functions will greatly assist in the understanding of model stability, AIC, BIC and model selection.

We have also demonstrated that graphical displays can be a useful aid in model selection problems for isotonic regression. We demonstrated that use of bootstrap model selection methodology using the m out of n bootstrap can be beneficial. However, in our experience the selection of m through our real world examples has minimal impact on the model selection aspect of the problem, that is choosing the degree q. In our extensive simulations moderate values of m performed best but in some instances a stratified bootstrap proved to be a more efficient and sensible approach which, if employed, would also impact on the selection of m. We have also shown that in some instances, using a constrained polynomial bounded over a semi-compact region, chooses a smaller degree polynomial than the corresponding unconstrained or monotone polynomials as in described Murray, Müller & Turlach (2013).

6

USING MONOTONE POLYNOMIALS FOR DETECTING INFLECTION POINTS

SUMMARY

While many fully parametric monotone regression models exist, in particular in the growth curve literature, such models may miss important features such as the number and location of roots, extrema and inflection points of (higher order) derivatives of a regression function. Using a new approach to fitting monotone polynomials to data as described in Chapter 2 and in (Murray, Müller & Turlach, 2015), we demonstrate the effectiveness of using monotone polynomials to detect inflection points. We use bootstrap methodology to determine the number and location of inflection points and show, through extensive simulations, that monotone polynomials are not only effective at estimating the number and location of inflection points when the underlying function is itself a monotone polynomial, but also when it is not. Our simulations show that the results are also reasonably robust to mis-specification of the degree of monotone polynomial.

Using two real world data sets we compare the smoothing spline approach of Firmin, Müller & Rösler (2011, 2012), to estimate the location of inflection points, with the use of monotone polynomials. We show that the latter provides more stable solutions and has the added benefit of simplistic implementation, with only one tuning parameter to determine, that is the degree of the polynomial, as opposed to smoothing spline based solutions, which have many parameters to be determined.

6.1 INTRODUCTION

In this chapter, we describe the results obtained in the last three months of candidature, which show the applicability of the work discussed in the previous chapters, with a specific focus on detecting features of higher order derivatives, for example inflection points. Inflection point problems are important issues in many physical applications and have particular prominence in the growth literature, for example human growth curves (see, for example, Mirman, 2014; Panik, 2014), plant growth, and even in financial applications such as growth of a company (for example defining the point where the company stops growing). However, to date, when data is involved, the problems have been solved for the majority of the time in one of two ways: either, by using the mathematical relationship derived from differential calculus and fitting the resulting (and often) non-linear models to the data or; by making use of the many smoothing spline techniques available when there is no underlying physical theory available. Such techniques have been applied in many situations with a varying degree of success with some software packages already available for such problems. For example the R packages `inflection` and `RootsExtremaInflections`, based on unpublished work at the time of writing (Christopoulos, 2014a,b), allow the estimation of *one* inflection point from a given set of data. More flexible, especially when the aim is specifically to look at growth curves, is the range of techniques employed in the package `Automatic Maxima Detection` (Dandurand & Shultz, 2010) with illustrations of the use of this software shown in Dandurand & Shultz (2011). The authors of this package demonstrate, using growth curve data, how the location of multiple inflection points can be determined. In particular they make use of the smoothing spline approach implemented in the `fda` R package (Ramsay, Wickham, Graves *et al.*, 2013). Consequently, the detection of inflection points are subject to choice of parameters to enter into the smoothing algorithms.

This apparent lack of simple but effective and reliable techniques to estimate the number and location of inflection points, in monotone increasing or monotone decreasing relationships, has led us to investigate the use of monotone polynomials for such purpose. In this chapter we describe through empirical investigations two scenarios:

1. Six different situations where inflection points are the focus and we know the data generating model;

2. Two real world scenarios, in neuroscience and early human growth, where the underlying literature suggests that the relationship between x and y must be monotone.

6.2 INFLECTION POINTS

We note that for a given monotonic polynomial function

$$p(x) = p(x; \boldsymbol{\beta}) = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_q x^q \tag{6.1}$$

the first and second derivatives are given by:

$$p'(x) = \beta_1 + 2\beta_2 x + 3\beta_3 x^2 + \cdots + q\beta_q x^{q-1} \tag{6.2}$$

and

$$p''(x) = 2\beta_2 + 6\beta_3 x + 12\beta_4 x^2 + \cdots + q(q-1)\beta_q x^{q-2} \tag{6.3}$$

and note that inflection points are defined by the location of the roots of the second derivative.

In order to determine the location of the inflection points using monotone polynomials we describe in this chapter a two step approach:

1. Using the bootstrapping methodology described in Chapter 3, select the 'best' degree of monotone polynomial and hence the number of inflection points.

2. With a second bootstrap determine the location of the inflection points by looking at modes in the bootstrap distribution of inflection points.

The remainder of this chapter is structured as follows: In Section 6.3, we describe simulated data used in this Chapter and in Section 6.4 we describe the results of our

numerical experiments using this simulated data, in which one can determine the number and location of inflection points through the fitting of monotone polynomials. In Section 6.5 we compare and contrast the use of monotone polynomials to other commonly used smoothing spline techniques with real world data. First, we use data from our motivating example (Firmin, Müller & Rösler, 2011, 2012) who utilised monotone smoothing spline techniques similar to those implemented by Dandurand & Shultz (2010) to locate inflection points; second we use data from the human growth curve literature. Finally, in Section 6.6 we provide some discussion and final comments.

We would like to emphasise that this particular chapter is focussed on the practical application and use of monotone polynomials in a real world scenario. We demonstrate that in certain situations our proposed monotone polynomial methodology works with simulated data from different underlying data generating functions both polynomial and non-polynomial. Subsequently we demonstrate the effectiveness in real world data and provide a comparison to the monotone smoothing spline approach. Mathematical development of theory and inference for inflection point detection using monotone polynomials is not discussed here and is beyond the scope of this thesis.

6.3 SIMULATED DATA EXAMPLES

To describe the effectiveness of using monotone polynomials to detect inflection points in data, we first generate six datasets; three based on a data generating model from underlying functions that are polynomials and monotone, and three based on monotone trigonometric functions. Each are there to provide us with a range of inflection points to detect, which in our simulated data is zero to five. To generate data with inflection points for monotone polynomials we define a function that has n_i roots, integrate twice whilst adding constants to ensure monotonicity. For the trigonometric functions we use their properties to ensure inflection points are at the desired locations. Using this process we generate our six datasets which are shown in Table 6.1.

The initial three (Equations (6.4)–(6.6)) are generated with the aim of describing how well monotone polynomials perform when the underlying data generating function is itself a monotone polynomial. The final three (Equations (6.7)–(6.9)) aim to determ-

Table 6.1: Description of simulated data for inflection point detection.

$f(x)$		Function Description	inflection points (#)	location of inflection points
$15 - x$	(6.4)	Negative decreasing linear relationship	0	NA
$\frac{1}{10000}(-5x^5 + 158.3x^4 - 1546.17x^3 + 4019.4x^2 - 5000x) + 14$	(6.5)	Monotone polynomial	3	$(1.1, 6.3, 11.6)$
$100000(-4.76x^7 + 187.3x^6 - 2744.7x^5 + 17897x^4 - 4505x^3) + 12$	(6.6)	Monotone polynomial	5	$(0, 2.4, 5.7, 8.9, 11.1)$
$2 + 12e^{-0.2x}$	(6.7)	Exponential decay model	0	NA
$3\pi - \frac{x}{2} + \cos\left(\frac{x-\pi}{2}\right)$	(6.8)	Monotone trigonometric function	2	$(0, 2\pi)$
$5\pi - x + \cos\left(x - \frac{\pi}{2}\right)$	(6.9)	Monotone trigonometric function	4	$(0, \pi, 2\pi, 3\pi)$

ine how well monotone polynomials perform when the underlying function is not of polynomial form. Furthermore, we note that the functions in (6.4) and (6.7) have zero inflection points. Anecdotal evidence has suggested that smoothing spline based methodology would, in such situations, tend to overestimate the number of inflection points.

All six of these functions are specifically designed to mimic our motivating example in addition to exploring the impact of fitting monotone polynomials to data generated from non-polynomial functions. We describe all these data with example simulated data points in Figure 6.1, which displays underlying data generating function with data superimposed, first derivative, and second derivative with the location of the inflection points indicated in each instance. The red lines corresponding to modes in the distribution of the first derivative, the red dotted lines to troughs in the distribution of the first derivative. Giving our aim of mimicking the motivational problem described in Firmin, Müller & Rösler (2011, 2012), we generate our design points such that $x_i = i - 1$, $i = 1, \ldots, 13$ (restricting the maximum degree polynomial that can be fitted to be 11 in all instances), with our model:

$$y_{ij} = f(x_i) + \epsilon_{ij}$$

where the errors are such that $\epsilon_{ij} \overset{i.i.d.}{\sim} N(0, 0.5^2)$ with $j = 1, 2, 3$ and all $n_i = 3$. In each case this gives an overall sample size of $n = 39$.

6.4 NUMERICAL EXPERIMENTS - RESULTS

We make extensive use of bootstrapping to examine the effectiveness of monotone polynomials for the estimation of inflection points, which includes choosing the 'best' degree, q, of the polynomial to be fitted following the approach described in Section 5.5.2. We use an m out of n stratified bootstrap, and for each bootstrap sample we calculate the prediction errors for fitted monotone polynomials of degrees $q = 1, \ldots, 9$, using:

$$PE^b(q) = u_b^{-1} \sum_{j=1}^{u_b} (y_{ij} - p(x_{ij}; \hat{\boldsymbol{\beta}}_q^{*b}))^2 \quad b = 1, \ldots, B. \tag{6.10}$$

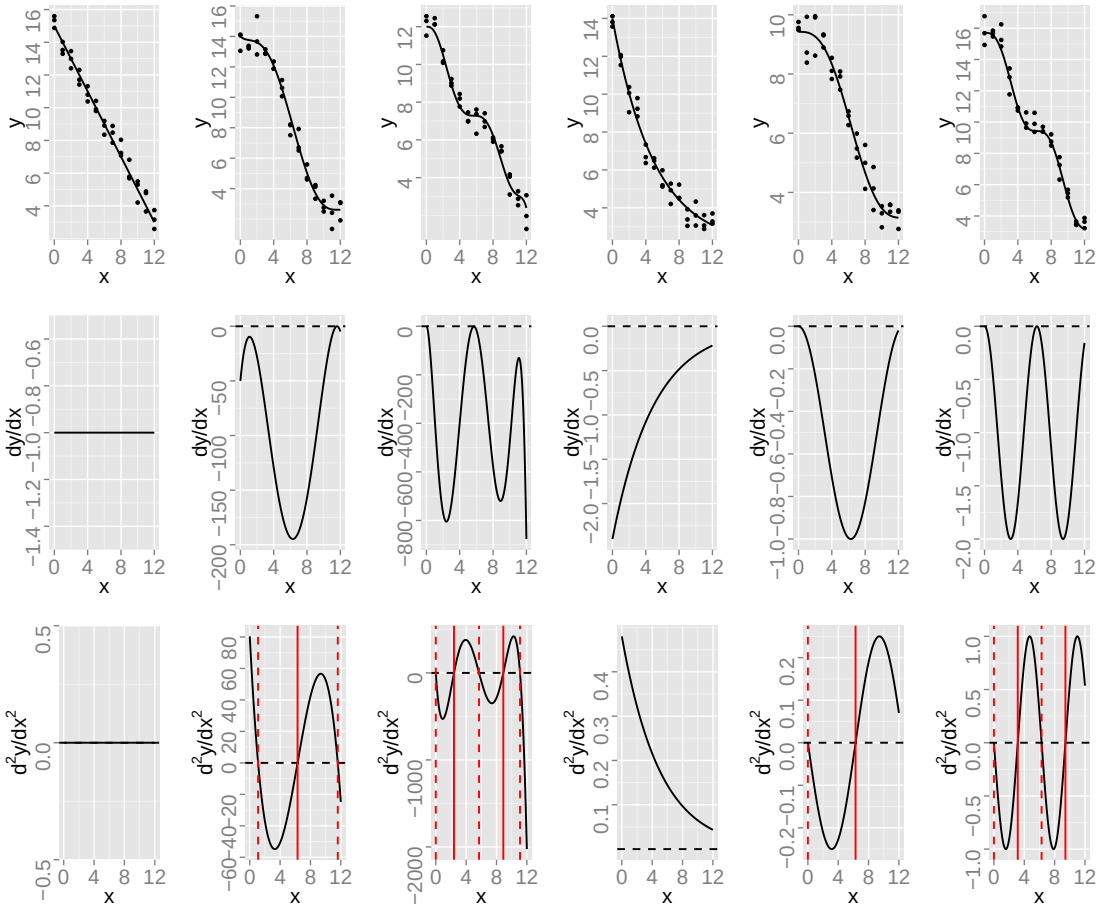

Figure 6.1: Simulated data (top tow), first and second derivatives (middle and bottom row respectively) for inflection point simulations based on (6.4)–(6.9) (from left to right columns). Vertical lines in second derivatives indicate the location of inflection points.

where $\hat{\beta}_q^{*b}$ is the estimated vector of coefficients for the monotone polynomial of degree q, based on the b^{th} bootstrap sample; and (x_{i_j}, y_{i_j}), $j = 1, \ldots, u \geq n - m$, are the (unselected) out-of-bag (x, y) pairs. For each bootstrap iteration the polynomial with degree $q^b = \arg\min PE^b(q)$, is selected as the 'best' model.

6.4.1 *Selecting the degree of monotone polynomial*

We present results initially for selecting the degree of polynomial. Using an m out of n stratified bootstrap, m ranging over 13, 20, 26, 33, n =39, selected to enable sensible stratification over the design points and thus maintaining the underlying design structure in our bootstrap samples. We simulated 1,000 data sets from each of the functions

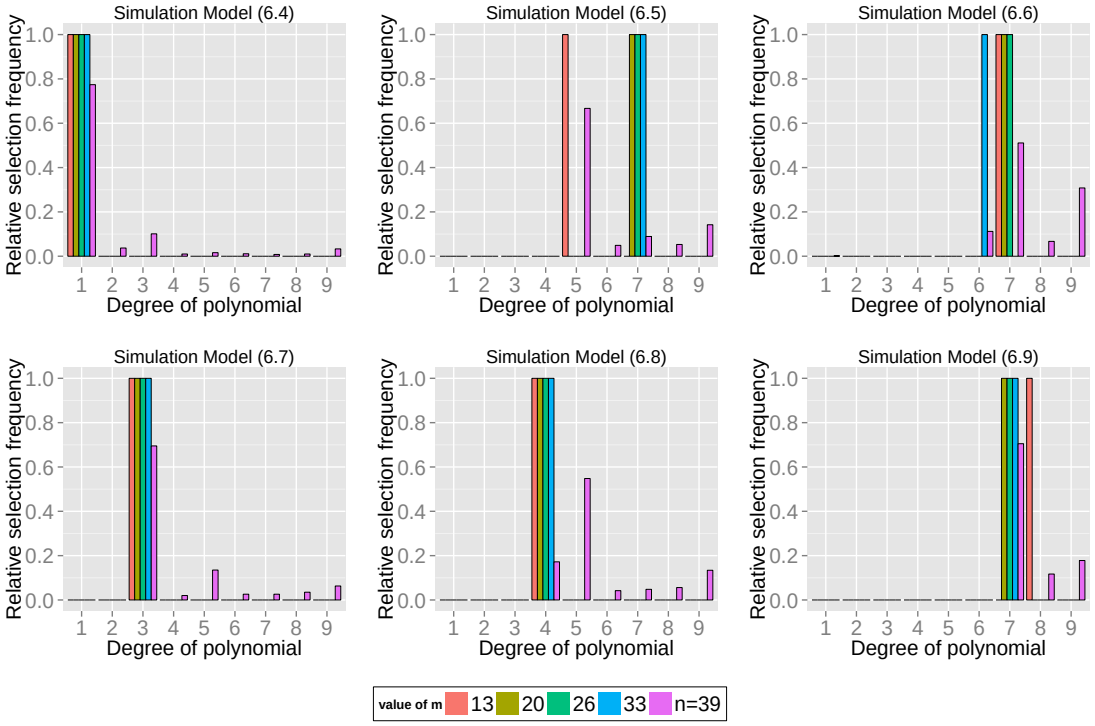

Figure 6.2: Model selection probabilities for simulated data with monotonicity constraint lower bounded at 0,using m out of n bootstrap and minimising prediction error for various values of m for the simulated data from monotone polynomial functions (6.4)–(6.6) and non-monotone polynomial functions (6.7)–(6.9).

described in (6.4)–(6.9), based on $B = 1,000$ m out of n stratified paired bootstrap samples for each and calculated the prediction error. For each simulated sample, for each of the six simulated data examples, we calculated the degree of polynomial that minimised the prediction error. The results of these are shown in two separate ways. First, considering all polynomials constrained to be monotone over the range $[a, \infty)$, (with $a = 0$ in this instance) thus allowing even degree polynomials. Second, we examined polynomials to be constrained to be monotone over the whole real line. The results from both these approaches are described in Figures 6.2 and 6.3 respectively.

The top row of Figure 6.2 describes the results for fitting monotone polynomials, over a semi-compact interval, to the simulated data simulated from polynomial functions (generated from (6.4)-(6.6)) with the bottom row the corresponding results for the non-polynomial functions (generated from (6.7)-(6.9)). We note that for the simple linear regression function (top left plot), regardless of the chosen m, we are able to accurately select the degree of polynomial as $q = 1$. This itself is useful when trying to determine

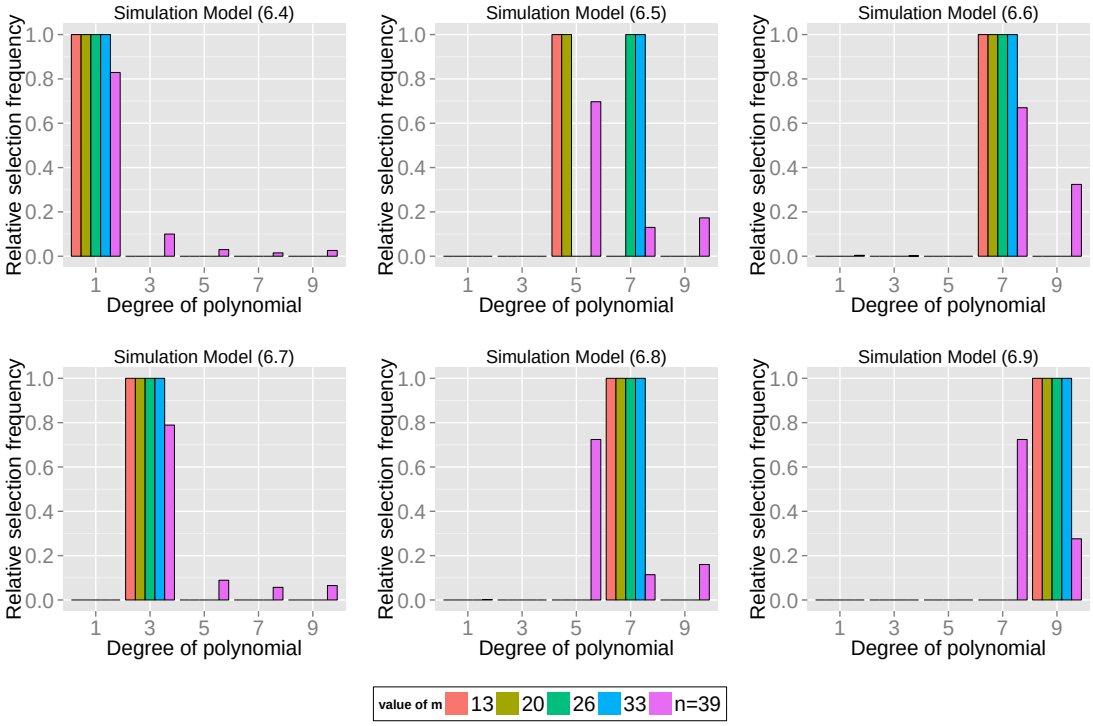

Figure 6.3: Model selection probabilities for simulated data with monotonicity constraint unbounded,using m out of n bootstrap and minimising prediction error for various values of m.

the number and location of inflection points, since when one is searching for inflection points, it seems sensible to answer the question of whether there are any at all.

The top middle panel describes the results when the simulated data was from a degree 5 polynomial. We see the choice of degree of polynomial is not consistent from simulation run to simulation run and is dependent on the size of m. For the smallest and largest values of m the degree 5 polynomial appears to provide the most suitable fit, whilst for moderate values of m a higher degree 7 polynomial appears more sensible. This may be reflective of the choice of the location of inflection points, with the lowest being close to the lower boundary of x values .

We note that when considering the underlying function being of degree 7 (top right panel), with inflection points not on the lower bound of the semi-compact interval, that we achieve consistent results for correct identification of polynomial degree through our model selection process for the lower range of values of m considered $(13, 20, 26)$, and a selection of a degree 6 polynomial when higher values of m are considered. This provides evidence that when selecting the degree of a polynomial one should

potentially use a range of values of m to examine robustness of the model selection process.

When considering the non-polynomial data generating functions in the bottom row of Figure 6.2 we note reasonably consistent results. The bottom left panel indicates that a cubic monotone polynomial was considered the best approximation of the exponential decay function, regardless of the magnitude of m. The middle panels suggest also reasonably consistent results in that increasingly larger degrees of monotone polynomials are needed to accurately describe data with increasing number of inflection points. In the middle panel we see that a degree 4 polynomial appears to provide a reasonable approximation to the trigonometric function with two inflection points, whilst the bottom right panel indicates a degree 7 or 8 monotone polynomial would provide a reasonable approximation with moderate values of m corresponding to degree 7.

Considering monotone polynomials over the whole real line we see similar results in Figure 6.3. For the polynomial function of degree 1 we consistently select a model of degree 1, whilst for the polynomial function of degree 5 we select either degree 5 or 7 and for the the polynomial function of degree 7 we select mainly degree 7, with the former and the latter selections made regardless of the size of m. For the non-polynomial functions again a consistent selection of polynomial is seen. The bottom left panel again indicates a degree 3 monotone polynomial, and the bottom middle and bottom right consistently suggest degree 7 and 9 monotone polynomial again regardless of m.

The results from these model selections are used in the next section to determine the numbers and locations of inflection points.

6.4.2 *Number and location of inflection points*

Whilst model selection reduces the problem of finding the location of inflection points, in some instances the model selection may not provide a clear solution on the 'best' degree of polynomial. Furthermore, if the set of models to select from includes monotone polynomials that are constrained to be monotone over both the whole real line and over a semi-compact interval, one clear solution to the model selection problem

may be more difficult to achieve. In data generated from (6.4) it is obvious that only polynomials of degree $q = 1$, regardless of the magnitude of m, should be considered. However, when the data is more complex, for example with the data generated from (6.5), which not only has more inflection points, but has the added difficulty of inflection points lying close to the boundary of the observed x values, we encounter more difficult problems. We will use this data as an example for the remainder of this section to demonstrate the flexibility of monotone polynomials for detecting inflection points.

Using the results provided from the model selection, we see that polynomials of degree 5 and 7, on both a semi compact interval of the from $[0, \infty)$ and over the whole real line, are viable solutions. In light of this we investigate all four of these possibilities to see the impact on the inflection point location.

Using bootstrapping over the range of values of m as specified earlier, we consider the number of inflection points and describe the results in detail for one of the simulated data sets. However, having reviewed numerous examples, these results are generalisable to other datasets. We note at this point that the total number of inflection points detected may not be the most useful representation of the effectiveness of this approach, especially when higher degree polynomials are used. In light of this we present in Table 6.2 several *useful* metrics: the total number of inflection points detected, the number inside the range of values of the design points, and the number within a close margin of being inside this range (in this instance we indicate this margin to be one unit and describe the number of inflection points within the range $[-1, 13]$). We note the following: in fitting a degree 5 monotone polynomial to this data (either over the whole real line or over a semi-compact interval lower bounded at 0), we consistently produce through our bootstrap samples, regardless of the size of m, three inflection points, as required from the model the data was generated. Furthermore, the vast majority of these inflection points are within the range of design point values or close to. For example, we see that with a degree 5 semi-compact interval monotone polynomial, and $m = 13$, out of the $1,000$ bootstrap samples all of them produced a model with three inflection points. Of these, 871 bootstrap samples had three inflection points inside the range of design point values and 994 had three inflection points within one unit of the range of design point values.

Similar results are observed for the degree 5 monotone polynomial constrained to be monotone over the whole real line. However, it is noticeable that the magnitude of m does have an impact when comparing these two models in that for low m the polynomial fit over the semi-compact interval has a higher proportion of inflection points inside the range of design point values whilst as we increase m (for example to $m = 33$ or $m = n$) the polynomial fit over the whole real line has more inflection points within the desired range. However, overall it is obvious that three inflection points are sensible for this data regardless of whether we constrain the polynomial to be monotone over the whole real line or over a semi-compact interval.

To demonstrate the difference in the four comparable models, Figure 6.4 shows the fitted curves for an example data set generated using (6.5). We note that all four models give a very similar fitted curve with the minor differences being observed for both degree 7 (red and green lines) but outside the range of values of the data. There was no difference between the two degree 5 polynomial fits, hence one black line is shown for both. Inflection points estimated are also in very similar positions. The two degree 7 polynomials (red and green lines) have almost identical inflection points whilst the degree 5 polynomials (black line) have marginally different locations to the degree 7 fits. Finally, not shown in this picture, are two additional inflection points for the degree 7 polynomials. In this example they both have two additional negative inflection point locations, at $(-28, -45)$ when the polynomial is constrained to be monotone over the whole real line, and at $(-13, -5.2 \text{ billion})$ when the polynomial is constrained to be monotone over the semi-compact interval.

This is further highlighted in Figure 6.5 where we show the distribution of inflection points for this data set, over a range of values of m using the m out of n stratified bootstrap, for the four different models under consideration. We note there is very little difference in the location of the inflection points regardless of which of the four different models are used. In fact the only major difference arises when the incorrectly specified degree 7 monotone polynomial is selected. In this instance the three inflection points are still identified however the bootstrap estimates of the location of the first inflection point (which in this case would be at $x = 1.1$) appears to be slightly more variable. This consistency between the methods suggests that even if the degree of

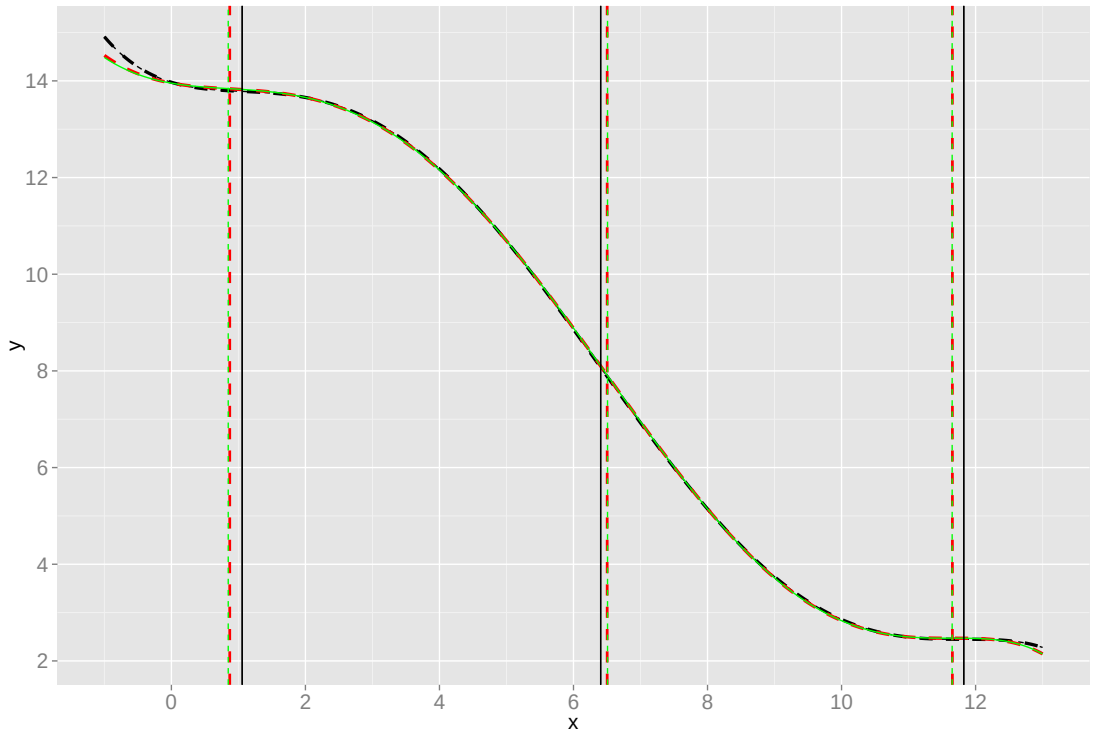

Figure 6.4: Fitted models for degree 5 and 7 monotone polynomials over the whole real line and semi-compact interval with inflection points based on an example data set generated from (6.5). Green lines monotone monotone polynomial fit (and inflection point location) constrained to be monotone over a semi-compact interval; red line degree 7 constrained to be monotone over the whole real line; black lines degree 5 fits and inflection points.

the polynomial is overestimated this methodology would still be useful for detecting inflection points within the range of values.

As an example we take one such bootstrap sample of our example dataset and note that after fitting a degree 7 polynomial, constrained to be monotone over the whole real line, we observe inflection points at $(-246, -162, 1.48, 6.28, 10.66)$, noting the first two points are located well beyond the range of observed x values and are irrelevant for applications such as ours. We also note that the three inflection points detected are similar to the true location of the inflection points from the data generating model, that is $(1.1, 6.3, 11.6)$. Again, we see consistency in these results when other datasets are considered, suggesting there exists some robustness to model mis-specification.

We make some general comments on the other data sets. First, we note that recovery of inflection points, for instances where the inflection points are not on the boundary of the x values, appears to be well performed by monotone polynomial data fitting and

Table 6.2: Distribution of number of inflection points (out of 1,000 bootstrap samples) by type of polynomial fitted and size of m used for m out of n bootstrap sampling based on an example data set generated from (6.5).

		Degree 5 Semi-Compact			Degree 7 Semi-Compact					Degree 5			Degree 7				
	Inf. Points	1	2	3	1	2	3	4	5	1	2	3	1	2	3	4	5
$m = 13$	Total	0	0	1000	0	0	455	0	545	0	0	1000	0	0	416	0	584
	Inside	6	213	871	14	340	643	3	0	5	203	792	5	216	779	0	0
	Close	0	6	994	0	181	801	18	0	0	4	996	0	50	949	1	0
$m = 20$	Total	0	0	1000	0	0	431	1	568	0	0	1000	0	0	403	0	597
	Inside	6	178	818	12	341	640	7	0	5	197	798	0	217	783	0	0
	Close	0	6	994	0	174	803	23	0	0	1	999	0	46	952	2	0
$m = 26$	Total	0	0	1000	0	0	451	1	548	0	0	1000	0	0	393	0	607
	Inside	0	189	811	13	333	651	3	0	0	203	797	3	207	789	1	0
	Close	0	2	998	0	178	808	14	0	0	3	997	0	38	960	2	0
$m = 33$	Total	0	0	1000	0	0	437	1	562	0	0	1000	0	0	416	584	
	Inside	4	209	787	6	311	679	4	0	6	197	797	1	194	805	0	0
	Close	0	1	999	0	167	813	20	0	0	3	997	0	47	951	2	0
$m = n$	Total	0	0	1000	0	0	457	0	543	0	0	1000	0	0	398	0	602
	Inside	2	218	780	7	342	645	6	0	2	205	793	0	197	803	0	0
	Close	0	1	999	0	179	802	19	0	0	3	997	0	33	964	3	0

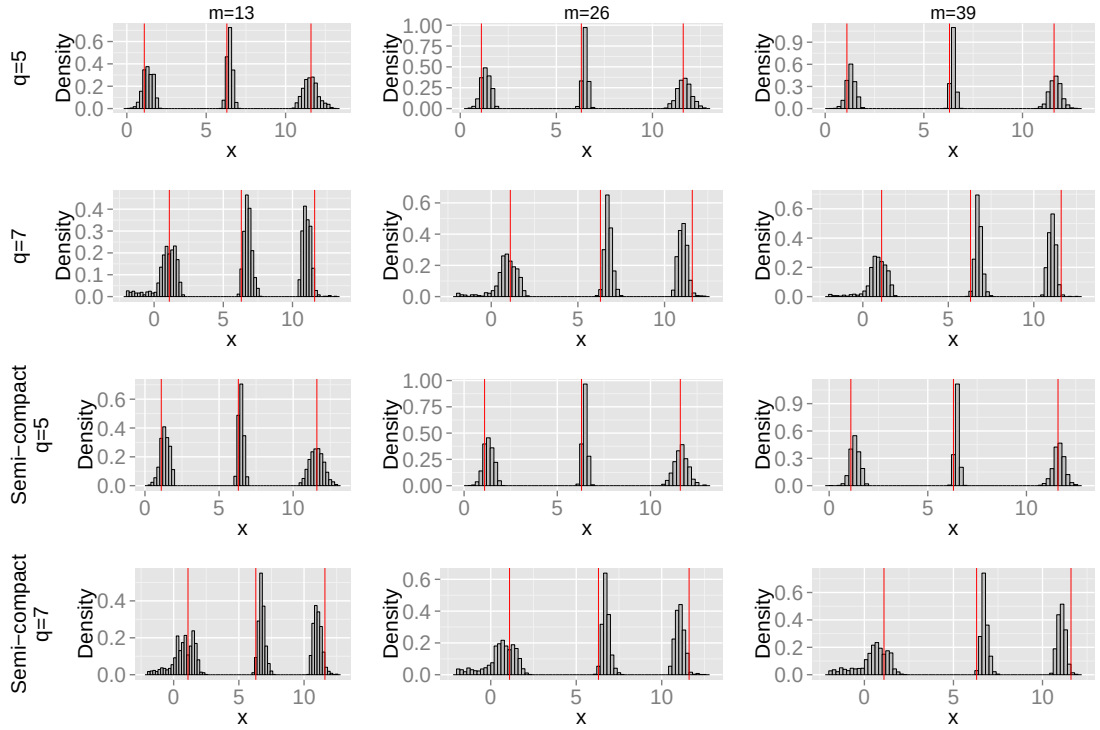

Figure 6.5: Distribution of inflection points at or near the range of values of the x data for an example data set generated from (6.5).

is reasonably robust to model mis-specification. Second, when inflection points on the boundary of the data have been specified in the data generating process, we note that the problem becomes more difficult. In these instances many simulations would not be able to detect the inflection point at the boundary. Third, the only simulated data set which provided some major problems was that of data generated from (6.7). Given the exponential decreasing nature of this data, it would appear that a polynomial representation, which does not have any inflection points was difficult to emulate. In fact in the 1,000 simulation runs for all values of $m < n$, a degree 3 polynomial was selected. Furthermore, using one simulated data set, approximately 80% of these simulations returned one inflection point inside the range of x values from the data.

Finally, an obvious extension using our methodology, is the estimation of confidence intervals for the location of inflection points . One relatively simple solution would be to fit mixtures of normals to processed bootstrap estimates (excluding outliers etc.) to determine the distribution of the inflection point locations. However, in order to do this we need to determine how to process the bootstrap estimates and address issues

such as: what to do with values outside the range of the data; how to define the *first* inflection point and; if the distribution of bootstrap estimates for each inflection point location overlap how does one consider the variability. Given there are many facets that need to be considered, we do not consider the problem further in this thesis.

6.5 REAL WORLD EXAMPLES TO DETERMINE THE NUMBER AND LOCATION OF INFLECTION POINTS

In this section we fit monotone polynomials to data with the aim of detecting inflection points, and compare the methods described earlier in this chapter using monotone polynomials, to those described in Firmin, Müller & Rösler (2011, 2012) who implicitly used Dandurand & Shultz (2010, 2011) through the R package `fda` (Ramsay, Wickham, Graves *et al.*, 2013). We use two real world data sets: first, the previously described brain function data (see Section 5.5.2 for more details) and second the well known and utilised Berkley guidance study growth data described in the introduction of this thesis.

6.5.1 *Firmin et al. brain function data*

Using the brain function data, with response MepL distribution (y) and explanatory variable delay time (x), we aim to determine the number and location of inflection points. Firmin, Müller & Rösler (2012) described a methodology that fits a smoothing spline to the (x, y) data using the `fda` package and the `smooth.monotone` function. Subsequently the fitted curve is evaluated at a fine grid of x values, the first and second order differences are taken in y with respect to x, to create numerical first and second derivatives. A search for modes (and troughs) in the distribution of the first derivative (or identification of roots of the second derivative) identify the corresponding inflection points. For a more thorough summary of their approach we refer to the supplementary material in Firmin, Müller & Rösler (2011). For one patient we initially compare

this approach with that of monotone polynomials, describe some limitations of the smoothing spline approach and finally provide a comparison for all patients.

6.5.1.1 *Patient X - Different fits*

We use one patient from the brain function data to initially describe the approach taken by Firmin, Müller & Rösler (2012) and one of the major drawbacks of their approach. In Figure 6.6 we show two fitted curves using the smoothing spline approach to the data. The first would generate two inflection points corresponding to modes in the distribution, the second would generate only one corresponding inflection point. The two plots in the left panel of Figure 6.6 demonstrate the difference in fits by 'tweaking' only one parameter in the smoothing algorithm. The right panel shows the dramatic impact it has had on the corresponding inflection points. We see with this simple change in the model fitted, the smoothing spline solution has been dramatically altered. Not only have the location of the inflection points changed but also the actual number of points. It is the non-robust nature of such a solution that has initially led us to fitting monotone polynomials to tackle such problems.

We consider this same problem, for the same patient data using a monotone polynomial approach. First, we demonstrate the simplicity of the problem graphically. Figure 6.7 shows the only decision to be made is on the degree of the polynomial, and this is a choice between a small number of competing models. In this case due to the limited number of design points, we consider only polynomials up to and including degree 9. Each of the curves shown demonstrate the solution to a monotone polynomial constrained to be monotone over the range $[0, \infty)$, using the sums of squares parameterisation. Note, using this parameterisation also allows us to consider even degree polynomials.

In order to determine the 'best' polynomial for this patients data we use the stratified bootstrapping methodology described previously, and use minimising the prediction error as the objective function. The top panel of Figure 6.8 shows bar charts for different values of m in the m out of n stratified bootstrap and demonstrates that a degree 6 monotone polynomial is most suitable (note this example was described previously in Section 5.5.2). The bottom panel shows the fitted curve with the location of the

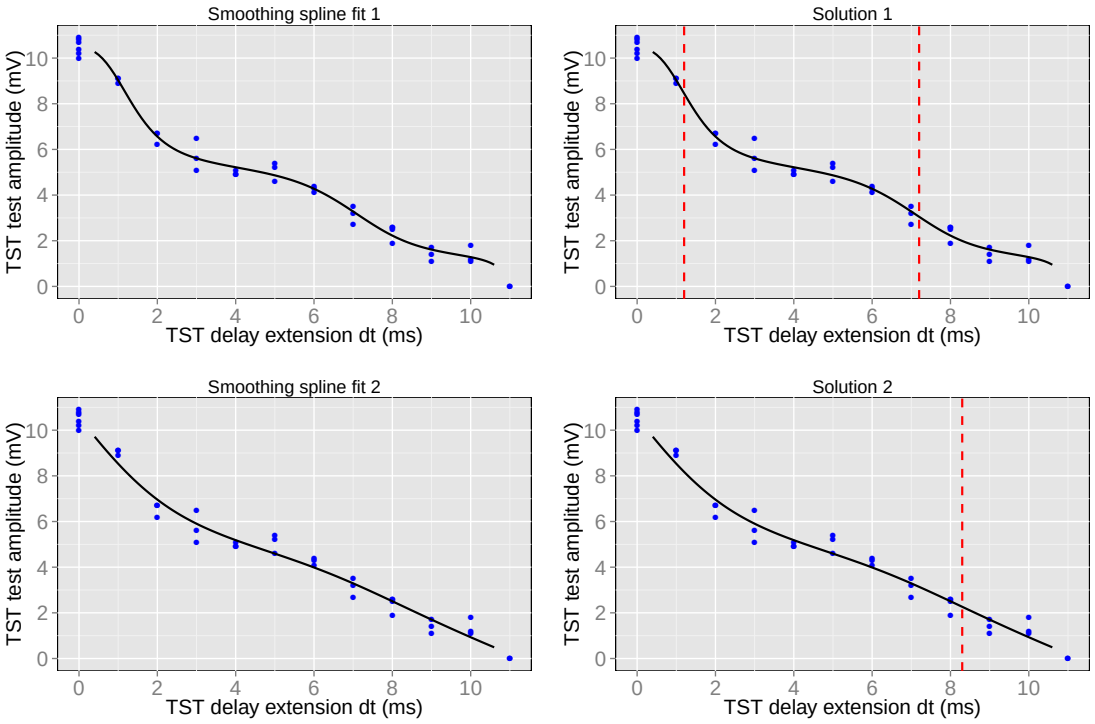

Figure 6.6: Example patient X data and two solutions to estimated the location of inflection points using the approach adapted by Firmin, Müller & Rösler (2011).

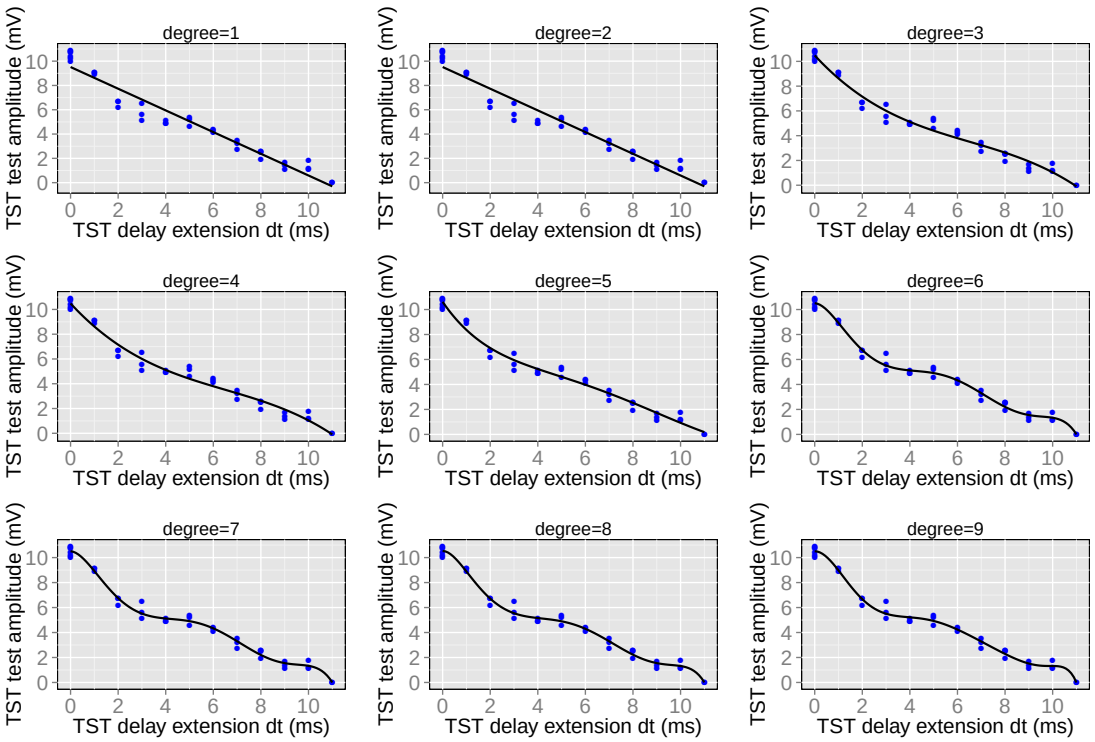

Figure 6.7: Different monotone polynomial fits for patient X.

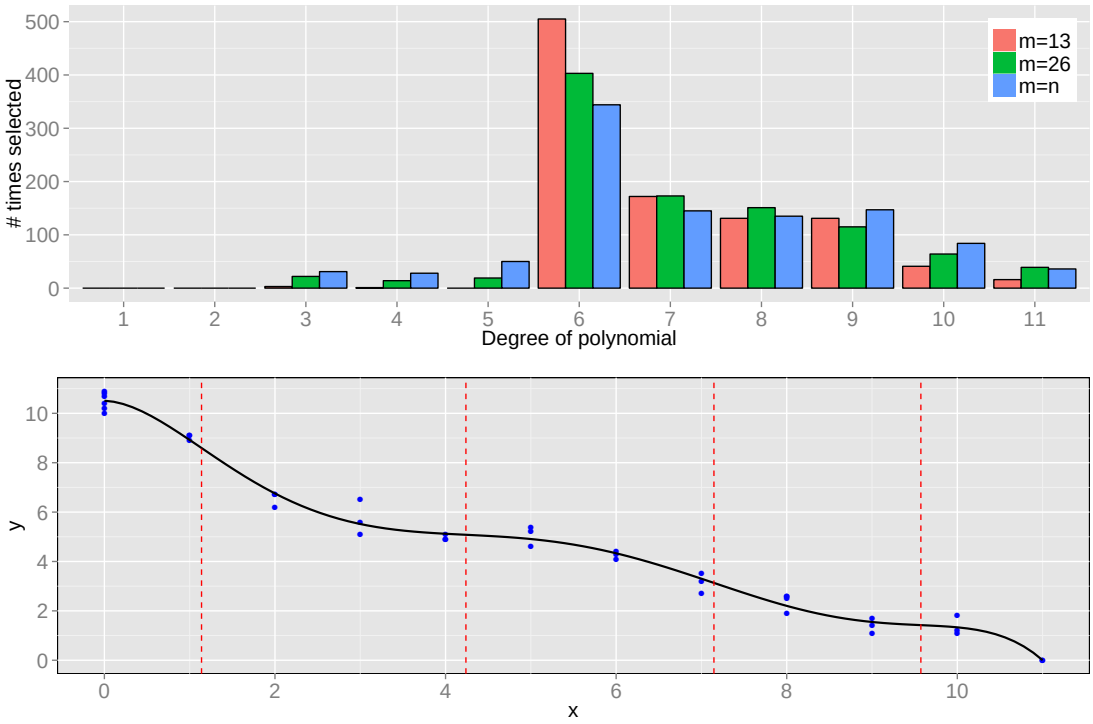

Figure 6.8: Model selection and fitted monotone polynomial curve for patient X with location of inflection points.

inflection points as determined by the 'best' fitting monotone polynomial to this data shown as vertical red dashed lines.

We note that four inflection points are detected, two corresponding to inflection points of local steepest decent. These are located at a similar position to that determined by Firmin, Müller & Rösler (2012) and can be seen more clearly in Figure 6.9 which provides the location of the inflection points detected (red vertical lines) and the bootstrap distribution of the location of such points. We note the clear separation in the four distinct modes, making the potential for further inference more simple. We finish this example by showing the comparison of the smoothing spline solution and the monotone polynomial solution in Figure 6.10 and note that not only are the fitted curves comparable in this instance, but the location of the inflection points corresponding to steep decent are at a similar location (1.14 and 7.15 for monotone polynomials and 1.20 and 7.20 for smoothing spline).

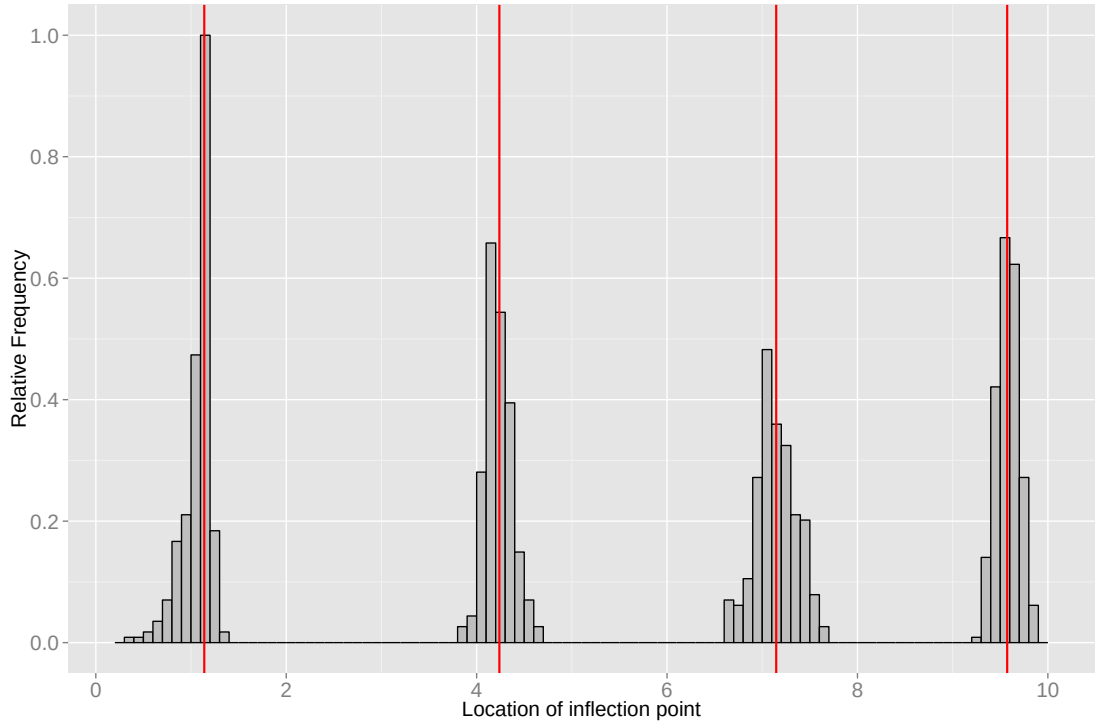

Figure 6.9: Bootstrap distribution of inflection points from fitted degree 6 monotone polynomial with estimated inflection points in red vertical lines.

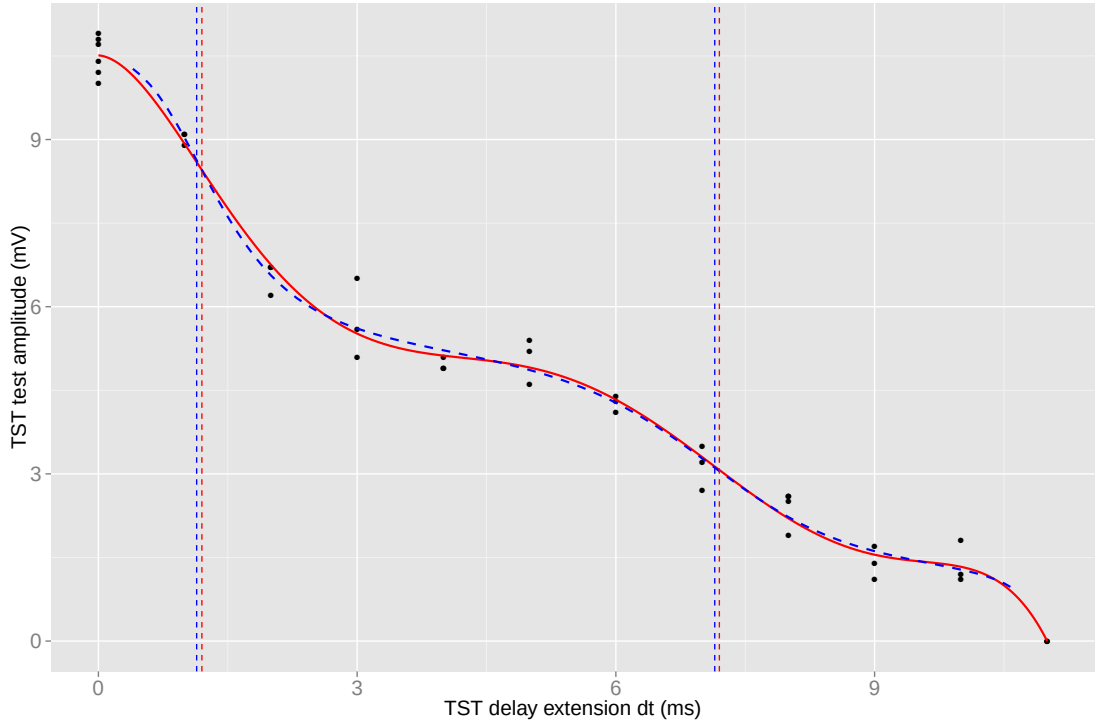

Figure 6.10: Fitted curves and inflection points (steep decent) locations for monotone polynomial fit and smoothing spline fit for patient X. Red: Monotone polynomial (deg 6); Blue: Monotone Smoothing spline.

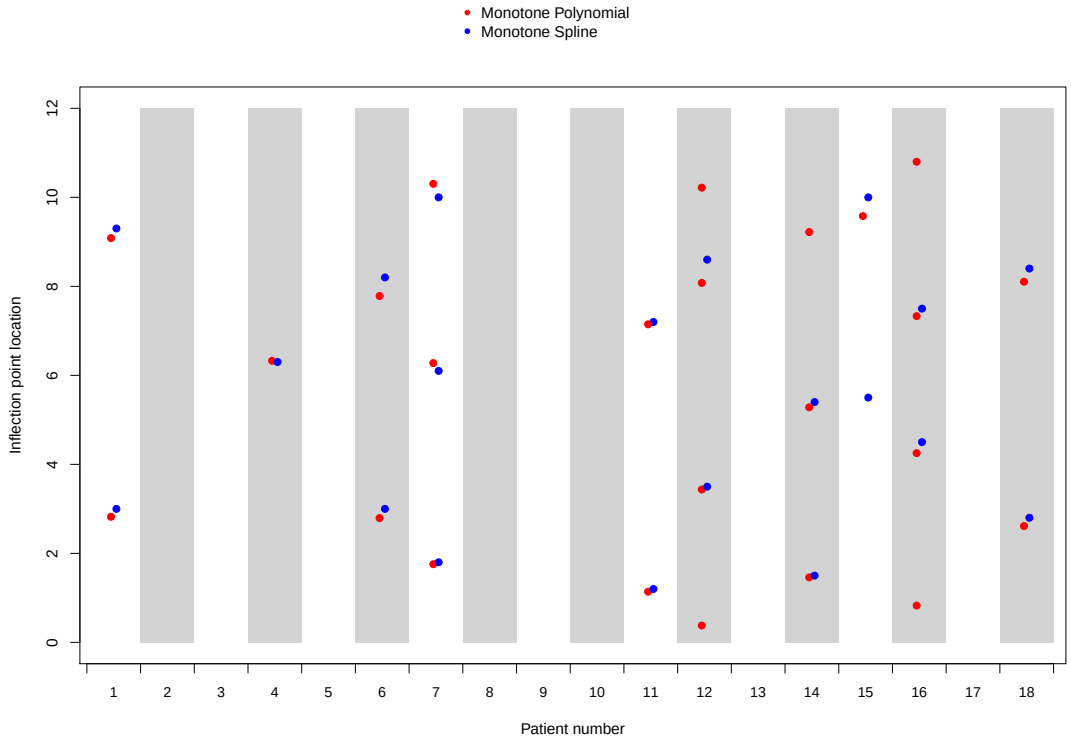

Figure 6.11: Comparisons of location of inflection points using monotone polynomials and monotone smoothing splines.

6.5.1.2 *Comparisons for the other patients*

Finally, we describe pictorially a simple comparison of the location of inflection points for the other patients in the brain function data. Figure 6.11 describes the location of inflection points for a variety of patients and we observe in many cases that the number and location of inflection points are similar. However, there are some differences, and there are some instances where no inflection points were detected, by either method.

6.5.2 *Growth curve examples from Berkley Guidance Study*

We finish this thesis where we started, by using the Berkley Guidance Study growth data, fitting monotone polynomials and monotone smoothing splines. We use the growth data of the 39 boys to examine the differences in monotone polynomial fits and the monotone smoothing spline fit using methodology described in the R package `fda` (Ramsay, Wickham, Graves *et al.*, 2013). We determined the degree of monotone polynomial using an m out of n non-parametric bootstrap for each boy with two choices

of $m = 20$ and $m = n = 31$ and determined the location of the inflection points by taking the roots of the second derivative of the fitted curve. We compared this to the solution provided by fitting monotone smoothers and evaluating the second derivative at a fine grid of values to determine the roots and location of inflection points. An example of the fitted curves and locations of inflection points is shown in Figure 6.12. We note, the blue curve denoting the smoothing spline solution and the black curve of the monotone polynomial solution (using $m = 20$ in the bootstrap model selection) are extremely close and both are good fits to the underlying data. However, the location of inflection points detected by the two competing models are not consistent. The monotone polynomial inflection point locations are denoted by the solid red vertical lines; the monotone smoothing spline inflection points by the dashed red vertical lines. In the left plot of Figure 6.12 we see that both solutions provide inflection points around 4.5, 11, and 14 years. However, the two additional inflection points are in different locations. The smoothing spline solution has another inflection point early whilst the monotone polynomial has an inflection point just after 6 years old. Similarly in the right panel of Figure 6.12 we see some very close agreement in the inflection points located around 12 and 15 years. However, again there is some discrepancy as to whether other inflection points are around 5-7 years as suggested by the smoothing spline solution, or on the edge of the solution space as suggested by the monotone polynomial solution. Of course it could be the case that neither is correct.

To complete this section we describe the number of inflection points detected for all 39 boys growth data using the monotone smoothing spline solution and the two versions of the monotone polynomial algorithm (with $m = 20$ and $m = 31$ for bootstrap model selection) in Figure 6.13.

Generally speaking, the $m = 31$ bootstrap gives more inflection points than when $m = 20$, which in turn gives more inflection points than the smoothing spline approach. We note that there could be several reasons for this. First, we have used a paired bootstrap and looked at minimising the prediction error as the objective function in order to determine the location of inflection points using monotone polynomials. Whilst both of these are useful in many instances, they may not be the *best* approach for this particular problem. The paired m out of n bootstrap does not retain the original design points in

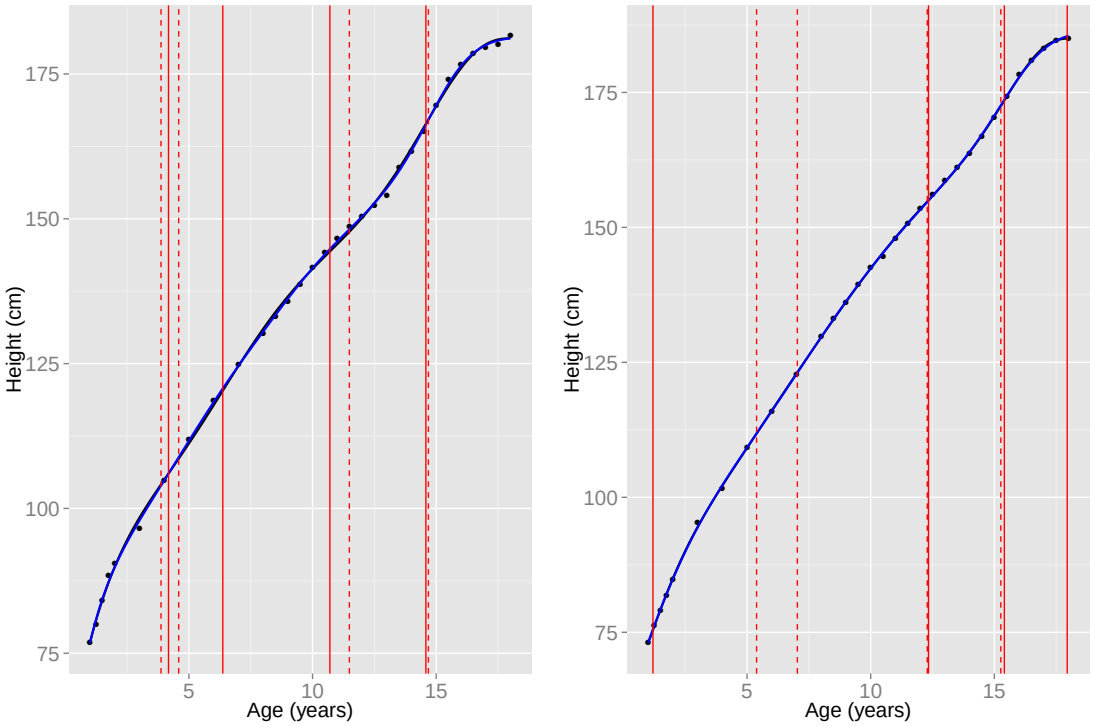

Figure 6.12: Comparisons of location of inflection points for two children from the Berkley Guidance Study using monotone polynomials (black curve and red solid vertical lines for inflection points) and monotone smoothing splines (blue curve and red dashed vertical lines for inflection points).

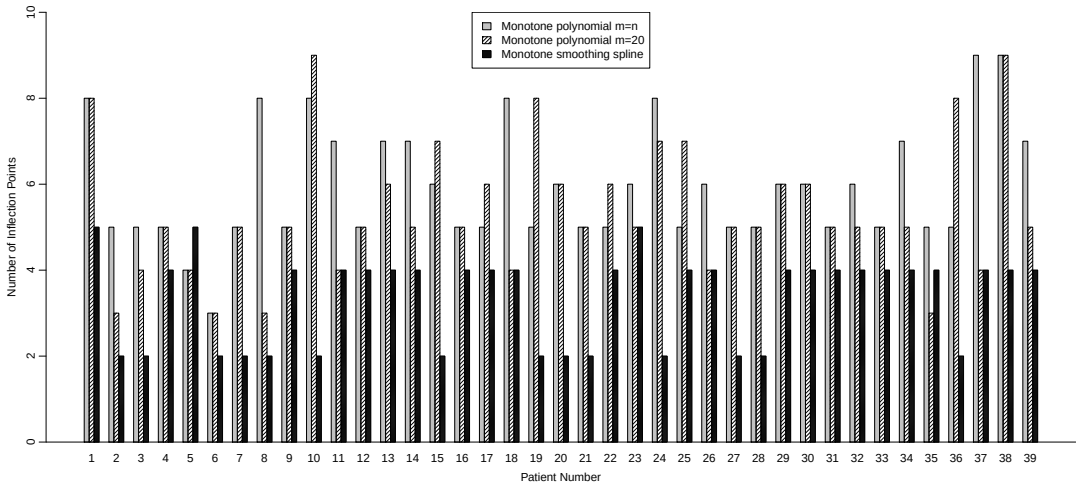

Figure 6.13: Comparisons of location of inflection points using monotone polynomials and monotone smoothing splines on children from the Berkley Guidance Study.

the growth data, whilst minimising the prediction error may not be the optimal solution when searching for inflection points. However, there are two clear conclusions: if our objective is to fit a curve to this data then the monotone polynomial solution is as reasonable as the monotone smoothing spline solution; second, decreasing m provides a solution more consistent with the monotone smoothing spline approach when it comes to number and location of inflection points. Whether the monotone polynomial solution is providing too many or too few inflection points is the subject of further investigation, as is the use of monotone polynomials in growth curve modelling.

6.6 CONCLUSIONS

We have described in this chapter methods for determining inflection points using monotone polynomials and have illustrated these with two real world examples. We have demonstrated through simulation studies, that monotone polynomials not only provide a viable solution to the location and number of inflection points whether the underlying curve is a monotone polynomial, or not. We have shown that even in situations where there are inflection points lying on the edge of the data range that monotone polynomials can manage, in many instances, to locate these inflection points. Extensive use of bootstrapping was employed to determine the degree of monotone polynomial and then provide a distribution for the location of inflection points.

We have limited our numerical experiments and comparisons at the moment to data which reflects our underlying motivational problem, that is, the brain function data described by Firmin, Müller & Rösler (2012). However, our final exercise using the Berkley Guidance Study growth data demonstrates that, with careful application, monotone polynomials could potentially be more widely used in a variety of situations. Furthermore, we have shown through our simulation exercises that the methods of monotone polynomials to determine the location of inflection points appears on the whole to be somewhat robust to model mis-specification of a higher degree, with the additional inflection points arising from this higher degree polynomial being fitted, quite frequently lying outside the range of data, thus deeming them irrelevant.

There remain open research questions arising from this work including in particular, more refined methodology to determine confidence intervals for the location of inflection points. However, we have suggested, with support from the bootstrap distributions of our inflection points, that mixtures of normals may be a suitable methodology to determine such estimates of variability.

A

APPENDIX A: EVALUATING OBJECTIVE FUNCTION AND ITS DERIVATIVES FOR MODELS (2.7) AND (2.8)

Following the same arguments described in Chapter 2 we illustrate the necessary calculations to evaluate the objective function and its derivatives.

A.1 MODEL PARAMETERISATION (2.7)

To calculate $\boldsymbol{\beta}$ for a given $\boldsymbol{\theta}$, we first build triples $(1, 2b_j, c_j^2 + b_j^2)$ which are the coefficients for the quadratic functions appearing in (2.7). By convoluting these triplets, we can calculate the coefficients $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{2K})^T$ of the polynomial

$$\gamma_0 + \gamma_1 t + \cdots + \gamma_{2K} t^{2K} = \prod_{j=1}^{K} \left\{ 1 + 2b_j t + \left(c_j^2 + b_j^2 \right) t^2 \right\}.$$

From $\boldsymbol{\gamma}$ we can readily calculate $\boldsymbol{\beta}$ as

$$\boldsymbol{\beta} = \left(\delta, \alpha\gamma_0, \alpha\frac{\gamma_1}{2}, \ldots, \alpha\frac{\gamma_{2K}}{2K+1} \right)^T.$$

Note that $\left(0, \gamma_0, \frac{\gamma_1}{2}, \ldots, \frac{\gamma_{2K}}{2K+1}\right)^T$ is the vector that contains the coefficient of the polynomial $\tilde{p}(x)$ in (2.9).

To minimise RSS numerically we also need first and second derivatives for a derivative based optimisation algorithm. These derivatives can easily be calculated in a similar manner. In general using (2.3) we see

$$\frac{\partial}{\partial \boldsymbol{\theta}} \text{RSS} = -2 \sum_{i=1}^{n} (y - p(x_i)) \frac{\partial}{\partial \boldsymbol{\theta}} p(x_i) \tag{A.1}$$

and from (2.9) we have

$$\frac{\partial}{\partial \delta}\mathrm{RSS} = -2\sum_{i=1}^{n}(y - p(x_i)), \tag{A.2a}$$

$$\frac{\partial}{\partial \alpha}\mathrm{RSS} = -2\sum_{i=1}^{n}(y - p(x_i))\tilde{p}(x_i), \tag{A.2b}$$

$$\frac{\partial}{\partial \tilde{\theta}_k}\mathrm{RSS} = -2\alpha\sum_{i=1}^{n}(y - p(x_i))\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(x_i), \tag{A.2c}$$

where $\tilde{\theta}_k$ is a component of the vector $\tilde{\boldsymbol{\theta}}$, that is one of the b_js or c_js. Previously we have discussed how $\tilde{p}(x_i)$ can be evaluated once γ is determined. The partial derivatives $\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(x_i)$ can be evaluated similarly. For example, if $\tilde{\theta}_k$ is b_{j_0}, then we build the triples $(1, 2b_j, c_j^2 + b_j^2)$ for $j \neq j_0$ and the triple $(0, 2, 2b_{j_0})$. After convoluting these K triples, the coefficients of the polynomial $\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(\cdot)$ can be readily calculated. Similarly, if $\tilde{\theta}_k$ is c_{j_0}, then we build the triples $(1, 2b_j, c_j^2 + b_j^2)$ for $j \neq j_0$ and the triple $(0, 0, 2c_{j_0})$. After convoluting these K triples, the coefficients of the polynomial $\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(\cdot)$ can be readily calculated.

Using (A.1) it follows that the Hessian is

$$\frac{\partial^2}{\partial \boldsymbol{\theta}\partial \boldsymbol{\theta}^T}\mathrm{RSS}(\boldsymbol{\theta}) = 2\sum_{i=1}^{n}\left(\frac{\partial}{\partial \boldsymbol{\theta}}p(x_i)\right)\left(\frac{\partial}{\partial \boldsymbol{\theta}}p(x_i)\right)^T - 2\sum_{i=1}^{n}(y - p(x_i))\frac{\partial^2}{\partial \boldsymbol{\theta}\partial \boldsymbol{\theta}^T}p(x_i) \tag{A.3}$$

and combining with (A.2) allows us to find formulae for the second partial derivatives of RSS, that is of $\frac{\partial^2}{\partial \theta_k\, \partial \theta_l}\mathrm{RSS}$. Some of these second derivatives involve second derivatives of the polynomial $\tilde{p}(x)$. Again, the coefficients of $\frac{\partial^2}{\partial \theta_k\, \partial \theta_l}\tilde{p}(x; \tilde{\boldsymbol{\theta}})$ can be determined by convoluting appropriately constructed triples. Though if either θ_l or θ_k is either α or δ, then, trivially, $\frac{\partial^2}{\partial \theta_k\, \partial \theta_l}\tilde{p}(x; \tilde{\boldsymbol{\theta}}) \equiv 0$. Also note that $\frac{\partial^2}{\partial b_j \partial c_j}\tilde{p}(x; \tilde{\boldsymbol{\theta}}) \equiv 0$, $j = 1, \ldots, K$.

These second partial derivatives are shown here for completeness. First we note the obvious relationship

$$\frac{\partial^2}{\partial \theta_k\, \partial \theta_l}\mathrm{RSS} \equiv \frac{\partial^2}{\partial \theta_l\, \partial \theta_k}\mathrm{RSS}$$

From (A.2a) we have

$$\frac{\partial^2}{\partial \delta^2}\text{RSS} \equiv 2n, \tag{A.4a}$$

$$\frac{\partial^2}{\partial \delta \partial \alpha}\text{RSS} = 2\sum_{i=1}^{n} \tilde{p}(x_i),. \tag{A.4b}$$

$$\frac{\partial^2}{\partial \delta \partial \tilde{\theta_k}}\text{RSS} = 2\alpha \sum_{i=1}^{n} \frac{\partial}{\partial \tilde{\theta_k}} \tilde{p}(x_i) \tag{A.4c}$$

From (A.2b) we have

$$\frac{\partial^2}{\partial \alpha^2}\text{RSS} = 2\sum_{i=1}^{n} \tilde{p}(x_i)^2, \tag{A.5a}$$

$$\frac{\partial^2}{\partial \alpha \partial \tilde{\theta_k}}\text{RSS} = -2\sum_{i=1}^{n} \left\{ (y_i - p(x_i)) - \alpha \tilde{p}(x_i) \right\} \frac{\partial}{\partial \tilde{\theta_k}} \tilde{p}(x_i). \tag{A.5b}$$

From (A.2c) we have

$$\frac{\partial^2}{\partial \tilde{\theta_k}^2}\text{RSS} = -2\alpha \sum_{i=1}^{n} \left\{ -\alpha \left(\frac{\partial}{\partial \tilde{\theta_k}} \tilde{p}(x_i) \right)^2 + (y_i - p(x_i)) \frac{\partial^2}{\partial \tilde{\theta_k}^2} \tilde{p}(x_i) \right\}, \tag{A.6a}$$

$$\frac{\partial^2}{\partial \tilde{\theta_k} \partial \tilde{\theta_j}}\text{RSS} = -2\alpha \sum_{i=1}^{n} \left\{ -\alpha \frac{\partial}{\partial \tilde{\theta_j}} \tilde{p}(x_i) \frac{\partial}{\partial \tilde{\theta_k}} \tilde{p}(x_i) + (y_i - p(x_i)) \frac{\partial^2}{\partial \tilde{\theta_k} \partial \tilde{\theta_j}} \tilde{p}(x_i) \right\}, \text{where } j \neq k. \tag{A.6b}$$

A.2 MODEL PARAMETERISATION (2.8)

To calculate $\boldsymbol{\beta}$ for a given $\boldsymbol{\theta}$, we first build triples $(b_j^2, 2b_j, 1 + c_j^2)$ which are the coefficients for the quadratic functions appearing in (2.8). By convoluting these triplets, we can calculate the coefficients $\boldsymbol{\gamma} = (\gamma_0, \ldots, \gamma_{2K})^T$ of the polynomial

$$\gamma_0 + \gamma_1 t + \cdots + \gamma_{2K} t^{2K} = \prod_{j=1}^{K} \left\{ b_j^2 + 2b_j t + \left(1 + c_j^2\right) t^2 \right\}.$$

From $\boldsymbol{\gamma}$ we can readily calculate $\boldsymbol{\beta}$ as

$$\boldsymbol{\beta} = \left(\delta, \alpha\gamma_0, \alpha\frac{\gamma_1}{2}, \ldots, \alpha\frac{\gamma_{2K}}{2K+1} \right)^T.$$

Note that $\left(0, \gamma_0, \frac{\gamma_1}{2}, \ldots, \frac{\gamma_{2K}}{2K+1}\right)^T$ is the vector that contains the coefficient of the polynomial $\tilde{p}(x)$ in (2.9).

To minimise RSS numerically we also need first and second derivatives for a derivative based optimisation algorithm. These derivatives can easily be calculated in a similar manner. In general using (2.3) we see

$$\frac{\partial}{\partial \boldsymbol{\theta}}\text{RSS} = -2\sum_{i=1}^{n}(y - p(x_i))\frac{\partial}{\partial \boldsymbol{\theta}}p(x_i) \tag{A.7}$$

and from (2.9) we have

$$\frac{\partial}{\partial \delta}\text{RSS} = -2\sum_{i=1}^{n}(y - p(x_i)), \tag{A.8a}$$

$$\frac{\partial}{\partial \alpha}\text{RSS} = -2\sum_{i=1}^{n}(y - p(x_i))\tilde{p}(x_i), \tag{A.8b}$$

$$\frac{\partial}{\partial \tilde{\theta}_k}\text{RSS} = -2\alpha\sum_{i=1}^{n}(y - p(x_i))\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(x_i), \tag{A.8c}$$

where $\tilde{\theta}_k$ is a component of the vector $\tilde{\boldsymbol{\theta}}$, that is one of the b_js or c_js. Previously we have discussed how $\tilde{p}(x_i)$ can be evaluated once γ is determined. The partial derivatives $\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(x_i)$ can be evaluated similarly. For example, if $\tilde{\theta}_k$ is b_{j_0}, then we build the triples $(b_j^2, 2b_j, 1 + c_j^2)$ for $j \neq j_0$ and the triple $(2b_{j_0}, 2, 0)$. After convoluting these K triples, the coefficients of the polynomial $\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(\cdot)$ can be readily calculated. Similarly, if $\tilde{\theta}_k$ is c_{j_0}, then we build the triples $(b_j^2, 2b_j, 1 + c_j^2)$ for $j \neq j_0$ and the triple $(0, 0, 2c_{j_0})$. After convoluting these K triples, the coefficients of the polynomial $\frac{\partial}{\partial \tilde{\theta}_k}\tilde{p}(\cdot)$ can be readily calculated.

Using (A.7) it follows that the Hessian is

$$\frac{\partial^2}{\partial \boldsymbol{\theta}\partial \boldsymbol{\theta}^T}\text{RSS}(\boldsymbol{\theta}) = 2\sum_{i=1}^{n}\left(\frac{\partial}{\partial \boldsymbol{\theta}}p(x_i)\right)\left(\frac{\partial}{\partial \boldsymbol{\theta}}p(x_i)\right)^T - 2\sum_{i=1}^{n}(y - p(x_i))\frac{\partial^2}{\partial \boldsymbol{\theta}\partial \boldsymbol{\theta}^T}p(x_i) \tag{A.9}$$

and combining with (A.8) allows us to find formulae for the second partial derivatives of RSS, that is of $\frac{\partial^2}{\partial \theta_k \partial \theta_l}\text{RSS}$. Some of these second derivatives involve second derivatives of the polynomial $\tilde{p}(x)$. Again, the coefficients of $\frac{\partial^2}{\partial \theta_k \partial \theta_l}\tilde{p}(x; \tilde{\boldsymbol{\theta}})$ can be determined by convoluting appropriately constructed triples. Though if either θ_l or θ_k is either α or δ, then, trivially, $\frac{\partial^2}{\partial \theta_k \partial \theta_l}\tilde{p}(x; \tilde{\boldsymbol{\theta}}) \equiv 0$. Also note that $\frac{\partial^2}{\partial b_j \partial c_j}\tilde{p}(x; \tilde{\boldsymbol{\theta}}) \equiv 0$, $j = 1, \ldots, K$.

These second partial derivatives are shown here for completeness. First we note the obvious relationship

$$\frac{\partial^2}{\partial\theta_k\,\partial\theta_l}\text{RSS} \equiv \frac{\partial^2}{\partial\theta_l\,\partial\theta_k}\text{RSS}$$

From (A.8a) we have

$$\frac{\partial^2}{\partial\delta^2}\text{RSS} \equiv 2n, \tag{A.10a}$$

$$\frac{\partial^2}{\partial\delta\partial\alpha}\text{RSS} = 2\sum_{i=1}^{n}\tilde{p}(x_i),. \tag{A.10b}$$

$$\frac{\partial^2}{\partial\delta\partial\tilde{\theta_k}}\text{RSS} = 2\alpha\sum_{i=1}^{n}\frac{\partial}{\partial\tilde{\theta_k}}\tilde{p}(x_i) \tag{A.10c}$$

From (A.8b) we have

$$\frac{\partial^2}{\partial\alpha^2}\text{RSS} = 2\sum_{i=1}^{n}\tilde{p}(x_i)^2, \tag{A.11a}$$

$$\frac{\partial^2}{\partial\alpha\partial\tilde{\theta_k}}\text{RSS} = -2\sum_{i=1}^{n}\left\{(y_i - p(x_i)) - \alpha\tilde{p}(x_i)\right\}\frac{\partial}{\partial\tilde{\theta_k}}\tilde{p}(x_i). \tag{A.11b}$$

From (A.8c) we have

$$\frac{\partial^2}{\partial\tilde{\theta_k}^2}\text{RSS} = -2\alpha\sum_{i=1}^{n}\left\{-\alpha\left(\frac{\partial}{\partial\tilde{\theta_k}}\tilde{p}(x_i)\right)^2 + (y_i - p(x_i))\frac{\partial^2}{\partial\tilde{\theta_k}^2}\tilde{p}(x_i)\right\}, \tag{A.12a}$$

$$\frac{\partial^2}{\partial\tilde{\theta_k}\partial\tilde{\theta_j}}\text{RSS} = -2\alpha\sum_{i=1}^{n}\left\{-\alpha\frac{\partial}{\partial\tilde{\theta_j}}\tilde{p}(x_i)\frac{\partial}{\partial\tilde{\theta_k}}\tilde{p}(x_i) + (y_i - p(x_i))\frac{\partial^2}{\partial\tilde{\theta_k}\partial\tilde{\theta_j}}\tilde{p}(x_i)\right\}, \text{where } j \neq k. \tag{A.12b}$$

B

APPENDIX B: FURTHER RESULTS FROM SIMULATION STUDIES CARRIED OUT IN CHAPTER 3

Bootstrap estimates using different bootstrap methodologies

We describe in Figures B.1 and B.2, bootstrap estimates and variability for the underlying data generating function $p_1(x)$ and note similar results for the other two functions. We note that regardless of the bootstrap methodology, very similar results. This is demonstrated visually through the patterns in row 1 (paired bootstrap), row 2 (residual bootstrap) and row 3 (weighted bootstrap), for each of Figures B.1 and B.2, being very similar, which show in each plot a histogram on top for the monotone polynomial bootstrap iteration estimates, and the bottom standard unconstrained least squares bootstrap iteration estimates both for data where the underlying fit using standard unconstrained least squares would be a monotone polynomial. Similarly rows 4 through 6 show consistent patterns for data where the underlying fit using standard unconstrained least squares would *not* be a monotone polynomial. For example, consider estimates of any β_i; we see the distribution of the bootstrap estimates to be very similar regardless of the bootstrap methodology regardless of the parameter one selects. This reasonable high degree of homogeneity has several ramifications. First, our chosen functions display reasonable levels of agreement regardless of the bootstrap method, suggesting that the bootstrap results, when fitting monotone polynomials to data, should be relatively robust to the bootstrap method used. Second, it suggests that one can simply select the most appropriate bootstrap technique to use when fitting monotone polynomials to data. (Note: there are no estimates using monotone polynomials for the residual *adjusted* bootstrap due to there being no defined *hat matrix*).

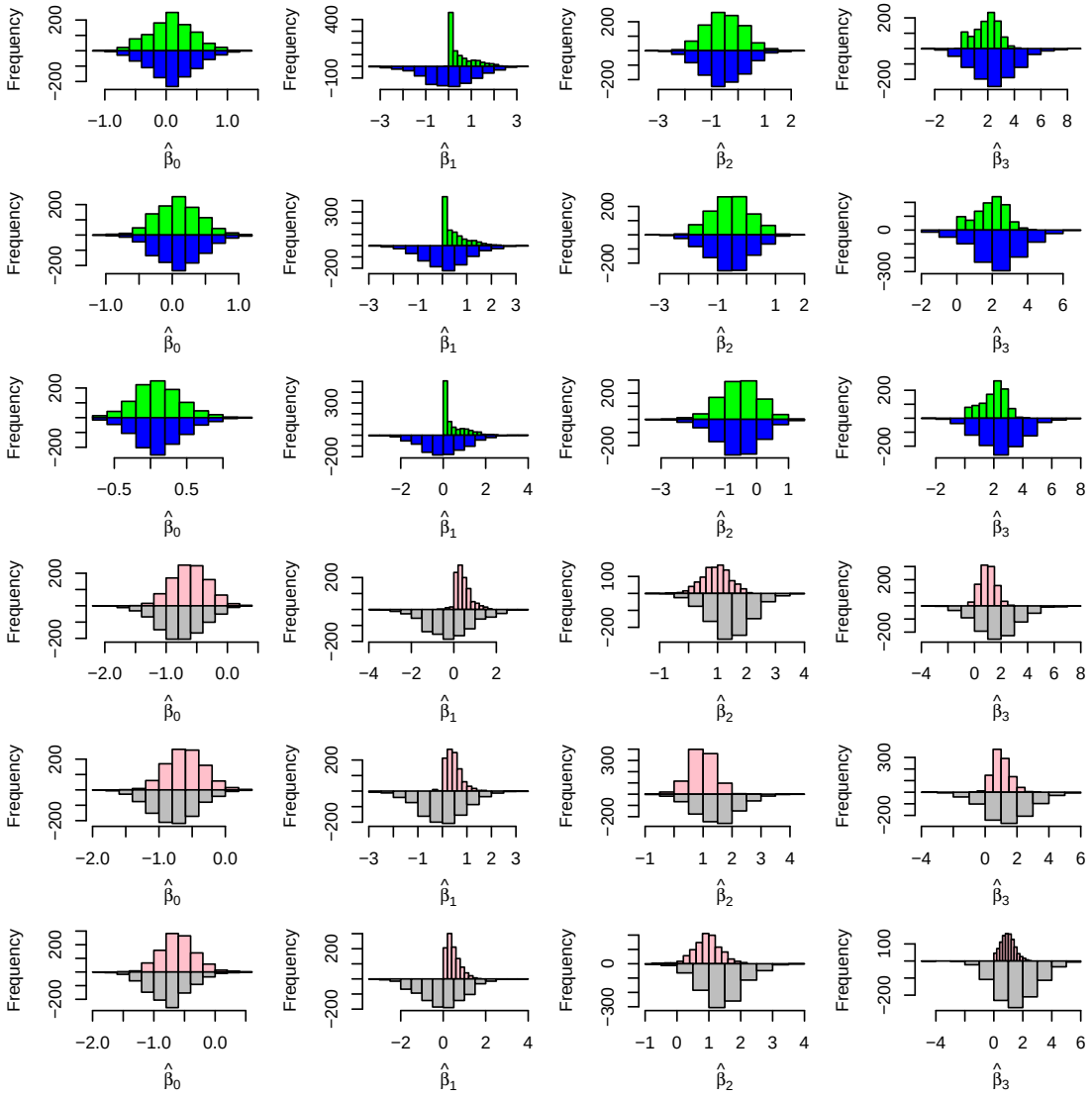

Figure B.1: Results of three bootstrap methods (non-parametric rows 1,4; residuals rows 2,5; weighted rows 3,6) using two data sets (least squares fit monotone rows 1-3; least squares fit not monotone rows 4-6) from the underlying polynomial function $p_1(x) = x^3$ for $n = 50$. Back to back histograms in each plots are bootstrap estimates of parameters from standard least squares estimation (bottom histogram) and using monotone polynomials (top histogram).

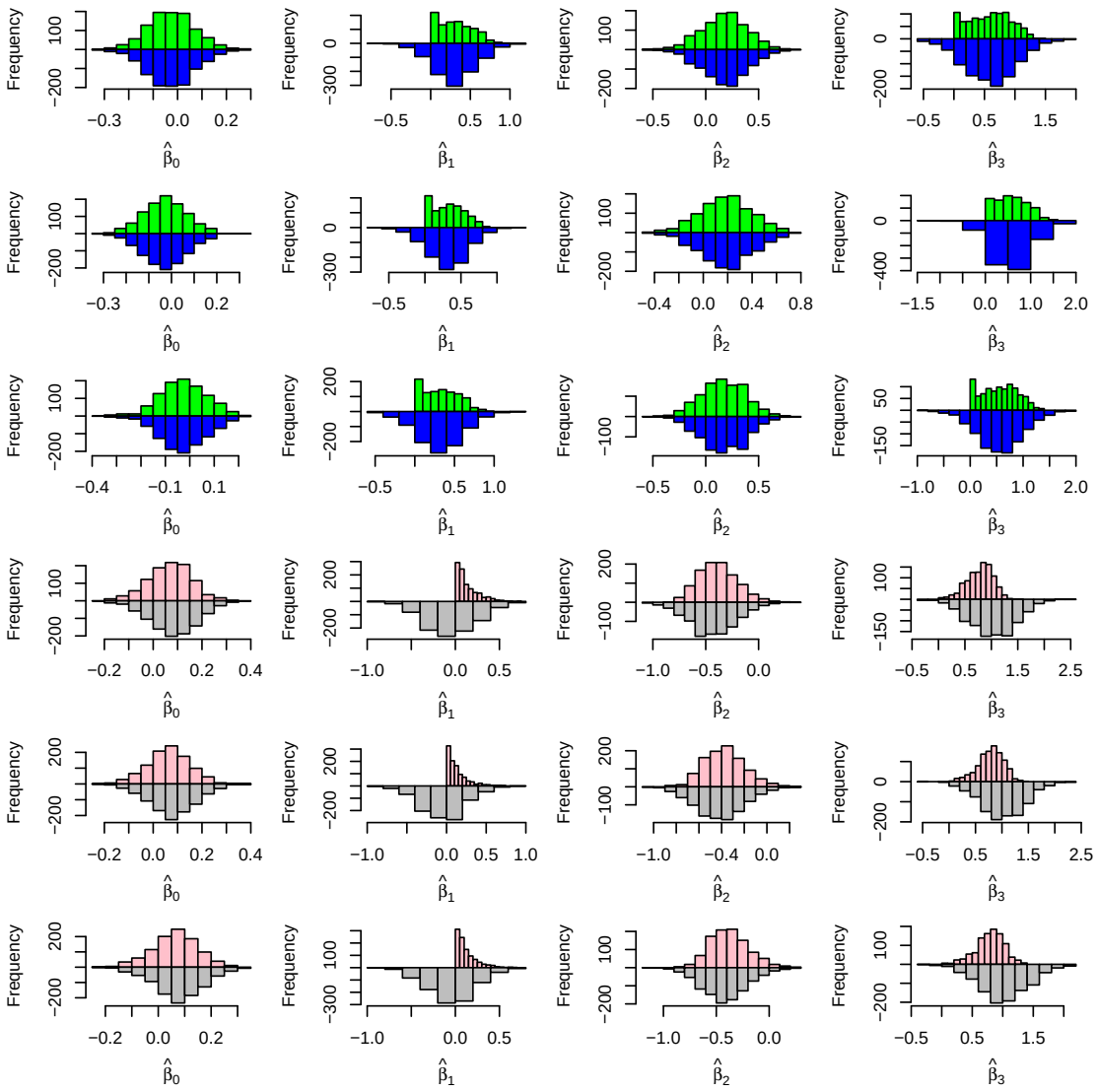

Figure B.2: Results of three bootstrap methods (non-parametric rows 1,4; residuals rows 2,5; weighted rows 3,6) using two data sets (least squares fit monotone rows 1-3; least squares fit not monotone rows 4-6) from the underlying polynomial function $p_1(x) = x^3$ for $n = 1,000$. Back to back histograms in each plots are bootstrap estimates of parameters afor standard least squares estimation (bottom histogram) and using monotone polynomials (top histogram).

C

APPENDIX C: ADDITIONAL EXAMPLE OF CURVATURE PLOT FROM CHAPTER 3

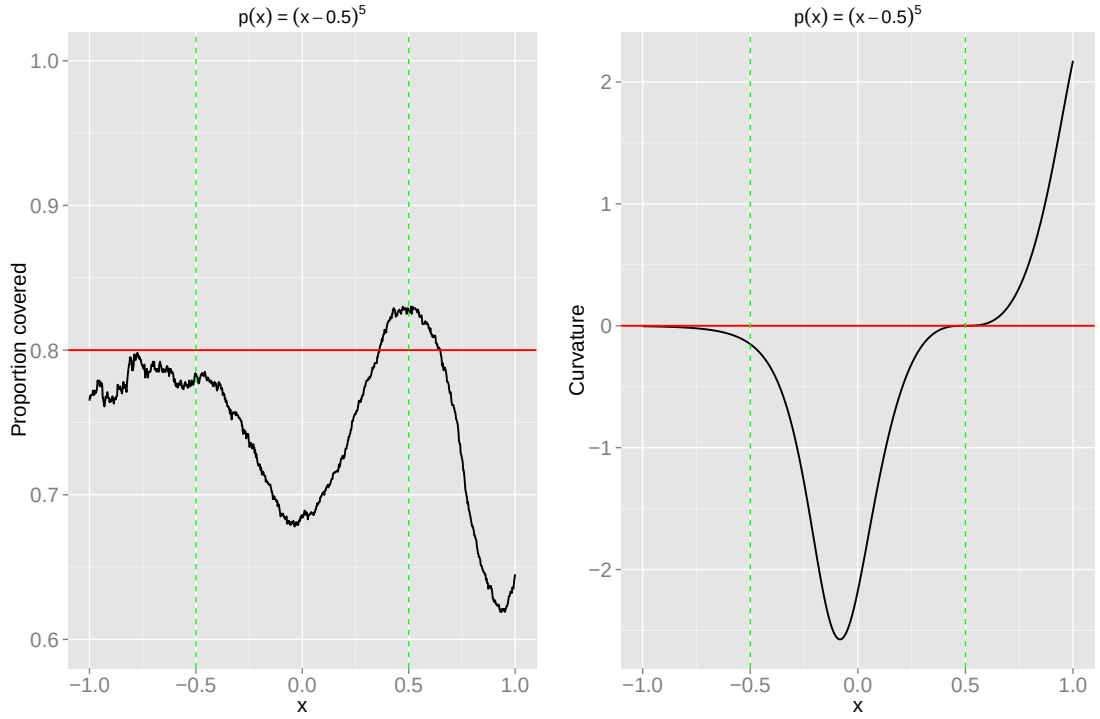

Figure C.1: Curvature and coverage probabilities for $p_4(x) = (x - 0.5)^5$ monotone polynomial using residuals bootstrap, with $m = n$.

BIBLIOGRAPHY

Akaike, H. (1973). *Information theory and an extension of the maximum likelihood principle. Proceedings of the Second International Symposium of Information Theory. In: Petrov, BN and Csáki, F. pp 267–281*. Budapest: Akadémiai Kiadó. (Cited on page 100.)

Barbe, P. & Bertail, P. (1995). *The Weighted Bootstrap*, *Lecture Notes in Statistics*, vol. 98. New York: Springer-Verlag. (Cited on pages 63, 65, and 106.)

Barlow, R.E., Bartholomew, D.J., Bremner, J.M. & Brunk, H.D. (1972). *Statistical Inference under Order Restrictions*. London: John Wiley & Sons. (Cited on page 53.)

Bates, D., Mullen, K.M., Nash, J.C. & Varadhan, R. (2011). *minqa: Derivative-free optimization algorithms by quadratic approximation*. URL `http://R-Forge.R-project.org/projects/optimizer/`. R package version 1.1.18/r530. (Cited on page 27.)

Box, G.E.P., Hunter, W.G. & Hunter, J.S. (1978). *Statistics for Experimenters*. New York: John Wiley & Sons. (Cited on page vii.)

Brickman, L. & Steinberg, L. (1962). On nonnegative polynomials. *The American Mathematical Monthly* **69**, 218–221. (Cited on page 25.)

Burnham, K.P. & Anderson, D.R. (2002). *Model Selection and Multimodel Inference – A Practical Information-theoretic Approach*. New York: Springer-Verlag, 2nd edn. (Cited on page 100.)

Chaillet, N., Nystrom, M., Kataja, M. & Demirjian, A. (2004). Dental maturity curves in finnish children: Demirjian's method revisited and polynomial functions for age estimation. *Journal of Forensic Sciences* **49**, 1324–1331. (Cited on page 128.)

Christopoulos, D.T. (2014a). Developing methods for identifying the inflection point of a convex/concave curve. ArXiv:1206.5478v2. (Cited on page 136.)

Christopoulos, D.T. (2014b). Roots, extrema and inflection points by using a proper Taylor regression procedure. Unpublished work available from ResearchGate. (Cited on page 136.)

Curtis, S.M. & Ghosh, S.K. (2011). A variable selection approach to monotonic regression with Bernstein polynomials. *Journal of Applied Statistics* **38**, 961–976. (Cited on page 11.)

Dandurand, F. & Shultz, T.R. (2010). Automatic detection and quantification of growth spurts. *Behavior Research Methods* **42**, 809–823. (Cited on pages 136, 138, and 150.)

Dandurand, F. & Shultz, T.R. (2011). Automatic maxima detection: A graphical user interface and a tutorial. *Tutorials in Quantitative Methods for Psychology* **7**, 21–31. (Cited on pages 136 and 150.)

Davison, A.C. & Hinkley, D.V. (1997). *Bootstrap Methods and Their Application*. Cambridge: Cambridge University Press. (Cited on pages 63, 64, 65, and 66.)

Demirjian, A., Goldstein, H. & Tanner, J.M. (1973). New system of dental age assessment. *Human Biology* **45**, 211–227. (Cited on pages 127 and 128.)

Dette, H., Neumeyer, N. & Pilz, K.F. (2006). A simple nonparametric estimator of a strictly monotone regression function. *Bernoulli* **12**, 469–490. (Cited on page 2.)

Dette, H. & Pilz, K.F. (2006). A comparative study of monotone nonparametric kernel estimates. *Journal of Statistical Computation and Simulation* **76**, 41–56. (Cited on page 2.)

Dette, H. & Studden, W.J. (1997). *The Theory of Canonical Moments With Applications in Statistics, Probability, and Analysis*. Wiley Series in Probability and Statistics, New York: John Wiley & Sons. (Cited on page 25.)

Draper, N. & Smith, H. (1998). *Applied Regression Analysis*. New York: John Wiley & Sons, 3rd edn. (Cited on pages 58 and 66.)

Dümbgen, L. (2003). Optimal confidence bands for shape-restricted curves. *Bernoulli* **9**, 423–449. (Cited on page 50.)

Efron, B. (1979). 1977 RIETZ Lecture - Bootstrap methods - Another look at the Jackknife. *The Annals of Statistics* **7**, 1–26. (Cited on page 60.)

Efron, B. & Tibshirani, R.J. (1993). *An Introduction to the Bootstrap*. New York: Chapman & Hall. (Cited on page 63.)

Elphinstone, C.D. (1983). A target distribution model for non-parametric density estimation. *Communications in Statistics - Theory and Methods* **12**, 161–198. (Cited on pages iii, 8, 10, 12, 17, 18, 23, 24, 37, 47, 87, 88, 89, and 90.)

Fausett, L.V. (2003). *Numerical Methods: Algorithms and Applications*. Upper Saddle River, NJ: Prentice Hall. (Cited on pages 19 and 26.)

Firmin, L., Müller, S. & Rösler, K.M. (2011). A method to measure the distribution of latencies of motor evoked potentials in man. *Clinical Neurophysiology* **122**, 176–182. (Cited on pages 4, 11, 39, 92, 124, 135, 138, 140, 150, and 152.)

Firmin, L., Müller, S. & Rösler, K.M. (2012). The latency distribution of motor evoked potentials in patients with multiple sclerosis. *Clinical Neurophysiology* **123**, 2414–2421. (Cited on pages 4, 11, 39, 92, 124, 135, 138, 140, 150, 151, 153, and 158.)

Flood, S.J., Mitchell, W.J., Oxnard, C.E., Turlach, B.A. & McGeachie, J. (2011). To evaluate the utility of smaller sample sizes when assessing dental maturity curves for forensic age estimation. *Journal of Forensic Sciences* **56**, 1604–1609. (Cited on page 128.)

Fox, J. & Weisberg, S. (2011). *An R Companion to Applied Regression*. Thousand Oaks CA: Sage, 2nd edn. (Cited on page 66.)

Friedman, J., Hastie, T., Höfling, H. & Tibshirani, R. (2007). Pathwise coordinate optimization. *The Annals of Applied Statistics* **1**, 302–332. (Cited on page 27.)

Friedman, J., Hastie, T. & Tibshirani, R. (2010). Regularisation paths for generalized linear models via coordinate descent. *Journal of Statistical Software* **33**, 1–22. (Cited on page 27.)

FRIEDMAN, J.H. & TIBSHIRANI, R. (1984). The monotone smoothing of scatterplots. *Technometrics* **26**, 243–250. (Cited on page 2.)

GOLDFARB, D. & IDNANI, A. (1982). Dual and Primal-Dual Methods for Solving Strictly Convex Quadratic Programs. In *Numerical Analysis, Proceedings, Cocoyoc, Mexico 1981*, ed. J.P. Hennart, *Lecture Notes in Mathematics*, vol. 909. Berlin: Springer-Verlag, pp. 226–239. (Cited on pages 16 and 46.)

GOLDFARB, D. & IDNANI, A. (1983). A numerically stable dual method for solving strictly convex quadratic programs. *Mathematical Programming* **27**, 1–33. (Cited on pages 16 and 46.)

HALL, P. (1989). Unusual properties of bootstrap confidence intervals in regression problems. *Probability Theory and Related Fields* **81**, 247–273. (Cited on page 76.)

HALL, P. & HOROWITZ, J. (2013). A simple bootstrap method of constructing nonparametric confidence bands for functions. *The Annals of Statistics* **41**, 1982–1921. (Cited on page 50.)

HALL, P. & HUANG, L.S. (2001). Nonparametric kernel regression subject to monotonicity constraints. *The Annals of Statistics* **29**, 624–647. (Cited on page 2.)

HAWKINS, D.M. (1994). Fitting monotonic polynomials to data. *Computational Statistics* **9**, 233–247. (Cited on pages iii, iv, 8, 10, 11, 12, 14, 15, 16, 17, 18, 24, 38, 40, 86, 87, 88, 89, 90, and 101.)

HAZELTON, M.L. & TURLACH, B.A. (2011). Semiparametric regression with shape constrained penalized splines. *Computational Statistics & Data Analysis* **55**, 2871–2879. (Cited on page 2.)

HECKMAN, N. & RAMSAY, J.O. (2000). Penalized regression with model-based penalties. *Canadian Journal of Statistics* **28**, 241–258. (Cited on page 4.)

HEINZMANN, D. (2008). A filtered polynomial approach to density estimation. *Computational Statistics* **23**, 343–360. (Cited on pages 8, 18, 22, 23, 27, and 37.)

HERITIER, S., CANTONI, E., COPT, S. & VICTORIA-FESER, M.P. (2009). *Robust Methods in Biostatistics*. Chichester: John Wiley & Sons. (Cited on page 132.)

JANSSEN, A. & PAULS, T. (2003). How do bootstrap and permutation tests work? *The Annals of Statistics* **31**, 768–806. (Cited on page 107.)

KARLIN, S. & STUDDEN, W.J. (1966). *Tchebycheff Systems: With Applications in Analysis and Statistics*. New York: John Wiley & Sons. (Cited on page 25.)

KONISHI, S. & KITAGAWA, G. (1996). Generalised information criteria in model selection. *Biometrika* **83**, 875–890. (Cited on pages 104 and 132.)

KRAUSE, A. & O'CONNELL, M. (2012). *A Picture is Worth a Thousand Tables. Graphics in Life Sciences*. London: Springer-Verlag. (Cited on page 101.)

LAWRANCE, I.C., MURRAY, K., BATMAN, B., GEARRY, R.B., GRAFTON, R., KRISHNAPRASAD, K., ANDREWS, J.M., PROSSER, R., BAMPTON, P.A., COOKE, S.E., MAHY, G., RADFORD-SMITH, G., CROFT, A. & HANIGAN, K. (2013). Crohn's disease and smoking: Is it ever too late to quit? *Journal of Crohn's and Colitis* **7**, e665–e671. (Cited on pages iv and 117.)

Loftus, G.R. (1983). A picture is worth a thousand p values: On the irrelevance of hypothesis testing in the microcomputer age. *Behavior Research Methods Instruments and Computers* **25**, 250–256. (Cited on page 101.)

Lokhorst, J. (1999). The LASSO and Generalised Linear Models. Honours project, Department of Statistics, The University of Adelaide, South Australia, Australia. (Cited on page 133.)

Magistris, M.R., Rösler, K.M., Truffert, A., Landis, T. & Hess, C.W. (1999). A clinical study of motor evoked potentials using a triple stimulation technique. *Brain* **122**, 265–279. (Cited on page 11.)

Magistris, M.R., Rösler, K.M., Truffert, A. & Myers, J.P. (1998). Transcranial stimulation excites virtually all motor neurons supplying the target muscle: A demonstration and a method improving the study of motor evoked potentials. *Brain* **121**, 437–450. (Cited on page 11.)

Mallows, C.L. (1973). Some comments on C_p. *Technometrics* **8**, 661–675. (Cited on page 101.)

Mammen, E. (1991). Estimating a smooth monotone regression function. *The Annals of Statistics* **19**, 724–740. (Cited on page 2.)

Mammen, E., Marron, J.S., Turlach, B.A. & Wand, M.P. (2001). A general projection framework for constrained smoothing. *Statistical Science* **16**, 232–248. (Cited on page 2.)

Marron, J.S., Turlach, B.A. & Wand, M.P. (1997). Local polynomial smoothing under qualitative constraints. In *Graph-Image-Vision*, eds. L. Billard & N.I. Fisher, *Computing Science and Statistics*, vol. 28. Fairfax Station, VA 22039–7460: Interface Foundation of North America, Inc., pp. 647–652. (Cited on page 2.)

McNamara, B.A. & Rosenwax, L.K. (2010). Which carers of family members at the end of life need more support from health services and why? *Social Science and Medicine* **70**, 1035–1041. (Cited on page 112.)

Meier, L., van de Geer, S. & Bühlmann, P. (2008). The group Lasso for logistic regression. *Journal of the Royal Statistical Society, Series B* **70**, 53–71. doi: 10.1111/j.1467-9868.2007.00627.x. (Cited on page 134.)

Meinshausen, N. & Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society, Series B* **72**, 417–473. (Cited on pages 101 and 106.)

Meyer, M.C. (2008). Inference using shape-restricted regression splines. *The Annals of Applied Statistics* **2**, 1013–1033. (Cited on pages 2, 6, 8, and 87.)

Meyer, M.C. (2012). Constrained penalized splines. *Canadian Journal of Statistics* **40**, 190–206. (Cited on pages 2, 6, 8, and 87.)

Meyer, M.C. & Liao, X. (2014). *cgam: Constrained Generalized Additive Model*. URL `http://CRAN.R-project.org/package=cgam`. R package version 1.2. (Cited on page 87.)

Miller, A. (2002). *Subset Selection in Regression*, vol. 95. Boca Raton: Chapman & Hall/CRC, 2nd edn. (Cited on page 99.)

MINNIER, J., TIAN, L. & CAI, T. (2011). A perturbation method for inference on regularized regression estimates. *Journal of the American Statistical Association* **106**, 1371–1382. (Cited on pages 106 and 107.)

MIRMAN, D. (2014). *Growth Curve Analysis and Visualization Using R*. The R Series, Boca Raton: Chapman & Hall/CRC. (Cited on pages 1, 3, and 136.)

MÜLLER, S., SCEALY, J.L. & WELSH, A.H. (2013). Model selection in linear mixed models. *Statistical Science* **28**, 135–167. (Cited on page 134.)

MÜLLER, S. & WELSH, A. (2005). Outlier robust model selection in linear regression. *Journal of the American Statistical Association* **100**, 1297–1310. (Cited on pages 65, 126, and 132.)

MÜLLER, S. & WELSH, A.H. (2009). Robust model selection in generalized linear models. *Statistica Sinica* **19**, 1155–1170. (Cited on pages 65, 106, 126, and 132.)

MÜLLER, S. & WELSH, A.H. (2010). On model selection curves. *International Statistical Review* **78**, 240–256. (Cited on pages 101, 104, 106, 110, and 132.)

MURRAY, K., HERITIER, S. & MÜLLER, S. (2013). Graphical tools for model selection in generalized linear models. *Statistics in Medicine* **32**, 4438–4451. (Cited on pages iv, 102, and 133.)

MURRAY, K., MÜLLER, S. & TURLACH, B.A. (2013). Revisiting fitting monotone polynomials to data. *Computational Statistics* **28**, 1989–2005. (Cited on pages iii, iv, 19, 50, 64, 84, 87, 88, 89, and 134.)

MURRAY, K., MÜLLER, S. & TURLACH, B.A. (2015). Flexible and fast monotone polynomial fitting. *Submitted* . (Cited on pages iii, iv, 50, 84, 87, 88, and 135.)

NELDER, J.A. & MEAD, R. (1965). A simplex method for function minimization. *The Computer Journal* **7**, 308–313. (Cited on page 27.)

OLIVE, D.J. (2007). Prediction intervals for regression models. *Computational Statistics & Data Analysis* **51**, 3115–3122. (Cited on page 74.)

OSBORNE, M.R. (1976). Nonlinear least squares — the Levenberg algorithm revisited. *Journal of the Australian Mathematical Society, Series B* **19**, 343–357. (Cited on page 27.)

PANIK, MICHAEL, J. (2014). *Growth Curve Modeling: Theory and Applications*. Hoboken: John Wiley & Sons. (Cited on pages 1, 3, and 136.)

PAPP, D. (2011). Optimization models for shape-constrained function estimation problems involving nonnegative polynomials and their restrictions. Phd thesis, Rutgers University, New Brunswick, NJ, USA. (Cited on page 25.)

PENTTILA, T.J. (2006). Personal communication. (Cited on page 19.)

POWELL, M.J.D. (2009). The BOBYQA algorithm for bound constrained optimization without derivatives. *Report No. DAMTP 2009/NA06*, Centre for Mathematical Sciences, University of Cambridge, UK. http://www.damtp.cam.ac.uk/user/na/NA_papers/NA2009_06.pdf. (Cited on page 27.)

PREECE, M.A. & BAINES, M.J. (1978). A new family of mathematical models describing the human growth curve. *Annals of Human Biology* **5**, 1–24. (Cited on page 3.)

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/. ISBN 3-900051-07-0. (Cited on pages v, 10, 48, and 98.)

Ramsay, J.O. (1988). Monotone regression splines in action (with discussion). *Statistical Science* **3**, 425–461. (Cited on page 4.)

Ramsay, J.O. (1998). Estimating smooth monotone functions. *Journal of the Royal Statistical Society, Series B* **60**, 365–375. (Cited on pages 2, 4, 6, 8, and 87.)

Ramsay, J.O. & Silverman, B.W. (2002). *Applied Functional Data Analysis: Methods and Case Studies*. New York: Springer-Verlag. (Cited on pages 4, 6, 8, 38, and 87.)

Ramsay, J.O. & Silverman, B.W. (2006). *Functional Data Analysis*. New York: Springer-Verlag, 2nd edn. (Cited on pages 4, 6, 8, and 87.)

Ramsay, J.O., Wickham, H., Graves, S. & Hooker, G. (2013). *fda: Functional Data Analysis*. URL http://CRAN.R-project.org/package=fda. R package version 2.4.0. (Cited on pages 4, 11, 87, 136, 150, and 155.)

Ratkowsky, D.A. (1990). *Handbook of Nonlinear Regression Models*. New York: Marcel Dekker. (Cited on page 1.)

Ronchetti, E. (1997). Robustness aspects of model choice. *Statistica Sinica* **7**, 327–338. (Cited on page 132.)

Ronchetti, E. & Staudte, R.G. (1994). A robust version of Mallows' C_p. *Journal of the American Statistical Association* **89**, 550–559. (Cited on page 132.)

Rosenwax, L.K., McNamara, B.A., Murray, K., McCabe, R.J., Aoun, S.M. & Currow, D.C. (2011). Hospital and emergency department use in the last year of life: A baseline for future modifications to end-of-life care. *Medical Journal of Australia* **194**, 570–573. (Cited on pages iv and 112.)

Roth, V. (2004). The generalized LASSO. *IEEE Transactions on Neural Networks* **15**, 16–28. (Cited on page 133.)

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics* **6**, 461–464. (Cited on page 100.)

Shao, J. (1994). Bootstrap sample size in non-regular cases. *Proceedings of the American Mathematical Society* **122**, 1251–1262. (Cited on pages 65 and 80.)

Shao, J. (1996). Bootstrap model selection. *Journal of the American Statistical Association* **91**, 655–665. (Cited on pages 54 and 125.)

Shao, J. (1997). An asymptotic theory for linear model selection. *Statistica Sinica* **7**, 221–264. (Cited on page 100.)

Shevchuk, I.A. (1993). Approximation of monotone functions by monotone polynomials. *Russian Academy of Sciences. Sbornik Mathematics* **76**, 51–64. (Cited on pages 54 and 88.)

Silvapulle, M.J. & Sen, P.K. (2005). *Constrained Statistical Inference: Inequality, Order and Shape Restrictions*. Hoboken, NJ: John Wiley & Sons. (Cited on page 53.)

SINIKSARAN, E. (2008). A geometric interpretation of Mallows' C_p statistic and an alternative plot in variable selection. *Computational Statistics & Data Analysis* **52**, 3459–3467. (Cited on page 101.)

SPJOTVOLL, E. (1977). Alternatives to plotting C_p in multiple regression. *Biometrika* **64**, 1–8. (Cited on page 101.)

STINE, R.A. (1985). Bootstrap prediction intervals for regression. *Journal of the American Statistical Association* **80**, 1026–1031. (Cited on pages 74, 77, and 79.)

TUDDENHAM, R.D. & SNYDER, M.M. (1954). Physical growth of california boys and girls from birth to eighteen years. *Publications in child development. University of California, Berkeley* **1**, 183–364. (Cited on page 6.)

TUKEY, J.W. & TUKEY, P.A. (1985). Computer graphics and exploratory data analysis: An introduction. In *In Proceedings of the Sixth Annual Conference and Exposition: Computer Graphics'*, vol. 85 3. pp. 773–785. (Cited on page 101.)

TURLACH, B.A. (2005). Shape constrained smoothing using smoothing splines. *Computational Statistics* **20**, 81–103. (Cited on page 2.)

TURLACH, B.A. & WEINGESSEL, A. (2011). *quadprog: Functions to solve Quadratic Programming Problems.* S original by Berwin A. Turlach, R port by Andreas Weingessel; R package version 1.5-4. (Cited on page 16.)

WEBER, N.C. (1984). On resampling techniques for regression models. *Statistics & Probability Letters* **2**, 275–278. (Cited on page 66.)