On smooth models for complex domains and distances

submitted by

David Lawrence Miller

for the degree of Doctor of Philosophy

of the

University of Bath

Department of Mathematical Sciences

September 2011

Summary

Spline smoothing is a popular technique for creating maps of a spatial phenomenon. Most smoothers use the Euclidean metric to measure the distance between data. This approach is flawed since the distances between points in the domain as experienced by the objects within the domain are rarely Euclidean. For example, the movements of animals and people are subject to both physical and political boundaries (respectively) which must be navigated. Measuring distances between the objects using the incorrect (Euclidean) metric leads to incorrect inference. The first part of this thesis develops a finite area smoother which does not suffer from this problem when the shape of the area is complex. It begins by rejecting the use of the Schwarz-Christoffel transform as a method for morphing complex domains due to its squashing of space. From there a method based on preserving within-area distances using multidimensional scaling is developed. High dimensional projections of the data are necessary to avoid a loss of ordering in the points. To smooth reliably in high dimensions Duchon splines are used. The model developed rivals the current best finite area method in prediction error terms and fits easily into larger models. Finally, the utility of projection methods to smooth general distances is explored.

The second part of the thesis concerns distance sampling, a widely used set of methods for estimating the abundance of biological populations. The work presented here introduces mixture formulation for the detection function used to model the probability of detection. The use of mixture models leads to flexible but monotonic detection functions, avoiding the unrealistic shapes which conventional methods are prone to. These new models are then applied to several existing, problematic data sets.

Acknowledgements

I would first like to thank my supervisor Simon Wood for three years of guidance and assistance in both technical and presentational matters. At many points my own pessimism would have lead to me abandoning an approach; the work presented here is testament to his continued input. My thanks also go to Simon Shaw who helped improve the work greatly via careful readings and comments. I would also like to thank the Engineering and Physical Sciences Research Council (EPSRC) for their financial support.

Without the work of my collaborators Len Thomas, Giampiero Marra and Luca Zanin a large portion of the thesis would not exist. In particular, I am indebted to my former officemate Giampiero Marra for a great many useful discussions over the past three years. In addition, Mark Taylor provided extremely useful comments on a first draft of the work in chapter 2.

Going further back in time, to before the work in this thesis had begun, I must thank the Centre for Research into Environmental and Ecological Modelling (CREEM) at the University of St Andrews (and Steve Buckland specifically) for employing me in summer placements for the four years that I was an undergraduate. Without the encouragement and inspiration of those working at CREEM, I am certain that I would not have embarked on a PhD.

On a personal note, Elle Dodd's continued moral support and encouragement (as well as patience and tolerance!) through the highs and lows of the past three years have been essential for both of our sanities. Finally I would like to thank my parents, without whom this would have been (quite literally) impossible.

Contents

List of Figures

List of Tables

Part I

Finite area smoothing

Chapter 1

Introduction to finite area smoothing

This chapter introduces the topics that will be addressed in the first six chapters regarding finite area smoothing. First, section 1.1 gives an introduction to smoothing using splines in two dimensions, section 1.2 then discusses the larger class of models to which the smoothers belong, section 1.3 addresses some more practical issues, then finally section 1.4 goes on to talk about existing approaches to spatial smoothing in a finite area.

1.1 Smoothing in two dimensions using splines

In ecological studies, it is typical that one of the covariates collected is the location at which the sample has been taken. Two possible uses for such data are considered here. First, location may be the only covariate collected, in which case estimating the spatial distribution of the phenomena in question is the goal (usually by plotting some kind of surface as a function of geographical coordinates). Alternatively, the spatial covariates may be used to remove spatial autocorrelation from the data, making the effects of non-spatial covariates clear and thus improving inference.

The following two examples highlight these two different objectives:

1. Chlorophyl levels in the Aral sea are monitored using satellite images. Each pixel in the image represents an area of 9 kilometres square on the Earth. However, the satellite images are noisy and so adjacent pixels can have vastly different measured levels of chlorophyll. One would expect the levels

to vary smoothly with location, so a model can be fitted to the image data which produces a smooth map of the chlorophyll concentration over the whole of the sea in an attempt to remove the noise. Figure 1-1 shows both raw and smoothed chlorophyll levels in the Aral sea (these data are revisited in section 4.6.4 and 5.3.2).

2. We wish to model the distribution of the North sea whiting population through space and time. In particular numbers of fish of age one were recorded by pulling a net up through the water column at a set of sample points. The sample locations and dates were recorded along with sea surface temperature, the identity and nationality of the ship that took the measurement and the depth of the sea bed at the sample location. Such a model can be used to draw inference about how the population has changed over time (for example, to see if overfishing is a problem or perhaps to see if there are reporting discrepancies between ships). The whiting's distribution in space and time is non-homogeneous (in particular it is known that yearlings tend to be found close to the shore) and failing to model this spatial heterogeneity could introduce bias in abundance estimates. Accurately modelling the spatial distribution is essential for reliable inference.

In both of these examples it is assumed that the phenomena in question (chlorophyll concentration and whiting density) vary smoothly according to their spatial location in the sense that the trend surface does not have large jumps as location changes. In many situations this assumption is biologically plausible.

There are many ways to construct models for the data described in the above examples, popular methods include kriging (Diggle and Ribeiro, 2007, Shabenberger and Gotway, 2005), kernel density estimation (Wand and Jones, 1994) and hierarchical Bayes models (Banerjee et al., 2003). Here the focus is on using *splines* (e.g. Wahba, 1990) for spatial smoothing via *additive models* (e.g. Hastie and Tibshirani, 1990).

Of particular interest here are smooth functions of space, and since the models are additive the emphasis is on situations akin to example 1 above, since if a method can be used in this context, it can also be included in models like those in example 2, simply as an additive component. For this reason, non-spatial covariates are ignored (for now).

Chapter 2 illustrates a situation akin to example 1 using administrative data from Italy and is based on the work in Marra et al. (2011).

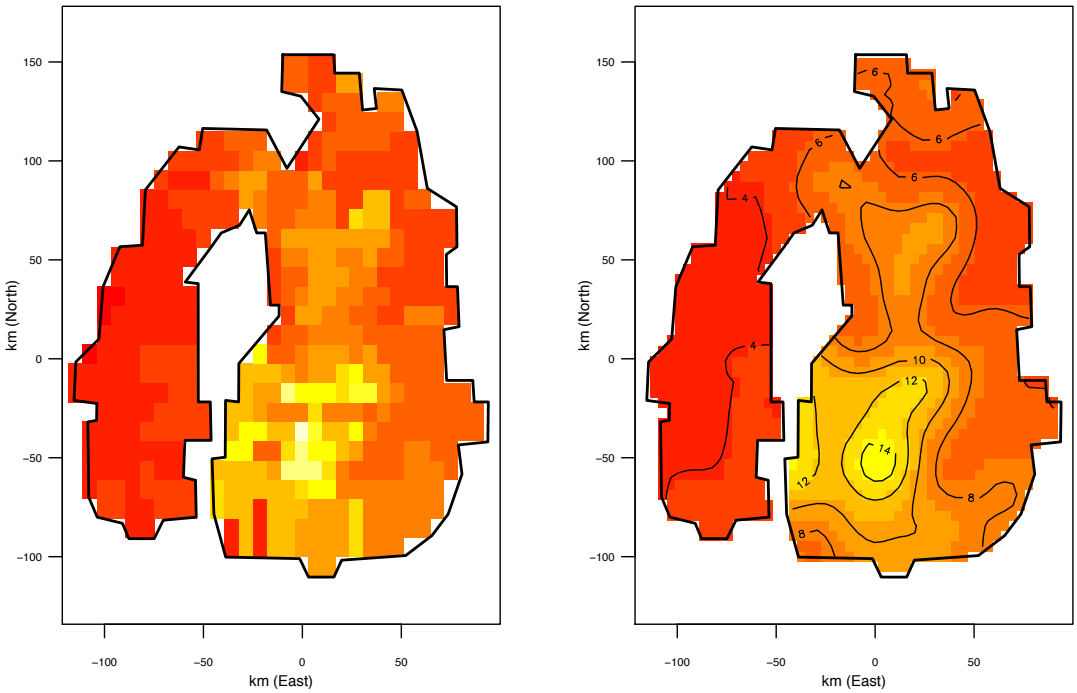

Figure 1-1: Left: raw chlorophyll levels in the Aral sea as recorded by the SeaWIFS satellite. Right: a smoothed version of the data. Further analysis of the data may be found in sections 4.6.4 and 5.3.2.

1.1.1 Basic setup

First, denote observations of the phenomenon of interest as z_i (in the examples above this would be the chlorophyll level or the yearling whiting catch at a particular point); i indexes the samples $i = 1, \ldots, n$, if there are n samples . Each z_i is the realisation of some random variable Z_i, where $Z_i = \mu_i + \epsilon_i$, where $\mu_i = \mathbb{E}(Z_i)$, the expected value of the i^{th} observation. Here ϵ_i is an error term and is assumed to be normally distributed with zero mean and some variance, σ^2. The spatial coordinates of the sample are also recorded, denote them $\mathbf{x}_i = (x_{i1}, x_{i2})$ (coordinates could be measured in latitude and longitude, or as kilometres North and East of some reference point, known as Northings and Eastings). The objective is to model the expected value of the response (μ_i) using the coordinates at which the data were collected.

Assuming that the phenomenon of interest varies smoothly in space is equivalent to saying that μ_i varies smoothly in space. Letting f be some smooth function, then as $\mu_i = f(\mathbf{x}_i)$:

$$z_i = f(\mathbf{x}_i) + \epsilon_i.$$

The observations are a sum of a smoothly varying spatial component and some random error. The problem is now how to estimate f.

One can imagine several possible ways of obtaining a suitable f. For example, one might simply work through a large book of mathematical functions, estimating parameters and finding the function that would fit the data best (for some definition of "best"). Alternatively one might estimate f as a kind of moving average of the values (for example LOESS, Cleveland and Devlin, 1988). The first option seems extremely time consuming (if it were even plausible) and the second will not give an "explicit" functional form at the end to slot into other procedures later. Rather than use either of these, a *basis function* representation is used for f. The idea is to build f out of a sum of J known functions, (b_js, say) scaled by coefficients (β_j, say) and then estimate these coefficients rather than the function as a whole. Mathematically:

$$f(\mathbf{x}_i) = \sum_{j=1}^{J} \beta_j b_j(\mathbf{x}_i). \tag{1.1}$$

Now some care must be taken in choosing both how many b_js are used (J) and their form. This is so that f sufficiently flexible over the whole of the domain that is to be smoothed over.

The next few sections present a brief introduction to how spatial smoothing using splines works, with a particular emphasis on the spatial case. However, it should be noted that at all times the models presented can be extended to higher (and lower dimensions) and that two dimensions are used for clarity and relevance. The primary references for the rest of this section are Wood (2006) and Ruppert et al. (2003), both provide excellent complementary introductions to the topic.

1.1.2 Smoothing with penalties

If f is very flexible it is possible that in estimating the β_js an f which tends toward interpolation of the data can be found. Interpolating the data is not useful since an f that simply jumps from datum to datum does not say any more about the spatial distribution than merely looking at the data. To obtain an f that interpolates the data, we can simply minimize the ordinary least squares (OLS) objective function. That is estimate the vector of coefficients, $\hat{\boldsymbol{\beta}}$, that

minimizes:

$$\sum_{i=1}^{n} \{z_i - f(\mathbf{x}_i)\}^2, \tag{1.2}$$

If the b_js are a sufficiently rich set of functions, this objective function does nothing to stop f simply interpolating the data (which would give a value of 0 in the above expression). To stop this from happening the "wigglyness" of f is penalized.

Penalizing the *wigglyness* (or *roughness*, Ruppert et al., 2003) of f makes sense since (as stated above) the belief is that the underlying phenomena is smooth. Mathematically, taking (1.2) and adding on a penalty based on the wigglyness gives:

$$\sum_{i=1}^{n} \{z_i - f(\mathbf{x}_i)\}^2 + \lambda \int \dots \int ||Pf(\mathbf{x})||^2 \mathrm{d}\mathbf{x}. \tag{1.3}$$

Here P is some derivative operator, for example the second derivatives (e.g. $P = \left(\frac{\partial^2}{\partial x_1^2}, \sqrt{2}\frac{\partial^2}{\partial x_1 x_2}, \frac{\partial^2}{\partial x_2^2}\right)$ in a 2-dimensional case). Integrating the derivatives over the whole space gives a measure of the wigglyness of the function, functions which vary a lot will lead to large values of the integral and hence have larger penalties. The exact form of P changes with the basis and dimensionality of the problem (as will be seen in the next section).

Depending on the situation, the penalty should have a different amount of influence on (1.3). The *smoothing parameter*, $\lambda(\geq 0)$, controls the trade-off between interpolation (which happens as $\lambda \to 0$, leading to no penalty) and fitting a simpler function (which happens as $\lambda \to \infty$, where all terms are penalized aside from those for which the integral evaluates to zero: those in the *nullspace* of the penalty). Figure 1-2 shows how different values of λ affect the fitted smooth function. Determining the value of λ will be covered in section 1.1.4. For now it is assumed that some optimal λ is known.

The expression for the penalty given in (1.3) looks like it might require a rather large amount of integration and as such would require a long time to compute however, it can be shown (Wood, 2006, p. 128) that the integral can be written as:

$$\int \dots \int ||Pf(\mathbf{x})||^2 \mathrm{d}\mathbf{x} = \boldsymbol{\beta}^{\mathrm{T}} \mathbf{S} \boldsymbol{\beta}, \tag{1.4}$$

where the ij^{th} element of $\mathbf{S}$ is the integral of the product of the appropriate derivatives (in the above example, second) of the i^{th} and j^{th} basis functions. Put

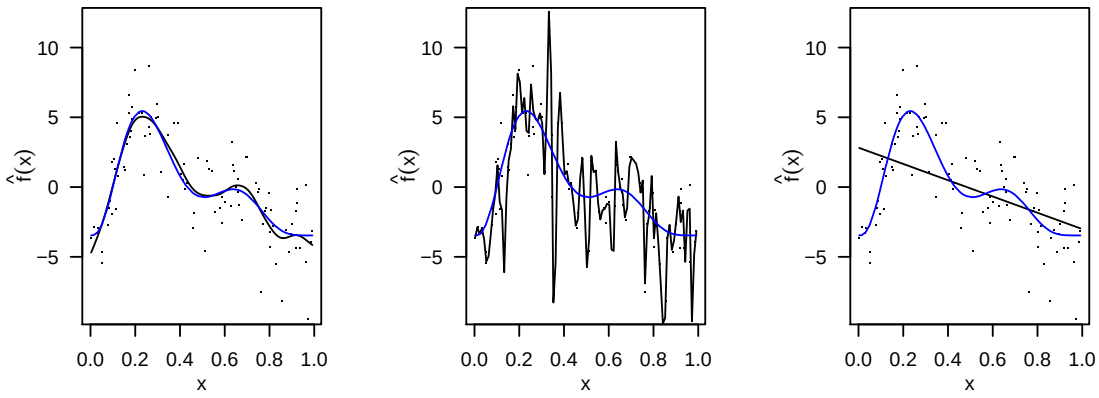

Figure 1-2: An example of how different values of λ affect the fitted smooth function of one variable. In the left panel, λ is chosen optimally by GCV (see section 1.1.4), in the middle panel $\lambda = 0$, tending towards an interpolating fit. In the right panel $\lambda = \infty$ leading to a straight line. In each panel the blue curve is the true function and the points are the data sampled from it with error.

more mathematically:

$$\mathbf{S}_{ij} = \int \dots \int \left(Pb_i(\mathbf{x})\right) \left(Pb_j(\mathbf{x})\right)^{\mathrm{T}} \mathrm{d}\mathbf{x}.$$

So in this case the penalty matrix $\mathbf{S}$ only needs to be computed once. Computation of the penalty is merely a case of calculating the quadratic form in (1.4).

1.1.3 Spline bases

So far all that has been said about the form of f is that it can be decomposed into a series of basis functions. Three bases are discussed here: *thin plate regression splines*, *P-splines* and *cubic splines*, which will be used in this thesis.

Thin plate regression splines

This section begins by discussing *thin plate splines* before going on to describe a computationally efficient version of the basis (*thin plate regression splines*) which will be used throughout the thesis. Thin plate splines are particularly useful in a spatial setting because they have a property known as *isotropy*: all directions have a common smoothing parameter so wigglyness in the x_1 direction has the

same weight in the penalty as in the x_2 direction (and so on through higher dimensions). This property is usually appropriate in a spatial setting, since there is nothing special about one geographical coordinate over another when it comes to the smoothness of the function to be fitted.

Thin plate splines were first proposed by Duchon (1977). Duchon begins by presenting the following penalty:

$$J_{m,d} = \int \dots \int_{\mathbb{R}^d} \sum_{\nu_1+\dots+\nu_d=m} \frac{m!}{\nu_1! \dots \nu_d!} \left(\frac{\partial^m f(x_1, \dots, x_d)}{\partial x_1^{\nu_1} \dots \partial x_d^{\nu_d}} \right)^2 \mathrm{d}x_1 \dots \mathrm{d}x_d, \qquad (1.5)$$

where m is the *derivative order*, d is the dimension of the data (in a spatial setting $d = 2$) and the $\nu_1, \dots, \nu_d$ terms simply ensure that derivatives are taken with respect to all the parameters in all of the necessary combinations.

Replacing the penalty in (1.3) with (1.5), the objective function is then:

$$\sum_{i=1}^{n} \{z_i - f(\mathbf{x}_i)\}^2 + \lambda J_{m,d}$$

It can then be shown that this objective function is minimized by a function of the form:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \delta_i \eta_{m,d}(r_i) + \sum_{j=1}^{M} \alpha_j \phi_j(\mathbf{x}), \qquad (1.6)$$

where $r_i = ||\mathbf{x} - \mathbf{x_i}||$ (the Euclidean norm of $\mathbf{x} - \mathbf{x_i}$) and the δ_is and α_j are parameters to be estimated. As in (1.1), f is decomposed into a sum of basis functions however, for a thin plate spline this summation is split into two parts: M polynomials that act over the whole of the data (the ϕ_js) and a set of *radial basis functions*, one centred at each datum (the $\eta_{m,d}$s). One can think of this as a global trend (in the 2-dimensional case, linear functions of the two coordinates) with extra flexibility provided by the radial basis functions.

The radial basis functions $\eta_{m,d}(r)$ are defined as:

$$\eta_{m,d}(r) = \begin{cases} \frac{(-1)^{m+1+d/2}}{2^{2m-1}\pi^{d/2}(m-1)!(m-d/2)!} r^{2m-d} \log(r) & d \text{ even}, \\ \frac{\Gamma(d/2-m)}{2^{2m}\pi^{d/2}(m-1)!} r^{2m-d} & d \text{ odd}. \end{cases}$$

where Γ is the gamma function. The ϕ_js are $M = \binom{m+d-1}{d}$ linearly independent polynomials of degree less than m which span the space of polynomials in $\mathbb{R}^d$; all of the ϕ_js are unpenalized as they lie in the nullspace of the penalty. It is also

important to note that to maintain continuity in f, $2m > d$; this means that the dimension of the nullspace increases rapidly with d (this will be discussed further in chapter 5).

This all looks rather complex, however in the 2-dimensional case, (1.5) looks much simpler. Setting $d = 2$ and $m = 2$, (1.5) is given as:

$$J_{2,2} = \int\int \left(\frac{\partial^2 f(x_1, x_2)}{\partial x_1^2}\right)^2 + 2\left(\frac{\partial^2 f(x_1, x_2)}{\partial x_1 \partial x_2}\right)^2 + \left(\frac{\partial^2 f(x_1, x_2)}{\partial x_2^2}\right)^2 \mathrm{d}x_1 \mathrm{d}x_2,$$

and f is:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \delta_i \eta_{2,2}(r_i) + \sum_{j=1}^{3} \alpha_j \phi_j(\mathbf{x}),$$

where:

$$\eta_{2,2}(r) = \frac{1}{8\pi} r^2 \log(r).$$

The nullspace of the penalty consists of three functions: $\phi_1(\mathbf{x}) = 1, \phi_2(\mathbf{x}) = x_1$ and $\phi_3(\mathbf{x}) = x_2$, which make no contribution to $J_{2,2}$. Figure 1-3 shows some examples of 2-dimensional thin plate basis functions.

The computational cost of fitting the model is cubed in the number of parameters, so fitting a model with one radial basis function per datum may prove impractical in some cases. To avoid this problem, one could either (*i*) select (perhaps randomly) some of the data and use only those points to create the basis and then use the full data to fit the model (i.e. changing the index of the summation in the first term of (1.6)) or (*ii*) select a (relatively small) representative set of points within the space covered by the data (though not necessarily data locations) which would be evenly spread out enough to create the basis (changing the $\mathbf{x}_i$s in the first summation in (1.6) – these points are known as *knots*). Both approaches have potential problems, namely: how many points should be chosen and where they should best be located? Both of theses approaches effectively reduce the size of the basis (changing the limit on the first summation in (1.6)), however both methods do this in a fairly arbitrary way. There is no objective measure of whether the points selected are "good".

Wood (2003) proposes a low-rank approximation to thin plate splines, referred to in this thesis as *thin plate regression splines*. Let the ij^{th} element of the $(n \times n)$ matrix $\mathbf{E}$ be the j^{th} radial basis function evaluated at the i^{th} datum (i.e. $\mathbf{E} = \eta_{m,d}(||\mathbf{x}_i - \mathbf{x}_j||)$), reducing the computations required to fit the model can be achieved by reducing the rank of $\mathbf{E}$. In the previous two approaches the

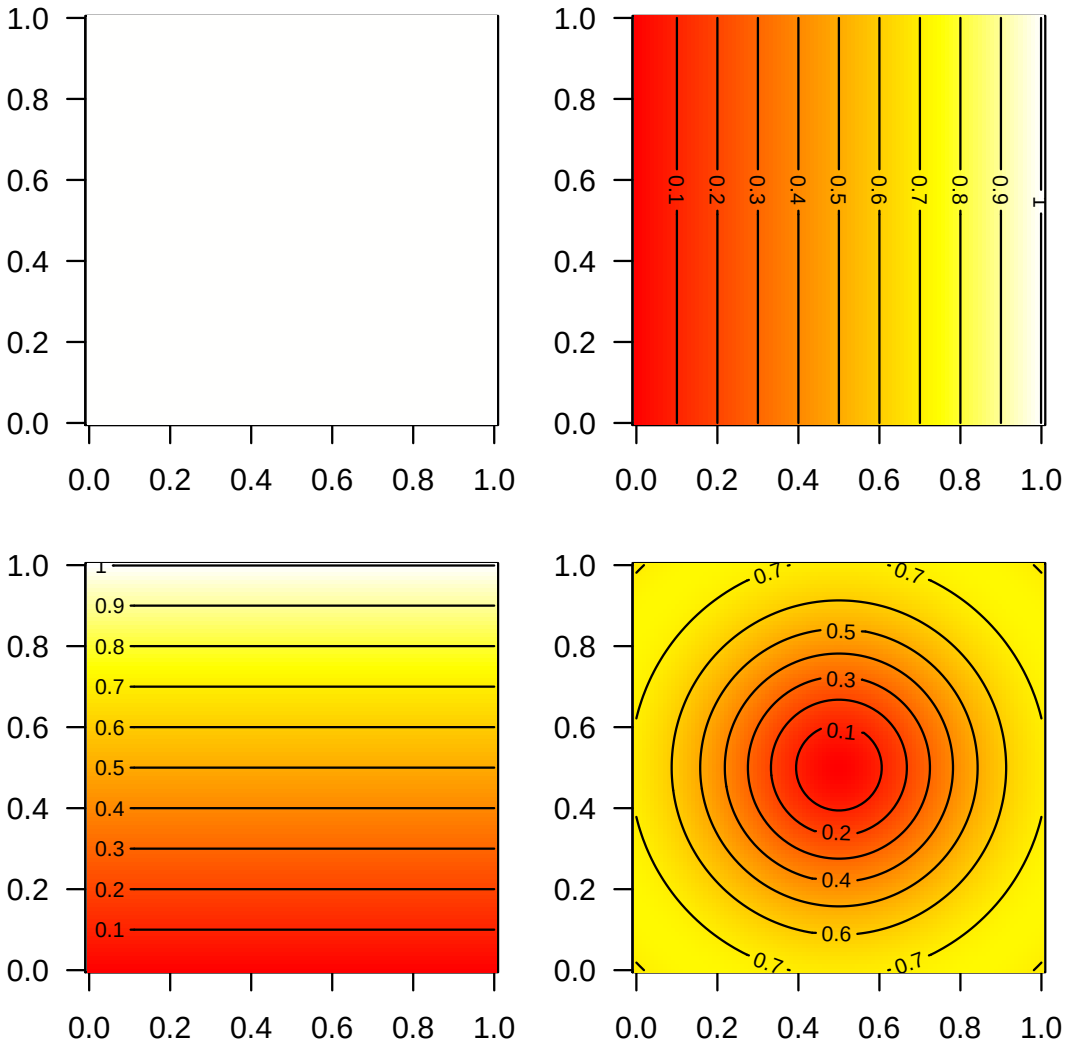

Figure 1-3: Example of a thin plate spline basis. The first three are $\phi_1(\mathbf{x}) = 1, \phi_2(\mathbf{x}) = x_1$ and $\phi_3(\mathbf{x}) = x_2$, which are in the nullspace of the $J_{2,2}$ penalty. The bottom right plot shows an example of a radial basis function centred on $(0.5, 0.5)$ with coefficient -100 (to put it on the same colour scale).

rank reduction was performed by removing columns from $\mathbf{E}$ (randomly sampling the data) or changing the number of columns by changing the location of the evaluations (using knots).

One way of reducing the size of $\mathbf{E}$ is by performing an eigen-decomposition (so $\mathbf{E} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}}$, where the columns of $\mathbf{U}$ are orthogonal eigenvectors and $\mathbf{\Lambda}$ is a diagonal matrix of eigenvalues decreasing in absolute value down the diagonal). Picking the k largest eigenvalues and truncating at that point (taking the first k columns of $\mathbf{U}$ and the top right $k \times k$ submatrix of $\mathbf{\Lambda}$) gives $\mathbf{E}_k (= \mathbf{U}_k \mathbf{\Lambda}_k \mathbf{U}_k^{\mathrm{T}})$. It can be shown that the reduced rank matrix $\mathbf{E}_k$ gives the best approximation to $\mathbf{E}$ (see Wood, 2003 for details). In practice, k is set to be "large enough" and further reduction in basis complexity is performed by penalization (see section 1.3.4). In the simulations and analyses in the following chapters k will be referred to as the "maximum basis size" to avoid confusion.

This low-rank approximation is the default implementation when using the "`tp`" basis in the `R` package `mgcv`. This package was used throughout the thesis.

P-splines

P-splines (Eilers and Marx, 1996) consist of B-spline basis function with discrete penalties, so before thinking about P-splines, the properties of B-splines are considered.

B-splines are simple local basis functions; they are simple in that all of the basis functions have the same shape and local in that they only have an effect near where they are centred (at the knots) – *compact support*. Taking (1.1), we replace the b_js with an $(m+1)^{\text{th}}$ order B-spline B_j^m (where the order is chosen). Note that the B_j^ms are only a function of one covariate, x, at this point but will be expanded to higher dimensions below. So, the basis representation of f given in (1.1) becomes:

$$f(x) = \sum_{j=1}^{J} \beta_j B_j^m(x).$$

The B_j^ms are defined recursively as:

$$B_j^m(x) = \frac{x - x_j^*}{x_{j+m+1}^* - x_j^*} B_j^{m-1}(x) + \frac{x_{j+m+2}^* - x}{x_{j+m+2}^* - x_{j+1}^*} B_{j+1}^{m-1}(x) \quad \text{for } j = 1, \ldots, J,$$

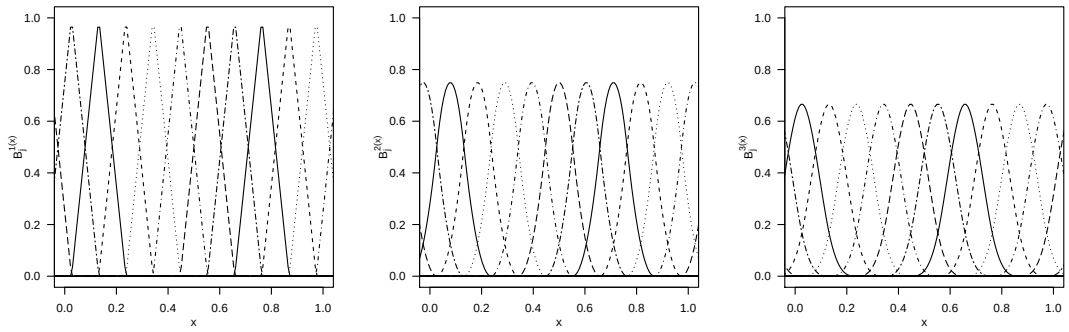

Figure 1-4: An example of B-spline basis functions for $m = 1, 2$ and 3 (from left to right) with evenly spaced knots (these are located at the peaks of the basis functions).

and

$$B_j^{-1}(x) = \begin{cases} 1 & x_j^* \leq x < x_{j+1}^* \\ 0 & \text{otherwise.} \end{cases}$$

The $J + m + 1$ knots x_j^* are evenly spaced over the x-axis and these, along with the order of the basis, determine how flexible f is. Each $B_j^m(x)$ is non-zero over the $m + 3$ adjacent knots. Contrary to their rather complex functional form, the functions shown in figure 1-4 are rather simple. From left to right the panels show B-splines bases with $m = 1, 2, 3$ for evenly spaced knots over $(0, 1)$.

P-splines take the B-spline basis and add a penalty structure. Because of their local nature, the penalty is somewhat different to the general penalty in (1.3) and is based on differences between adjacent bases. Since the B_j^ms are defined locally, smoothness only needs to be enforced on neighbouring basis functions. The objective function given in (1.3) then becomes:

$$\sum_{i=1}^{n} \{z_i - f(x_i)\}^2 + \lambda \sum_{j=1}^{J-1} (\beta_{j+1} - \beta_j)^2,$$

if squared second differences are used (this could be a higher order difference, see Eilers and Marx, 1996). Such a penalty is very fast to compute, since many of the elements of the penalty matrix, $\mathbf{S}$, are zero due to the local nature of the basis functions.

Cubic splines

Cubic splines are another a univariate basis and, like P-splines they require the specification of knots. Cubic splines are made up of sections of cubic polynomials which are continuous (up to second derivatives) at the joins. The objective function for a cubic spline is:

$$\sum_{i=1}^{n} \{z_i - f(x_i)\}^2 + \lambda \int_{x_1^*}^{x_J^*} \left(\frac{\partial^2 f(x)}{\partial x^2} \right)^2 \mathrm{d}x,$$

where $\{x_j^* : j = 1, \ldots, J\}$ are J knots. This gives rise to the set of cubic polynomials that form the basis. Such a basis has many possible parametrizations; here the "cardinal" parametrisation is used (the basis functions take the value one at one knot and zero at all others). Letting $\beta_j = f(x_j^*)$ and $\delta_j = \frac{\partial^2 f(x)}{\partial x^2}|_{x=x_j^*}$ (so one can then write the δ_js as a function of the β_js), the parametrization gives the following form for f:

$$\begin{aligned} f(x) =& \frac{x_{j+1}^* - x}{x_{j+1}^* - x_j^*}\beta_j + \frac{x - x_j^*}{x_{j+1}^* - x_j^*}\beta_{j+1} \\ &+ \left\{ \frac{\left(x_{j+1}^* - x\right)^3}{x_{j+1}^* - x_j^*} - (x_{j+1}^* - x_j^*)(x_{j+1}^* - x) \right\} \frac{\delta_j}{6} \\ &+ \left\{ \frac{\left(x - x_j^*\right)^3}{x_{j+1}^* - x_j^*} - (x_{j+1}^* - x_j^*)(x - x_j^*) \right\} \frac{\delta_{j+1}}{6} \qquad \text{if } x_j^* \leq x \leq x_{j+1}^*. \end{aligned}$$

Summing over j then yields an implicit set of J b_js as in (1.1). The cubic spline basis is then parameterized in terms of its values (and the values of its derivatives) at the knots. This setup may seem odd but does lead to the spline having directly interpretable parameters (which is not the case for, say, thin plate splines). An example of a basis function is given in figure 1-5.

Further details may be found in Wood (2006, pp. 122-126, pp. 149-151).

Cyclic cubic splines

The above spline basis may be extended to a *cyclic cubic spline* by imposing the constraint that $f(x_1) = f(x_k)$ and that the values of the first and second derivatives must also match. This specifies that the spline must "join up" at each end. The form of f is the same as before, but there is one less coefficient to estimate (since the first and last are the same).

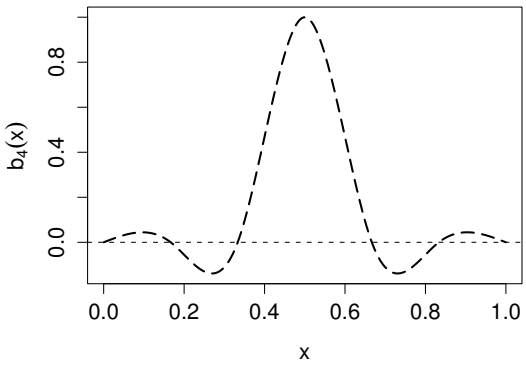

Figure 1-5: An example of a cubic spline basis function with a knot at 0.5. Figure taken from Wood (2006, p. 151).

Tensor products

Both P-splines and cubic splines are defined only in one dimension, however, it is possible to build 2-dimensional (and higher) smooths from *tensor products* of 1-dimensional bases. This is made possible by thinking of each 1-dimensional basis as a marginal of the higher dimensional smooth. A spatial smooth of, say x_1 and x_2 can be constructed by first writing down the basis expansions for the marginal smooths of x_1 and x_2 (in general terms, since this applies to all splines not just P-splines and cubic splines, see section 2.2.2):

$$f_{x_1}(x_1) = \sum_{j=1}^{J} \beta_j b_j(x_1), \quad f_{x_2}(x_2) = \sum_{k=1}^{K} \delta_k d_k(x_2). \tag{1.7}$$

where the δ_k and d_k are analogous to β_j and b_j and assuming basis sizes of J and K (of course, J and K may be the same size). To make f_{x_1} vary with x_2 we can then simply make the β_js a function of x_2, the simplest way of doing this would be to define:

$$\beta_j(x_2) = \sum_{k=1}^{K} \delta_{jk} d_k(x_2).$$

so then f_{x_1,x_2} would is defined as:

$$f_{x_1,x_2}(x_1, x_2) = \sum_{j=1}^{J} \sum_{k=1}^{K} \delta_{jk} d_k(x_2) b_j(x_1).$$

Finally, the penalty for such a smooth can be written:

$$\int\int \lambda_{x_1} \left\{P_{x_1} f_{x_1,x_2}(x_1,x_2)\right\}^2 + \lambda_{x_2} \left\{P_{x_2} f_{x_1,x_2}(x_1,x_2)\right\}^2 \mathrm{d}x_1 \mathrm{d}x_2,$$

where P_{x_1} and P_{x_2} are derivative operators with respect to x_1 and x_2 respectively (or the equivalent discrete penalty in the P-spline case).

Although only a very simplistic example is given here, tensor product splines provide an extremely useful tool, allowing for extra dimensions to be added to models using different bases. The use of a different smoothing parameter for each direction allows for *anisotropic* smoothing, so that covariates that are measured on different scales (for example temperature and length) may be combined into one tensor product smooth, avoiding the assumption that the degree of smoothing required is the same in both directions. In particular this can be useful when constructing a spatiotemporal smooth: for example using a thin plate spline for the spatial part of the smooth (so the spatial part of the model is isotropic) then taking a tensor product of that with a cubic spline basis (or another 1-dimensional thin plate spline) for the temporal effect (so a different amount of smoothing can be used for each direction). This is the setup that will be used in chapter 2 for the Italian data.

1.1.4 Smoothing parameter selection by GCV

The objective function given in section 1.1.2 only allows the estimation of $\hat{\boldsymbol{\beta}}$ for some given λ (or $\boldsymbol{\lambda}$); finding an optimal smoothing parameter has not yet been addressed. A simple and effective way to find $\hat{\lambda}$ is to assess how well the model performs on data which were not in the sample – i.e. assessing the prediction error of the model. The *leave-one-out cross validation* score (LOOCV, see section 1.3.3) does exactly this by fitting the model to all but one of the data and calculating the difference between the prediction of the excluded datum and its true value (LOOCV is also known as *ordinary cross validation*, OCV). Rather than fitting n models to the data (one for each excluded datum), the *generalized cross validation* (GCV) score can be used, which can be written as follows, allowing for easy computation:

$$\mathcal{V}_g = \frac{n||\mathbf{z} - \hat{\mathbf{f}}||^2}{\{n - \mathrm{tr}(\mathbf{A})\}^2}, \tag{1.8}$$

where tr($\mathbf{A}$) indicates the trace of $\mathbf{A}$, the hat (or influence) matrix for the smoother (see section 1.3.4), $\hat{\mathbf{f}} = \left(\hat{f}(\mathbf{x}_1), \hat{f}(\mathbf{x}_2), \ldots, \hat{f}(\mathbf{x}_n)\right)^{\mathrm{T}}$ is the vector of fitted values for the model (i.e. evaluations of $\hat{f}$ at each of the data points). Numerical minimization of $\mathcal{V}_g$ with respect to λ (which enters $\mathcal{V}_g$ via $\mathbf{A}$) gives the optimal smoothing parameter ($\hat{\lambda}$). Further details are given in Wood (2006, pp. 134–137).

There are, of course, other ways to select optimal smoothing parameters. In the unidimensional case, a simple grid search for λ is possible, however this obviously becomes computationally burdensome once more than one optimal smoothing parameter is required. When the scale parameter is known the *Un-Biased Risk Estimator* (UBRE) can be used in place of GCV (Craven and Wahba, 1979). UBRE can be thought of as an estimate of the expected mean square error: the expected, squared, distance between the truth and the estimated model (see section 1.3.1, below, for further discussion of the MSE). Further discussion of smoothing parameter selection is given in sections 4.5, 4.6 and 4.7 of Wood (2006).

Details of how smoothing parameter selection and estimation of $\hat{\boldsymbol{\beta}}$ are combined into a fitting procedure are given in section 1.2.3.

1.2 Extending to more complex models

So far only smooths of two geographical coordinates with normal errors in the response have been discussed. This section gives an overview of some extensions to these models.

1.2.1 Covariates

Although in the above, the focus has been on including on geographical coordinates as explanatory variables, other covariates (or combinations of covariates) can be included in an additive way. The notation in (1.3) and (1.4) is simply extended in this case and all of the above results hold, simply by defining $f(\mathbf{x}) = \sum_k f_k(\mathbf{x}^{(k)})$ for the k smooth functions f_k of corresponding covariates $(\mathbf{x}^{(k)})$. The smoother matrix $\mathbf{S}$ is replaced by $\mathbf{S} = \sum_k \lambda_k \mathbf{S}_k$ where each $\mathbf{S}$ corresponds to an f_k and multiple smoothing parameters (the λ_k) must be estimated. With such additive models identifiability becomes an issue since each of the f_k can only be found up to some additive constant (since one could add a value, a

to f_1, say and then subtract it from f_2 leading to the same model as if a were not included). To get around this problem *identifiability constraints* are used. By enforcing a sum to zero constraint on the value of each function at the observed covariates the model is sure to be identifiable.

1.2.2 Higher dimensional smooths

As well as including covariates as additive components, they can also be included as extra dimensions via tensor products (or directly into the basis in the thin plate regression spline case). When using thin plate splines, the order of the derivatives in the penalty (m) can be changed. Indeed, it is required that the derivative order changes according to the number of dimensions that smoothing takes place in ($2m > d$). Using a derivative order that is too low can lead to a non-smooth f, which is clearly undesirable. Smoothing in high dimensions can be unreliable and numerically tricky but not impossible (as will be seen later in chapter 5).

1.2.3 Generalized additive models

Up until now only additive models with normal errors have been considered. If other exponential family response distributions are used with the models described above, we call this a *generalized* additive model (GAM) and in that case we may model $\eta_i = g(\mu_i)$ where g is a *link function* (in the same sense as the GLM case, see Hardin and Hilbe, 2002, p. 8) and η_i is the linear predictor. In order to estimate $\hat{\boldsymbol{\beta}}$, the *penalized iteratively re-weighted least squares* (PIRLS) algorithm must be used.

To use PIRLS, one must first think of the GAM as a penalized GLM. Consider first the usual GLM model matrix $\mathbf{X}$. By appending the basis evaluations of each datum as columns of $\mathbf{X}$ (i.e. $\mathbf{X} := (\mathbf{X}, \mathbf{X}^*)$, where the ij^{th} element of $\mathbf{X}^*$ is $b_j(\mathbf{x}_i)$), the smooth terms in the model can be included in the usual model matrix setup. The PIRLS algorithm is as follows (Wood, 2006, p. 138):

First define $\eta_i = \mathbf{X}_i\boldsymbol{\beta}$ (where $\mathbf{X}_i$ is the i^{th} row of $\mathbf{X}$) as the linear predictor such that $\mu_i = g^{-1}(\eta_i)$ and the variance function, $V(\cdot)$, such that $\text{Var}(Z_i) = \phi V(\mu_i)$ (ϕ is the scale parameter, see Wood, 2006, p. 62). At iteration k the PIRLS algorithm is:

1. Given the current linear predictor estimate and mean response vectors ($\boldsymbol{\eta}^{[k]}$

and $\boldsymbol{\mu}^{[k]}$, respectively) calculate the diagonal weight matrix with the following elements:

$$\mathbf{W}_{ii}^{[k]} \propto \frac{1}{V(\mu_i^{[k]})g'(\mu_i^{[k]})^2},$$

and the pseudodata:

$$s_i = g'(\mu_i^{[k]})(z_i - \mu_i^{[k]}) + \mathbf{X}_i\boldsymbol{\beta}^{[k]},$$

stored in the n-vector $\mathbf{s}$. $g'(\mu_i^{[k]})$ is the first derivative of the link function with respect to $\mu_i^{[k]}$.

2. Minimize

$$||\sqrt{\mathbf{W}^{[k]}}(\mathbf{s}^{[k]} - \mathbf{X}\boldsymbol{\beta})||^2 + \sum_j \lambda_j \boldsymbol{\beta}^{\mathrm{T}}\mathbf{S}_j\boldsymbol{\beta}$$

with respect to $\boldsymbol{\beta}$, giving $\boldsymbol{\beta}^{[k+1]}$ and hence allowing the calculation of $\boldsymbol{\eta}^{[k+1]}$ and $\boldsymbol{\mu}^{[k+1]}$.

This procedure is iterated to convergence of $\boldsymbol{\beta}$ for the given smoothing parameter(s).

The "hierarchical" optimisation method of Wood (2011) is used throughout this thesis to find $\hat{\boldsymbol{\beta}}$ and $\hat{\boldsymbol{\lambda}}$. That is: at each iteration a smoothing parameter is selected to optimize the (GCV) score, which in the generalized case is given by:

$$\mathcal{V}_g = \frac{nD(\hat{\boldsymbol{\beta}})}{\{n - \mathrm{tr}(\mathbf{A})\}^2},$$

where $D(\hat{\boldsymbol{\beta}})$ is the model *deviance* (defined as the saturated log-likelihood minus the log-likelihood, evaluated at the current parameter values, all multiplied by 2ϕ) (see Wood, 2006, p. 178). This optimal $\boldsymbol{\lambda}$ then implies a set of model coefficients (the best $\boldsymbol{\beta}$ for that $\boldsymbol{\lambda}$) which are found using PIRLS, the algorithm then proposes a new $\boldsymbol{\lambda}$ based on the derivatives of $\mathcal{V}_g$ with respect to $\log \boldsymbol{\lambda}$ at this point (the log scale is used to ensure that $\boldsymbol{\lambda}$ remains positive). This continues until convergence.

Other GAM fitting methods

It has been observed (Reiss and Ogden, 2009) that the GCV score can sometimes have problems with multiple minima (i.e. the optimisation can get stuck at a non-optimal λ). Reiss and Ogden (2009) also show that GCV tends to give more

variable estimates for the smoothing parameter(s). An alternative to minimizing the GCV score ($\mathcal{V}_g$) is to use a likelihood-based method such as approximate REML or ML, which are less prone to this problem (Wood, 2011). The key to understanding restricted maximum likelihood or (marginal) maximum likelihood methods is the realisation that the estimates of the coefficients (the $\hat{\boldsymbol{\beta}}$) are the posterior modes of the distribution of $\boldsymbol{\beta}|\mathbf{z}$ if $\boldsymbol{\beta} \sim N(\mathbf{0}, \phi\mathbf{S}^{-})$ (where $\mathbf{S}^{-}$ the generalized matrix inverse of $\mathbf{S}$ and ϕ is the scale parameter, as above). When the $\boldsymbol{\beta}$ are considered to be random variables, smoothing parameters can then be thought of as variance parameters and a marginal likelihood can then be formed (this is known as the *random effects formulation*). This likelihood can then be optimized with respect to λ and put in the place of the GCV score ($\mathcal{V}_g$) in hierarchical procedure above. At each iteration the optimal $\boldsymbol{\beta}$ is found via PIRLS.

Ruppert et al. (2003, pp. 120-123) provide an interesting discussion of automatic smoothing parameter selection. In conclusion they state that there is no clear "best" method for selecting smoothness but rather that both methods are imperfect in different ways.

There are many other ways of fitting GAMs, such as backfitting (Hastie and Tibshirani (1990), see also section 5.1), Markov chain Monte Carlo (MCMC, e.g. Fahrmeir et al. (2004)) or integrated nested Laplace approximations (INLA, Rue et al. (2009)). The hierarchical procedure of Wood (2011) (using either GCV or REML/ML) was chosen over these other methods for several reasons. Backfitting and MCMC are quite computationally expensive, especially when larger models are used (see Wood (2006), pp. 213-215 for why this is the case for backfitting). MCMC-type approaches can be improved by exploiting sparse bases (like P-splines) however, thin plate regression splines (and their nice properties like isotropy) cannot be used. INLA can be very fast, however when many terms are included it becomes computationally expensive (Rue et al. (2009) suggest more than 10 become problematic).

Ruppert et al. (2009) gives an overview of developments in the area of semiparametric regression in general during the 2003–2007 period.

1.3 Smoothing in practice

This final section deals with two topics not covered above: assessing model performance and the practical implementation of the methods detailed above.

1.3.1 Mean squared error

Simulation studies will be used extensively to evaluate the proposed methods. In order to determine the performance of the methods, some metric must be chosen. Mean squared error (MSE) is a standard approach to assessing the performance of a model, in terms of its prediction error. The MSE of the fitted model $\hat{f}$ is defined as:

$$\text{MSE}(\hat{f}) = \mathbb{E}\left[\left\{\hat{f}(X) - f(X)\right\}^2\right],$$

which (if there are n_p prediction points) can be estimated as:

$$\widehat{\text{MSE}}(\hat{f}) = \frac{1}{n_p}\sum_{i=1}^{n_p}\left\{\hat{f}(\mathbf{x}_i) - f(\mathbf{x}_i)\right\}^2, \tag{1.9}$$

where $f(\mathbf{x}_i)$ is the "true" values of f at the prediction points, $\mathbf{x}_i$. In the spatial simulations in the coming chapters, the prediction points will be a relatively dense set of locations to test how well $\hat{f}$ models the true surface.

To test to see if two models' MSEs are significantly different, a paired Wilcoxon signed rank test (e.g. Wetherill, 1982, pp.173-175) is used to analyse the results from simulations. When MSE is mentioned from now on, it will be with reference to $\widehat{\text{MSE}}$ in (1.9).

1.3.2 The Brier score

The Brier score (Brier, 1950) is useful when the response variable is binary (a more general version of the score for m-ary variables can also be used). In this case the probability of observations belonging to a particular class (0 or 1 in the binary case) is measured, so it makes sense to assess the model performance using the probabilities rather than just the classifications. The score is similar in form to the MSE:

$$\text{BS} = \frac{1}{n}\sum_{i=1}^{n}\left\{\hat{f}_p(\mathbf{x}_i) - z_i\right\}^2 \tag{1.10}$$

where subscript f_p indicates that the functions return values on the probability scale, rather than on the scale of the linear predictor. The z_is are the n, m-chotomous data. The Brier score has the advantage of offering a more granular measure of the model errors, since probabilities will be continuous on $(0, 1)$ rather than dichotomous (or m-chotomous) class predictions.

1.3.3 Leave-one-out cross validation

When analysing non-simulated data (where the true values are unknown) it is still sometimes necessary to quantify the out-of-sample error in predictions. This can be achieved using leave-one-out cross validation (LOOCV). The LOOCV score is calculated as:

$$\text{LOOCV} = \frac{1}{n}\sum_{i=1}^{n}\left\{\hat{f}^{-i}(\mathbf{x}_i) - z_i\right\}^2, \tag{1.11}$$

where $\hat{f}^{-i}$ is the model fit to all of the data except the i^{th}, so one can think of the LOOCV score as a measure of fit to unseen data.

1.3.4 Effective degrees of freedom

The hat or influence matrix, $\mathbf{A}$, is the matrix such that $\hat{\boldsymbol{\eta}} = \mathbf{A}\mathbf{z}$. It has the rather useful property that taking the trace of $\mathbf{A}$ ($\text{tr}(\mathbf{A})$) gives the *effective degrees of freedom* (EDF) of the model. The EDF gives a measure of the complexity of the fitted model. The higher the EDF, the more complex the model. The rationale for this is by analogy to the linear model. In that case we know that the degrees of freedom is simply the length of $\boldsymbol{\beta}$ (minus any identifiability constraints) which is the same as the value of $\text{tr}(\mathbf{A})$ if the smoothing parameters are all set to zero. It can also be shown that the minimum value of the EDF is $\text{rank}(\sum_i \mathbf{S}_i)$ and that the EDF varies smoothly between these two values with the smoothing parameters (Wood, 2006, p. 170–171).

The EDF can be a useful tool when it comes to model choice and model diagnostics; if models seem to have similar performance then looking at the EDF may give a reason to choose one over another (if one is simpler). In most cases the basis dimension is set as an upper bound, the smoothing penalty suppresses parts of the model (see section 1.1.3). Therefore basis dimension is not a major concern provided that it is not set too low (Wood, 2006, p. 161).

1.3.5 `mgcv`

Throughout this this thesis the software package `mgcv` by Simon Wood is used (although some bespoke software was needed, see section 6.2). The package is free (GPL) software for the language R (R Development Core Team, 2011). The library gives a simple, extensible collection of fitting routines, bases and diagnostics.

1.3.6 Summing up

This section has hopefully given a brief introduction to generalized additive models in the setting of spatial smoothing. The next section goes into the particulars of the problem that will be address in the first part of the thesis: finite area smoothing.

1.4 Finite area smoothing

1.4.1 Overview of finite area smoothing

As we have seen so far, splines are a flexible way to perform spatial smoothing in two dimensions. To recap, a typical application consists of a response modelled as a function of its spatial coordinates. The estimated function can then be used to perform inference. This may simply consist of creating maps of the phenomenon (see chapter 2 for an example) or as part of a larger model, taking into account nuisance spatial effects. Finite area smoothing concerns the situation in which the domain over which this smoothing takes place is bounded.

When the geographical region has a *complex boundary*, features from one part of the domain can unduly influence other parts. An example of non-convexity would the case where the polygon has some peninsula-like feature(s) so that there is a gap between two parts of the domain. Of course this would only be problematic if there were notably different observed values on either side of such a feature. Given that there is some scientific motivation as to why those parts of the domain should not affect each other, features such as peninsulae give rise to a phenomenon known as *leakage*.

Leakage occurs when a smoother inappropriately links two parts of a domain (Wood et al., 2008). The phenomenon is problematic since it causes the fitted surface to be mis-estimated; this can then lead to incorrect inference (e.g. bias). Leakage can be seen in figure 1-6 where the high values in the upper half of the domain leak across the gap to the lower values below and vice versa.

The problem of leakage arises because of the way in which the smoother measures how near objects are to one another. Most smoothing techniques use the Euclidean metric to measure the distance between data. Clearly though, this approach is flawed: biological populations do not conform to Euclidean geometry in their movement patterns and hence their observed positions will reflect this. Just as whales do not uniformly distribute themselves across sea and glacier,

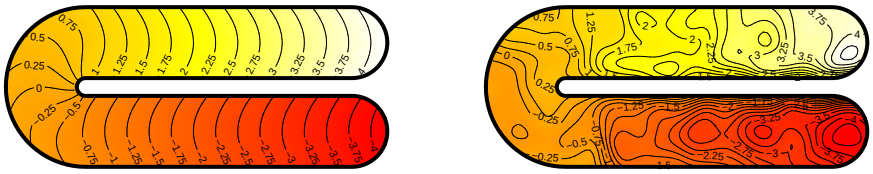

Figure 1-6: An example of leakage. A thin plate regression spline was fit to data sampled from the function on the left, the model smooths across the gap in the middle of the domain (right.)

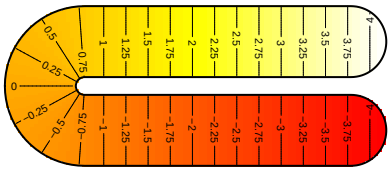

Figure 1-7: The horseshoe function as it appeared in Ramsay (2002).

fish do not lay their eggs on land. Natural and man-made barriers carve up the landscape (and seascape), partitioning biological populations; spatial models should take this into account.

The response may be smooth, just not necessarily over $\mathbb{R}^2$ (Wang and Ranalli, 2007). Modelling the structure of the domain correctly by embedding the extra information relating to the shape of the boundary (whether this be implicitly or explicitly) allows models to represent this smoothness.

1.4.2 Ramsay's horseshoe function as a benchmark for finite area smoothing

Ramsay (2002) proposes a function which can be used to benchmark new approaches to 2-dimensional smoothing. The function takes the form of a horseshoe shape which is flat across the domain has a gradient along the domain's major axis. This can be seen in figure 1-7. Wood et al. (2008) modifies the test function by adding curvature across the minor axis of the shape (left plot in figure 1-6). This was added in order to avoid the horseshoe function lying in the nullspace of their model's penalty, making the problem too easy for their method. It is the second shape that will be used for simulations here and shall be referred to as the *Ramsay horseshoe* throughout.

As mentioned above, when the smoothing problem is specified in terms of Euclidean distance, the model takes the distance between the points in the two arms of the horseshoe as the distance over the gap in-between them, rather than the distance along the major axis of the shape. This causes the high function values from one side to contaminate the other side (and the low to contaminate the high).

1.4.3 Previous approaches to leakage

The cause of leakage can be characterized in two ways: either the smooth does not respect the boundary of the domain, or the smooth does not take into account the geometry of the domain (in particular with regard to the distance between points within the domain). Previous work in this area has been to combat leakage along these two lines. Work of Ramsay (2002) and Wood et al. (2008) both use a partial differential equation (PDE) boundary condition approach to try to prevent leakage, whereas Wang and Ranalli (2007) and Eilers (2006) modify the way that inter-point distances are measured in order to avoid smoothing across boundaries. These four main works are now summarized.

FELSPLINE

Ramsay (2002) proposes finite element L-splines (FELSPLINEs). The L-spline penalty is similar to the one in (1.3):

$$\int_{\Gamma} (L_p f)^2 \mathrm{d}\mathbf{x}. \tag{1.12}$$

Integration is performed over Γ (the domain over which the smoothing is to take place) and L_p is a roughness operator defined as:

$$L_p = \Delta^p + c_{p-1}\Delta^{p-1} + \cdots + c_1\Delta + c_0 I. \tag{1.13}$$

Here I is the identity operator, the $(c_0, \ldots, c_p)$ are constants and Δ is the Laplacian (sum of second derivatives with respect to x_1 and x_2). This can be thought of as simply replacing the P operator in (1.3) and changing the integration domain.

In order to find f Ramsay takes a finite element approach. First triangulating the domain, then constructing a set of bivariate quadratic polynomial basis functions over each triangle, specifying that there be continuity over the edges of the triangles. By taking the FELSPLINE objective function and transforming

it into a variational form (as one would for a PDE), the approximation to the minimizer of the objective function is found.

Since the triangulation and hence the penalty of the FELSPLINE is only calculated over the domain, and the continuity is specified over neighbouring cells, the method prevents leakage. However, although FELSPLINE does not exhibit leakage on the original horseshoe (figure 1-7), in practice the model makes unrealistic physical assumptions. The boundary conditions of FELSPLINE specify that the gradient is zero, along normals to the boundary. This is not always physically realistic. Wood et al. (2008) show that by modifying the test function on horseshoe domain (see section 3.3.1), the FELSPLINE performance begins to falter.

FELSPLINE does not offer a realistic physical model and is therefore not a viable solution to the finite area smoothing problem in general.

Geodesic low-rank thin plate splines

Wang and Ranalli (2007) adopt a "within-area distance" formulation for thin plate splines. They choose to use the geodesic distance between two points, that being the shortest path within the domain. This gives a definition of how near objects are in the domain. The within-area distances are used in the radial basis functions in place of the Euclidean norm (section 1.1.3). The approach is referred to as *geodesic low-rank thin plate splines* (GLTPS).

To calculate the distances, Wang and Ranalli first create a complete, weighted, undirected, graph (G, say) with a data point at each vertex and the distance between each pair of vertices as the weights on the edges. They then find the restricted graph of G, G_k, in which each vertex is only connected to its k nearest neighbours. With this new, restricted graph the geodesic distances between each pair of vertices can be calculated using Floyd's algorithm (Floyd, 1962).

As the authors point out, the quality of the approximation is dependent on the size of the data set and its density. At low densities the estimated geodesic distance will tend towards the Euclidean, at high densities the approximation tends, asymptotically toward the true geodesic distance (Bernstein et al., 2000). Even if dense enough data were available, the method will be rather slow since Floyd's algorithm is cubic in the number of vertices (the size of the data set).

Taking these points into account, GLTPS appears cumbersome, slow and dependent on dense data.

Soap film smoother

The soap film smoother (Wood et al., 2008) uses a rather simple physical model to prevent leakage from occurring. First, consider the domain boundary to be made of wire, then dip this wire into a bucket of soapy water, you will then have a soap film in the shape of your boundary. Consider the wire to lie in the $x_1 - x_2$ plane and the "height" of the soap film at a given point to be the functional value of the model (i.e. in the z direction). This film is then distorted smoothly by moving it toward the data, while minimising the surface tension in the film. The film is bounded by the wire, so smoothing across any peninsulae (and thus leakage) is not possible by construction. It has been assumed so far that the wire has the correct values for the boundary; fortunately it is possible to estimate the wire's height simultaneously with the rest of the film so the whole smooth is totally data-dependent.

Fitting a model using the soap film smoother consists of first finding a set of basis functions which are solutions to a series of partial differential equations (PDEs) which define the film. Since these functions arise from PDE with boundary conditions, the resulting functions respect those boundary conditions by default. In a similar way to the thin plate spline, the soap film basis functions can be separated into two parts: one which deals with the interior of the boundary (equivalent to the radial basis functions) and the other, which performs a similar function to the linearly independent polynomials in the thin plate spline: controlling what the smooth looks like when the function is completely smooth (i.e. when the penalty evaluates to zero). In two dimensions, the thin plate spline will reduce to a plane when the penalty is zero (since a plane is completely smooth according to the penalty) however, for the soap film the surface should respect the boundary conditions.

Although mathematically elegant, the soap film smoother is a rather complex and computationally expensive model (since the series of PDEs must be solved to create the basis before the model can be fitted). One must also pick knots for the soap film smoother to use, introducing an element of arbitrariness into the fitting process. The model also treats the boundary as something special, it is not clear that this is always appropriate (in particular thinking of ocean-based studies where some of the boundaries are coastlines but others are essentially arbitrary).

Although not perfect, the soap film smoother does not have any obvious major technical flaw, unlike the unrealistic physical assumptions that FELSPLINE

makes. It is also already implemented in the R package `soap` unlike GLTPS, which does not have an easily available software package. For these reasons it will be used throughout the thesis as the "gold standard" against which the methods proposed will be measured.

Since the soap film smoother will play a key part in the thesis as a benchmark for other methods, chapter 2 is a dedicated introduction to the method. The chapter is set in the context of a case study, investigating the spatiotemporal distribution of resident foreigners in Italy. Section 2.2 illustrates the basis construction as well as how to incorporate the soap film smoother in a larger spatiotemporal model using a tensor product (section 1.1.3).

Domain morphing

An alternative approach to treating the boundary as something special is to transform the space in which the points lie to a different domain which is more suitable for smoothing. For example, with Ramsay's horseshoe, it seems intuitive to simply bend the horseshoe into a long strip and then smooth on that domain.

Indeed, Eilers (2006) proposed using the *Schwarz-Christoffel transform* for this very purpose (the author independently came to the idea in 2008). The basic idea is to find a function that takes points in the domain the data lie in and maps them to another domain in which smoothing is easier. Using the Schwarz-Christoffel transform for smoothing will be investigated in chapter 3.

Outside of the smoothing spline and GAM literature, transformation-based methods have also been suggested, in particular, when using a kriging approach (see Venables and Ripley (2002, pp. 425-430) for a concise introduction, Shabenberger and Gotway (2005) or Diggle and Ribeiro (2007) for a thorough treatment). Kriging consists of modelling the spatial correlation between points via the *semivariogram* and a spatial trend via a mean function (similar to the linear predictor in the smoothing case, although there are flavours of kriging where the mean is considered constant and/or known). Semivariogram models assume that the correlation between points is related to the distance between the points but not their position (this is known as *stationarity*, and comes in varying degrees, Shabenberger and Gotway (2005, pp. 42-44)). When the boundary of the domain has a complex shape, the correlation between points is likely to vary with distance within the domain rather than the Euclidean distance between points. Simply substituting within-area distances into the semivariogram will lead to an invalid semivariogram (section 6.1; Curriero, 2006)

Several authors have suggested the use of some kind of transformation of the data points in space in order to maintain stationarity by approximating within-area distances with equivalent Euclidean distances via *multidimensional scaling* (Løland and Høst, 2003; Jensen et al., 2006; Curriero, 2006). The use of multidimensional scaling to project these distances ensures that the semivariogram remains positive or conditionally negative definite (which is required to have a valid semivariogram, Curriero (2006)).

Using multidimensional scaling as a transformation of a spatial domain is investigated in chapters 4 and 5. A comparison between the geostatistical implementations and the methods developed in this thesis is given in section 6.1, once the proposed methods have been fully explained.

Creating some kind of mapping between the space in which the data lies and the space in which conventional smoothers perform well is convenient. Relying on existing, tested methodology is clearly appealing. Transformation-based approach also benefit from not treating the boundary as a special in the basis setup. The properties that such a mapping would require to be useful for smoothing will be investigated in subsequent chapters.

Chapters 3, 4 and 5 investigate the combination a transformation of space and conventional smoothers to solve the problem of leakage in finite area smoothing. The next chapter attempts to solidify the concepts presented so far (smoothing, penalties, leakage, tensor products) as well as highlight some of the potential pitfalls when modelling complex data. The chapter applies a spatiotemporal model of legal immigrants in Italy using a tensor product of a soap film smoother basis (for space) and a cubic spline basis (for time).

Chapter 2

Modelling the spatiotemporal distribution of the incidence of resident foreign population in Italy

Augustin et al. (2009) demonstrated the utility of a tensor product of a thin plate regression spline (for space) and a cubic spline (for time) in spatiotemporal smoothing. Using the soap film smoother described in chapter 1 (in place of the thin plate regression spline), a spatiotemporal smooth is used to model the distribution of the incidence of the resident foreign population in Italy. This chapter presents the first application of this approach to spatiotemporal smoothing in a complicated geographic region.

The work presented here is adapted from the article "Modelling the spatiotemporal distribution of the incidence of resident foreign population in Italy" by G. Marra (Statistical Science, University College London), D. L. Miller and L. Zanin (Prometeia, Bologna), accepted for publication in *Statistica Neerlandica*. Each of the authors contributed equally to the article.

2.1 Introduction

2.1.1 Background

Many European countries have recently experienced a substantial increase in the proportion of immigrants in the population (Manning, 2010). In particular

this has increased since the introduction of new EU members in 2004 and 2007 (Kahanec and Zimmermann, 2009).

Immigration statistics calculated at a national level do not provide information on the local spatial and temporal distribution of the phenomenon. This information may be of crucial importance for planning local policies. Using Italian data (at a municipal level) for the period 2003-2008, a tensor product smoother combining a cubic regression spline basis for time and a soap film spline basis for space can be used to create spatiotemporal maps. These maps could then be effectively used by policy makers to decide the allocation of economic resources at both a local and national level.

A *municipality* (in Italian *comune*) is the lowest level administrative subdivision in Italy. Municipalities then make up provinces (of which there are 110) which in turn make up regions (of which there are 20).

2.1.2 Data

The *resident foreign population* includes all people (born in Italy or abroad) who declare their citizenship not to be Italian. The change in resident foreign population at a municipal level, at the end of a year, is determined by

$$RFP^{31^{st}} = RFP^{1^{st}} + (I_1 + I_2 + I_3 + I_4) - (D_1 + D_2 + D_3 + D_4 + AC), \quad (2.1)$$

where $RFP^{31^{st}}$ indicates the total number of resident foreigners at the end of December and $RFP^{1^{st}}$ the number of resident foreigners on the 1^{st} of January. I_1 denotes the number of people whose parents are foreigners (at least one of them being resident in the municipality), I_2 the number of foreign citizens who asked to transfer their residence from another Italian municipality to the current one, I_3 those who asked to transfer their residence from abroad, and I_4 refers to recording operations due to other reasons (e.g. foreigners mistakenly deleted from the registry of the municipality, because they were temporarily missing). D_1 represents the number of resident foreigners who died during the year, D_2 those who moved to a different municipality, D_3 those who moved abroad, D_4 refers to cancellations for other reasons (e.g. foreigners deleted from the registry of the municipality, because they were not present), and AC denotes those resident foreigners who obtained Italian citizenship during the year.

Given a specific area such as municipality (although this is often calculated on provincial, regional or national level too), a measure of density of resident

foreign population is the percentage *incidence of resident foreigners* (IRF), given as the ratio of the number of resident foreigners (the RFP) to the total resident population multiplied by 100 (e.g. Lowell, 2007). This gives a simple demographic indicator for comparing different areas of a country in terms of number of resident foreigners per 100 resident inhabitants. According to ISTAT (the Italian government's statistical office), the IRF as calculated on a national level has grown substantially, from 2.7% in 2002 to 6.5% in 2008.

Recently ISTAT has integrated the official statistics with a new public database (available via `http://demo.istat.it`), mainly based on administrative sources. The database gives the number of resident foreigners (calculated as above) in each of almost 8100 municipalities. Note that this does not include foreigners entering the country illegally, this issue is not addressed here. The quantification of illegal immigrants is an open topic of discussion in the studies of international migrations (e.g. Strozza, 2004). Several approaches have been proposed in literature to quantify this, but all are subject to criticism (especially with regard to the magnitude of errors in estimates). Here only official data on the resident foreign population are considered. The ISTAT data, at a municipal level, (the highest resolution available at this time) are an excellent resource for creating maps.

Figure 2-1 shows plots of the raw IRF data at a municipal level over the years 2003-2008 (these are, of course, averaged over a grid for plotting purposes). The features that stand out clearly are the marked difference between north and south, and the increase in the IRF over time. However, these maps make it difficult to detect any local spatial structure in the incidence. For example, the Po Valley (northern Italy) has relatively high levels of IRF in 2008 but it is not really possible to clearly identify any particular areas of polarization. This can be problematic for policy makers in the central public administration who wish to allocate economic resources as efficiently and effectively as possible. Previous studies analysing the distribution of resident foreign population have employed descriptive analysis and/or to what amounts to complex linear modelling approaches (e.g. Fonseca, 2008, Longhi et al., 2010). Using the soap film smoother, high resolution spatiotemporal maps of the IRF can be created.

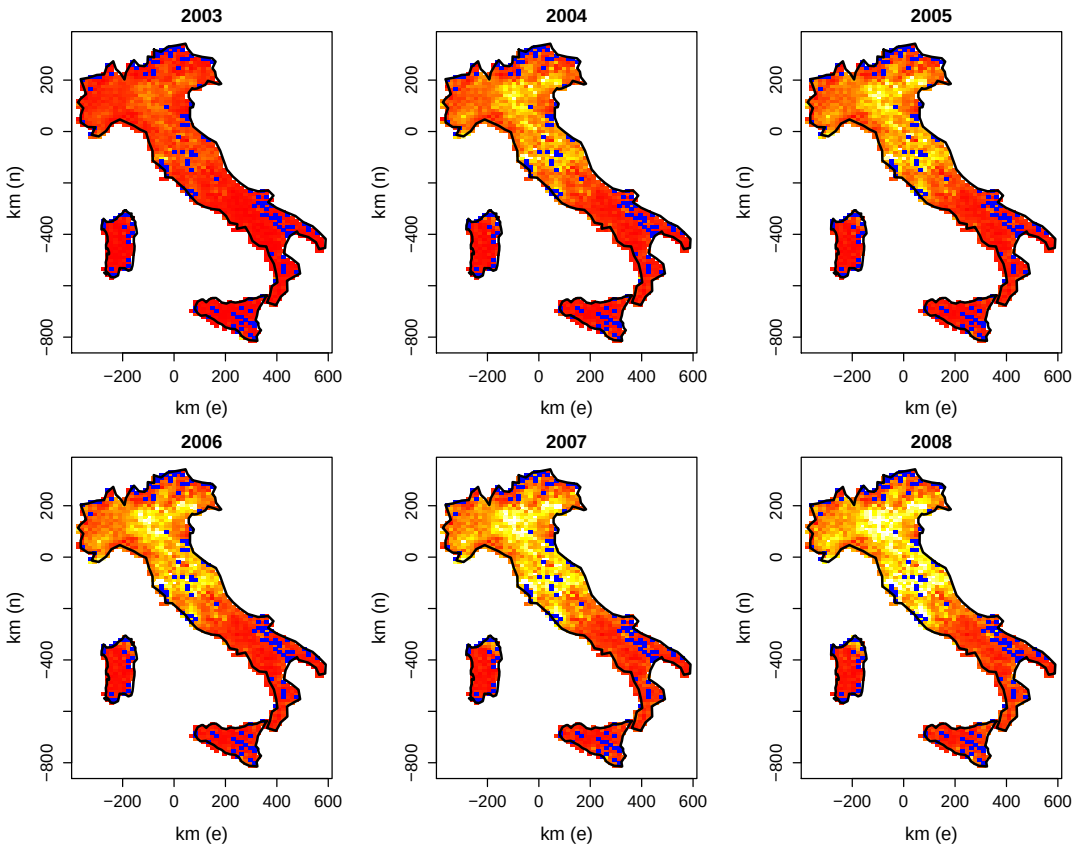

Figure 2-1: Empirical maps of the percentage incidence of resident foreigners in Italy over the years 2003-2008. These were obtained using ISTAT data at a municipal level. The incidence is given as the ratio of the number of resident foreigners to the total resident population multiplied by 100. The colour scale ranges from an incidence of 0 (dark red) to an incidence of 12 (white). Cells where there was no data are coloured blue (see figure 2-2 for comparison).

2.2 The model

There are currently no relevant economic covariates available at a municipal level, the response (IRF) is modelled using a smoother which is a function of only the spatial coordinates and time. The model is then used to create smoothed maps of the geographical area of interest over time. Two points are noteworthy here. First, since the coastline of Italy has a complex boundary, leakage (as described in section 1.4) is likely to occur; inappropriately linking parts of the domain could have serious policy implications. Second, looking at figure 2-1, there appears to be a strong temporal interaction (those places with high incidence increase their incidence over time) so the model used should account for a space-time interaction. These two goals can be achieved by using a generalized additive model incorporating a three-dimensional tensor product smoother combining a cubic regression spline basis (section 1.1.3) for the temporal trend and a soap film smoother (Wood et al., 2008 and section 1.4) for the spatial component.

2.2.1 Model specification

As explained in the introductory section, the IRF is given as the ratio of the number of resident foreigners to the total resident population multiplied by 100. The IRF may therefore only take positive values (but can be zero). Letting f be a smooth function, the proposed model is as follows

$$\log\left\{\mathbb{E}(\texttt{irf}_{it})\right\} = f(\texttt{year}_t, \texttt{n}_i, \texttt{e}_i),\ \texttt{irf}_{it} \sim \text{Tweedie}\left\{\mathbb{E}(\texttt{irf}_{it}), \phi\mathbb{E}(\texttt{irf}_{it})^p\right\}, \quad (2.2)$$

at municipality $i = 1, \ldots, 8094$ and year $t = 2003, \ldots, 2008$. ϕ is a dispersion parameter and the log link function ensures positive fitted values. $\texttt{irf}_{it}$, $\texttt{n}_i$, and $\texttt{e}_i$ represent the variables percentage IRF, Northing and Easting, respectively. f is a multidimensional smooth function of `year`, `n` and `e` which models the joint effect of these variables on `irf`. Northing and Easting are as described in section 1.1.1 (in this case the offsets used were 11.5 longitude, 44 latitude), thus ensuring that the spatial part of the smoother is isotropic (which would not be the case if latitude and longitude were used, since lines of latitude become closer at the poles). A Tweedie distribution was used to model the response. Prior to using the Tweedie the Gamma distribution was used, however when analysing diagnostic plots substantial structure was still present in the redisiduals. The Tweedie distribution also allows exact zeros in the data, which the Gamma

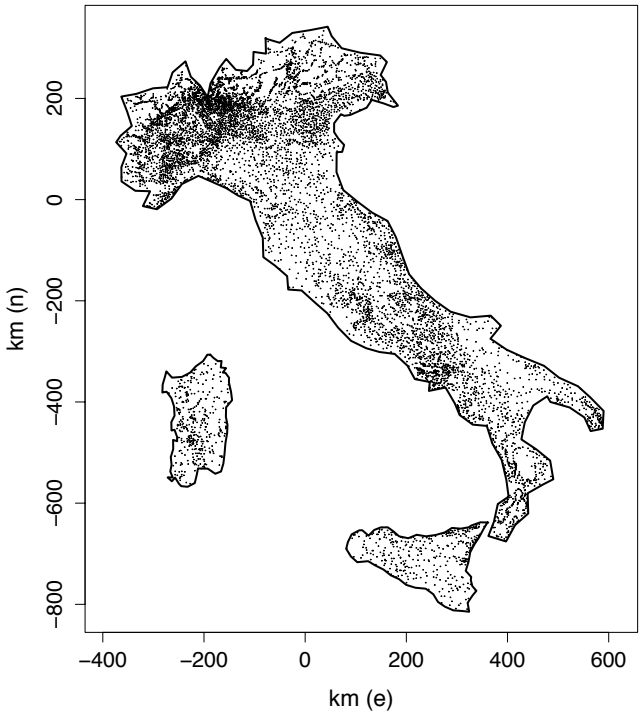

Figure 2-2: Raw data locations for the incidence of resident foreigners with the boundaries of Italy, Sardinia and Sicily. Each point is the location of the centroid of a municipality.

distribution does not (this avoided adding 1×10^{-6} to the response, which was necessary in the Gamma case). Tweedie distributions are a special case of an exponential dispersion model and include, for example, the normal ($p = 0$), Poisson ($p = 1$) and gamma ($p = 2$) distributions (Jørgensen, 1987). For $1 < p < 2$ Tweedie distributions can be represented as Poisson mixtures of gamma distributions, with mass at zero but otherwise continuous on the positive reals. The Tweedie distribution is implemented in the R package `mgcv`, making it an easy to use alternative. Dunn and Smyth (2005) provides a survey of published applications stressing the utility and flexibility of this class of distributions.

Note that the data were collected per municipality but enter into the model as points which are located at the centroids of the municipalities (see figure 2-2). This change of support from areal to point data is not problematic and is desirable. The aim is to produce smoothed maps; using areal data would yield maps that are step functions for each municipality hence making it more difficult to see patterns in the data. To make the maps as easily interpretable as possible, smoothing of points is a clear choice.

The technical details of the construction of the smoothers is now covered. First, building on the general formulation given in section 1.1.3, the construction of a three-dimensional tensor product smoother is shown, using a one-dimensional temporal smoother and a two-dimensional spatial smoother. The details of the construction of the two-dimensional smoother, using the soap film smoother are then given. The final section details the construction of temporal trend estimates.

2.2.2 A three-dimensional tensor product smoother for time and space

The tensor product smooth used here is similar to the one shown in section 1.1.3, however it differs in two ways. First, one of the components is a two-dimensional, isotopic spatial smooth (f_{year}) and the other component (f_{space}) is a marginal one-dimensional smooth of time. Second, the size of each basis is different.

Omitting the subscripts i and t for simplicity, the temporal smooth f_{year} and spatial smooth f_{space} can be written in terms of their basis decompositions (as in (1.1)):

$$f_{\text{year}}(\texttt{year}) = \sum_{j=1}^{J} \xi_j b_j(\texttt{year}) \quad \text{and} \quad f_{\text{space}}(\texttt{n}, \texttt{e}) = \sum_{r=1}^{R} \varphi_r d_r(\texttt{n}, \texttt{e}),$$

where the $b_j(\texttt{year})$ and $d_r(\texttt{n}, \texttt{e})$ are known cubic regression spline and soap film basis functions (respectively), with corresponding parameters ξ_j and φ_r and spline dimensions J and R.

In order to set up a three-dimensional tensor product smoother for time and space it is necessary for $f_{\text{year}}(\texttt{year})$ to vary smoothly with the spatial dimensions. This can be achieved by allowing the parameters ξ_j to vary smoothly with `n` and `e`. Using the spline set-up for $f_{\text{space}}(\texttt{n}, \texttt{e})$ we may write (analogously to section 1.1.3):

$$\xi_j(\texttt{n}, \texttt{e}) = \sum_{r=1}^{R} \varphi_{jr} d_r(\texttt{n}, \texttt{e}),$$

which results in:

$$f(\texttt{year}, \texttt{n}, \texttt{e}) = \sum_{j=1}^{J} \sum_{r=1}^{R} \varphi_{jr} d_r(\texttt{n}, \texttt{e}) b_j(\texttt{year}).$$

To calculate the penalty, first let J_{year} and J_{space} be measures of the wigglyness

of the functions f_{year} and f_{space} respectively. For f_{year}, the second-order cubic spline penalty evaluates $J_{\text{year}}(f_{\text{year}}) = \int \left(\partial^2 f_{\text{year}}/\partial \texttt{year}^2\right)^2 \texttt{dyear}$ (section 1.1.3). The penalty for the soap film smoother used for the spatial part of the tensor product is covered in the next section; for ease of explanation it is treated as a black box here.

An overall penalty for the tensor product smoother can be obtained by applying the penalties of $f_{\text{space}}(\texttt{n}, \texttt{e})$ to the varying coefficients of the marginal smooth $f_{\text{year}}(\texttt{year})$, $\xi_j(\texttt{n}, \texttt{e})$,

$$\sum_{j=1}^{J} J_{\text{space}} \left\{\xi_j(\texttt{n}, \texttt{e})\right\},$$

and the penalties of $f_{\text{year}}(\texttt{year})$ to the varying coefficients of the marginal smooth $f_{\text{space}}(\texttt{n}, \texttt{e})$, $\varphi_r(\texttt{year})$,

$$\sum_{r=1}^{R} J_{\text{year}} \left\{\varphi_r(\texttt{year})\right\}.$$

It follows that the penalty of $f(\texttt{year}, \texttt{n}, \texttt{e})$ can be written:

$$\lambda_{\text{space}} \sum_{j=1}^{J} J_{\text{space}} \left\{\xi_j(\texttt{n}, \texttt{e})\right\} + \lambda_{\text{year}} \sum_{r=1}^{R} J_{\text{year}} \left\{\varphi_r(\texttt{year})\right\}, \tag{2.3}$$

where as usual, λ_{space} and λ_{year} are the smoothing parameters for time and space respectively.

Section 1.1.3 gives the details of the cubic spline basis used to model the temporal part of the smooth. The next section shows how f_{space} and J_{space} are constructed.

2.2.3 The soap film smoother

Since there is no particular reason to believe that the resident foreign population should be continuous across physical boundaries such as the Mediterranean Sea (at best there are merely potential resident foreigners in the sea), a smoother which takes into account of the fact that the borders of Italy represent both physical barriers should be used. Although there is no reason *a priori* to believe that there will be particularly troublesome leakage (see section 1.4) here, mitigating against the issue by use of appropriate basis function choice from the outset is the most sensible course of action.

As mentioned in section 1.4.3, the soap film smoother (Wood et al., 2008) uses

a rather simple physical model to prevent leakage from occurring. First, consider the domain boundary to be made of wire, then dip this wire into a bucket of soapy water; a soap film with the same shape as the boundary will have then formed. Now consider the wire to lie in the `n-e` plane and the height of the soap film at a given point to be the functional value of the model. This film is then distorted smoothly by moving it vertically toward each datum locally, while minimizing the surface tension in the film as a whole. Mathematically, the domain (Γ) is bounded by some polygon (B). For the islands of Sicily and Sardinia is their coastlines and for for the mainland is its coastline along with the border with France, Switzerland, Austria and Slovenia to the north. Note that a separate model was used for each of the mainland and Sardinia and Sicily (see section 2.3 for why this was necessary).

The soap film smoother basis is quite similar in form to the thin plate regression spline basis given in section 1.1.3. The basis is split into two sets of functions to form a smoother that respect the necessary boundary conditions. The first basis is used for the smoothing within the region of interest, Γ; the second is to deal with the boundary, B. These two sets of basis functions are then summed to form:

$$f_{\text{space}}(\mathtt{n},\mathtt{e}) = \sum_{j=1}^{J} \alpha_j a_j(\mathtt{n},\mathtt{e}) + \sum_{k=1}^{K} \gamma_k g_k(\mathtt{n},\mathtt{e}),$$

where the γ_k and α_j are the parameters to be estimated. One can think of the $a_j(\mathtt{n},\mathtt{e})$ as an offset dictated by the estimated boundary conditions on B (similar to the linearly independent polynomials in the thin plate spline, although it is important to note that these functions are not planes) and the sum of the $g_k(\mathtt{n},\mathtt{e})$ as the smooth function to the data inside Γ (analogous to the radial basis functions in the thin plate spline). Unlike thin plate regression splines however, the soap film basis requires the specification of K knots inside Γ and J boundary knots on B (here a grid is used for the internal knots and the boundary knots are equally spaced along B, further detail on the setup used is given in section 2.3 and table 2.1). For convenience later, the second sum is labeled as f_{int} (the part of f with knots inside Γ). The rest of this section shows how these bases are constructed.

For the internal part of the smoother we first find a set of functions $\rho_k(\mathtt{n},\mathtt{e})$. These are each solutions to the Laplace's equation in two dimensions

$$\frac{\partial^2 \rho}{\partial \mathtt{n}^2} + \frac{\partial^2 \rho}{\partial \mathtt{e}^2} = 0,$$

except at one of the knots $(\mathtt{n}_k^*, \mathtt{e}_k^*)$. Then, solving Poisson's equation in 2-dimensions

$$\frac{\partial^2 g_k}{\partial \mathtt{n}^2} + \frac{\partial^2 g_k}{\partial \mathtt{e}^2} = \rho_k(\mathtt{n}, \mathtt{e}), \tag{2.4}$$

for k indexing the K knots. When the boundary condition $\rho_k(\mathtt{n}, \mathtt{e}) = 0$ is applied, the set of basis functions for the soap film smoother $g_k(\mathtt{n}, \mathtt{e})$ are found. The partial differential equations (PDEs) are solved numerically using a multi-grid solver (e.g. Press et al., 1992, pp. 862-880), see the appendix of Wood et al., 2008 for further technical details.

To find the height of the function at each point around the boundary a cyclic spline basis is used to construct $f_{\text{bnd}}(r)$. This $f_{\text{bnd}}(r)$ will have the expansion

$$f_{\text{bnd}}(r) = \sum_{j=1}^{J} \alpha_j \delta_j(r), \tag{2.5}$$

where r is the distance along the boundary, the α_j are parameters and $\delta_j(r)$ are cubic spline basis functions. The basis functions have the same form as the cubic spline discussed in section 1.1.3, but to ensure that the spline is cyclic the value of the function at the first knot is constrained to be the same as that at the last knot up to their second derivatives. Note that the values of the α_j are not of interest at this stage, only the basis expansion. The basis functions $a_j(\mathtt{n}, \mathtt{e})$ can be found by solving (2.4) for $\rho_k(\mathtt{n}, \mathtt{e}) = 0$ with the boundary condition resulting from setting $\alpha_j = 1$ (and all other α_i to zero) in (2.5), using the same methods as for the $g_k(\mathtt{n}, \mathtt{e})$ above. Each $a_j(\mathtt{n}, \mathtt{e})$ can then be thought of as the function with a peak at the corresponding knot on B, which are smooth across the whole of Γ.

The set of basis functions of f_{int} have been found, as well as the boundary-induced-smooth which acts as a base for the soap film smoother. Although this seems like a rather esoteric setup, all the procedure above is effectively doing is setting up a basis such that standard penalized regression techniques can be used. Just as one might choose one spline basis over another for some property it possesses, the soap film basis has the property that it obeys the (estimated) boundary conditions of the region that are to be smoothed over.

In section 2.2.2 the spatial penalty term was simply referred to as a single quantity, however (as was the case with the basis) the penalty is split into two parts: one for the cyclic smoother around the boundary and the other for the internal smoother. Hence, the spatial part of the penalty given in (2.3) is expressed

as:

$$\lambda_{\text{space}} \sum_{j=1}^{J} J_{\text{space}} \left\{ \xi_j(\mathtt{n}, \mathtt{e}) \right\} = \lambda_{\text{int}} J_{\text{int}} + \lambda_{\text{bnd}} J_{\text{bnd}}.$$

Rather than estimating λ_{space}, the two smoothing parameters λ_{int} and λ_{bnd} are estimated and these control the smoothness of the spatial part of the model. The two interior and boundary penalties are now considered individually.

The isotropic interior penalty term is calculated as

$$J_{\text{int}} = \int_{\Gamma} \Big(\frac{\partial^2 f_{\text{int}}}{\partial \mathtt{n}^2} + \frac{\partial^2 f_{\text{int}}}{\partial \mathtt{e}^2} \Big)^2 \mathrm{d}\mathtt{n}\mathrm{d}\mathtt{e}.$$

Note that the integration occurs only over Γ rather than over the whole of $\mathbb{R}^2$ as would usually be the case. Also, there is no mixed derivative term, and the whole integrand is squared rather than each term individually (in contrast with, say, a thin plate regression spline penalty given in (1.5)). This allows the `n` and `e` terms' derivatives to be traded off against each other so that the nullspace of the penalty is infinite dimensional. This permits those functions in the nullspace to be sufficiently wiggly to meet any boundary conditions. Note the similarity to the FELSPLINE penalty in section 1.4.3.

The penalty for the cyclic spline running about the boundary, used to estimate the α_j, is calculated as

$$J_{\text{bnd}} = \int_{B} \Big(\frac{\partial^2 f_{\text{bnd}}}{\partial r^2} \Big)^2 \mathrm{d}r.$$

where the basis functions are the cubic regression splines defined above, but with the additional condition that the function's values and their first and second derivatives at the first and last knot must be equal. See e.g. Wood, 2006, p. 149 for more details.

The smooth function for $f_{\mathtt{space}}$ obtained using the construction outlined above is rotationally invariant. Two smoothing parameters are estimated for the soap film (one for the interior and one for the boundary) but each controls the smoothness for both geographical directions at once (Wood et al., 2008). However, the tensor product smoother using the cubic regression spline basis for time and a soap film for space is not isotropic in space-time. This anisotropy is desirable since, as explained in Section 2.2.1, the measurements of space and time are on different scales so it would not make sense to estimate one smoothing parameter for both the spatial and temporal components (e.g. Wood, 2006, p. 162).

In all, two smoothing parameters are estimated for the soap film (λ_{int} and

λ_{bnd}), and one for the temporal component ($\lambda_{\texttt{year}}$). Models were run using both REML and GCV for smoothness selection (with no major differences between the two methods). The results presented here are those for REML as described in section 1.2.3.

To summarize, the basis functions $a_j(\texttt{n}_i, \texttt{e}_i)$ and $g_j(\texttt{n}_i, \texttt{e}_i)$, and the penalties J_{bnd} and J_{int} have been found. Soap basis functions and penalties can be obtained using the the R package `soap` which implements the ideas discussed in this section. The `soap` package along with `mgcv` were used to construct the model.

2.2.4 Variance and trend estimation

Taking a Bayesian view it is possible to construct intervals using the posterior distribution of the model parameters. The interesting feature of these intervals is that, since they include both a bias and a variance component, they have good observed *frequentist* coverage probabilities across the function (Marra and Wood, 2011). Given a large sample size (and assuming asymptotic normality of the MLE; Wood (2006, pp. 192–194)), the posterior distribution is given as

$$\boldsymbol{\beta}|\texttt{irf} \dot{\backsim} N(\hat{\boldsymbol{\beta}}, \mathbf{V}_{\boldsymbol{\beta}}), \tag{2.6}$$

where $\hat{\boldsymbol{\beta}}$ is the maximum penalized likelihood estimate of (all of) the smoother's parameters, $\boldsymbol{\beta}$ (i.e. concatenating all of the parameters discussed above into one vector). $\mathbf{V}_{\boldsymbol{\beta}}$ is of the form $(\mathbf{X}^{\mathrm{T}}\mathbf{W}\mathbf{X} + \mathbf{S}^{*})^{-1}\phi$, $\mathbf{X}$ contains the columns associated with the regression spline bases used to set up the model, $\mathbf{W}$ and $\mathbf{z}$ are the weight matrix and the pseudodata vector at convergence of the algorithm used to fit the penalized model and ϕ is the scale parameter (as in section 1.2.3). Given result (2.6), confidence intervals for linear functions of the parameters can be found easily. Intervals for non-linear functions of the model coefficients can be conveniently obtained by simulation from the posterior distribution of $\boldsymbol{\beta}$. To aid interpretation of the results, trend estimates for some given areas of interest can be produced using the predictive distribution of $\texttt{irf}_{it}$, using the method proposed in Augustin et al. (2009). This approach can be implemented as follows:

1. Repeat the following steps for $b = 1, \ldots, N_b$, where N_b represents the number of random draws.

   (a) Simulate a random $N(\hat{\boldsymbol{\beta}}, \hat{\mathbf{V}}_{\boldsymbol{\beta}})$ and call the resulting coefficient vector $\boldsymbol{\beta}_b$.

Region	Interior knots	Cyclic spline basis size	Cubic spline basis size
Mainland	41 (14×14)	20	6
Sardinia	12 (5×6)	8	6
Sicily	12 (6×6)	10	6

Table 2.1: Basis sizes per region for the smooth functions to be fitted to the Italian data. For the interior (soap film) knots, the numbers in brackets show the initial grid, the other number gives the number of knots actually used (those inside the boundary).

(b) Calculate $\widehat{\mathbb{E}(\texttt{irf}_{it})} = \widehat{\texttt{irf}}_{itb} = \exp(\mathbf{X}^*_{it}\boldsymbol{\beta}_b)$, where $\mathbf{X}^*_{it}$ is evaluated at the observed values.

(c) For a given area of interest a and year t, calculate

$$\widehat{\overline{\texttt{irf}}}^a_{tb} = \frac{1}{n_a}\sum_{i=1}^{n_a}\widehat{\texttt{irf}}_{itb},$$

where n_a represents the number of observations in area a.

2. Produce the required summary statistics, in this case median, lower and upper 95% quantiles, for the temporal trend $\widehat{\overline{\texttt{irf}}}^a_{tb}$.

Small values for N_b are typically tolerable. In practice, N_b can be set to 100. Increasing this value does not change the results which are presented in the next section.

2.3 Results

The model was implemented using the basis sizes (for the cubic and cyclic regression splines) and interior knots (for the soap film) given in table 2.1. Note that separate models were used for each of mainland, Sicily and Sardinia since scenarios with multiple islands are not currently supported in the `soap` package.

The parameter for the Tweedie distribution (p in (2.2)) was set to 1.2, this was chosen by trial and error to produce the best possible residual plots. Candy (2004) show that the p parameter maybe optimized in a GLM however this is rather more complex in the GAM case. It would be possible to perform a simple grid search over values for p, however it is also the case that the value of p is only important to the first decimal place, so a manual search was not overly time consuming

(Williams et al., 2011). The estimate for ϕ was equal to 0.905. Residual analysis was carried out following an approach similar to that of Chandler (2005). Figure 2-3 gives some diagnostics for the proposed model. Overall, the first two sets of boxplots do not show long-term trends, but suggest the presence of some spatial residual structure. The normal Q-Q plot shows some curvature in the upper tail. The scale-location plot indicates the presence of some heteroskedasticity, although the LOESS smooth running though the plot (grey line) does not indicate a relationship between absolute residuals and predicted values; the highlighted grey points correspond to the exact zeros in the data. The presence of extreme values in all diagnostic plots suggest under-estimation of the IRF on occasion. Descriptive analysis revealed that some municipalities have considerably higher IRF levels compared to those in neighbouring municipalities. This is mainly due to the specific socio-economic features of attractiveness of each municipality which need to be accounted for in order to avoid under-estimation. Unfortunately, as pointed out in the introduction, currently no relevant economic data are available at a municipal level and hence such information can not be incorporated in the model. The deviance explained was 57%.

As a check, the proposed model was fitted again but replacing the response variable with the residuals. The resulting maps suggested the presence of some residual structure in those municipalities characterized by very high IRF levels; this was expected given the presence of extreme residuals evidenced in the diagnostic plots. Sensitivity analysis showed that the results do not change if more basis functions are used for the smooth term in the model.

The proposed model was compared with one in which the spatial and temporal effects were simply additive. As in Augustin et al. (2009), Schwarz's Bayesian Information Criterion (BIC, see e.g. Burnham and Anderson, 2002, p. 286) was used to compare the two models; the latter yielded an increased BIC. Visually comparing the maps produced by these two models, the proposed model contained details that could not be seen in the model where the spatial and temporal effects were additive.

Despite the lack of fit corresponding to those municipalities characterized by very high levels of the IRF, the analysis above suggests that the proposed model is able to capture the overall spatiotemporal structure present in the data. Should some relevant economic variables become available at a municipal level, the model specification and, as a consequence, its explanatory power could be considerably improved.

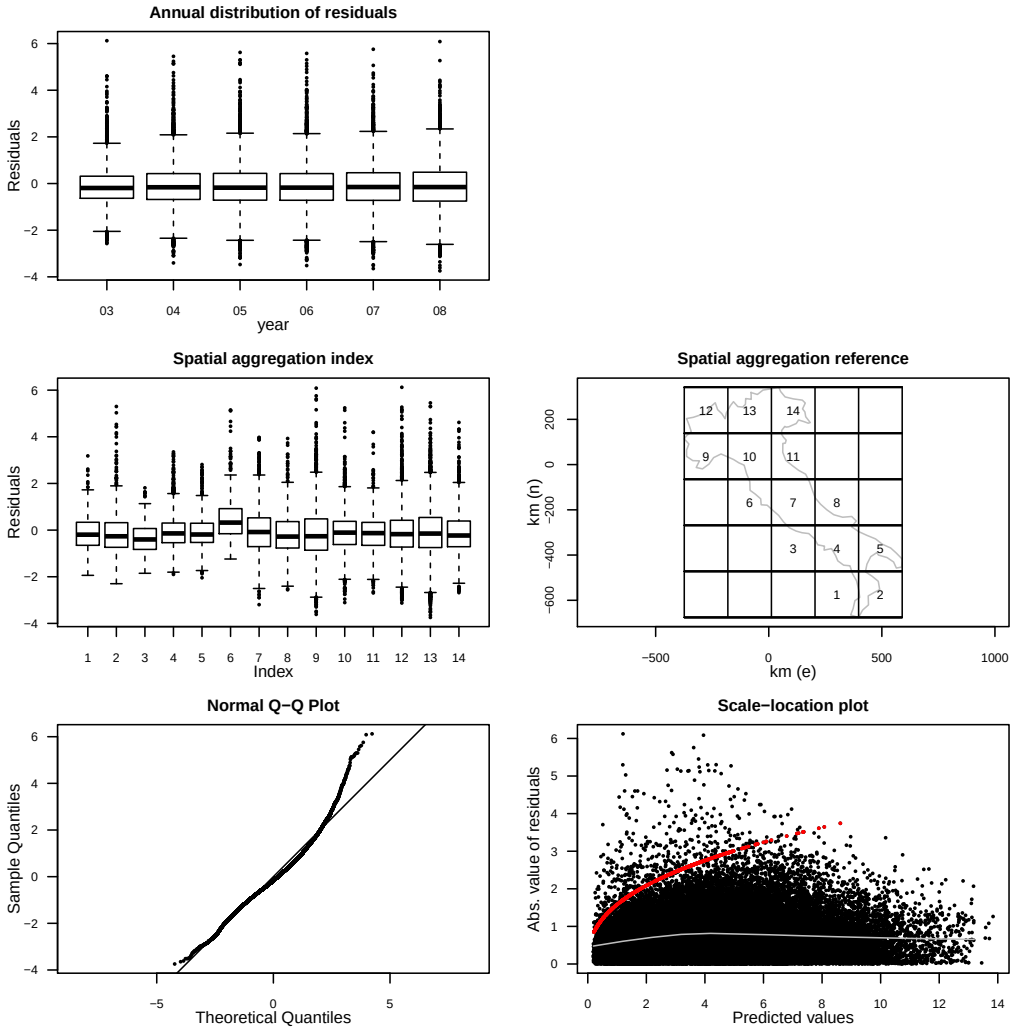

Figure 2-3: Deviance residual diagnostics from the spatiotemporal model of the incidence of resident foreigners. Row-wise, left to right, from top: (i) yearly distributions, (ii) distributions of residuals grouped according to squares on a 5x5 grid which fell inside the boundary, (iii) the layout of this grid, (iv) normal Q-Q plot, (v) absolute residuals versus predicted values, with LOESS curve (grey line), the line of grey points correspond to exact zeros in the data.

2.3.1 Spatiotemporal maps

As mentioned in the introductory section, Italy's national IRF was 6.5% in 2008. Obviously, this number says nothing about specific local areas with particularly high or low levels of resident foreigners. Moving to smaller aggregations (say, the regional or provincial level) the same argument holds: some information is lost. This kind of low resolution statistic can mask smaller-scale heterogeneity in the population. Our aim here is to capture exactly this spatial heterogeneity.

At first glance, figure 2-1 shows the stark contrast between north and south. The maps also highlight the inhomogeneous spatial and temporal distribution of the IRF. However, as pointed out above, smoothed maps can give a much better picture of the distribution of the IRF. Smoothing the maps allows for the separation of signal and noise in the data, giving an idea of the overall structure of the phenomena, as well as providing a trend information via a temporal component of the model. Figure 2-4 shows smoothed spatiotemporal maps of the IRF in Italy for the period 2003-2008, obtained using the approach described in the previous sections. The maps show that in 2003 northern and central Italy have the highest levels IRF. Over the subsequent years (2004-2008), the incidence spreads, forming four main areas where resident foreigners tend to live. Focusing on 2008, the most popular areas are in north Italy, specifically the Emilia-Romagna and Lombardy regions (with captials Bologna and Milan, respectively). These, although popular in 2003, appear to have increased their incidence. The second most attractive area is made up of the central Italian regions of Tuscany, Umbria, Marche and Lazio. The final two areas are composed of Liguria and Piedmont, and Veneto, Friuli-Venezia Giulia and Trentino-Alto Adige, in the northwest and northeast of the country, respectively. There is also an interesting growth in the incidence at the Swiss and Austrian borders (from about 2% in 2003 to about 4-5% in 2008). Resident foreigners seem less attracted by the regions and islands of southern Italy. Returning to the Po Valley area, the maps in figure 2-4 allow us to identify the presence of two large, well defined polarization patterns in the IRF, the first in the centre of the Po Valley and the second near Venice. The maps in figure 2-1 do not reveal these patterns.

Figure 2-5 shows the estimated temporal trends for both the full Italian territory and broken down by area with 95% confidence intervals (using the approach in section 2.2.4). The trends for northern and central Italy are similar. However, those for southern Italy and the islands are much flatter. Northern and central Italy also show a faster growth in incidence. Such differences are supported by

the confidence intervals. Overall, these trends reflect what we see in the maps in figure 2-4, that there is a significant difference between northern/central Italy and the south of the country and its islands. Note that, since the approach outlined in section 2.2.4 cannot account for the presence of residual correlation structure, one would expect the obtained confidence intervals not to be exactly representative than those that might be produced when adjusting for such correlation. Indeed, we see here that the confidence intervals are very tight. Using a procedure which accounts for the correlation would likely widen the intervals, especially in the south and islands where there is less data. The most obvious approach would be to fit a generalized additive mixed model (Wood, 2006, chapter 6) with a carefully chosen correlation structure. Unfortunately convergence problems prevented such a model from being fitted.

The results presented in this section can be useful for policy-makers who may want to allocate economic resources as efficiently and effectively as possible, supporting, for instance, policies and services needed for the integration of resident foreigners. Such policies relate to a number of services such as: admission to education, access to the public health service, professional training, services supporting the match of labor supply and demand. For sociologists and demographers these maps may represent a new way to model spatiotemporal demographic changes and display the results graphically.

2.4 Conclusions

This chapter has shown an application of the soap film smoother as part of a larger spatiotemporal model. It has also hopefully shown that the models defined in chapter 1 can be usefully applied to spatial smoothing as well as illustrating the modelling process.

The smoothed maps show in figure 2-4 and the trends in figure 2-5 show that modelling the incidence of resident foreigners properly can give a much better picture of the distribution of the IRF as compared to just looking at the empirical maps in figure 2-1. The maps clearly show which parts of Italy are more popular with resident foreigners and the temporal trends gave an indication of the differences between different areas over time (in particular north and south). This information may be crucial for policy-makers who may want to allocate economic resources as efficiently and effectively as possible for supporting (for example) policies and services needed for the integration of resident foreigners.

Should some relevant economic variables become available at a municipal level, logical extensions incorporating covariate effects could be considered. This might assist in determining what further factors affect the distribution of resident foreigners in Italy and further aid decision making.

Using a tensor product of the soap film with the cubic regression splines allowed for the full interaction between the spatial and temporal components of the model. This approach is relatively simple to implement thanks to the ease of developing such models using off-the-shelf software (in particular the R packages `mgcv` and `soap`). There were, however, some difficulties. Specifically, it was not possible to build a model for both the mainland and islands simultaneously, which would allow for the estimation of a single temporal smooth for the whole of the country. It is also unfortunate that it was not possible to fit a generalized additive mixed model to take into account the correlation in the data, this was due to numerical issues (in particular the optimization procedure failed to converge).

This analysis has given some idea of the practical considerations from the point of view of an investigator when smoothing over regions with complex boundaries. Being able to code models in a familiar environment makes the process of developing a model much easier (writing the model definition in R for the model used here is only marginally more complicated than that for a more standard GAM). Keeping within the GAM framework allows for the usual diagnostic procedures to be used, making model checking straightforward. On an extremely practical level, it was apparent that the amount of time that it takes to fit the model becomes a serious consideration during development (it took 50 minutes to fit the three models on a MacBook Air 2,1 with a 1.86Ghz Intel Core 2 Duo processor and 2GB of RAM). Waiting for models to fit or to fail to converge can cause frustration and only acts as a barrier to wide-adoption. The methods developed in the coming chapters should ensure that model fitting does not take too long.

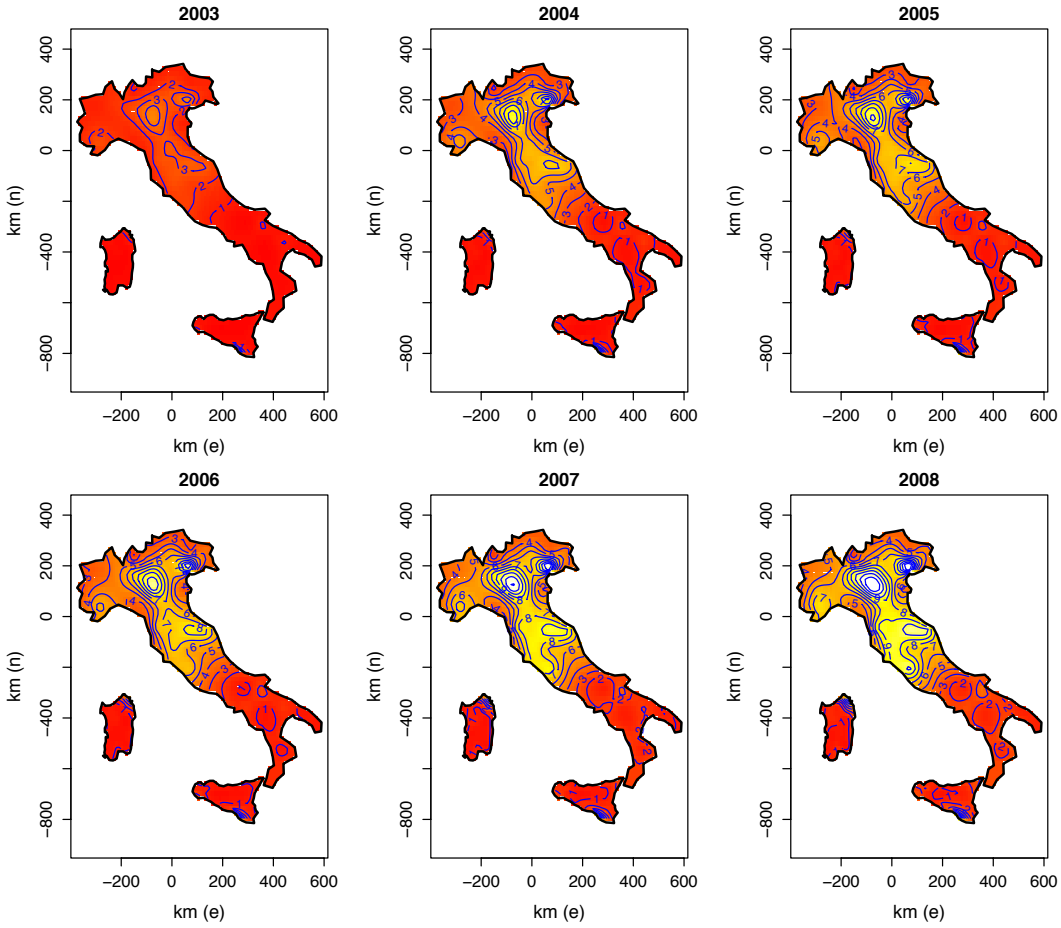

Figure 2-4: Spatiotemporal maps of the percentage incidence of resident foreigners in Italy over the years 2003-2008. These were obtained using model (2.2) with a tensor product smoother based on a cubic regression spline basis for time and a soap film spline basis for space, and ISTAT data at a municipal level. Predictions were made over those points lying inside the study region from a 100 by 100 grid. The incidence is given as the ratio of the number of resident foreigners to the total resident population multiplied by 100. The colour scale ranges from an incidence of 0 (dark red) to an incidence of 12 (white). Blue lines indicate contours separated by a one unit change in incidence.

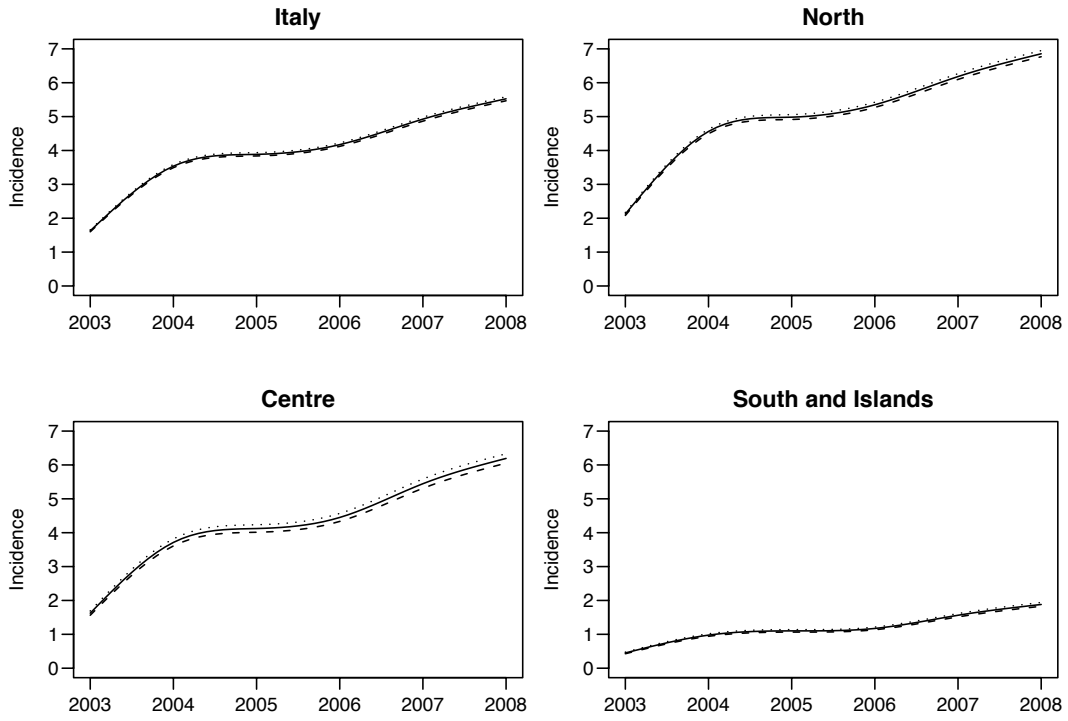

Figure 2-5: Temporal trends in incidence of resident foreigners over the study period for Italy (top left), followed by trend estimates for north, central and south and islands areas with 95% confidence intervals. North was defined as those points in the prediction grid above -20 km north, central as between -20 km and -300, and south and islands (including Sardinia and Sicily) as below -300.

Chapter 3

Using the Schwarz-Christoffel transform to morph domains for finite area smoothing

3.1 Introduction

The example of the Ramsay horseshoe in section 1.4 shows that leakage can be problematic for spatial smoothing. The horseshoe is a rather simple example for leakage, however it does encapsulate the most important feature of leakage: a gap with differing response on each side. It is not so simple to be unrealistic however: comparing the horseshoe to the Aral sea example seen in section 1.1, there are many similarities. Starting with a simple example such as the horseshoe should reveal more fundamental problems without getting bogged-down in issues relating the the overall complexity of the domain.

Leakage is caused by the smooth not respecting the boundary of the domain of interest, so transforming the domain in such a way that the smoother does not need to respect the boundary should reduce leakage. In the case of the Ramsay horseshoe it seems rather obvious what we would like to do: straightening the domain out to be a rectangle seems like the most logical course of action. Objects within the horseshoe surely experience the domain in this way (i.e. their coordinate system is based on the major and minor axes of the shape, rather than Cartesian coordinates) and within-domain distances are well approximated by a rectangle with length equal to the major axis length of the horseshoe and width the same as the horseshoe. As we shall see through the rest of the chapter, the

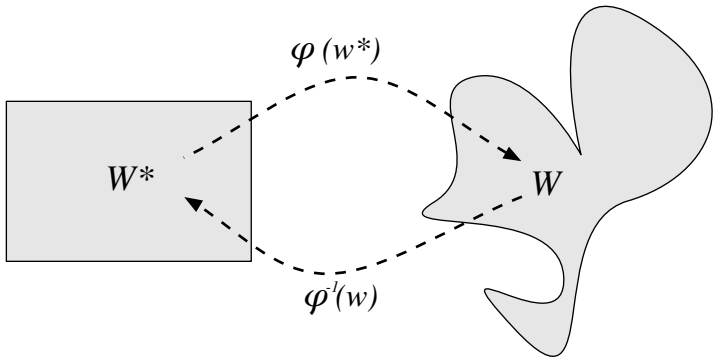

Figure 3-1: Diagram showing the φ^{-1} mapping from an arbitrary shape (W; where the data were collected) to the rectangle (W^*; where smoothing can be performed reliably). φ maps from the rectangle to the arbitrary shape. φ and φ^{-1} will be referred to as the forwards and backwards mappings respectively (c.f. section 3.2.2).

Schwarz-Christoffel transform allows for exactly such a morphing of the domain.

This chapter investigates the efficacy of using a conformal mapping to transform the domain in which we wish to perform smoothing. The mapping takes points in the domain of the data (W, which is the interior of Γ) to a domain on which it is easier to smooth (W^*). In particular the utility of the Schwarz-Christoffel transform is examined (elaborating on Eilers, 2006).

Given some region that it is difficult to smooth over, one approach is to transform the domain in which the problem resides to some known shape. So, for example, one could transform a region into a rectangle, circle or other familiar shape to avoid leakage via a mapping. Figure 3-1 shows φ mapping from the rectangle to an arbitrary shape and φ^{-1} mapping in the other direction. It seems natural to treat the domain as if it were made of silly putty and simply squash the region into the shape required to perform analysis, especially in the case of the Ramsay horseshoe. A transformation-based approach is appealing since it allows the use of existing techniques in the transformed domain; once it's a familiar shape, the domain can be smoothed using a splines just as one would when the domain does not have a complicated boundary.

The Schwarz-Christoffel mapping takes a specified shape and maps it to an arbitrary polygon. The most common domains to transform from are: (i) the upper half-plane, (ii) a rectangle and (iii) the unit disc. This is achieved in the upper half-plane case by taking points on the real line and mapping them to the vertices of the polygon (see figure 3-2). This can be thought of as "unwrapping"

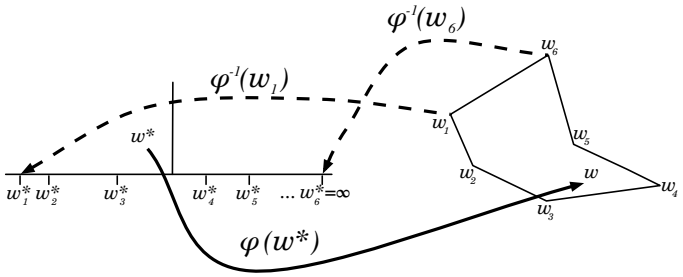

Figure 3-2: An example of mapping and arbitrary point from the upper half-plane to a point on a polygon (solid line). Dashed lines show the inverse map of the vertices of the polygon (w_k) to the prevertices (w_k^*) via φ^{-1} for $k = 1$ and $k = 6$. Note that w_6 is mapped to ∞ on the real line.

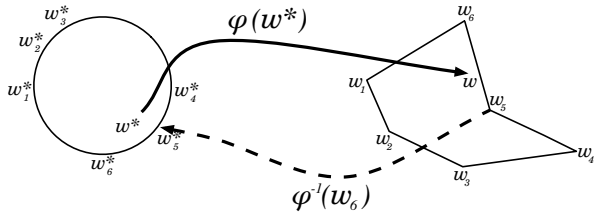

Figure 3-3: An example of mapping an arbitrary point on the unit disc to a point on a polygon (solid line). The dashed line shows the inverse map of one of the vertices of the polygon (w_6) to its corresponding prevertex (w_6^*) via φ^{-1}.

the polygon onto the real line. For the unit disc case, points on the circle bounding the unit disc map to vertices on the polygon (see figure 3-3). The rectangular case is somewhat similar to the unit disc in that extra points are added to the boundary. Once the mapping has been performed, those points lying inside the polygon are also moved around, creating a new (non-uniform) distribution of space. These three domains are discussed in more detail in section 3.2.2.

Proposed procedure

The central idea here is to transform the domain with the complicated boundary to one that is less complicated, then smooth in this transformed domain. The procedure is as follows:

1. Determine the domain over which we would like to smooth, W. This could be the polygon which bounds the region or a simplified version of it.

2. Compute the Schwarz-Christoffel transform of W (the domain with the

complicated boundary) to get W^* (the nicely shaped domain, in which we want to smooth). We then obtain the functions φ and φ^{-1}, which map between the two domains.

3. Map the co-ordinates of the data in W to W^*, via φ^{-1}.

4. Smooth the data in W^* using the methods in chapter 1.

5. Transform back to W; perform any further inference, create heatmaps, etc.

The second section of this chapter explains the technical details of the mapping. The third section gives results of some simulations and the final section summarizes the results of these simulations and draws conclusions about the utility of the method.

3.2 Technical details

This section gives some of the mathematical and computational details required to calculate the Schwarz-Christoffel mapping. The primary reference is Driscoll and Trefethen (2002), which covers almost all aspects of the Schwarz-Christoffel transform.

3.2.1 Nomenclature

The polygon is first defined formally along with its associated quantities, as they will be referred to throughout the rest of the chapter.

A polygon, Γ, is a collection of vertices $w_1, w_2, \ldots, w_K$ and interior angles $\alpha_1\pi, \alpha_2\pi, \ldots, \alpha_K\pi$. For convenience define $w_{K+1} = w_1$ and $w_0 = w_K$. Numbering of vertices is anti-clockwise. The angles are such that $\alpha_k \in (0, 2]$ and we require:

$$\sum_{k=1}^{K}(1 - \alpha_k) = 2. \tag{3.1}$$

The external angle, $\theta_k\pi$, is given by $(1 - \alpha_k)\pi$ (see figure 3-4).

Both points in the domain and the vertices of the polygon are represented as complex numbers (e.g. for some point w, $w = x_1 + ix_2$). The boundary of the polygon is denoted by Γ. Two domains have already been mentioned: W and W^*, denoting the original domain (inside Γ) and the transformed domain (for example, the plane, unit disc or rectangle), respectively. The vertices of Γ are

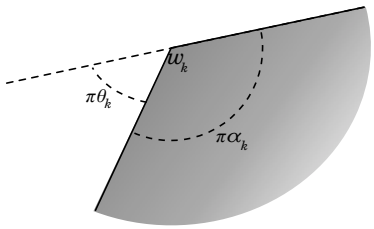

Figure 3-4: The internal angle $\pi\alpha_k$ is associated with the vertex w_k. The external angle is $\pi\theta_k$. Shading indicates the inside of the polygon.

denoted as w_k and the vertices of the transformed boundary are denoted as w_k^* (the *prevertices*). In general, a point in the polygon's original domain will be denoted as w and in the transformed domain as w^*.

The function φ is a mapping from the transformed domain to the polygon (i.e. $\varphi : W^* \mapsto W$). The inverse mapping function, φ^{-1}, is used to take points from the polygon to one of: the unit disc, rectangle or half-plane ($\varphi^{-1} : W \mapsto W^*$). See figure 3-1.

3.2.2 Schwarz-Christoffel Mapping

There are many possible domains that can be mapped from (see Driscoll and Trefethen, 2002, section 4 for many examples). This section looks at the mathematical formulation for the upper half-plane, unit disc and rectangle. The three mappings discussed here are either canonical (in the case of the half-plane) or considered to be useful in a smoothing context (the other two).

For the purposes of smoothing we are interested in the function φ^{-1} (i.e. the function that goes from the domain in which the data were collected to the transformed one), φ must be found before calculations can be made with its inverse (see section 3.2.4). In the literature φ is referred to as the *forwards map* and φ^{-1} as the *backwards map*.

The forwards map, φ, is determined up to translation, scaling, and rotation by the prevertices (see below). The *Schwarz-Christoffel parameter problem* is the task of efficiently finding the prevertices (hence φ). This is a complex problem since the prevertices are solutions to non-linear equations. The Schwarz-Christoffel parameter problem is discussed in section 3.2.3. For the rest of this section the prevertices (the w_k^*s) are assumed to be known.

The rest of the section discusses the mathematical form of the mappings for the upper half-plane, unit disc and rectangle. In each case the mapping takes

the form of an integral which must be evaluated numerically (since there are not closed-form solutions). Accurate and fast (since often many evaluations are needed) computation of these integrals is covered in Driscoll and Trefethen (2002, pp. 27–29) and Howell and Trefethen (1990). Methods for finding and computing Schwarz-Christoffel transformations are implemented in the *SC Toolbox* package for MATLAB written by Tobin A. Driscoll.

The upper half-plane

When mapping from the upper half-plane to W, first set $\varphi(\infty) = w_K$ without any loss of generality. Driscoll and Trefethen (2002, p. 10) then give the following formula:

$$\varphi(w^*) = A + C \int_{w_0^*}^{w^*} \prod_{k=1}^{K-1} (\zeta - w_k^*)^{\alpha_k - 1} \, d\zeta. \tag{3.2}$$

Here A and C are complex constants determined once the w_k^* (which all lie on the real line) have been calculated. These control the scaling, translation, and rotation of the transform.

Mathematically, the base point of the integration does not matter, as it only affects the value of A (Driscoll and Trefethen, 2002, p. 3). However, when calculating $\varphi(w^*)$ numerically, w_0^* is usually chosen to be the prevertex nearest to the point w^*. This is primarily to avoid numerical problems (such as avoiding singularities, see Driscoll and Trefethen (2002, p. 27-29)) but choosing a w_0^* near to w^* also makes the computation faster.

Although setting $\varphi(\infty) = w_K$ does not make any difference in a mathematical sense, it does mean that the density of the points mapped into the upper half-plane is rather odd from a smoothing perspective. Given two adjacent points near w_K, their spacing on the upper half-plane is huge (since w_K is mapped to ∞) in comparison to two adjacent points near any other vertex. For this reason mapping from the upper half-plane is not pursued further here.

Unit disc

The formula for the unit disc looks very similar to that for the upper half-plane but the product now runs over all K prevertices (which are complex numbers, Driscoll (1996)). The integrand is simply a constant multiple of the upper half-

plane case (see also Driscoll and Trefethen, 2002, p. 12):

$$\varphi(w^*) = A + C \int_{w_0^*}^{w^*} \prod_{k=1}^{K} \left(1 - \frac{\zeta}{w_k^*}\right)^{\alpha_k - 1} d\zeta. \tag{3.3}$$

As above, A and C are complex constants responsible for scaling, translation, and rotation, and w_0^* is the base point of the integration.

Rectangle

The rectangle mapping is slightly different in its calculation to the two above mappings. The mapping proceeds in two stages: first mapping from the rectangle to the upper half-plane and then mapping from upper half-plane to W can be calculated (prevertices lie on the real line, as described for the upper half-plane case).

For the rectangle case, the four vertices of Γ which will correspond to the four corners of the rectangle must be specified. The mapping from the rectangle to the upper half-plane is exactly defined by the Jacobi elliptic sine function, sn (further information may be found in Bronshtein et al. (2003, p. 701), Abramowitz and Stegun (1972, p. 567–586) and the appendix of Howell and Trefethen (1990)). The Jacobi elliptic sine function maps the corners of the rectangle to the points $-\gamma^{-1/2}$, -1, 1 and $\gamma^{1/2}$ on the real line (these are marked in order as A, B, C and D in figure 3-5). γ is defined by the aspect ratio of the rectangle and hence which corners of the rectangle map to which vertices of Γ (the computation is covered in depth in the appendix of Howell and Trefethen (1990) and Driscoll and Trefethen (2002, p. 50)). We now have sn which maps from the rectangle to the upper half-plane, so all that is required is to fine the Schwarz-Christoffel mapping from the upper half-plane to the polygon, W. This is performed exactly as described above.

The computation of this map is expensive due to the evaluation of the elliptic function (Driscoll and Trefethen, 2002, p. 49). Howell and Trefethen (1990) propose a shortcut by mapping to the strip, i.e. log-transforming the points in the upper half-plane to the infinite strip ($0 < \text{Im } z < 1$; Driscoll and Trefethen (2002, p. 44)). The Schwarz-Christoffel formula to map from the strip is:

$$\varphi(w^*) = A + C \int_{w_0^*}^{w^*} \exp\left[\frac{\pi}{2}\left(\alpha_- - \alpha_+\right)\zeta\right] \prod_{k=1}^{K} \left[\sinh \frac{\pi}{2}(\zeta - w_k^*)\right]^{\alpha_k - 1} d\zeta.$$

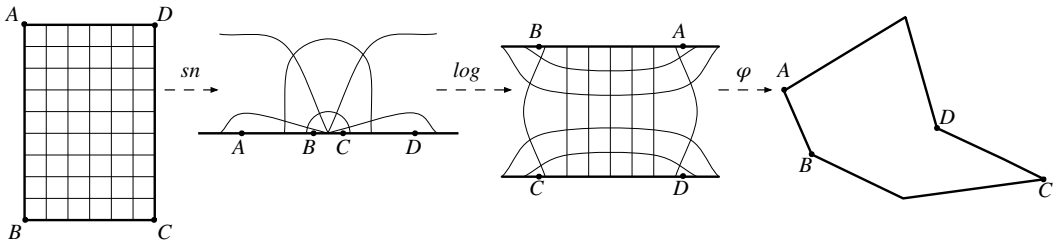

Figure 3-5: An example of the series of mappings required to go from the rectangle to an arbitrary polygon using the Schwarz-Christoffel transform. From left to right: the corners of the rectangle are mapped to points on the real line using the Jacobi elliptic function; using a log transformation, points in the upper-half plane are mapped to the strip (this is purely for computational convenience, this step could be skipped and the Schwarz-Christoffel transform from the upper half-plane to the polygon found); finally, points are mapped from the strip to the polygon via Schwarz-Christoffel transform.

where α_- and α_+ are the so-called divergence angles which allow for each end of the strip to map to a different shape (otherwise the mapping will be symmetrical along this line Re $z = \frac{1}{2}$), sinh is the hyperbolic sin function. The mapping requires that one prevertex be fixed to the origin. This is how the mapping is computed in the MATLAB package *SC Toolbox*, used below.

Figure 3-5 shows the series of mappings required, including the use of the mapping from the upper half-plane to the strip for computational convenience.

3.2.3 Computation of the Schwarz-Christoffel mapping

To compute the map, the prevertices, w_k^* must be found. As mentioned above, the prevertices are solutions to complicated, non-linear equations and must be found numerically. This section illustrates a simple algorithm for finding the prevertices (the primary reference is Driscoll and Trefethen (2002, p. 23–27)).

The complex constants (A and C) in the above equations control scaling, translation, and rotation and can be computed once the prevertices are found. The w_k^* are found by iteratively by mapping those points back to the polygon to give an approximation to Γ, Γ'. To measure the quality of approximation of Γ' to Γ the following set of equations are used:

$$F_k = \frac{|\varphi^{-1}(w_{k+1}) - \varphi^{-1}(w_k)|}{|\varphi^{-1}(w_2) - \varphi^{-1}(w_1)|} - \frac{|w_{k+1}^* - w_k^*|}{|w_2^* - w_1^*|} \qquad \text{for } k = 3, \ldots, K-1. \tag{3.4}$$

Here $|\varphi^{-1}(w_{k+1}) - \varphi^{-1}(w_k)|$ is the distance between the k^{th} and $(k+1)^{\text{th}}$ vertex. This is found by integrating along the line between the points within W. See also section 3.2.3.

Intuitively, we are comparing the side lengths of the true polygon with its approximation in order to measure how well Γ' approximates Γ at each iteration (Saff and Snider, 1993, A-3). Both of these measures are scaled by the distance between the first two vertices (in their respective domains).

Note that (3.4) does not include the vertex w_K. By theorem 3.1 of Driscoll and Trefethen, 2002, p. 24 a polygon is precisely defined by its angles and its vertices not including w_K (since if the direction of the edges leaving w_1 and w_{K-1} are known, the point where they meet may be found). It is for this reason, in the upper half-plane case, that w_K can be mapped to ∞ without loss of generality.

Also note that (3.4) does not include w_1 or w_2 in the numerator on the right hand side. This is due to all vertices (and hence w_1 and w_2) being rescaled, rotated, and translated by the complex constants, A and C, in the Schwarz-Christoffel formula.

In practice some of the prevertices are fixed. For the rectangle case, the vertices of Γ which will map to which vertices of the rectangle must be specified (Driscoll and Trefethen, 2002, p. 48). In the unit disc case w_K^*, w_{K-1}^* and w_{K-2}^* are fixed as $w_K^* = 1$, $w_{K-1}^* = -i$ and $w_{K-2}^* = -1$ (Driscoll and Trefethen, 2002, p. 24).

The scaling factor, C, may be calculated using:

$$C = \frac{|\varphi^{-1}(w_2) - \varphi^{-1}(w_1)|}{|w_2 - w_1|}. \tag{3.5}$$

A is the image of the base point of the integration and is usually written as w_0. For computational reasons this is usually the prevertex nearest to the point which is to be mapped, w^* (Driscoll and Trefethen, 2002, p. 27).

Sketch of an algorithm to calculate the Schwarz-Christoffel mapping

1. Accept inputs:
    - $w_1, \ldots, w_K$ (the vertices of Γ),
    - K (the number of vertices),
    - $\alpha_1, \ldots, \alpha_K$ (the internal angles at each vertex, divided by π),

- $w_1^*, \ldots, w_K^*$ (initial values for the prevertices),
- ϵ (tolerance for convergence).

2. Define the objective function, F_k, as:

$$F_k = \frac{|\varphi^{-1}(w_{k+1}) - \varphi^{-1}(w_k)|}{|\varphi^{-1}(w_2) - \varphi^{-1}(w_1)|} - \frac{|w_{k+1}^* - w_k^*|}{|w_2^* - w_1^*|}, \qquad \text{for } k = 3, \ldots, K-1,$$

   (a) Use steepest descent and then Newton's method to minimize each of the F_ks with respect to w_k^* for each k.

   (b) Update $w_1^*, \ldots, w_K^*$.

   (c) Go back to (a) unless $|F_k| < \epsilon \quad \forall k$.

3. Calculate C and A as detailed above.

4. Return values for $w_1^*, \ldots, w_K^*$, C and A.

Starting values for the algorithm are evenly spaced vertices around the edge of the disc/rectangle or, in the case of the plane, along the real line. Not including those vertices specified as being fixed, above.

3.2.4 Moving between W and W^*

Forwards map

Calculating the forwards map is simply a case of evaluating φ at the necessary points. Definitions of the forwards mappings are given in section 3.2.2.

Backwards map

To calculate the backwards mapping, there are two possible approaches: (i) using Newton's method to solve the equation $\varphi(w^*) - w = 0$ and (ii) solving the initial value problem (IVP):

$$\frac{dw^*}{dw} = \frac{1}{\varphi^{-1}(w^*)} \quad \text{and} \quad \varphi^{-1}(w_0) = w_0^*. \tag{3.6}$$

In practice a combination of these methods are used. Solving (3.6) approximately gives the starting values for the Newton iterations which are significantly faster (since φ^{-1} is cheaper than φ to compute) (Driscoll and Trefethen, 2002, p. 29).

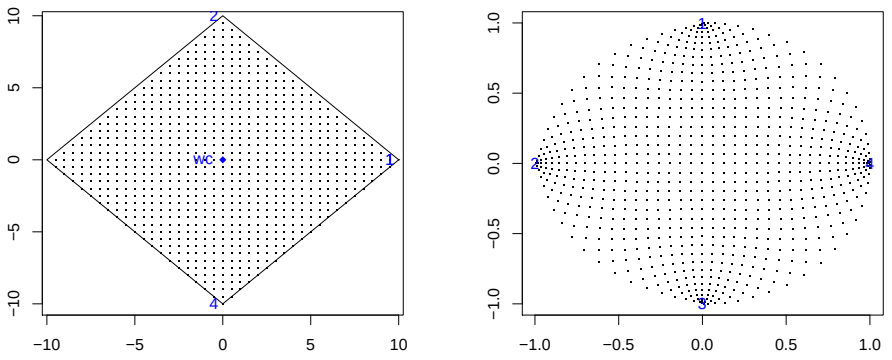

Figure 3-6: A regular grid of points over the square region (left). The right panel shows the mapping of these points under the Schwarz-Christoffel transformation to the unit disc using the CRDT method described in section 3.2.5.

The only problem with this is that the path from w_0 to the point to map, w, must lie entirely inside the polygon. Whether this is true is not known, since after the mapping has been computed the only known points are the vertices (at which the IVP is singular). So, to combat this, all points on the path are checked sequentially. This computation, although inelegant, is fast compared to the IVP/Newton iterations.

An example of using the backwards map to find the transformed co-ordinates from a square to the unit disc is given in figure 3-6. An irregular nonagon is given in figure 3-7. Note that in these two diagrams the CRDT method described in section 3.2.5 was used rather than the method given in section 3.2.3 where the three vertices are fixed on the circle.

3.2.5 Crowding

The crowding problem

When the polygon is elongated or has many vertices, the mapped vertices may be positioned too closely in the transformed domain. In elongated regions prevertices can be located exponentially close such that they are indistinguishable in finite precision arithmetic (Howell and Trefethen, 1990). This effect is referred to as *crowding* and can be observed in figure 3-8.

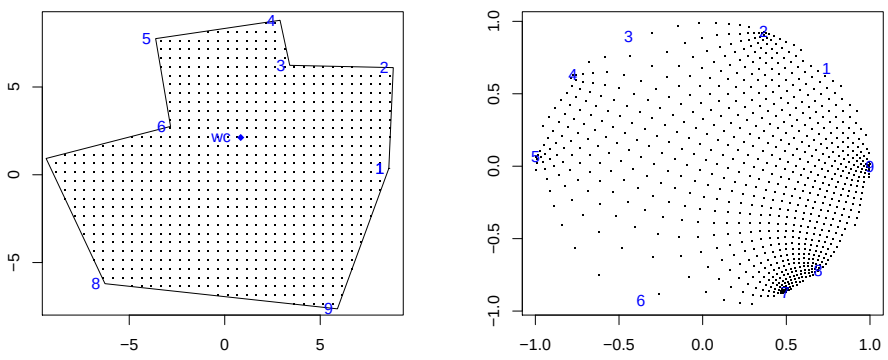

Figure 3-7: A regular grid of points over region bound by a irregular nonagon (left). The right panel shows the mapping of these points under the Schwarz-Christoffel tranformation to the unit disc using the CRDT method described in section 3.2.5

Fixing crowding

If the crowding is caused by Γ being elongated then a primitive fix is to map to an elongated domain such as the rectangle or plane. This approach is suggested in Howell and Trefethen (1990), however, as they point out, this does not eliminate all crowding and problems can still occur when there are acute peninsulae in the polygon. Mapping to an elongated domain also does not fix problems which occur when mapping from T- or H-shaped domains (so-called "multiply elongated" domains).

In order to combat this problem more effectively, Driscoll and Vavasis (1996) propose the CRDT (cross-ratios of the Delaunay triangulation) algorithm (see below). The CRDT algorithm replaces step 2 in the algorithm described in section 3.2.3.

In the solution to the Schwarz-Christoffel parameter problem given in section 3.2.2, conditions on the side lengths and orientations of the polygon are enforced. The CRDT algorithm imposes conditions about quadrilateral sections of the polygon and the diagonals of the polygon. First a Delaunay triangulation of the domain is performed and then pairs of triangles are merged into quadrilaterals. A measure is then defined (the *cross-ratio*) which specifies a set of non-linear equations to be solved. These equations enforce the constraint that the cross-ratio in mapped polygon comes out correctly.

Driscoll and Vavasis (1996) also note that each set of prevertices has $K - 3$

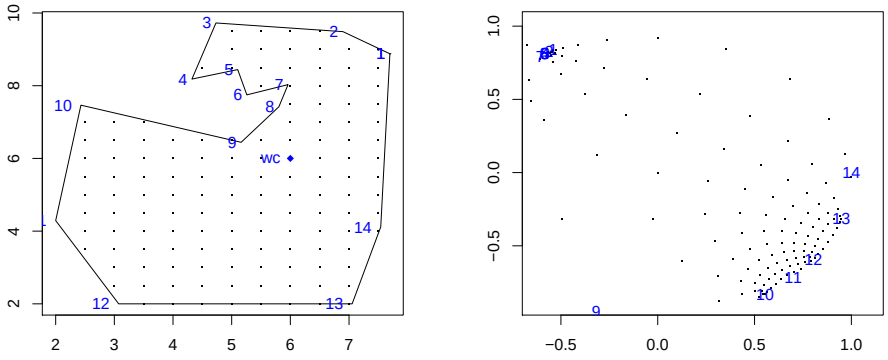

Figure 3-8: An example of crowding. Note that prevertices 1 through 8 are mapped almost to a singularity in the right panel.

degrees of freedom, hence there is a three parameter family of possible vertex arrangements that all map to the same polygon. The most stable of these embeddings should be used. This idea can be extended by noting that the polygon is identical when additional vertices are added between the current ones, provided that the internal angle associated with the new vertex is π. These extra vertices also do not change the Schwarz-Christoffel formula since in (3.3) $\alpha_k = 1$ for an angle of π. Adding these extra vertices gives control over the aspect ratio of the mapping.

Putting all of these ideas together gives a replacement for the algorithm described in section 3.2.3: the CRDT algorithm (Driscoll and Trefethen, 2002, pp. 30-39). First adding edges to the polygon with internal angle π to remove elongated parts of the domain and once the domain is triangulated the cross-ratio is found:

$$\rho(a, b, c, d) = \frac{(d-a)(b-c)}{(c-d)(a-b)}, \tag{3.7}$$

for each of the $K-3$ quadrilaterals in the polygon (where a, b, c, d are prevertices). Then, analogously to (3.4) a series of equations are set up specifying that the cross-ratios remain the same in the polygon and the transform of the rectangle back to the polygon. Then solving for the values of ρ for the original domain in the same manner as we solved for side lengths in the original problem.

In figure 3-9 the CRDT method is used with a rectangular domain, crowding has been alleviated to some degree. The point density, as well as vertex density seems to be more uniform than in figure 3-8.

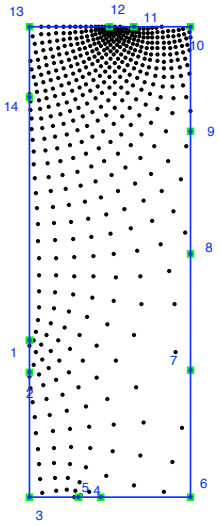

Figure 3-9: The mapping of the irregular domain featured in figure 3-8 using the CRDT method mapping to a rectangle. The crowding is now much less severe.

The downside of using CRDT is that it may add too many vertices to maintain the aspect ratio, so the algorithm takes longer to run than the one specified in section 3.2.3 since it tends to be cubic in the number of vertices (Driscoll, 2005).

3.3 Simulation experiments

In order to test the efficacy of the Schwarz-Christoffel transform for the purposes of smoothing over complex regions, a series of simulation experiments were performed. The *SC Toolbox* for MATLAB was used to transform and map the points. Smoothing was then performed in R using the packages `mgcv` and `soap`.

3.3.1 Ramsay horseshoe

The first set of simulations were run using the Ramsay horseshoe (see section 1.4.2). In order to map as simple a domain as possible, initially a bounding box was used as the W domain and mapped to the rectangle via the `evalinv()` function in the *SC Toolbox*. The bounding box is shown in figure 3-10.

A sample of 1000 points was then taken from the horseshoe and noise added to the data. First the coordinates of the sampled points were expressed as complex numbers of the form (i.e. $w = x_1 + ix_2$). The sample was then mapped into W^*, creating a new set of coordinates ($w^* = x_1^* + ix_2^*$). Smoothing was then performed over the responses in the W^* domain using the `gam()` function in `mgcv` having

Figure 3-10: The horseshoe with its bounding box. The vertices marked 1, 4, 5 and 8 were mapped to the corners of the rectangle.

taken the real and imaginary parts of each data points to be its coordinates.

The models that were fitted to the samples were as follows:

1. *soap*: the soap film smoother with 32 internal knots and 30 boundary knots.

2. *sc+ps*: Schwarz-Christoffel transform of the horseshoe's bounding polygon to the rectangle, P-splines with a 6×10 grid of knots.

3. *sc+tp*: Schwarz-Christoffel transform of the horseshoe's bounding polygon to the rectangle, thin plate regression splines with a maximum basis size of 30.

4. *tprs*: thin plate regression splines with a maximum basis size of 30.

Figure 3-11 shows the true function and typical realisations of: the fit given by a thin plate regression spline on the transformed domain, a P-spline fit on the transformed domain and the fit given by the soap film smoother. Looking at the heat maps one can see that the general shape of the horseshoe function is clearly being reproduced by all methods (especially in comparison to that in figure 1-6). However, on the transformed domains the curvature across the minor axis is not captured.

Table 3.1 shows the settings for noise level and sample size of the simulations. Figure 3-12 shows boxplots of the logarithm of the MSE for the simulations. The Schwarz-Christoffel transform yields results which are comparable, if not better, than the soap film. At a sample size of 100 performance degrades for all methods.

The fits on the transformed domain and the soap film have much smaller MSE than the thin plate regression spline, except in one case, Schwarz-Christoffel with P-splines for sample size 100 and noise level 0.3. Looking closer at the results for this model, there were two results where the MSE was 54.68 and 437.99

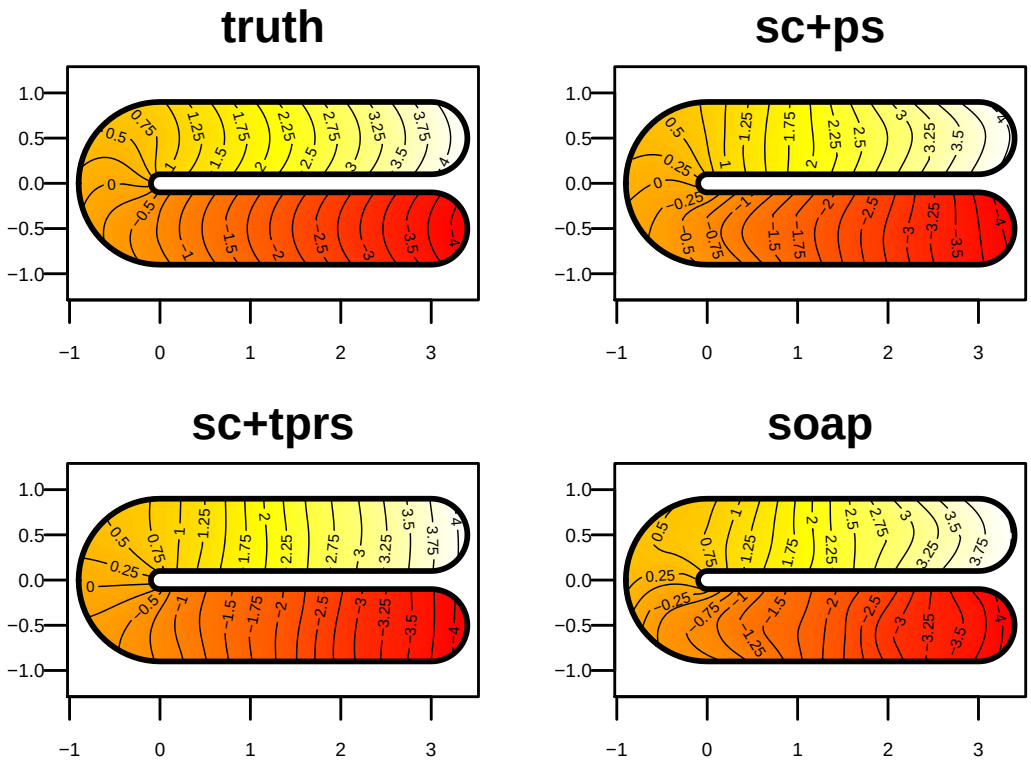

Figure 3-11: A typical set of predictions using P-splines on the transformed domain (top right, "sc+ps"), thin plate regression splines on the transformed domain (bottom left, "sc+ps") and soap film smoother (bottom right, "soap") for the Ramsay horseshoe (top left, "truth"). Sample size was 250, the standard deviation of the Gaussian noise added to the samples was set to 1.

which, when removed, put the MSE back to 0.03003, which seems much more reasonable. These results were presumably due to the P-splines' gridded knot setup, giving a poor fit where there was not enough data. This shows one of the many disadvantages of a knot-based approach. These results were removed for the plotting of the boxplots.

Although this seems initially encouraging, it is worth bearing in mind at this point that in domains like the horseshoe it is obvious what the transform should be. This can be seen by looking at the predicted values for the model in the W^* domain. Figure 3-13 shows predictions from the fitted surface alongside the true values when the domain has been put into its "natural" coordinate system. The coordinate system for the fitted values is the Schwarz-Christoffel transformed coordinates and for the true values it is the major and minor axes of the shape (i.e. one axis along the central curve of the shape and the other perpendicular to that). In the plot, the strong linear trend along the major axis of the horseshoe

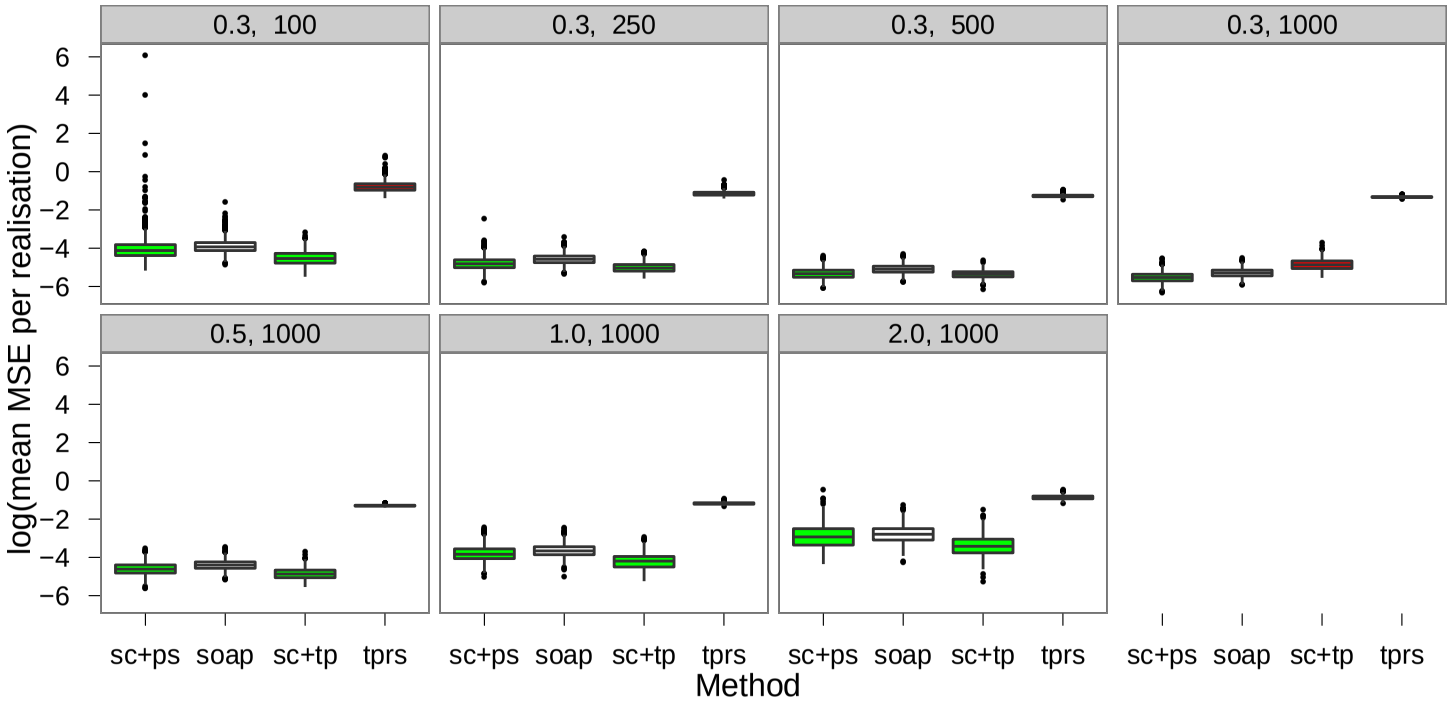

Figure 3-12: Boxplots of the logarithm of the per-realisation MSE for the simulations on the Ramsay horseshoe for the Schwarz-Christoffel transform method using P-spline ("sc+ps") and thin plate regression splines ("sc+tp"), the soap film smoother ("soap") and thin plate regression splines ("tprs") at varying (noise level, sample size) pairings. Colour codings give the results of a Wilcoxon signed rank test on the paired MSEs, between the soap film and the other methods; red indicates significantly worse results, green significantly better and white no significant (at the 0.01 level) difference. In all cases thin plate regression splines were significantly worse than the soap film smoother.

Sample size	Noise level
1000	0.3
500	0.3
250	0.3
100	0.3
1000	0.5
1000	1
1000	2

Table 3.1: Setup for the simulations using the Schwarz-Christoffel transform for the Ramsay horseshoe. Noise level is the number a random deviate from a standard Normal distribution was multiplied by before being added to the value from true test function.

can be seen in both cases, making this a rather simple smoothing problem for the method. The smooth is being calculated in a close approximation to its natural domain. Such an approximation would not be as easy to find for a less regular, more realistic domain.

Alternate Ramsay horseshoe

The second domain tested was the alternate version of the Ramsay horseshoe from Wood et al. (2008). For this domain there is a gorge running along the major axis of the horseshoe (see figure 3-14). The same simulation setup was used as for the first domain (see table 3.1 and model list, bove).

Simulation results for the alternative horseshoe begins to see the soap film creep ahead of the transformation method. The boxplots in figure 3-15 show that the combination of Schwarz-Christoffel transform and P-splines performs better than the soap film when the sample size is low, however at higher noise levels and sample sizes the soap film smoother begins to out-perform the transform method.

Figure 3-16 shows that the Schwarz-Christoffel mapping with P-splines captures the overall structure of the shape better than the soap film smoother, which is rather patchy in its reproduction of the alternate horseshoe. Although the P-spline fit ignores the gradient at the ends of the shape. The Schwarz-Christoffel with thin plate regression spline basis appears worst here, not capturing any of the main features of the domain.

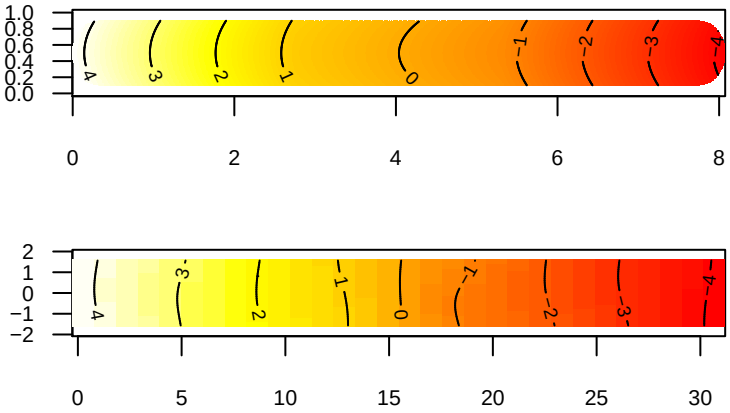

Figure 3-13: Heat map of the true values of the modified Ramsay horseshoe projected into its natural domain (left) and the predicted values of the fit given using the Schwarz-Christoffel transform and then smoothed using a thin plate regression spline (right).

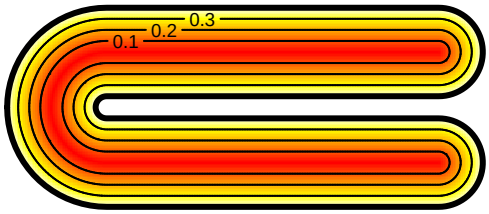

Figure 3-14: The alternate version of the Ramsay horseshoe. Unlike the previous horseshoe there is a gorge running along the major axis of the shape.

The effect of the Schwarz-Christoffel transform on the domain

Using the Schwarz-Christoffel transform before smoothing gives an increase in performance for some smoothers in some situations but not others. To investigate the effect of the distortion on the domain, the mapping of a straight line in the W domain can be plotted along with its equivalent line in the W^* domain. It is also useful to look at the response along that line in both the transformed and untransformed coordinate systems and see how this compares to looking at the response in the horseshoe's natural coordinate system.

Figure 3-17 shows a line along the centre of the horseshoe and its equivalent line in the transformed domain. We can see from this that a line that is straight in the domain (in the sense that it runs along the major axis, keeping the same

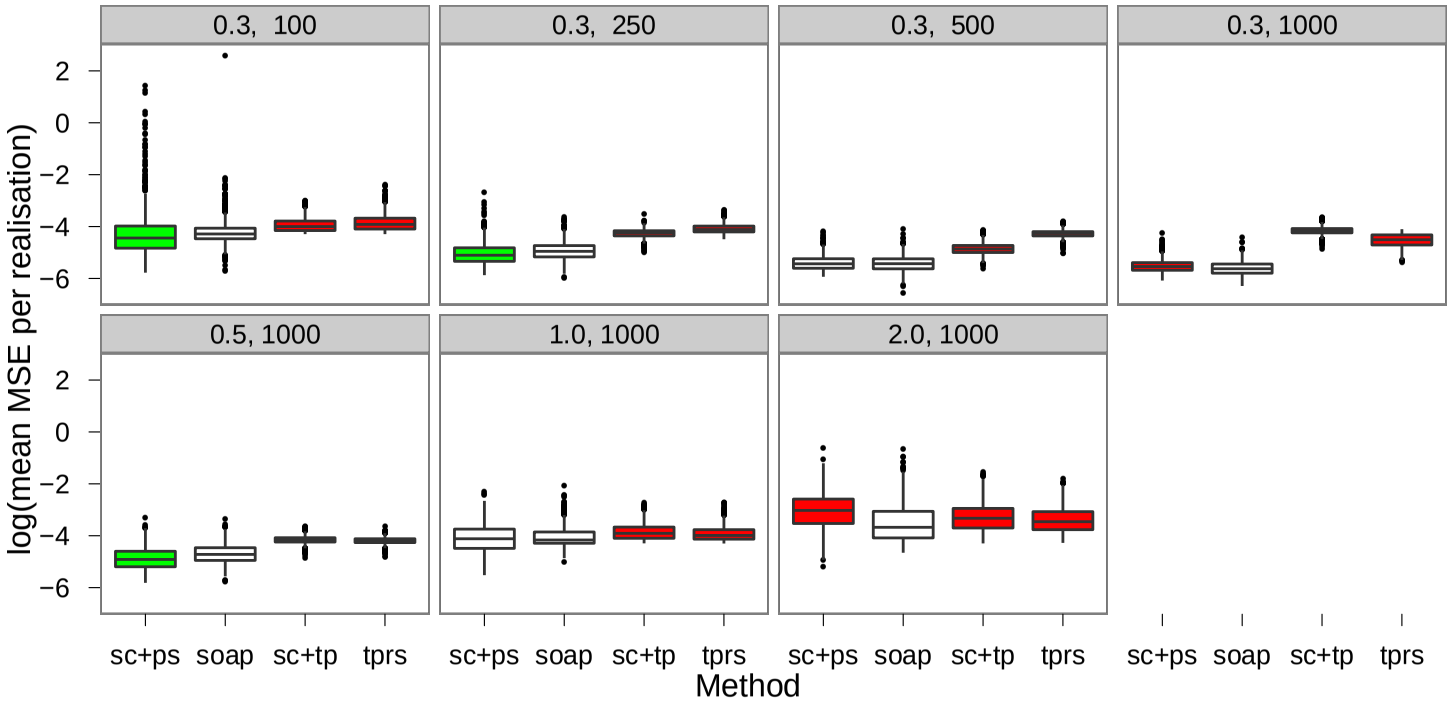

Figure 3-15: Boxplots of the logarithm of the per-realisation MSE for the simulations on the alternate Ramsay horseshoe for the Schwarz-Christoffel transform method using P-spline ("sc+ps") and thin plate regression splines ("sc+tp"), the soap film smoother ("soap") and thin plate regression splines ("tprs") at varying (noise level, sample size) pairings. Colour codings give the results of a paired Wilcoxon signed rank test to detect the difference in MSE between the soap film and the other methods; red indicates significantly worse results, green significantly better and white no significant difference (at the 0.01 level).

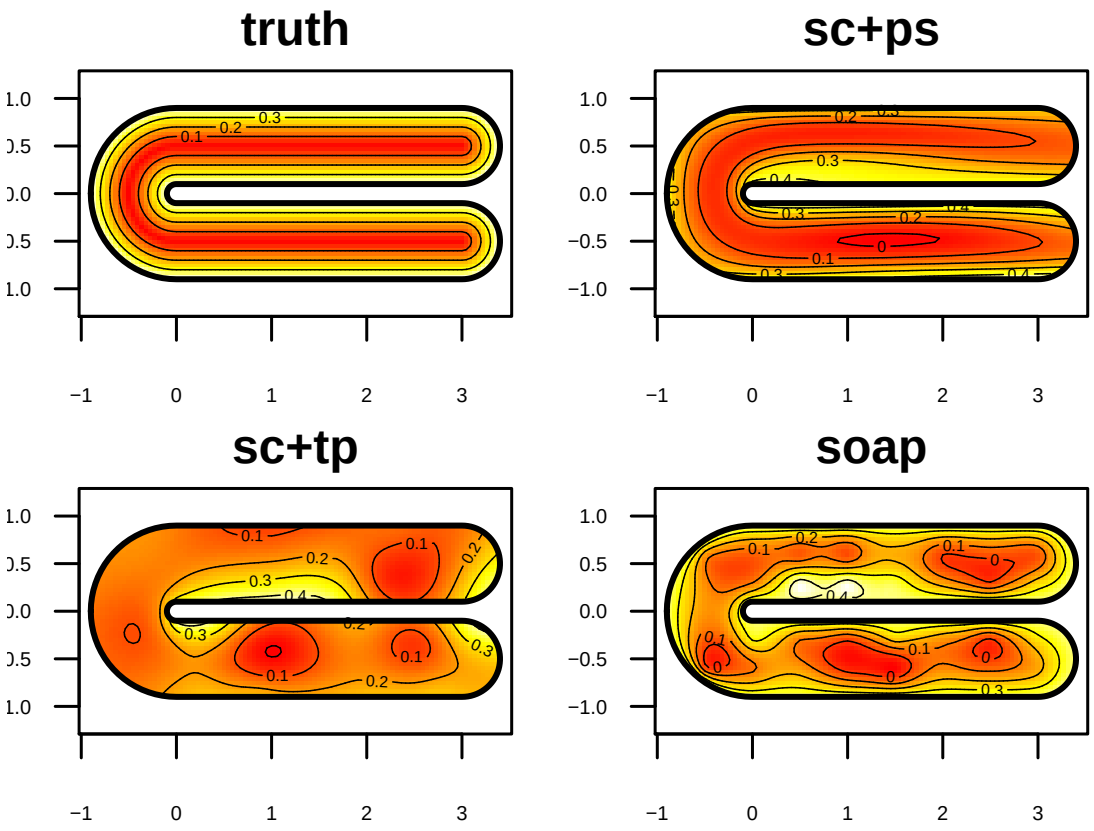

Figure 3-16: Typical realisations of the fit to the alternate Ramsay horseshoe for each method. Clockwise from top left: the original figure ("truth"), the function estimated by the Schwarz-Christoffel transform with P-splines ("sc+ps"), function estimated by the Schwarz-Christoffel transform with thin plate regression splines ("sc+tp") and finally the soap film smoother ("soap"). Additive noise level was 1.

distance from either side of the horseshoe) has a bump in it in the transform. The curvature does not appear to be particularly extreme in this case, however, one can imagine that this could get significantly worse for regions with more complicated boundaries.

Figure 3-18 shows the evaluations of the horseshoe function along the line plotted against three coordinates. The first plot shows the function evaluations on the W domain as a response to change in x_2. The second on the W^* domain, as a response to x_2^*, in the transformed coordinate system. The final plot is in the horseshoe's natural domain, i.e. the value of f as a function of distance along the major axis of the shape. From these plots one can see the quality of approximation to the natural domain of the horseshoe the Schwarz-Christoffel transform provides. Only two minor kinks occur in the line. Looking at where

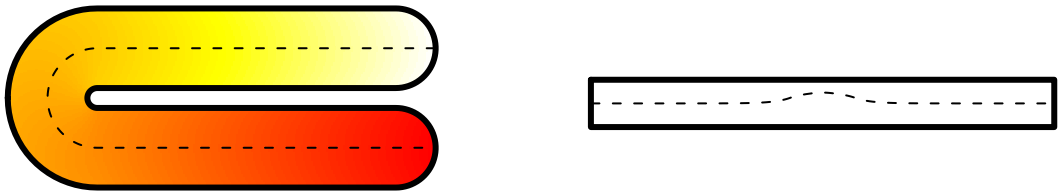

Figure 3-17: Mapping of a straight line along the major axis of the Ramsay horseshoe to its position in the "unwrapped" domain.

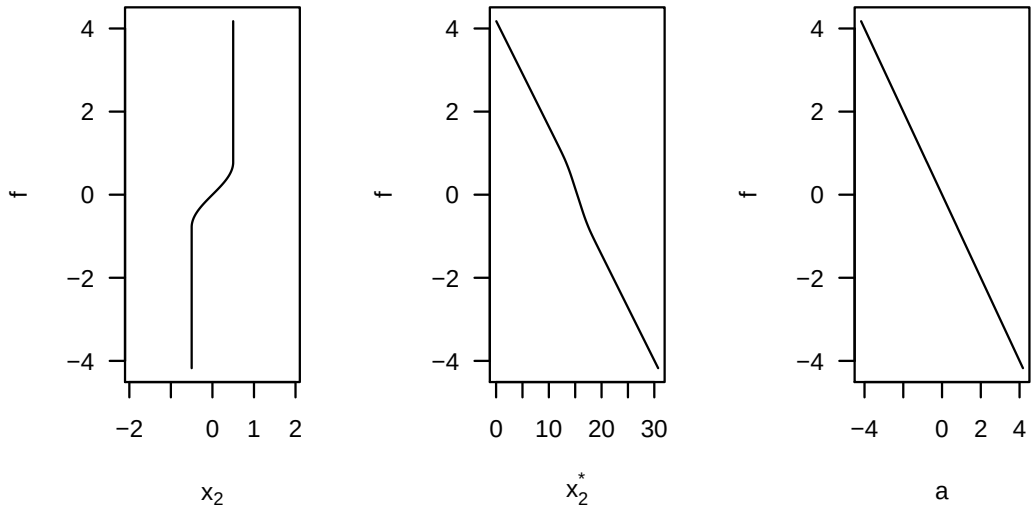

Figure 3-18: Plots of the horseshoe function against the y axis for (left) the untransformed horseshoe, (middle) the shape under the Schwarz-Christoffel transform and, (right) the function evaluation against the major axis.

the kinks occur, they correspond exactly to those kinks in figure 3-17.

Figure 3-19 shows analogous plots to figure 3-18 for the alternate Ramsay horseshoe and backs up this hypothesis. The second panel shows the mapping of the x_2 component of the centreline against the response and the third panel shows the same in the horseshoe's own domain. The two plots appear to be indistinguishable, aside from the change in scale on the horizontal axis; this may account for the failure of the P-splines to model the gradient at the "ends" of the shape see in figure 3-16.

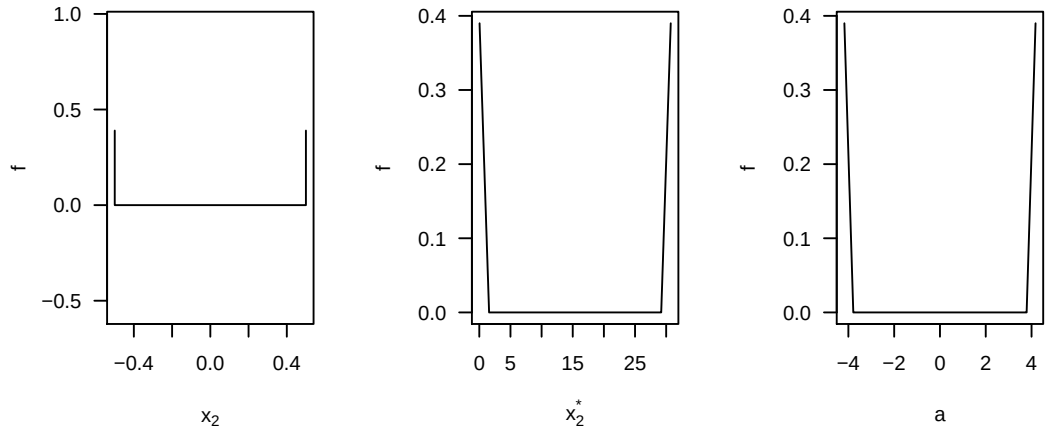

Figure 3-19: Plots of the alternate horseshoe function against the y axis for (left) the untransformed horseshoe, (middle) the shape under the Schwarz-Christoffel transform and, (right) the function evaluation against the major axis.

3.3.2 Peninsula domain

Looking at a more realistic domain highlights potential problems with using the Schwarz-Christoffel transform in practice. The domain in the top left corner of figure 3-20 shows a domain which replicates some of the features of a coastline.

The CRDT algorithm was used for the Schwarz-Christoffel mapping, extra vertices introduced into the polygon with angle π, are shown, along with the mapping in figure 3-21. From this figure one can see that although there is no crowding in the numerical sense discussed in section 3.2.5, there is significant "bunching-up" of the vertices. This would indicate that odd artefacts might be introduced into the smooth.

Several combinations of vertex mappings and knots were tried. The result of this was that the vertices chosen to map to the corners of the rectangle were those shown in pink in figure 3-21. For the soap film smoother a 15 by 15 grid of internal knots was used, of which 109 were inside the region. The cyclic spline around the boundary used 49 knots. Clearly this is the kind of domain in which the soap film smoother performs well (when the chance of leakage is high).

The models used in the simulation were:

1. *soap*: the soap film smoother with 109 internal knots and 49 boundary knots.

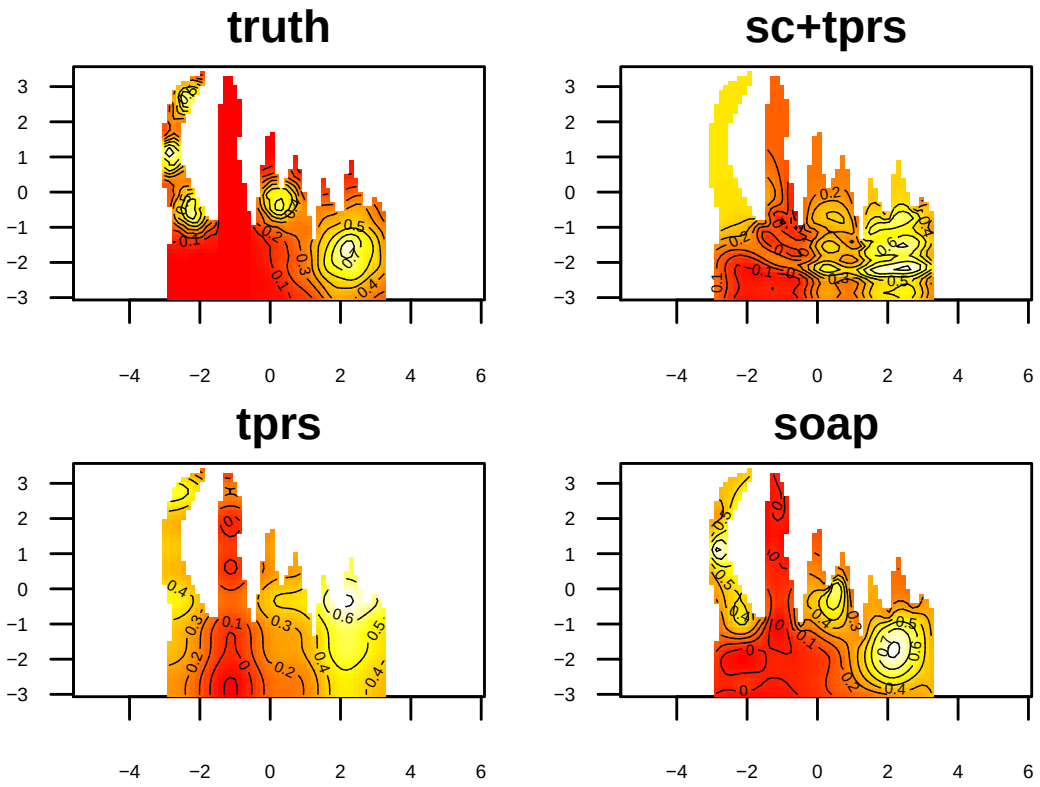

Figure 3-20: The peninsulae domain. From top left, clockwise: truth, fit from Schwarz-Christoffel transform with thin plate regression spline, soap film smoother fit, and thin plate regression spline fit for typical realisations. Sample size was 500 and the noise level was 0.02. For the Schwarz-Christoffel transformed domains, the rectangular mapping was used.

2. *sc+tprs*: Schwarz-Christoffel transform of the peninsula domain to the rectangle, thin plate regression splines with a maximum basis size of 49.

3. *sc+tprs box*: Schwarz-Christoffel transform of the peninsula domain's bounding polygon to the rectangle, thin plate regression splines with a maximum basis size of 49.

4. *tprs*: thin plate regression splines with a maximum basis size of 49.

Looking at the realisations in figure 3-20, one can see that the Schwarz-Christoffel transform causes a huge distortion in the surface. The contour lines in the upper right pane of the figure show how the distortions have pushed the data points around, causing a very bad fit. It also looks as though there is a ridge across the smooth over the domain, which is clearly unwanted. The thin plate regression spline and the transform method both smooth over the details in

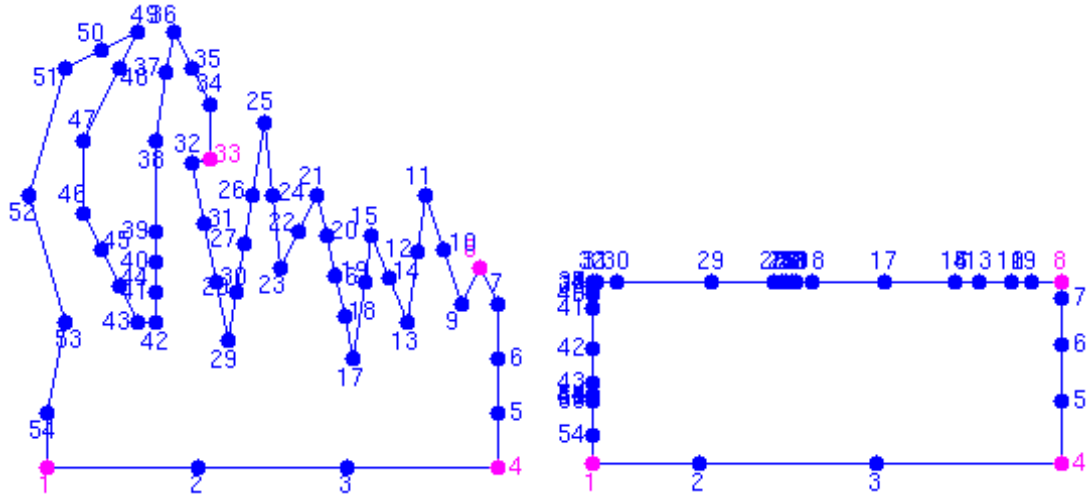

Figure 3-21: Mapping of the polygon to rectangle for the double peninsula domain example. Note the extra vertices added by the CRDT algorithm.

the first peninsula. The soap film smoother is vastly superior to either method in this situation as can be seen in the boxplots in figure 3-23.

In the horseshoe simulations the shape of the domain was simplified by using a bounding polygon. Figure 3-23 shows the results of using the bounding polygon shown in figure 3-24 with the Schwarz-Christoffel mapping (again mapping to the rectangle) as an attempt to reduce the distortions seen in figure 3-20. There is no significant increase in performance by attempting to use the bounding polygon and in fact the variability in the MSE appears to have increased. Although this did iron out some of the artefacts in the smooth, figure 3-25 shows that new, undesirable, features appear. It appears that the Schwarz-Christoffel transform does not perform well in such a situation.

3.4 Conclusions

Domain morphing-type techniques undeniably show some promise when it comes to the problem of smoothing over complex regions. Yet the Schwarz-Christoffel mapping is not the correct transformation to use for all domains. What has been seen here is that the mapping is too prescriptive in its morphing of the domain. There is no reason to think that a particular domain should always be transformed to, it should just be a convenient shape to smooth over. The restriction to only use the unit disc, rectangle, etc. constrains the technique, causing phenomena which are as bad (if not worse) than the leakage we initially sought to avoid.

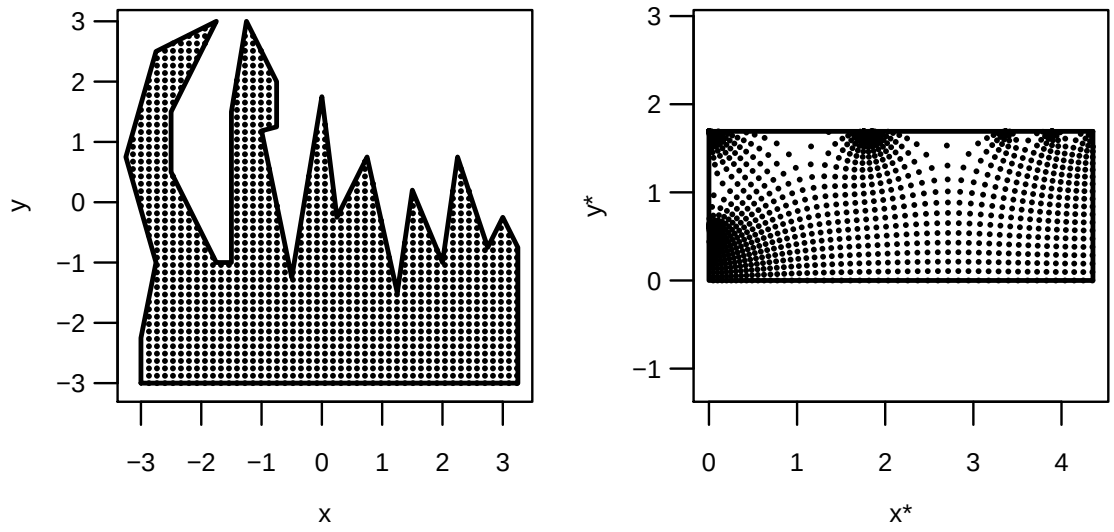

Figure 3-22: Change in the density of points between the mapped and unmapped spaces for the double peninsula domain. Areas with particularly high density in the right panel correspond to vertices in the left.

Crowding (in a technical sense) can be avoided by using the CRDT algorithm, but the squashing together of the points caused by using the Schwarz-Christoffel transform causes huge problems when smoothing. This fundamental problem of mapping between domains is exacerbated by using a restricted set of shapes to map into. It would be preferable to minimize the distortion to the distribution of points in space and using a less regular domain for W^*, rather than having a pre-specified domain, which causes the distribution o of the points to become concentrated at a few points.

The point map in figure 3-22 shows that the Schwarz-Christoffel transform can clearly separate parts of the domain, drawing apart those areas of the domain which were causing the leakage previously. In the case of the Ramsay horseshoe, the transformation that is needed is obvious; problems occur when the transformation is not obvious. With the peninsula domain, it is not clear what an appropriate domain to map into would look like even if the Schwarz-Christoffel transform could map to an arbitrary shape. Certainly, the rectangle or unit disc are not the shapes which immediately come to mind.

Fortunately, all is not lost. This brief foray into conformal mapping has provided plenty of insights into how smoothers will act under mapping schemes. It has also given an appreciation of the criteria which dictate the utility of a transformation-based method, especially with regard to how smoothing behaves

when space is distorted.

With these points in mind, the next step is to find a general mapping scheme for domains with complicated boundaries which is not too prescriptive, causes minimum distortions to the distribution of space but minimises the effects of leakage. This is the subject of the following three chapters.

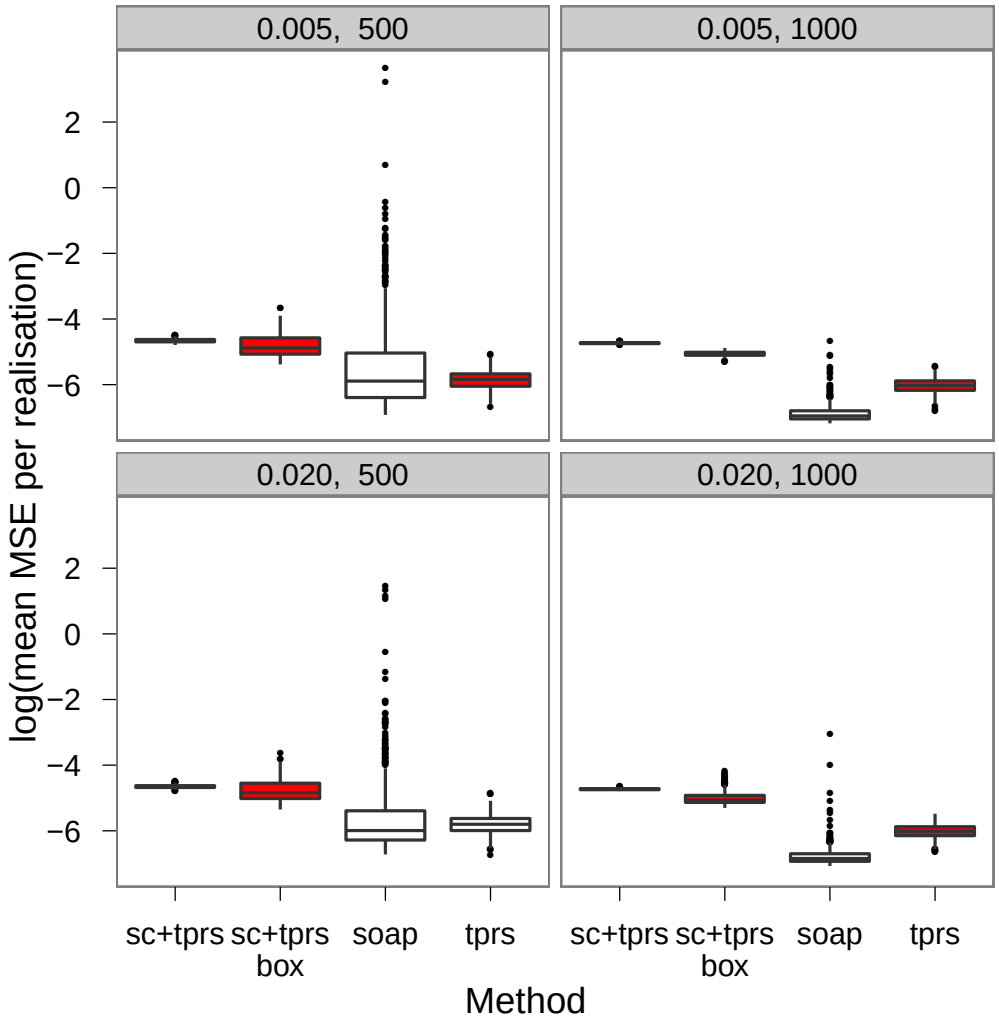

Figure 3-23: Boxplots of the logarithm of the MSE averaged over 1253 prediction points for 500 replicates for various (noise level, sample size) pairs. The models fitted were: the rectangle Schwarz-Christoffel mapping with thin plate regression splines ("sc+tprs") and the bounding box mapped to the rectangle ("sc+tprs box"), alongside the soap film smoother ("soap") and thin plate regression spline ("tprs"). Colours indicate the results of a paired Wilcoxon signed rank test on whether the MSEs were significantly different from the soap film smoother's; red indicates significant different and worse MSE, white non-significant (at the 0.01 level).

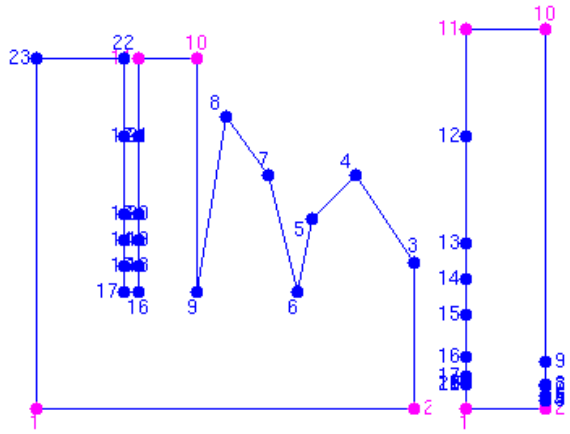

Figure 3-24: The CRDT mapping of the bounding box to the rectangle for the peninsula domain. Pink vertices correspond to those mapped to the corners of the rectangle.

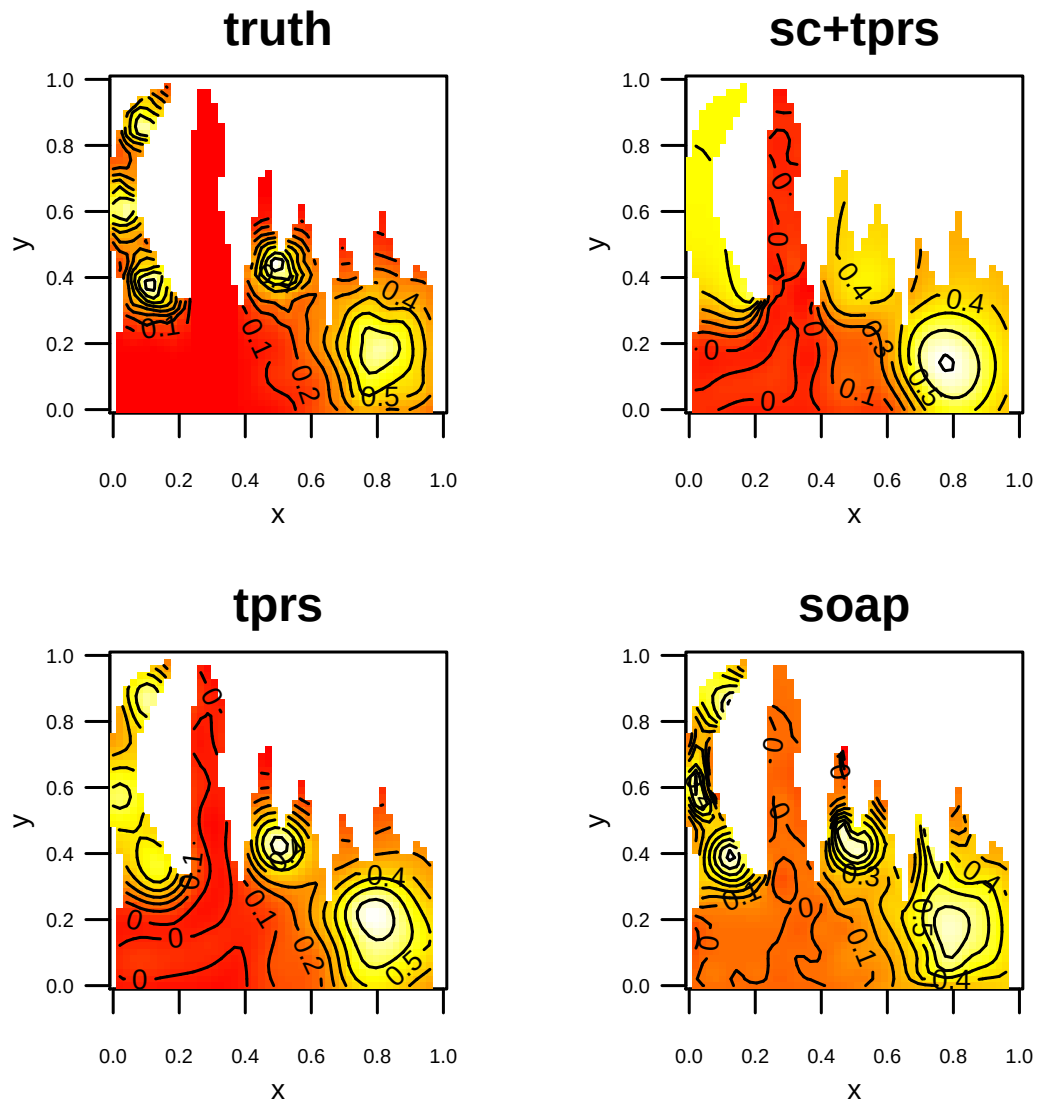

Figure 3-25: The domain with two peninsulae. Truth and typical realisations for (clockwise from top right) a thin plate regression spline fit to the Schwarz-Christoffel transformed domain using the bounding box, the soap film smoother, and a thin plate regression spline on the untransformed domain. Sample size was 500 and the noise level was 0.02.

Chapter 4

Multidimensional scaling for domain-dependent finite area smoothing

4.1 Introduction

Following the work of the previous chapter, the objective is now to find a mapping of the data into a new space which causes minimum local distortion to the points in that new space, while still moving the points apart enough to avoid leakage. At the same time the mapping must also effectively separate those parts of the original domain which are subject to leakage. This balance (between making sure the model overcomes leakage whilst at the same time does not cause artefacts in the smooth) is essential to the success of any transformation-based approach. The Schwarz-Christoffel transform is clearly not flexible enough for general use since it dictates the transformed domain from the outset. A method that depends on the shape of original domain may have more promise.

4.1.1 Proposition

Multidimensional scaling (MDS) or, as it is often referred to, principal coordinates (PCO) (Gower, 1966) is a method commonly used in multivariate analysis. It is closely related to techniques such as principal components analysis (PCA, Chatfield and Collins, 1980, p. 200) and canonical correspondence analysis (CCA, ter Braak, 1986). The starting point for MDS is a matrix of distances, representing some kind of dissimilarity between observations. This distance could be

calculated from the data, for example ideological distance between politicians measured using NOMINATE scores (Cameron, 2009, p. 225), or could instead be distances that occur in the data naturally through experimental setup, for example comparative distances between stimuli response in a psychophysical experiment (Torgerson, 1952). The focus here is obviously on geographical distances.

MDS takes this matrix of distances and projects the data in such a way that Euclidean inter-point distances in the projection are approximately the same as the distances in the matrix (Chatfield and Collins, 1980, p. 187). If the matrix of distances is of rank n then the projection can be in $n-1$ or less dimensions; a projection into 2 dimensions is a typical choice, since it is easily visualised. For this reason one can also think of MDS as a dimension reduction technique, finding a projection of a data cloud into lower dimensional space, while still retaining information about the dissimilarities between the points.

When MDS is performed on some categorised set of dissimilarities (as is often the case in social science and psychology) it is referred to as non-metric MDS, where as on a continuous scale it is known as metric MDS. Discussion here will focus on metric MDS.

Multidimensional scaling provides a way to transform a domain in a similar way to Schwarz-Christoffel. Given the set of distances between points in a domain, we can project those points into a configuration such that the distances between those points are approximately preserved. Now, if the Euclidean metric were to be used to calculate the distances between the points then the result from the projection would be identical (up to rotation and translation) to the starting point configuration (provided that the projection had the same number of dimensions as the original data). However, if it were possible to use a metric that took into account the distance within the boundary (a *within-area distance*) then the Euclidean distances between points in the resulting configuration would be (approximately) the same as the within-area distances. This would lead to distances used by the basis functions of the smoother to be approximately the within-area distances.

Justification for this approach is as follows. In many spatial applications within-area distances is meaningful, given that there is some reasoning behind why certain parts of the domain should not affect one another. Biological populations respect the intrinsic structure of these domains and in general do not respect Euclidean geometry in their movement patterns (for example, they move around obstacles, avoid predators and track prey). When within-area distances

are meaningful, it makes sense to include the structure of the domain in the model, rather than somewhat arbitrarily choose Euclidean geometry and discard this extra information. However, as literature on smoothing is firmly based in a Euclidean context, it would be preferable to perform the smoothing in Euclidean space. In this case the approximation to Euclidean space afforded by an MDS projection of the within-area distances offers a bridge between these two requirements.

4.1.2 Proposed procedure

First take the sample locations $\mathbf{x}_i = (x_{1i}, x_{2i})$ (as in section 1.1.1) of the i^{th} point with response z_i (in general $\mathbf{x}_i$ could be k-vector, although 2-vectors of geographical coordinates are used throughout this chapter). The proposed procedure is as follows:

1. Obtain the MDS configuration for the domain using some representative set of points over the area in question. The only use of the MDS locations obtained in this step is to find the initial MDS configuration; they are discarded afterward. Representative points could be a sparse grid over the domain or a subset of $\{\mathbf{x}_i : i = 1 \dots n\}$. More detail and justification is provided in section 4.2.3, below.

2. Using the MDS configuration obtained above along with Gower's interpolation (see section 4.2.2) to obtain the location of the sample in the MDS configuration: $\{\mathbf{x}_i^*, z_i : i = 1 \dots n\}$.

3. Smooth $\{\mathbf{x}_i^*, z_i : i = 1 \dots n\}$ using a penalised regression spline.

4. To predict at a location $\mathbf{x}_j$ in the original domain, use Gower's interpolation to obtain the point's location in the MDS space: $\mathbf{x}_j^*$. Predict $\hat{f}(\mathbf{x}_j)$.

Here, finding an MDS configuration of a set of points consists of: *i*) calculating the within-area distances between the points, *ii*) forming the distance matrix and, *iii*) actually performing the MDS projection; the details of each step are covered in the following sections. This approach is referred to as MDS+RS (MultiDimensional Scaling with Regression Splines) throughout the chapter.

The rest of this chapter is structured as follows: in section 4.2 a technical overview of MDS is given, along with technical details of how the MDS configuration is calculated; section 4.3 focuses on how the within-area distances are found;

section 4.4 shows some examples of this method on simulated data. Sections 4.5 and 4.6 show some improvements to the initial method and further simulations, section 4.7 details remaining problems. Finally, section 4.8 draws the chapter to a conclusion and lays out areas of further work for the next chapter.

4.2 Technical details

The basic concept behind MDS as used here is to take the data, calculate their within-area inter-point distances and then find their locations in a new coordinate system based on those inter-point distances. Their new locations are determined by finding the eigen-decomposition of the (centred) matrix of distances between points. First a description of the MDS procedure when Euclidean distances are used is given, followed by the justification for the use of the same procedure when using within-area distances.

4.2.1 Finding the new point configuration

First define d_{ij} as the distance between the points i and j. These are used to form a symmetric $n \times n$ matrix, $\mathbf{D}$, with ij^{th}element d_{ij}^2. For the moment let us ignore the issue of how distances are calculated and assume that d_{ij} is simply the Euclidean distance between points i and j.

Diaconis et al. (2008) gives a clear definition of the algorithm (due to Schoenberg, 1935 and Torgerson, 1952) for finding the new locations of points, which is outlined below. Further detail is given in Krzanowski (1990), pp. 104–108 and Chatfield and Collins (1980), pp. 189–200.

First suppose that the original data locations (the $\mathbf{x}_i$s) are unknown, let us find a spatial configuration which reproduces the same distance matrix as the $\mathbf{x}_i$s, then show that there is a simple relation between these two configurations. The n locations in MDS space, $\mathbf{x}_i^*$ (for $i = 1, \ldots, n$), form the rows of an $n \times p$ matrix, $\tilde{\mathbf{X}}^*$ (these new locations reside in p-dimensional space, which we can find lower dimensional projections of; see below). Now let $\mathbb{S} = \tilde{\mathbf{X}}^* \tilde{\mathbf{X}}^{*\mathrm{T}}$, so $\mathbb{S}$ is a matrix of scalar products of the point vectors, i.e. the ij^{th}element of $\mathbb{S}$ is:

$$\mathbb{S}_{ij} = \mathbf{x}_i^* \left(\mathbf{x}_j^*\right)^{\mathrm{T}}, \tag{4.1}$$

where $\mathbb{S}$ is an $(n \times n)$ matrix and $\mathbf{x}_i^*$ is as above. Note that $\tilde{\mathbf{X}}^*$ may only be found up to a translation and rotation (this does not alter the Euclidean distances).

The centroid of the points in $\tilde{\mathbf{X}}^*$ is at the origin, so the sum of each column of $\tilde{\mathbf{X}}^*$ is zero.

We now wish to relate $\mathbf{D}$ to $\mathbb{S}$. First, note that that ij^{th}element of $\mathbf{D}$ is

$$d_{ij}^2 = (\mathbf{x}_i^* - \mathbf{x}_j^*)(\mathbf{x}_i^* - \mathbf{x}_j^*)^{\mathrm{T}} = \mathbf{x}_i^* \left(\mathbf{x}_i^*\right)^{\mathrm{T}} + \mathbf{x}_j^* \left(\mathbf{x}_j^*\right)^{\mathrm{T}} - 2\mathbf{x}_i^* \left(\mathbf{x}_j^*\right)^{\mathrm{T}}. \tag{4.2}$$

Using (4.1), (4.2) can be written as:

$$\mathbf{D} = \mathrm{diag}\left(\mathbb{S}\right)\mathbf{1}^{\mathrm{T}} + \mathbf{1}\mathrm{diag}\left(\mathbb{S}\right)^{\mathrm{T}} - 2\mathbb{S}, \tag{4.3}$$

where $\mathbf{1}$ is an $n \times 1$ vector of 1s and $\mathrm{diag}\left(\mathbb{S}\right)$ is the $n \times 1$ vector of diagonal elements of $\mathbb{S}$.

Define:

$$\mathbf{H} = \mathbf{I} - \frac{1}{n}\mathbf{1}\mathbf{1}^{\mathrm{T}}, \tag{4.4}$$

where $\mathbf{I}$ an $n \times n$ identity matrix and $\mathbf{1}\mathbf{1}^{\mathrm{T}}$ is an $n \times n$ matrix of 1s.

By pre- and post-multiplying any matrix by $\mathbf{H}$ the matrix is double centred (such that row and column means are 0). Pre- and post-multiplying (4.3) by $\mathbf{H}$ yields:

$$\mathbf{HDH} = -2\mathbf{H}\mathbb{S}\mathbf{H}. \tag{4.5}$$

The contributions from the first two terms on the right hand side of (4.3) are zero since the rows of $\mathrm{diag}\left(\mathbb{S}\right)\mathbf{1}^{\mathrm{T}}$ and the columns of $\mathbf{1}\mathrm{diag}\left(\mathbb{S}\right)^{\mathrm{T}}$ are constant. Since $\mathbb{S}$ is already centred so $\mathbf{H}\mathbb{S}\mathbf{H} = \mathbb{S}$. Rearranging, the following relation between $\mathbb{S}$ and $\mathbf{D}$ holds:

$$\mathbb{S} = -\frac{1}{2}\mathbf{HDH}. \tag{4.6}$$

Having found a relation between $\mathbf{D}$ and $\mathbb{S}$, we can now find the relation between $\mathbb{S}$ and $\tilde{\mathbf{X}}^*$. This is simply a case of finding the eigen-decomposition of $\mathbb{S}$. The eigen-decompostion is useful since we wish to decompose the space based on the directions of largest variation (those that contribute to $\mathbb{S}_{ij}$ the most).

Finding the eigen-decomposition of $\mathbb{S}$, we obtain $\mathbb{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}}$. Here $\mathbf{U}$ is the $n \times n$ matrix with orthogonal columns which are the eigenvectors of $\mathbb{S}$ and $\mathbf{\Lambda}$ is the $n \times n$ diagonal matrix of eigenvalues of $\mathbb{S}$ (in descending absolute value). An $\tilde{\mathbf{X}}^*$ satisfying (4.1) may then be computed as:

$$\tilde{\mathbf{X}}^* = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}. \tag{4.7}$$

To summarize the above procedure, given that we have a distance matrix $\mathbf{D}$, the

following steps can be performed to find the MDS projection of the data that $\mathbf{D}$ was calculated from:

1. Finding the matrix $\mathbb{S} = \frac{1}{2}\mathbf{HDH}$.
2. Eigen-decomposing $\mathbb{S}$, to obtain $\mathbb{S} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^{\mathrm{T}}$.
3. Computing $\tilde{\mathbf{X}}^* = \mathbf{U}\mathbf{\Lambda}^{\frac{1}{2}}$.

The aim here is to smooth in two dimensions, and in general multidimensional scaling is performed to reduce the dimensionality of the data, so we must now reduce the dimensionality of $\tilde{\mathbf{X}}^*$. To represent the space using two dimensions the directions with the two largest eigenvalues are chosen and the others discarded. These two largest eigenvalues and their associated eigenvectors constitute the two largest sources of variation in distance (since they are the two largest contributions to $\mathbb{S}$) this gives the two dimensional representation of the data.

Defining $\mathbf{X}^*$ to be the $n \times 2$ dimensional matrix found by truncating $\tilde{\mathbf{X}}^*$ to its first two columns, a 2-dimensional representation has been obtained. More generally, the k-dimensional MDS representation of the space can be found by taking the first k columns of $\tilde{\mathbf{X}}^*$.

In summary, to calculate the MDS configuration of a set of points (given their inter-point distances) we merely need to double centre the matrix of distances, perform an eigen-decomposition on the resulting matrix, and finally truncate the eigen-decomposition and find the new point set.

For finite area smoothing, d_{ij} should be the shortest distance between the points i and j such that the path remains within the domain. Calculation of the inter-point distances is covered in section 4.3, however it is important to first justify the use of these steps when non-Euclidean distances are used. Given that the distances in $\mathbf{D}$ obey the triangle inequality, the n points from which $\mathbf{D}$ is computed may be represented by MDS in $(n-1)$-dimensional Euclidean space (at worst, see Gower (1968)). In the case when the distances in $\mathbf{D}$ are shortest within-area distances, one can think of the points as residing in a higher number of dimensions in such a way that the distances between them are Euclidean.

4.2.2 Gower's interpolation

Given the setup in section 4.1.2, once the MDS configuration has been found further points will need to be inserted into our MDS representation. For example

when further data is collected, or in order to predict over points not in the initial grid. In this case we would like to insert those new points into the configuration given by MDS. This can be performed by Gower's interpolation (Gower, 1968).

Say we have some point, x_{new} and we wish to find its location, x^*_{new} in the current MDS configuration. The position of x^*_{new} is at a Euclidean distance from the points in $\mathbf{X}^*$ which is approximately the same as the (within-area) distance between x_{new} and the points in the non-transformed space, $\mathbf{X}$.

Note that here it is assumed that a 2-dimensional projection has been used in the initial MDS configuration, Gower's interpolation remains valid for the case in which the initial MDS projection is k-dimensional.

Gower's interpolation formula

We may find the position in the transformed space, x^*_{new} , of some new datum x_{new} in the original space using:

$$\tilde{x}^*_{\text{new}} = \frac{1}{2}\mathbf{\Lambda}^{-1}(\tilde{\mathbf{X}}^*)^{\mathrm{T}}\mathbb{D}. \tag{4.8}$$

Here $\mathbf{\Lambda}$ $(n\times n)$ and $\tilde{\mathbf{X}}^*$ $(n\times p)$ are as above, $\mathbb{D}$ $(n\times 1)$ is the vector of centred within-area distances from the points in the original configuration to x_{new}. The resulting vector, $\tilde{x}^*_{\text{new}}$, is a p-vector, so again we can truncate to a k-vector in a similar way to above (taking the first two entries gives the 2-dimensional projection).

In Gower (1968) the i^{th} element of $\mathbb{D}$ is defined as $-(d^2_{i,\text{new}} - \text{diag}(\tilde{\mathbf{X}}^*\tilde{\mathbf{X}}^{*\mathrm{T}})_i)$, with $d^2_{i,\text{new}}$ being the squared distance from the i^{th} point to the new point. The centring is given by the diagonal elements of $\tilde{\mathbf{X}}^*\tilde{\mathbf{X}}^{*\mathrm{T}}$ i.e. the squared distances from the original points to the centroid of the MDS configuration. To avoid confusion (and to emphasise that the full $\tilde{\mathbf{X}}^*$ matrix is used, rather than its truncated version), it may be easier to think of the expression for the i^{th} element of $\mathbb{D}$ as $-(d^2_{i,n+1} - \text{diag}(\mathbb{S})_i)$. Since $\mathbb{S}$ is already known, this expression is more sensible to use for computation as it doesn't imply any extra matrix multiplication.

Gower's interpolation extends simply to the case when m new points are inserted by having $\mathbb{D}$ as an $n \times m$ matrix (so that the resulting $\tilde{x}^*_{\text{new}}$ is an $p \times m$ matrix that can be truncated to the first k columns, as above).

4.2.3 Using grids to compute the initial MDS configuration

Gower (1968) shows that performing MDS on a dataset is equivalent to performing MDS on a reduced set of points and then inserting the remaining points when the Euclidean metric is used to calculate the distances. However in the within-area distance case there can the potential problems if the reduced set of points does not encapsulate enough information about the domain. The first section here addresses this problem. The second section investigates the new configuration of points in a similar way to check for the problematic squashing of space seen when the Schwarz-Christoffel transform was used in the previous chapter.

The only place that the boundary enters the model is through the MDS configuration and in turn, the MDS configuration's only influence from the boundary is via the distances in $\mathbf{D}$. In a spatial setting one can imagine the case in which samples were not taken from one part of the domain of interest (for example, there may be no observations in a particular peninsulae) and in that case the resulting MDS configuration would differ from a configuration when data from the whole domain was used. Ensuring that different analyses on the same domain of interest yield consistent results is extremely important.

When Euclidean distances are used to calculate $\mathbf{D}$, the criterion needed so that the resulting space is the same (in the sense given above, that insertion into a reduced point set is the same as mapping the complete point set) is that there is one more point used to create the MDS configuration than there are dimensions in the space in which they reside (provided the points are not collinear) (de Silva and Tenenbaum, 2004). There is no similar criteria for within-area distances and it is unclear what form such a criterion would take.

A simple example of this problem is shown in figure 4-1. Here, a regular grid has been generated inside a T-shape (top left panel). The point configuration found by using the full set of points and within-area distances is given in the top right panel. Sampling only the "head" or "tail" of the T and using those points to generate the MDS configuration, then inserting the other points leads to the plots in the bottom left for head and bottom right respectively (red points are inserted into the black configuration). One can see that the configuration of the inserted points looks warped compared to the configuration in the top right.

Although the cases shown in figure 4-1 are somewhat pathological, looking at more reasonable situations still shows that the results can vary a considerable

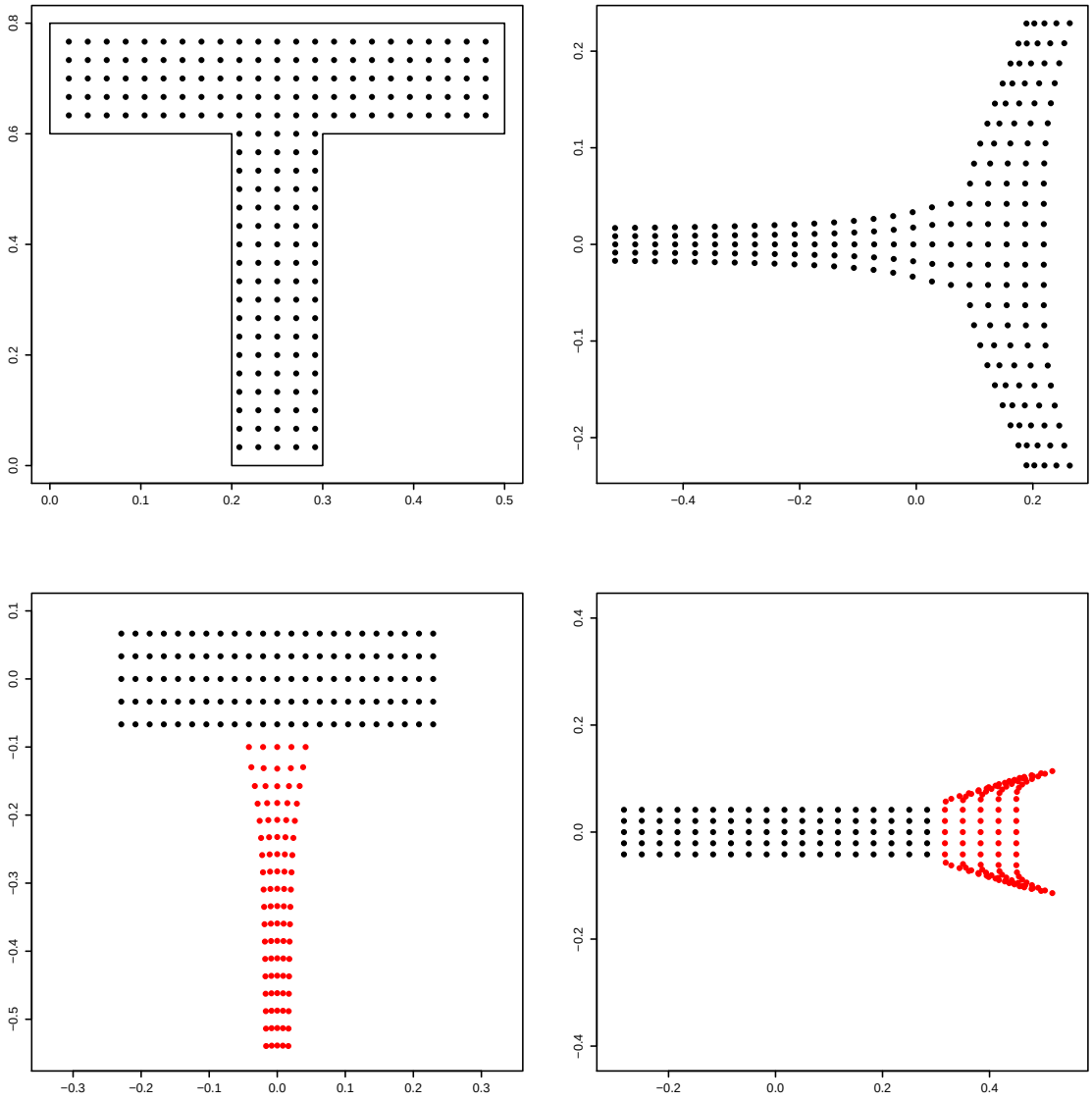

Figure 4-1: Data generated inside a T-shape (top left) is fed into MDS at once (top right). When either the head or tail of the T is used for the original MDS configuration and the other points inserted, the shape produced is distorted (bottom row). In the bottom row the black points give the initial configuration and the red points are those inserted later.

amount. In figure 4-2 the black and green points make up the original MDS configuration; the five green points are chosen at random. The red points are then inserted. As can be seen in these four typical realisations, the shape of the MDS space is dependent on those points used to create the initial MDS configuration.

This problem can be rectified by using an appropriately spaced grid over the domain to calculate the eigen-decomposition, thus ensuring that the whole domain is covered. Provided that the grid is fine enough to catch all of the important features in the boundary of the domain, the problems above should not arise.

4.3 Finding the within-area distances

So far it has been assumed that the matrix of distances D is known. Using Euclidean distances would lead to the original configuration of points being recovered (up to translation and rotation) provided the projection dimension was the same as that of the original set of points. As was pointed out in section 1.4, Euclidean distances are the cause of leakage in the first place, since they do not reflect the distances that must be travelled between points residing in the domain of interest. This section describes (what the author believes to be) a novel algorithm to find shortest paths within a given domain.

Note that paths between point pairs in *simple* polygons (i.e. those polygons without holes) are considered. Although this limits the types of domains that can be addressed, it does make the shortest path algorithm simpler, since the shortest path is unique.

There are several methods available to calculate within-area distances, however given that the domain of interest is any arbitrary simple polygon this limits the number that are applicable. The two most promising possibilities are the geodesic methods used by Wang and Ranalli (2007) (see also section 1.4.3) and the A* algorithm (Hart et al., 1968) which can be thought of as a generalisation of Dijkstra's algorithm (Dijkstra, 1959). However, both of these algorithms rely on the discretization of the domain of interest. As stated in section 1.4.3, this discretization of the domain is undesirable since the results then become dependent on the resolution of the discretization of the domain, even if a high enough resolution can be used the computational cost becomes prohibitively expensive for such methods. It would be preferable to have an elementary algorithm (i.e.

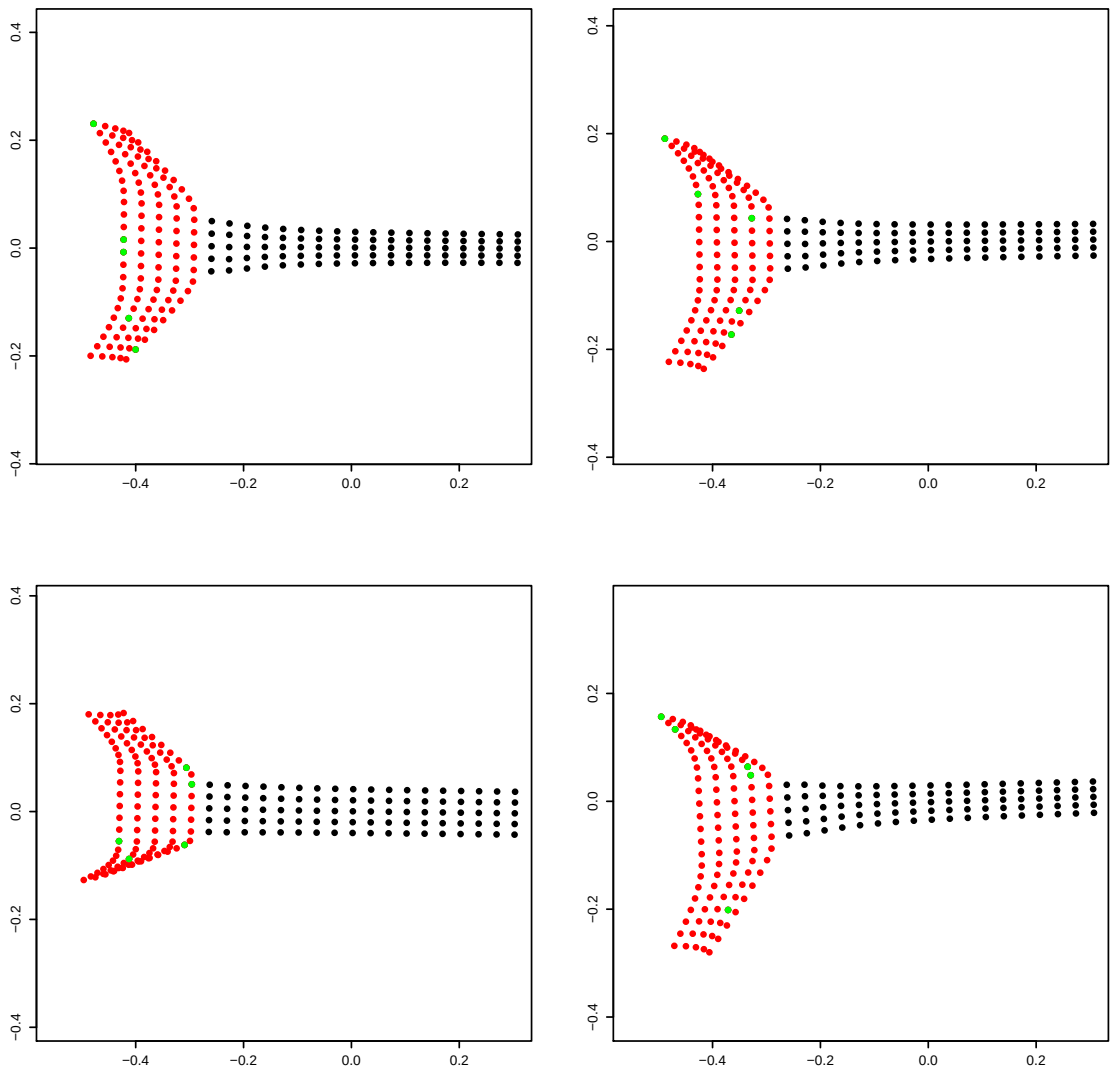

Figure 4-2: Using the T-shape in figure 4-1 (top left), the tail (black points) of the T was used with 5 randomly sampled (green) points in the head. The head (without the 5 green points) was then inserted into the MDS configuration (red). As can be seen from these four realisations, the output varies greatly depending on the points sampled.

one not relying on complex data structures or extensive theory) for finding the shortest path within the polygon.

The algorithm is defined as follows, it may be helpful to look at the example in figure 4-3 while reading.

Let the domain boundary be some polygon, Γ. Given that there is no direct path within the domain between two points (p_1 and p_2, say), the algorithm proceeds as follows to create a path, $\mathcal{P}$, which is an ordered set of vertices:

1. (INIT) Start by drawing a line between p_1 and p_2 (figure 4-3, (i)). Start the path as the lines from p_1, p_2 to their nearest intersection with the boundary of Γ (p_1^1, p_2^1, say). Then form two paths. The first path from p_1^1 to p_2^1 ($\mathcal{P}_1$) contains the vertices of Γ found moving along the boundary from p_1^1 to p_2^1. The second ($\mathcal{P}_2$), is found by taking the path from p_1^1 to p_2^1 in the other direction around the boundary, ie. the vertices of Γ not in the first path. It is easy to see that $\{\mathcal{P}_1 \cup \mathcal{P}_2\} \setminus \{p_1^1, p_2^1\} = \Gamma$. The DELETE step (below) is then performed on $\mathcal{P}_1$ and $\mathcal{P}_2$, removing any superfluous vertices. Finding the length of $\mathcal{P}_1$ and $\mathcal{P}_2$ and choosing the shorter ($\mathcal{P}^*$), the initial path is formed as $\mathcal{P} = (p_1, p_1^1, \mathcal{P}^*, p_2^1, p_2)$.

   In figure 4-3, (iii), $\mathcal{P}_1$ is marked in green and is chosen to form the initial path, $\mathcal{P} = (p_1, p_1^1, \mathcal{P}_1, p_2^1, p_2)$, as $\mathcal{P}_1$ is shorter than $\mathcal{P}_2$, in red.

2. (DELETE) Given a triple of vertices, $(v_i, v_{i+1}, v_{i+2}) \in \mathcal{P}$, if the line between v_i and v_{i+2} is shorter than the path (v_i, v_{i+1}, v_{i+2}) and the line between v_i and v_{i+2} lies inside Γ then delete v_{i+1} (figure 4-3, (iv) and (vi)). The entire path is iterated over ($i = 1, \ldots, N-2$, if there are N vertices in $\mathcal{P}$) deleting all superfluous vertices until there are no changes in successive runs.

   For example in figure 4-3 (iii), v_2 is deleted from $\mathcal{P}$ because the path straight between v_1 and v_3 is shorter, and within Γ.

3. (ALTER) Given a triple of vertices $(v_i, v_{i+1}, v_{i+2}) \in \mathcal{P}$, if the candidate replacement path $\mathcal{P}_{ID}$ is shorter than the path (v_i, v_{i+1}, v_{i+2}) then replace (v_i, v_{i+1}, v_{i+2}) with $\mathcal{P}_{ID}$ (figure 4-3, (v)). The candidate replacement path, $\mathcal{P}_{ID}$, is calculated by running INIT with p_1 and p_2 replaced by v_i and v_{i+2}.

   For example in figure 4-3 (iv), the path (v_1, v_2, v_3) is longer than the path $\mathcal{P}_{ID} = (v_1, v_2^1, v_3)$ (green dashed line in (iv)) so the former is replaced with the latter in $\mathcal{P}$. The path created by INIT is marked as $\mathcal{P}_I$ in (iv) in red.

4. (ITER) Iterate further DELETE and ALTER steps (in pairs) until there has been no change in $\mathcal{P}$ from one run to the next (i.e. convergence) (figure 4-3, (vi)).

Of course, if there is a direct path between p_1 and p_2 then the Euclidean distance between the points can be used and the above algorithm is not run.

Although it was not possible to theoretically prove that the algorithm will always converge to the shortest path, it is clear at least that the algorithm will always converge (since this only requires that there be no change in the path for two consecutive iterations). Extensive simulations showed that the algorithm gave sensible results.

4.4 Simulation experiments

In order to investigate the efficacy of MDS+RS, a series of simulation experiments were performed. In all cases the results for MDS+RS were compared to those of the current best method (the soap film smoother) and the standard approach that does not account for leakage (thin plate regression spline).

The R packages `mgcv` and `soap` were used for smoothing. Bespoke software was used to find the within-area distances and to perform Gower's interpolation (written by the author, see section 6.2). MDS projections were performed using the `cmdscale()` function in R. In all cases smoothing parameter estimation was performed using GCV (see section 1.1.4).

4.4.1 The Ramsay horseshoe

If a general method is to be useful it must first perform well on even a simple case such as the modified Ramsay horseshoe since it clearly illustrates the problem of leakage.

Setup

For the horseshoe, samples of 250 points were taken and normal noise added (for 3 different standard deviations: 0.1, 1 and 10). (These are the settings used in Wood et al., 2008.) Three methods were applied to the data. Predictions were then made over 718 points (including the sample locations). 200 realisations were generated and the effective degrees of freedom (EDF) and MSE recorded for each replicate. The three methods used were as follows:

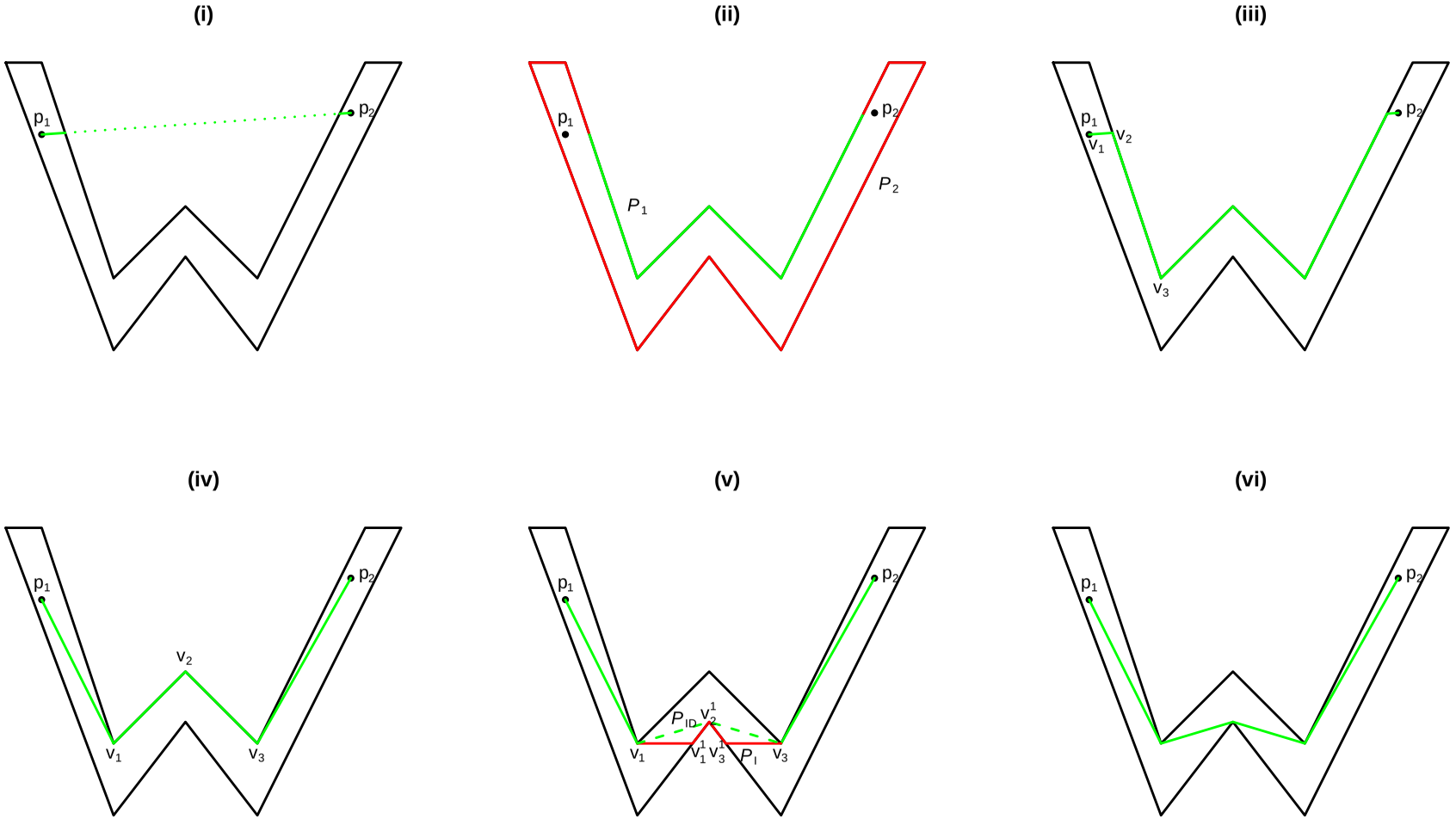

Figure 4-3: The green lines in (i) to (vi) show the steps forming the shortest path as the algorithm progresses from initial state to final, shortest path (bottom right). See section 4.3.

1. *Thin plate spline*: bivariate thin plate regression spline with basis size 100.

2. *Soap film smoother*: 32 knots evenly spread over a grid over the domain, cyclic spline on the boundary was of basis size 39.

3. *MDS+RS*: Used a thin plate regression splines of basis dimension 100. The initial MDS grid was made of points from 10×20 grid that lay inside the horseshoe.

Note that due to time and computational restrictions, the boundary was reduced from the 160 vertex polygon in the `fs.boundary()` function in `soap` to a 21 vertex polygon by only using every 8^{th} vertex. This should not cause a major difference in results even if the soap film used the full boundary and MDS+RS used only the reduced set of edges, since the objective is to allow the smoother to get a broad idea of the topology of the domain, rather than the minutiae of the boundary features. This is especially true given that the large number of boundary vertices are due to the attempt to approximate the curved parts of the shape, rather than other features such as peninsulae. In the simulations presented here, the soap film smoother and MDS+RS used the same boundary.

Results

Predictions from a typical realisation can be seen from figure 4-4, where the noise level was 1 with a sample size of 250. Both the soap film smoother and MDS+RS are able to reproduce the main features of the true horseshoe function due to their ability to respect either the boundary (in the case of the soap film) or the geometry of the domain (in the case of MDS+RS). The thin plate regression spline shows leakage as expected. When the noise level is high the MDS+RS outperforms the soap film smoother in MSE terms (and is less variable).

The EDFs in figure 4-5 show that MDS+RS fits a less complex model than the thin plate regression spline on average, and for the two higher error situations, has a lower EDF than the soap film. Given that this is coupled with a lower MSE, it appears that MDS+RS simultaneously yields both a more accurate and less complex model than the soap film for the horseshoe when there is a high level of noise. When noise is lower, the soap film and MDS+RS MSEs are still roughly of the same order. Figure 4-6 shows the logarithm of the per-realisation average MSE for each of the models at each error level.

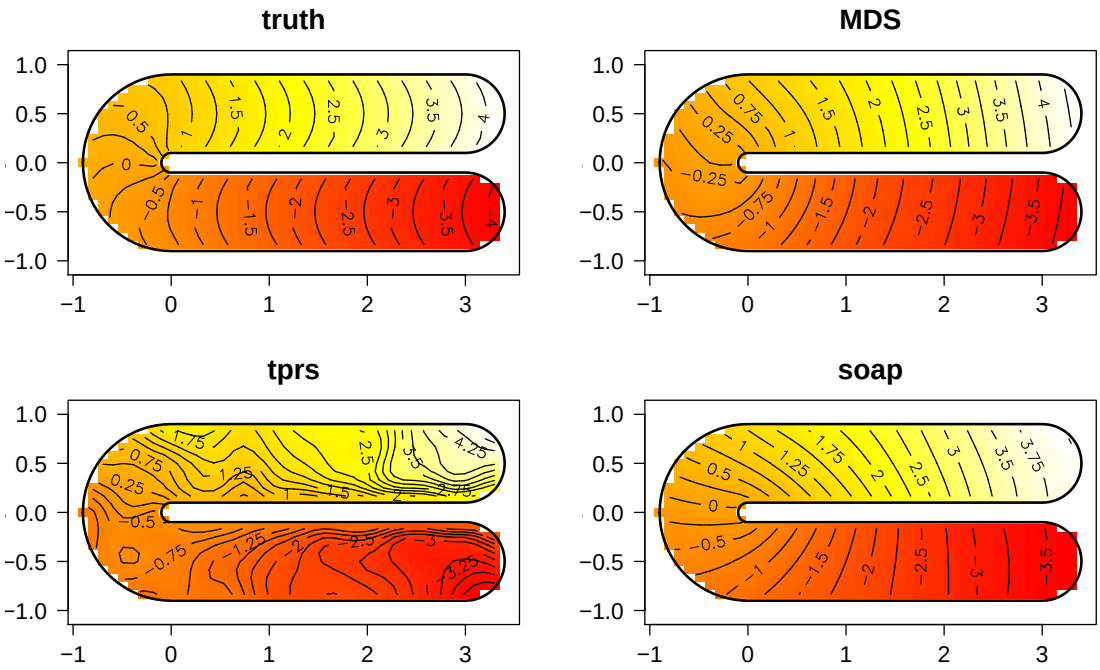

Figure 4-4: From top left clockwise: truth for the (modified) Ramsay horseshoe ("truth"), MDS+RS ("MDS"), the soap film smoother ("soap") and thin plate regression splines ("tprs") when 250 points sampled with noise level set to 1.

Just as when the Schwarz-Christoffel transform was used to morph the domain, it is interesting to see what has happened to the distribution of points in space. Figure 4-7 shows the effect of the transform on a regular grid of points (left) when they are projected into MDS space (right). The projection has also succeeded in parting the two arms of the horseshoe, reducing leakage (as can be seen in the realisations in figure 4-4).

4.4.2 Peninsula domain

The Ramsay horseshoe is an easy domain to smooth over since it is clear that a transformation should be parting the two arms of the domain. In practice however, which parts of the domain should be separated the most may not be so clear cut. For this reason a more realistic, complex domain would provide a better insight into the efficacy of the method. The domain shown in figure 4-8 is a modification of the peninsula domain seen in section 3.3.2 with the three high density areas in the far left peninsula replaced with a single continuous gradient, making it a slightly easier domain. The domain was modified because it was found in the simulations in section 3.3.2 that it was difficult to verify visually that the

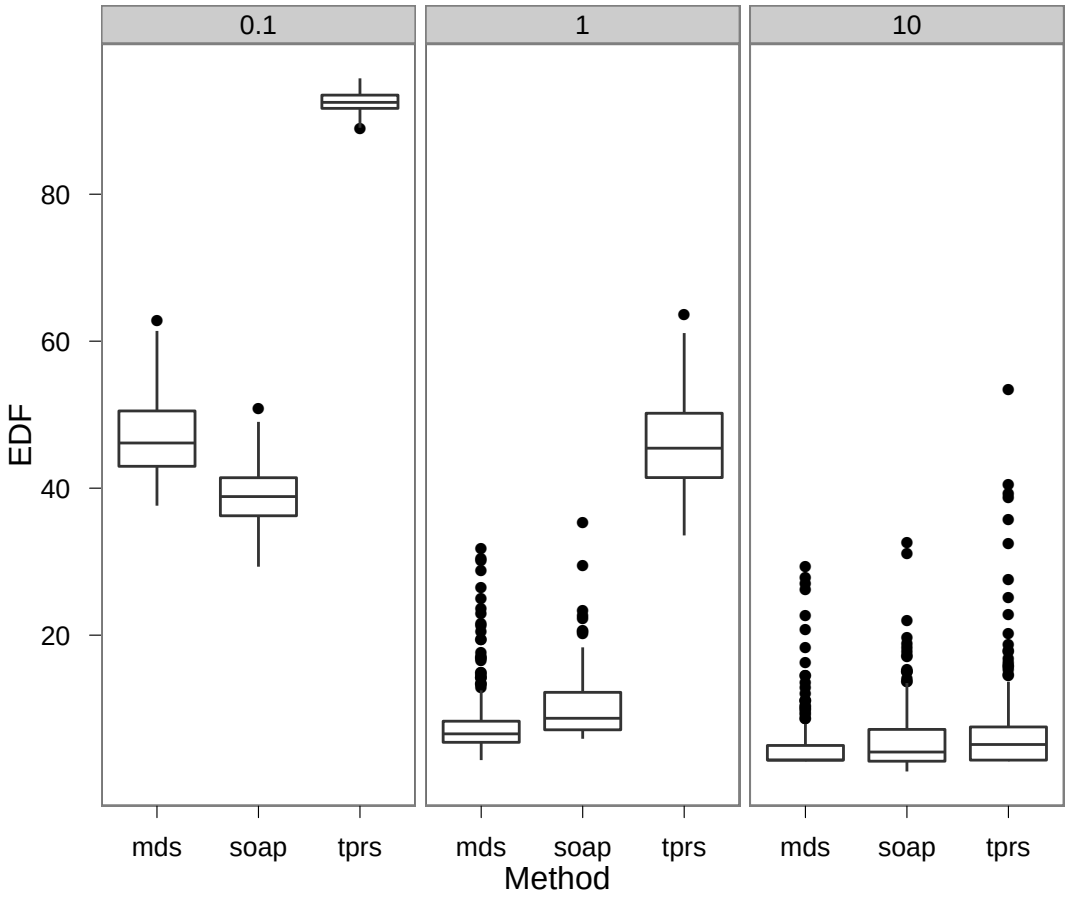

Figure 4-5: Boxplots of the EDF per realisation of the Ramsay horseshoe for MDS+RS ("mds"), the soap film smoother ("soap") and thin plate regression spline ("tprs") for noise levels 0.1, 1 and 10.

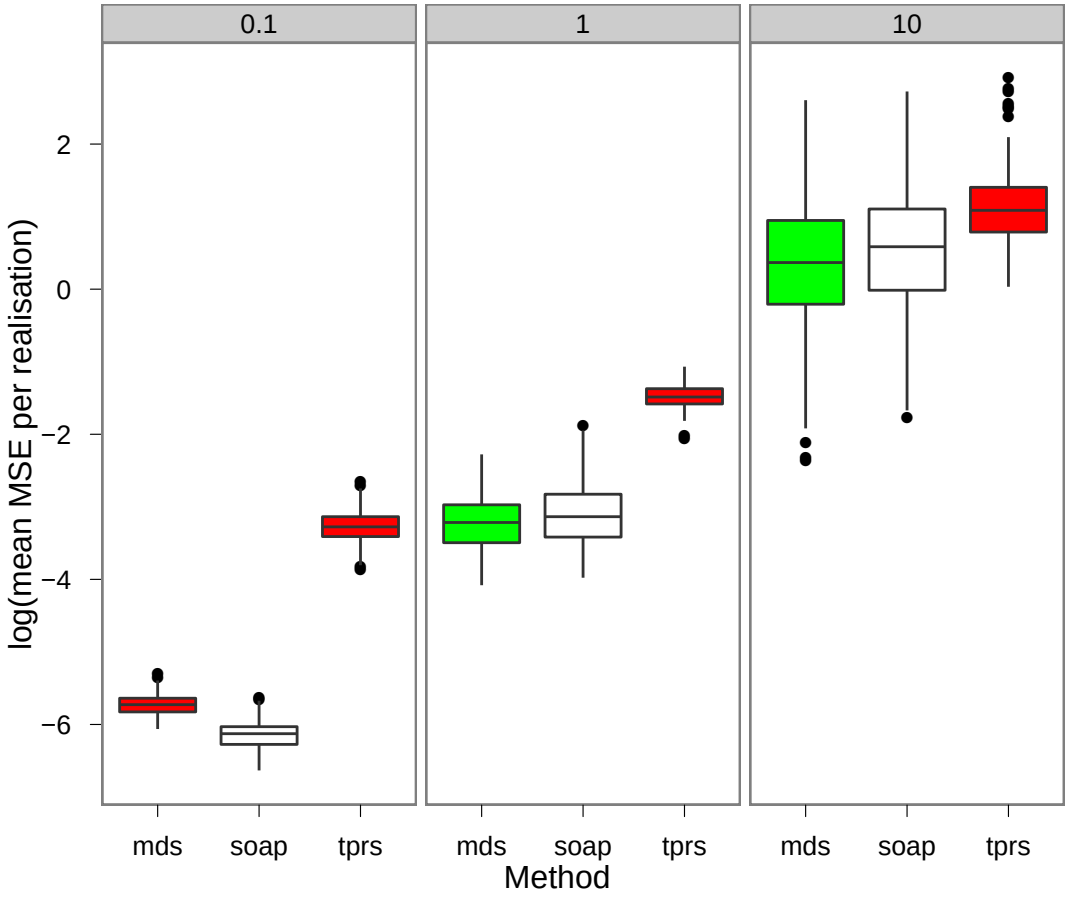

Figure 4-6: Boxplots of the logarithm of the MSE per realisation of the Ramsay horseshoe for MDS+RS ("mds"), the soap film smoother ("soap") and thin plate regression spline ("tprs") for noise levels 0.1, 1 and 10 (left to right). As previously a paired Wilcoxon signed rank test showed that MSEs for MDS+RS and thin plate regression spline were significantly different from the soap film smoother (at the 0.01 level) where the colours above indicate whether the MSE was better (green) or worse (red).

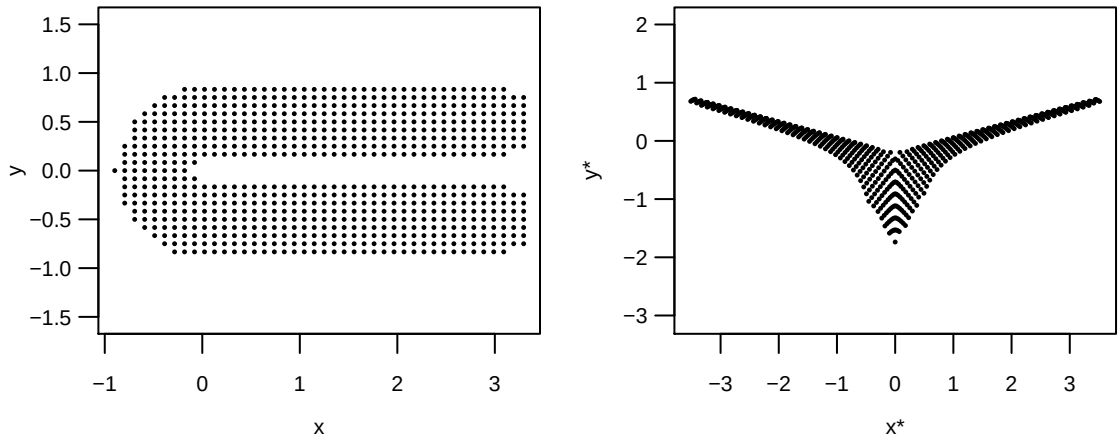

Figure 4-7: A regular grid over the Ramsay horseshoe (left) and its projection into MDS space (right).

smoother estimated the peaks in the peninsula correctly; a single gradient is easier to check by eye.

Setup

The simulations consisted of 200 realisations of 250 samples from the surface in figure 4-8. Normal noise was added at three levels 0.35, 0.9, and 1.55 (corresponding to signal-to-noise ratios (SNRs) of 0.95, 0.75 and 0.5, respectively. SNRs were calculated as the mean squared correlation between true function value and the truth with error added). Mean squared error over 1253 prediction points (which included the sample points) was calculated and recorded, along with EDF for each model. The models fitted were:

1. *Thin plate regression spline*: bivariate thin plate regression spline with maximum basis size 100.

2. *Soap film smoother*: cyclic spline on boundary of basis size 60, 109 internal knots evenly spaced on a grid over the domain.

3. *MDS+RS*: after transform a bivariate thin plate regression spline with maximum basis size 100. The initial MDS grid was 10×10 (48 points were inside the domain).

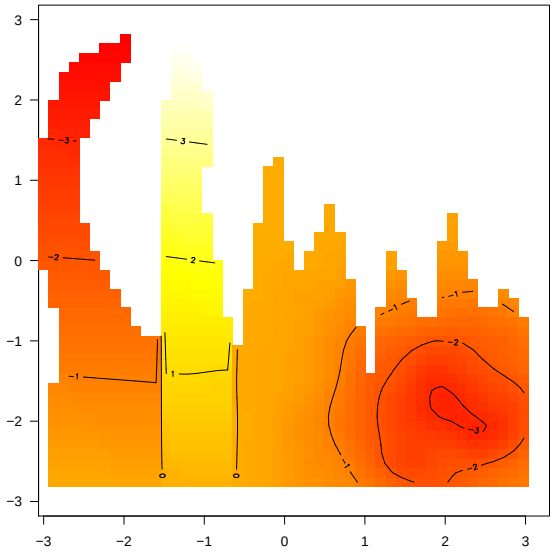

Figure 4-8: True function for the domain with multiple peninsulae.

Results

Looking at a typical realisation in figure 4-9 (noise level = 0.9, SNR = 0.75, sample size 250), the thin plate regression spline shows signs of leakage across the two large peninsulae, whereas MDS+RS and the soap film do not. The thin plate regression spline does, however, reproduce the peak in the lower right much more faithfully, the other two smoothing over it. In this realisation, MDS+RS deals with the values inside the peninsula a little better than the soap film smoother (the contour lines are more similar to those in the true function). On the other hand, the soap film captures the shape of the lower right peak slightly more accurately.

Figures 4-10 and 4-11 show boxplots of the MSE and EDF for the models above. The soap film smoother consistently has a statistically significantly lower MSE (at the 0.01 level). The soap film also tends to fit simpler models than the other two approaches except at the highest noise level.

As with the Ramsay horseshoe, it is interesting to see what the projection into MDS space has done to the distribution of the points in the domain. Figure 4-12 shows points in the domain in Euclidean space and MDS space. There appears to be some high concentrations of points in the far left peninsula and in the right side in the MDS space. This is due to the projection of the points into 2-dimensional space, which can be easily seen in figure 4-13 where the points have been projected into 3-dimensional space. The 3-dimensional projection also shows that there is

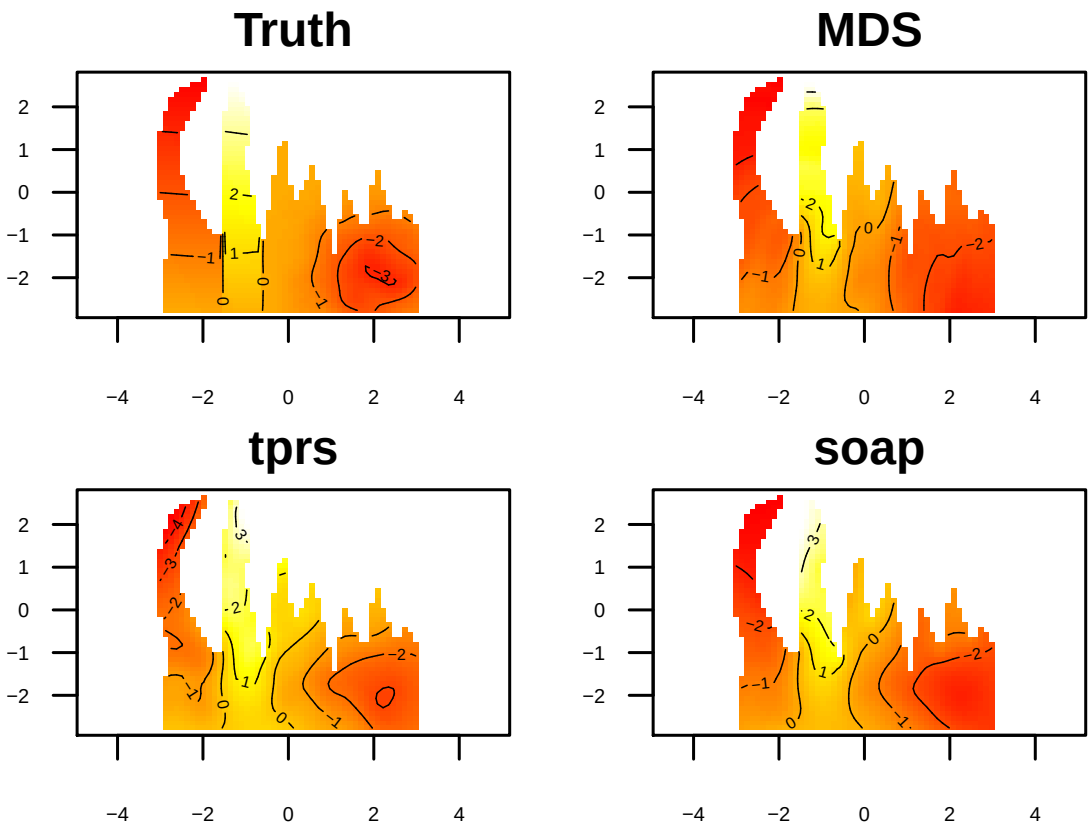

Figure 4-9: A typical realisation of fits from the multiple peninsulae domain when the noise level was set to 0.9 (SNR = 0.75) and the sample size was 250. The prediction grid was of size 1253. Clockwise from top left: the true function, prediction from: MDS projection smoothed with thin plate regression spline, the soap film smoother and thin plate regression spline.

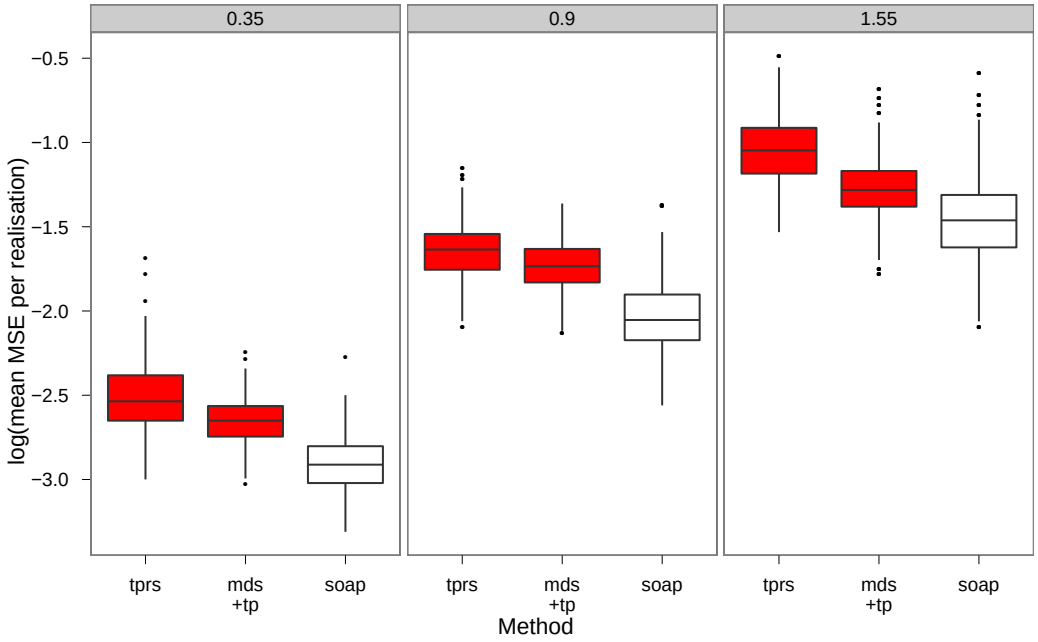

Figure 4-10: Boxplots of the logarithm of the MSE per realisation of the peninsula domain for the MDS approach ("mds+tp"), soap film smoother ("soap") and thin plate regression spline ("tprs") for noise levels 0.35, 0.9 and 1.55. A paired Wilcoxon signed rank test shows that the MSEs for the two other models were significantly different from the soap film smoother (and worse) at the 0.01 level.

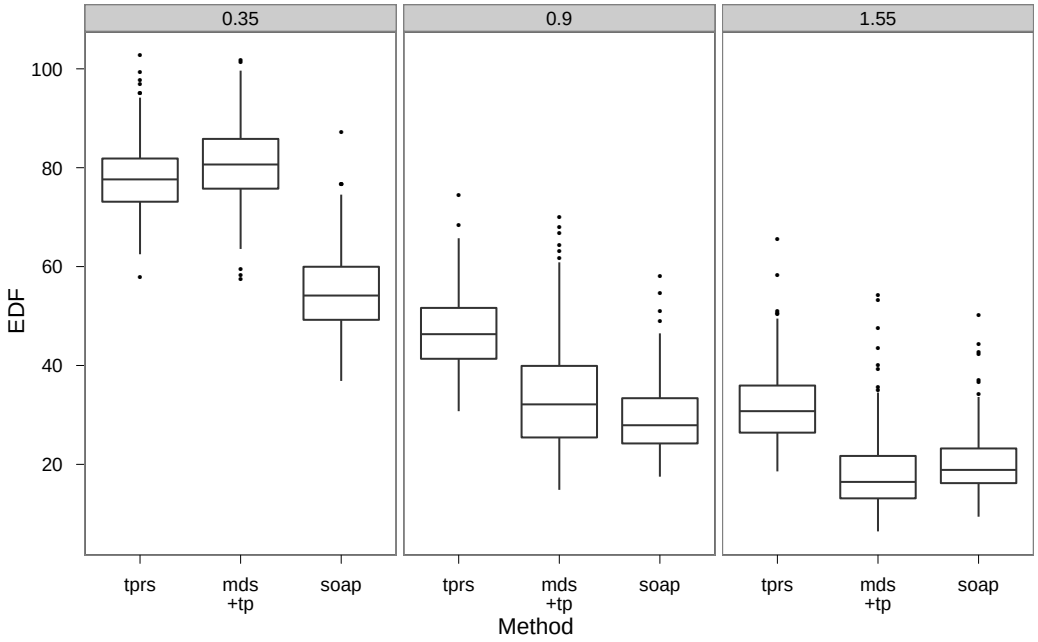

Figure 4-11: Boxplots of the EDF for each model per realisation of the peninsula domain for the MDS approach("mds+tp"), soap film smoother ("soap") and thin plate regression spline ("tprs") for noise levels 0.35, 0.9 and 1.55.

separation between the smaller peninsulae in higher dimensions that cannot be seen in the 2-dimensional projection.

This high point density in the right side of the MDS space could be the reason for the poor reproduction of the function in that region seen in figure 4-9. There appears to have been a severe breakdown in isotropy in this part of the domain (and in the left peninsula), which the thin plate regression spline does not handle well. This must be accounted for in the smooth if accurate models are to be built.

4.4.3 Areas for improvement

From this set of simulations areas for improvement to MDS+RS can be seen. First, the above problem of the accuracy of the model (in terms of faithfully reproducing the function) needs to be addressed. Second, the calculation of the within-area distances by the algorithm given in section 4.3 has considerable computational cost, even in comparison to the soap film smoother basis setup (see also section 2.4). Table 4.1 shows the average timings for running MDS+RS, thin plate regression spline and soap film smoothers over the peninsula domain.

In order to be useful to practitioners MDS+RS must perform at least as well

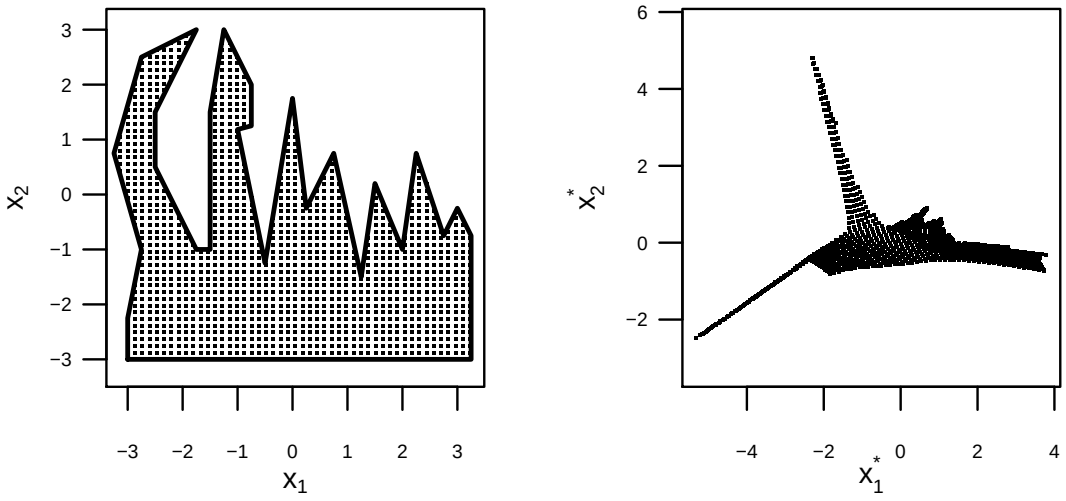

Figure 4-12: A regular grid over the peninulae domain (left) and its projection into 2-dimensional MDS space (right).

	MDS+RS	Soap film	Thin plate spline
Fit	84.852	24.678	0.402
Prediction	155.400	29.395	0.125

Table 4.1: Average time (in seconds) to fit and predict on a realisation of the peninsula domain for the three models considered above. Times are averaged over 100 realisations. For each realisation a sample of size 250 was taken, then a 1253 values were predicted.

as the soap film smoother in terms of accuracy (i.e. low MSE) and preferably take comparable computational time for model fitting. The next two sections address these issues.

4.5 Making MDS+RS faster

Calculating MDS by Lanczos iteration

The R command used to perform the multidimensional scaling, `cmdscale`, uses the routine `eigen` in order to perform the requisite matrix eigen-decomposition. This routine will calculate a full eigen-decomposition of the matrix, even if only the first k eigenvalues and/or eigenvectors are required. Using Lanczos iteration, only the first k eigenvalues (in numeric or algebraic size order) will be calculated.

The Lanczos procedure works by iteratively building a symmetric $i \times i$ tridi-

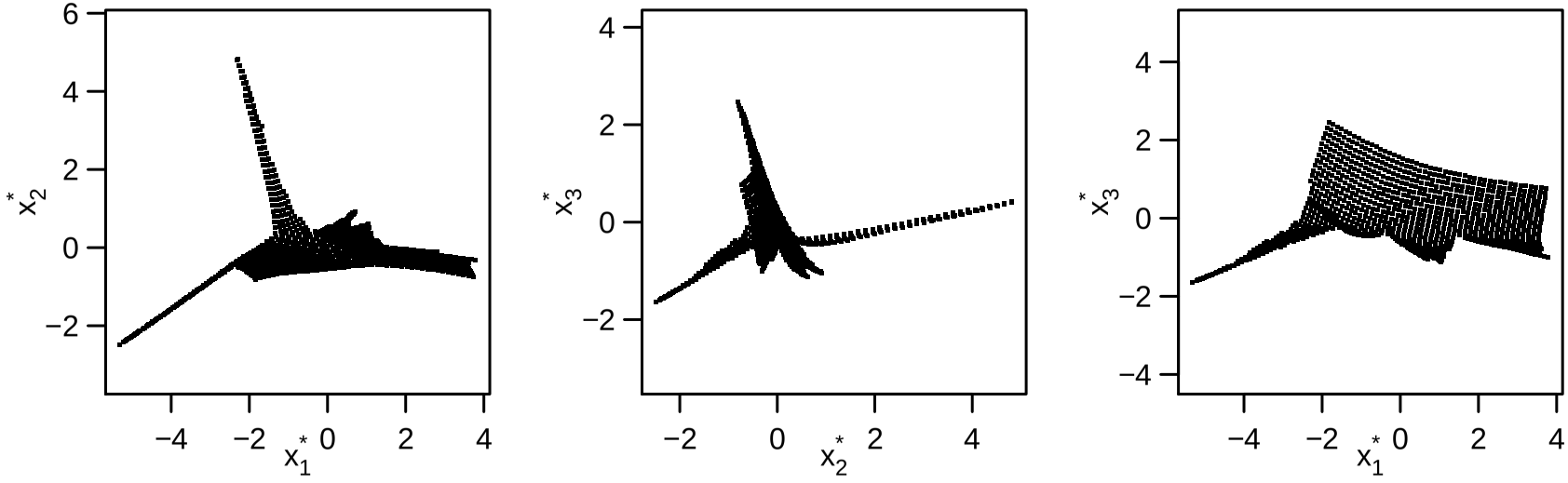

Figure 4-13: The peninsula domain projected into 3-dimensional MDS space. The plots show combinations of axes, note that the first panel is the same as the 2-dimensional projection shown in figure 4-12.

agonal matrix (at the i^{th} iteration) with eigenvalues which are approximately the same as the i largest eigenvalues of the original matrix. Further detail is given in Wood (2006, pp. 335-337).

The `igraph` library for R provides an interface to the C++ package `ARPACK++` which implements the Lanczos procedure. Replacing the `cmdscale` command with one that uses the `ARPACK++` interface provided by `igraph` will decrease the number of computations needed, thus making the calculation of the eigenvalues and vectors faster.

A quick benchmark shows that `ARPACK++` can compute the first two eigenvalues and vectors faster than just using `eigen` when the eigen-decomposition to be computed is of a large matrix. Generating a 1000 by 1000 symmetric matrix of normal random deviates with mean 0 and variance 1000, then performing an eigen-decomposition takes 1.68 seconds using `ARPACK++` and 3.26 seconds using `eigen` (averaged over 100 runs). This advantage drops once the matrix size is around 100 by 100 and the cost of calling the C++ code begins to dominate; in this case `ARPACK++` takes 0.037 seconds and `eigen` takes 0.034 (averaged over 100 runs). Given that the disadvantage is in the order of hundredths of a second and the advantage is a two-fold decrease in computational time, it makes sense to use the `ARPACK++` code in all cases.

Partial path calculation

It is often the case that the points for which the within-area distances are required form a grid (the initial MDS grid setup, or when prediction points need to be found). This grid setup can be exploited since there are many sets of paths that are rather similar. These paths may perhaps only differ in their final vertex. When this is the case much computational time is wasted calculating similar paths, it would be useful to exploit this problem and use it to increase the speed of the path calculation.

By appending the points between which the within-area distance is required to either end of one of a series of pre-calculated base paths, then optimizing this new path using the DELETE and ALTER steps as before, it is hoped that the computational time will be reduced. Using base paths will hopefully removes the expensive calculation in the middle of the path, where perhaps the bulk of the interactions with the boundary take place.

The algorithm is as follows, with notation and routines (INIT, DELETE, ALTER and ITER) identical to those in section 4.3:

1. Begin by creating a sparse grid of within the simple polygon Γ and calculate the (M, say) non-Euclidean within-area paths between all pairs of points in the grid, as in section 4.3. Store these paths as $\mathcal{P}_1, \ldots, \mathcal{P}_M$.

2. For each unique pairing of p_i and p_j in the full data set, calculate the path using one of the following:

   (a) Find a $\mathcal{P}_k$ such that the path between p_i and one end of $\mathcal{P}_k$ and p_j and the other end of $\mathcal{P}_k$ is Euclidean within Γ. Join p_i and p_j onto the appropriate ends of $\mathcal{P}_k$ and alternate between DELETE and ALTER steps until convergence.

   (b) If no $\mathcal{P}_k$ can be found calculate the path between p_i and p_j as in section 4.3.

Note that those paths between points in the sparse grid which are Euclidean are not stored since it is always at least as expensive to store, add and optimise those paths then calculating them from scratch. If the required path is Euclidean anyway, then retrieving a Euclidean path, adding in p_i and p_j, and then iterating over ALTER and DELETE steps to make it both the shortest and a Euclidean path will take longer than just creating a Euclidean path to begin with. If the path between p_i and p_j is non-Euclidean then the non-Euclidean part of the path must lie outside $\mathcal{P}_k$ (by definition, if $\mathcal{P}_k$ were Euclidean) and therefore will take the same number of operations to find the boundary crossing points and calculate the shortest path around the feature locally as it will to calculating the whole path from scratch.

Simulation - Lanczos and partial path calculation improvements

Taking both the Lanczos procedure and the partial path calculation together, a simulation was run to find the improvements in terms of computational time for the double peninsulae domain. Average time for both model fitting and prediction are given in table 4.2 for 100 realisations.

The differences between the first two columns are striking. The partial path calculation has dramatically reduced the computational time for the calculation of the entries of the distance matrix, making it faster than the soap film smoother for the model fitting, and reducing the prediction time to a third of its previous value. The soap film smoother's prediction is relatively low since the bulk of the computational time is spend on solving the PDEs necessary to find the basis

	MDS+RS	MDS+RS(pp)	Soap film	Thin plate
Fit	84.852	18.653	24.678	0.402
Prediction	155.400	53.511	29.395	0.125

Table 4.2: Average time (in seconds) to fit and predict on a realisation of the peninsula domain for the three models considered above. Times are averaged over 100 realisations using R's built-in `system.time` function. For each realisation a sample of size 250 was taken, then a 1253 values were predicted. For the MDS+RS columns pp indicates the cases where the partial paths were precalculated with the Lanczos procedure used to find the eigen-decomposition of the distance matrix, those not marked use the algorithm given in section 4.3 and did not use the Lanczos procedure.

functions (see section 2.2.3). The thin plate regression spline times are shown to give a comparison for the time actually taken to fit the model, the remaining time for MDS+RS is taken up by calculating the distances and performing the MDS.

Now that the computational time has been considerably reduced, the next task is to improve MDS+RS's fit to the data.

4.6 Using penalty adjustments to correct for squashing

Figures 4-12 and 4-13 show that the points in the peninsulae domain have been squashed together in some areas. In particular, this occurs in the peninsulae themselves. Since the motivation here is to improve estimates in the peninsulae, it would be extremely unfortunate if the transformation was detrimental to the smoother's performance in these areas (albeit in a different way). The squashing is similar to that seen in section 3.2.5 when the Schwarz-Christoffel transform was used to transform the domain, although not as severe as crowding seen, the squashing together of points can still be problematic (as was seen in section 3.3.2).

Uneven point density can cause problems for the smoother since the measure of smoothness will change. Squashing space together may make the data appear more variable than it in fact is (see figure 4-14 for a 1-dimensional example). Moving into two dimensions, an isotropic smooth seems like an unrealistic choice if the point densities are different in each direction. One might expect that the amount of smoothing required would be a function of the density of points at that location.

The higher MSEs shown in figure 4-10 might be explained by the change in density of points in the domain after it has been transformed into MDS space. In this case adjusting the thin plate spline penalty in order to take into account the change in point density in MDS space might improve performance.

An alternative approach would be to use a tensor product basis (of, for example, P-splines as was used in section 3.3) where there is a separate smoothing parameter for each dimension. However, thin plate regression splines have other appealing properties (in particular the low-rank approximation avoids knot placement issues, see section 1.1.3). A tensor product smooth may also still run into problems in each dimension with regard to the squashing of space. For these reasons, this section explores the utility of penalty adjustments based on point density for thin plate regression splines.

4.6.1 Overview

Wood (2000) shows that if one of the covariates of a thin plate spline (x_2 say) is transformed such that $x_2' = x_2/k$, then $f(x_1, x_2'k)$ will give the same fit as $f(x_1, x_2)$ (ie. the fit will be the same under the new coordinates), if the penalty is changed to:

$$\int\int_{\mathbb{R}^2} \left(\frac{\partial^2 f}{\partial x_1^2}\right)^2 + 2k\left(\frac{\partial^2 f}{\partial x_1 \partial x_2}\right)^2 + k^3\left(\frac{\partial^2 f}{\partial x_2^2}\right)^2 \mathrm{d}x_1\mathrm{d}x_2, \tag{4.9}$$

from the usual thin plate regression spline penalty (see section 1.1.3).

This approach will only handle a linear rescaling in one dimension; in the case of the MDS distortions, non-linear re-scalings in two dimensions must be addressed. To generalise (4.9) to the non-linear two-dimensional case a function of the coordinates must be found which gives the change in density for each point in the domain. Denote such a function $\mathcal{L}^*(x_1, x_2)$.

Including the function in the integral should allow the penalty to be adapted according to the degree to which space has been squashed, thus getting around the spatial inhomogeneity which appears to be affecting the model. The calculation of $\mathcal{L}^*(x_1, x_2)$ is elaborated on below.

Given that the function $\mathcal{L}^*(x_1, x_2)$ is known, the penalty is given as:

$$\int\int_{\Gamma'} \mathcal{L}^*(x_1, x_2)\left\{\left(\frac{\partial^2 f(x_1, x_2)}{\partial x_1^2}\right)^2 + 2\left(\frac{\partial^2 f(x_1, x_2)}{\partial x_1 \partial x_2}\right)^2 + \left(\frac{\partial^2 f(x_1, x_2)}{\partial x_2^2}\right)^2\right\}\mathrm{d}x_1\mathrm{d}x_2. \tag{4.10}$$

Note the change of integration domain from $\mathbb{R}^2$ to Γ', the transformed domain. This is because $\mathcal{L}^*(x_1, x_2)$ is not defined outside of Γ'.

Before moving straight to the 2-dimensional case, the method was tested in one dimension.

4.6.2 Penalty adjustments in one dimension

Before implementing this approach in full, a 1-dimensional test was run. The function:

$$g(x) = 0.2x^{11}\left\{10(1-x)\right\}^6 + 10(10x)^3(1-x)^{10}, \tag{4.11}$$

over the range [0,1] was used. The function was then split into four sections ($[0, 0.4]$, $(0.4, 0.6]$, $(0.6, 0.8]$ and $(0.8, 1]$) and in each of these sections x values were multiplied by 20,1,0.05 and 1, respectively (effectively contracting or expanding space). The function and its squashed form are shown in the top left and right panels (respectively) of figure 4-14. The function was evaluated at 100, equally spaced, points over the interval $[0, 1]$. These points were then used as the sample (with no noise added), a thin plate regression spline was fitted, and predictions made at same points yielding the blue lines in the lower two plots. The left plot shows the predictions in the transformed space and the right in the original space. The green line was produced using the same procedure but with a thin plate regression spline with the adjusted penalty. As can be seen from the plot, the fit has been improved greatly, but how is the adjustment calculated?

Penalty adjustment calculation

For the moment let us take f to be a 1-dimensional smooth function. The formula for the ij^{th} element of the penalty matrix given in (1.4) can be adapted in the following way:

$$\mathbf{S}_{ij} = \int_a^b \mathcal{L}^*(x)\frac{\partial^2 b_i(x)}{\partial x^2}\frac{\partial^2 b_j(x)}{\partial x^2}\mathrm{d}x = \int_a^b \mathcal{L}^*(x)b_i''(x)b_j''(x)\mathrm{d}x,$$

where b_j is the j^{th} basis function of f and letting a prime indicate differentiation with respect to x. The integral can then be approximated by the midpoint rule as:

$$\mathbf{S}_{ij} = \frac{b-a}{K}\sum_{k=1}^{K}\mathcal{L}^*(x_k)b_i''(x_k)b_j''(x_k) \quad \text{for} \quad x_k = a + \frac{(k-0.5)(b-a)}{K}, \tag{4.12}$$

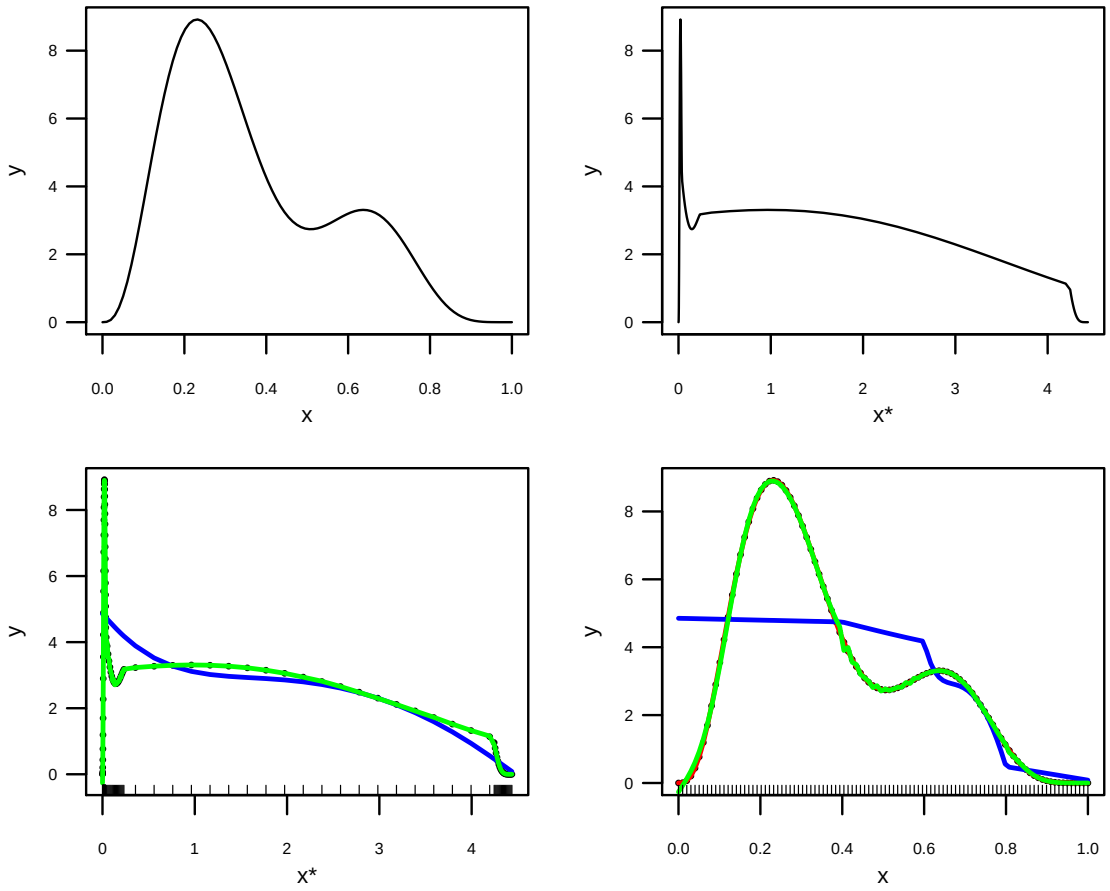

Figure 4-14: Using penalty adjustments to fit a regression spline to (4.11) after it has been squashed. The function in the top left is squashed to the form in the top right. The bottom left plot shows the fit from a thin plate regression spline (blue) and a thin plate regression spline with adjusted penalty (green) in the transformed space. The bottom right shows the same fit in the untransformed space. Clearly, the penalty adjustment improves the fit.

for $k = 1, \ldots, K$. Second derivatives are evaluated by finite differences in the usual manner:

$$b_i''(x) = \frac{b_i(x + 2\epsilon) - 2b_i(x + \epsilon) + b_i(x)}{\epsilon^2}. \tag{4.13}$$

For the sake of efficiency, a $K \times J$ matrix, $\mathbf{D}$, is calculated with kj^{th} element:

$$\mathbf{D}_{kj} = \sqrt{\mathcal{L}^*(x_k)} b_j''(x_k), \tag{4.14}$$

for x_k as above. Then $\mathbf{S}$ may be calculated as:

$$\mathbf{S} = \frac{b - a}{K} \mathbf{D}^{\mathrm{T}} \mathbf{D}.$$

Note that in previous chapters, the penalty was computed analytically, rather than numerically in `mgcv` so these calculations were unnecessary. However, since a different penalty is evaluated here, it must be calculated numerically.

In this example $\mathcal{L}^*(x)$ was simply calculated using the inverse of the cube of the (known) factor by which the relevant part of the domain (given above) was squashed.

Checking that the adjustment works

Figure 4-14 shows that the adjustment faithfully reproduces $g(x)$ for the zero error case, fitting a much more sensible model than the standard thin plate regression spline. To check that this is true more generally, the smoothing parameter (λ) was specified (rather than being automatically selected) so that the models with modified and unmodified penalties would have the same EDF. Figure 4-15 shows such an experiment. Using (4.11) and adding standard normal noise (multiplied by 0.4), the smoothing parameter was set so that the EDF would be 71, 19 and 42 (working down the diagram). The plots show that the adjustment deviates from truth at most as badly as the vanilla thin plate regression spline but overall corrects some of the departures from the truth, even in presence of error when the model flexibility is restricted.

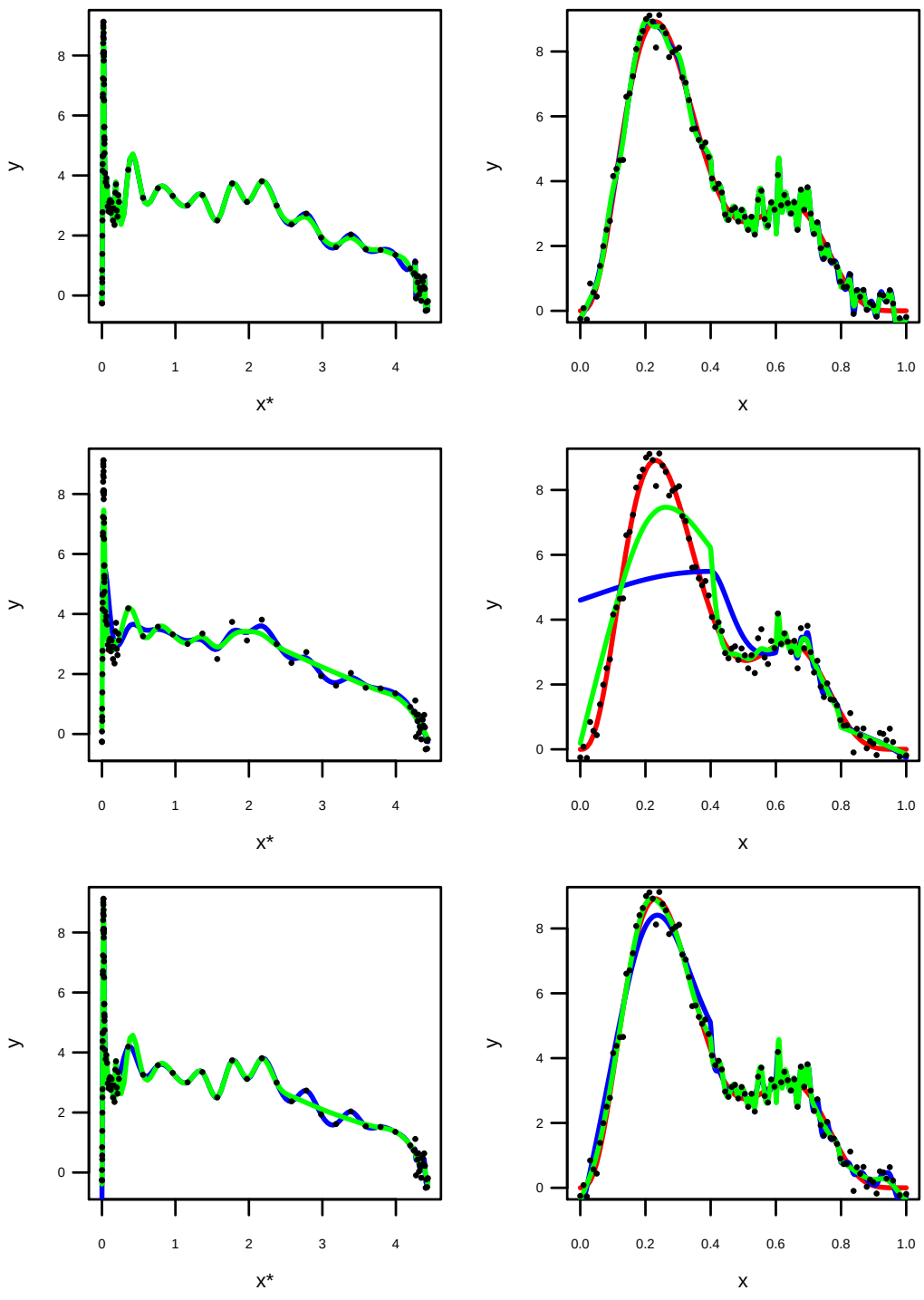

Figure 4-15: Predictions in transformed and untransformed (left and right columns respectively) for thin plate regression spline (blue line) and penalty adjusted thin plate regression spline (green line) fits to the function in (4.11) when the smoothing parameter was pre set to give EDF of 71, 19, and 42 (top to bottom).

4.6.3 Penalty adjustments in two dimensions

Using a similar procedures as for one dimension, the two dimensional case can be addressed. Again looking at the ij^{th} element of $\mathbf{S}$:

$$\mathbf{S}_{ij} = \int\int_{\Gamma'} \mathcal{L}^*(x_1, x_2) \left(\frac{\partial^2 b_i(x_1, x_2)}{\partial x_1^2} \frac{\partial^2 b_j(x_1, x_2)}{\partial x_1^2} + 2 \frac{\partial^2 b_i(x_1, x_2)}{\partial x_1 \partial x_2} \frac{\partial^2 b_j(x_1, x_2)}{\partial x_1 \partial x_2} + \frac{\partial^2 b_i(x_1, x_2)}{\partial x_2^2} \frac{\partial^2 b_j(x_1, x_2)}{\partial x_2^2} \right) \mathrm{d}x_1 \mathrm{d}x_2.$$

Matrices analogous to (4.14) can be constructed using the finite differences from (4.13) for differentials x_1 and x_2 individually and

$$\frac{\partial^2 b_i(x_1, x_2)}{\partial x_1 \partial x_2} = \frac{b_i(x_1 + \epsilon, x_2 + \epsilon) - b_i(x_1 + \epsilon, x_2) - b_i(x_1, x_2 + \epsilon) + b_i(x_1, x_2)}{\epsilon^2},$$

for the cross term. The matrices then take the form:

$$[\mathbf{D}_{x_1}]_{kj} = \sqrt{\mathcal{L}^*(x_{1k}, x_{2k})} \frac{\partial^2 b_j(x_{1k}, x_{2k})}{\partial x_1^2},$$

$$[\mathbf{D}_{x_2}]_{kj} = \sqrt{\mathcal{L}^*(x_{1k}, x_{2k})} \frac{\partial^2 b_j(x_{1k}, x_{2k})}{\partial x_2^2},$$

$$[\mathbf{D}_{x_1 x_2}]_{kj} = \sqrt{\mathcal{L}^*(x_{1k}, x_{2k})} \frac{\partial^2 b_j(x_{1k}, x_{2k})}{\partial x_1 \partial x_2}.$$

So $\mathbf{S}$ may then be written as:

$$\mathbf{S} = \mathbf{D}_{x_1}^{\mathrm{T}} \mathbf{D}_{x_1} + \mathbf{D}_{x_1 x_2}^{\mathrm{T}} \mathbf{D}_{x_1 x_2} + \mathbf{D}_{x_2}^{\mathrm{T}} \mathbf{D}_{x_2}.$$

Where the partial derivative evaluation points (x_{1k} and x_{2k}) now form a grid for the integration to be calculated. First defining x_{1k} and x_{2k} analogously to (4.12):

$$x_{1k} = a_{x_1} + \frac{(k - 0.5)(b_{x_1} - a_{x_1})}{K}, \quad x_{2k} = a_{x_2} + \frac{(k - 0.5)(b_{x_2} - a_{x_2})}{K},$$

for $k = 1, \ldots, K$. The integration grid may then be constructed as points in the x_1 direction:

$$\{x_{11}, x_{11}, x_{11}, \ldots, x_{12}, x_{12}, x_{12}, \ldots, x_{1K}, x_{1K}, x_{1K}\},$$

and x_2 direction:

$$\{x_{21}, x_{22}, x_{23}, \ldots, x_{2K}, x_{21}, x_{22}, x_{23}, \ldots, x_{2K}, \ldots\}.$$

Finally, those (x_{1k}, x_{2k}) that do not lie inside the boundary in MDS space are removed leaving only those points that lie inside (so that the integration is performed over Γ').

Finding $\mathcal{L}^*$

Up to this point it has been assumed that the factors by which space has been stretched are known. In practice, this is not the case and they dependent on the transformation. A simple, heuristic way to calculate $\mathcal{L}^*(x_1, x_2)$ is to think of the degree to which space has been stretched or squashed as a change in the density of the points in space. By estimating the density of the points over a grid in MDS space (by simply counting the number of points from a grid in the data space are mapped to a particular square in a grid in the MDS space), an approximation to the density change over the whole domain can be found.

The general idea is to make $\mathcal{L}^*(x_1, x_2)$ a function of the change in density of points caused by projecting the original space into MDS space. Assuming that the density in the untransformed space is 1 everywhere, it is only necessary to calculate the density in MDS space. Having calculated the point density in MDS space, $\mathcal{L}^*(x_1, x_2)$ is just some function of this density.

Although this method of approximating the density change is somewhat ad hoc, it should be able to highlight the large-scale changes in density which are causing the most problems when smoothing in MDS space. The point maps in figure 4-12 show that the density change in density is relatively smooth and does not have any sudden jumps, so a relatively coarse approximation to the density should suffice.

In order to find calculate the point density in MDS space, the following steps are performed:

1. A grid in the original space is mapped into the MDS space. Given how computationally demanding using a dense grid would be, a sparse grid was used and then interpolated. This consisted of taking 10 equally spaced points on each side of the square in the sparse grid and drawing lines between points on opposing sides. Extra points were then added where the lines crossed (along with those points lying on the boundary of the square itself).

2. The interpolated points were then used to estimate the overall point density in MDS space by simply counting the number of points there were in each of a set of squares made from the integration grid. The count per cell is a function of location and is denoted $\mathcal{L}(x_1, x_2)$. An example is shown in the bottom right plot of figure 4-16 for the double peninsulae domain.

3. Then define $\mathcal{L}^*(x, y)$ as the function

$$\mathcal{L}^*(x_1, x_2) = \frac{1}{\{\mathcal{L}(x_1, x_2) + 1\}^{3/2}}.$$

   Adding one to the denominator avoids division by zero when evaluating $\mathcal{L}^*(x_1, x_2)$. The fact that $\mathcal{L}^*(x_1, x_2)$ is a piecewise function should not be too worrying since the aim here is to address the broader problems with the change in spatial density, not the fine-gained details.

Note that the power is now $\frac{3}{2}$. This is since the contraction/expansion in each direction individually is not known, but rather the overall change. As such $\frac{3}{2}$ is used rather than the cubic on x and y and unitary on the cross term. Several other options were also tested (including changing the power and removing $+1$ in the denominator) however it was found that the above formulation provided the best results and so only those results are shown here.

Checking that the adjustment works

In order to make sure that the adjustment works in both the known and unknown contraction/expansion case, a small simulation was run. In this case a surface consisting of two bivariate normal distributions (mean vectors $(0, -0.5)$ and $(0, 0.5)$, covariance matrix diagonal entries $(0.2, 0.1)$) were sampled from (sample size 300) and then noise added from a normal$(0, 0.05)$ distribution. The surface was then divided into its four constituent quadrants about the origin and squashed according to the following factors (in (x_1, x_2) pairs, in order top left, top right, bottom left, bottom right): ((0.3,5), (1,5), (0.3,1), (1,1)). This is equivalent to what happened in the 1-dimensional case above, but per-dimension (so for x_1, the values are replaced by $x_1/0.3$ and for x_2 by $x_2/5$ and so on).

The samples were then used to fit a standard thin plate regression spline model, thin plate regression spline with adjusted penalty (with the factors above used as the values of $\mathcal{L}^*(x_1, x_2)$) and a thin plate regression spline with $\mathcal{L}^*(x_1, x_2)$ estimated from the density of the grid once it was transformed into MDS space.

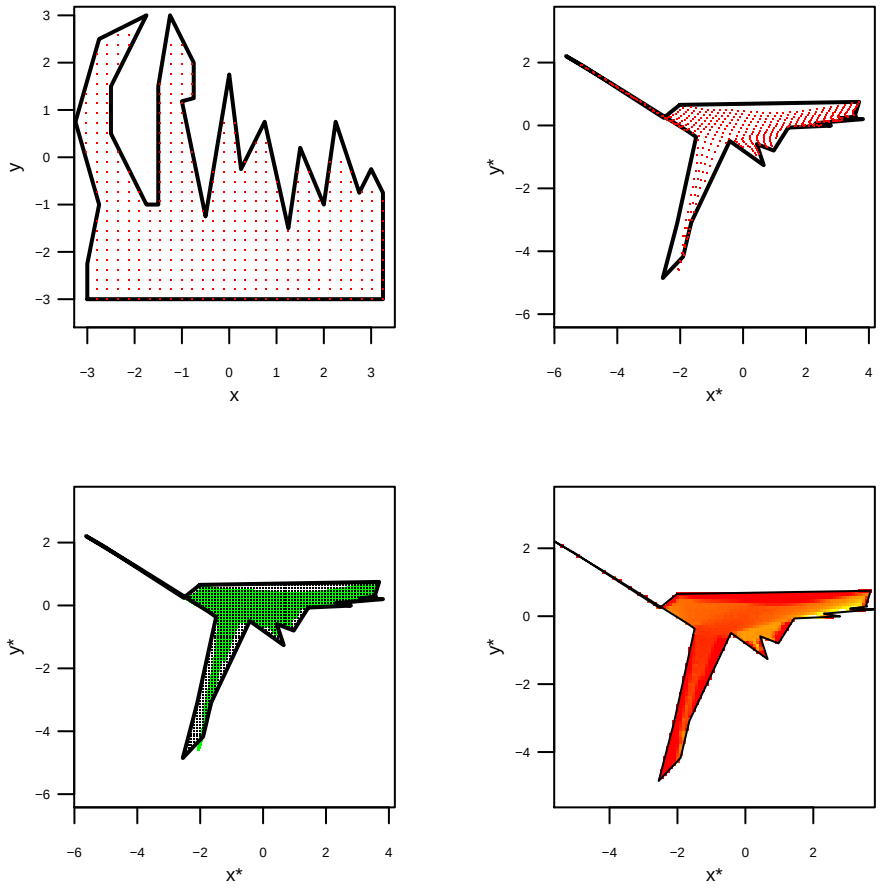

Figure 4-16: The grids used to calculate $\mathcal{L}^*(x_1, x_2)$ for the double peninsulae domain. The red grid in the top left figure is mapped to the red grid in the top right panel. The red points in the top right are then used as the basis for the interpolation in the bottom left. The number of green points in each of the squares made from the black points in the bottom left plot are used to calculate the spatial density in that square. The heat map in the bottom right shows the values of $\mathcal{L}(x_1, x_2)$ (i.e. the density of the green points), here red is low density, yellow is high. Note that some of the points lie outside of the boundary in the MDS projections. This is due to the boundary in MDS space being the straight line interpolant of the vertices of the boundary in the original space.

The mean squared error between the truth and prediction over a dense (50 by 50) grid was then calculated.

The simulation results show that there is a decrease in MSE when the expansion/contraction of the space is taken into account (MSEs were: 2.355, 2.30 and 2.305 for thin plate regression splines, known stretch and estimated stretch, respectively). Unfortunately this didn't offer the same visual improvement as the 1-dimensional case.

The next section puts the adjusted penalty approach to the test on the peninsulae domain seen previously and the Aral sea data set discussed in section 1.1.

4.6.4 Wider simulations and real data

Peninsulae domain

Using the same setup as in section 4.4.2, for each error level (0.35, 0.9, and 1.55), 200 realisations were generated. From these 250 samples were drawn to fit the model, predictions were made over a grid of 1253 points with MSE and EDF recorded per model for each simulation.

The models that were fitted were:

1. *tprs*: thin plate regression spline with basis size 140.
2. *mds+tp*: MDS+RS using a thin plate regression spline with basis size 140.
3. *mds 3D*: MDS+RS using a 3-dimensional thin plate regression spline with basis size 140. Here MDS was used to project the data into three dimensions rather than two.
4. *mds+adj*: MDS+RS using a thin plate regression spline with basis size 140, with penalty adjustments.
5. *soap*: soap film smoother using 109 internal knots evenly spaced on a grid over the domain, with boundary basis size 60.

The results from mds+adj are actually worse than those from just mds+tp in all but the lowest noise case. Figure 4-17 shows boxplots of these results.

A Wilcoxon signed rank test, matching pairs between realisations showed that there was a significant difference between the MSE of each model and the soap film smoother. As can be seen from figure 4-17, soap outperforms all of the other methods on this domain, with MDS 3D coming in second.

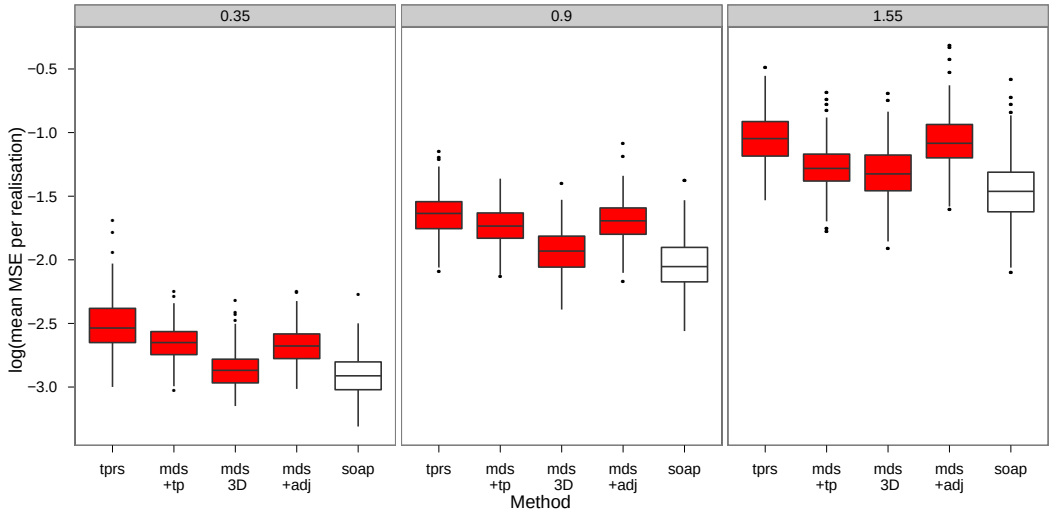

Figure 4-17: Logarithm of per realisation average mean squared error for the double peninsulae domain. Models are in groups of five for each error level (0.35,0.9,1.55). In all cases, a Wilcoxon signed rank test showed that MSEs for all models were significantly different from the soap film smoother (at the 0.01 level).

Adding just one more dimension improves the MSE more than using the complicated adjustment terms described above. Perhaps such an approach deserves further attention. The plots in figure 4-13 show that projecting into an additional dimension allows for further separation of both a large and small peninsulae, which should avoid leakage as well as perhaps help with the anisotropy (since in the extra dimension the far left peninsula has a greater width).

As would be expected, the method using the adjusted penalty had a lower EDF than the other methods aside from the vanilla thin plate regression splines (see figure 4-18), since it is penalizing more heavily. Interestingly the model using the 3-dimensional projection also has a low EDF, showing that there may be some utility in using such a method especially considering its relatively low MSE.

It seems that the penalty adjustments have not been useful for this domain. It is especially interesting to see that the addition of one dimension provides much better models than a penalty-based approach.

Finally, note that there were three realisations omitted (for all models) in figures 4-17 and 4-18. In these realisations the soap film smoother failed to fit the model due to knot placement. These can be safely removed as, in practice, the computer would inform the user that the knot placement was not appropriate and the knot layout could be altered.

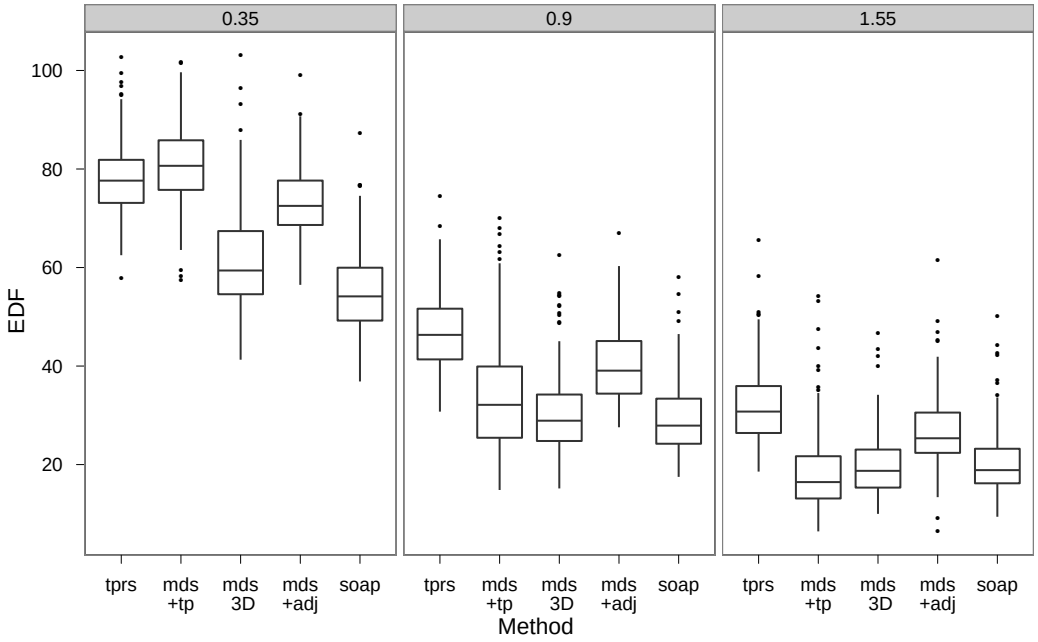

Figure 4-18: Per realisation EDFs for the double peninsulae domain. Models are in groups of five for each error level (0.35,0.9,1.55).

Aral sea

The Aral sea is located between Kazakhstan and Uzbekistan. It has been steadily shrinking since the Soviet government diverted the sea's two tributaries in order to irrigate the surrounding desert during the 1960s. The NASA SeaWifs satellite collected data on chlorophyll levels in the Aral sea (see also section 1.1 and Wood et al., 2008) over a series of 8 day observation periods from 1998 to 2002. The 496 data are averages of the 38^{th} observation period. Smooths were fitted to the spatial coordinates (Northings and Eastings) with the logarithm of chlorophyll concentration as the response (and assuming that the response was Gamma distributed as in Wood et al., 2008).

A thin plate regression spline, MDS+RS and soap film were all fitted to the data. In summary the setup for each model was:

1. *tprs*: thin plate regression splineswith basis size 70.
2. *soap*: the soap film smoother using a 12 by 12 grid of knots (74 were inside) and a boundary smooth with basis size 49.
3. *mds*: MDS+RS using thin plate regression splines with basis size 70.

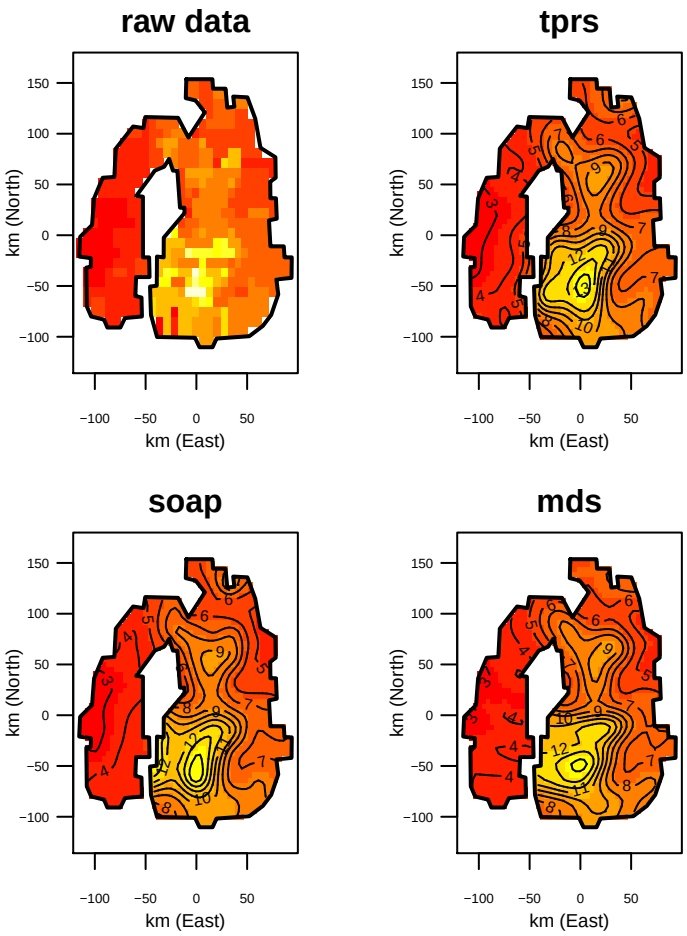

Figure 4-19: Raw data and predictions from the models fitted to the Aral sea chlorophyll data. Clockwise from top left: raw data, thin plate regression spline, soap film smoother, and MDS+RS.

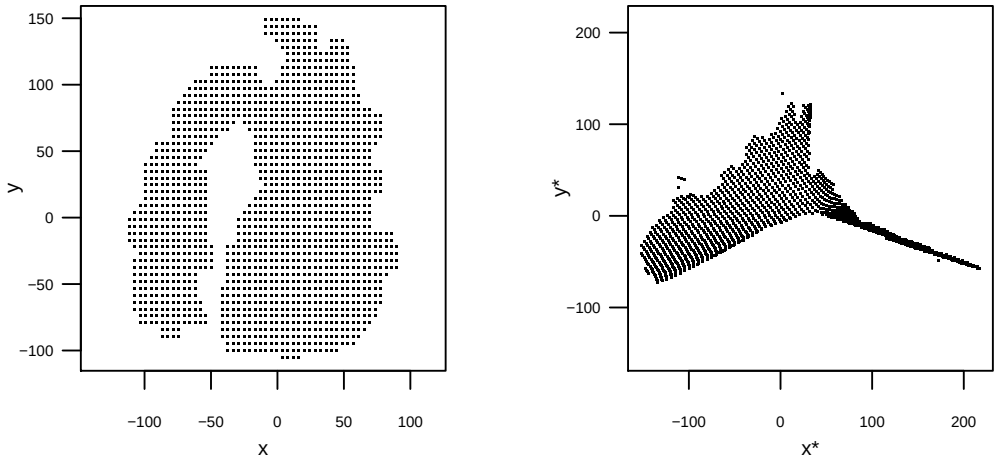

Figure 4-20: The prediction points for the Aral sea data set (left), with their projection into MDS space (right).

The models were then used to predict over a grid of 496 points to create the heat maps shown in figure 4-19. The fits are broadly similar, with the thin plate regression spline showing some signs of leakage around (-50,-50). Both MDS+RS and the soap film smoother do not have this problem. The contour lines for all of the models look roughly the same in the main part of the sea, but in the smaller lobe, MDS+RS is rather different from both the soap film smoother and thin plate regression spline.

Although the leakage is avoided, there appear to be some strange artefacts in the smooth. Ovals of higher chlorophyll appear in the smaller lobe when the MDS+RS is fit to the data, along with contours close to the far left of the smaller lobe. Looking at a point plot in MDS space (figure 4-20) reveals why this might be happening. As can be seen from the figure, the smaller lobe has been severely squashed which will clearly have an adverse effect on the smoother.

The MDS+RS with adjusted penalty was also used to fit the model, using the same basis as above. The predicted surface given by the model is shown in figure 4-21. Again, the same artefacts are clearly visible in the smaller lobe of the region.

As in the peninsula case above, a 3-dimensional MDS projection was also used and a thin plate regression spline fitted. The artefacts are less prominent and the surface looks much more like the one given by the soap film smoother. Again, the 3-dimensional projection shows much promise especially given the minimal extra

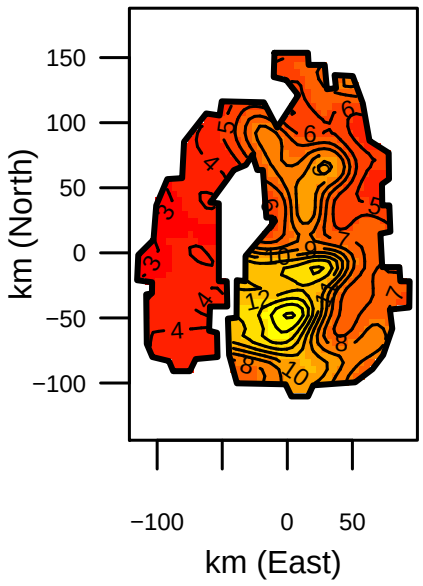

Figure 4-21: Predictions for the Aral sea using MDS+RS with adjusted penalty.

cost to running the additional model (if the within-area distances are already calculated, only the MDS projection needs to be calculated, and the thin plate regression spline fitted).

4.7 Problems with the methodology so far

At this point two outstanding issues must be addressed. The first is that, following the experiments above, there are issues with artefacts in the smooths produced by MDS+RS when a low dimensional projection is used. The effect of these artefacts is decreased when the projection is taken into higher dimensions, however, the artefacts are still present when a 3-dimensional projection is used and high dimensional smoothing can be rather tricky.

What follows is an explanation of why the artefacts occur, and how high dimensional smoothing can help. This section explores these two problems, explains what is going wrong and sets out what is needed for a solution.

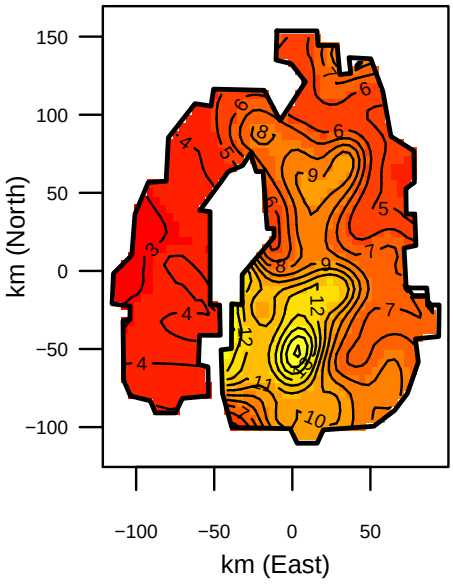

Figure 4-22: Predictions for the Aral sea using a 3-dimensional projection into MDS space with MDS+RS.

4.7.1 Why adjusting the penalty is not the solution

The simulations above show that the adjusted penalty scheme does not offer any advantage over using MDS+RS with the standard penalty. Before running the simulations, several different functions of the MDS point density ($\mathcal{L}^*$) were compared. All resulted in worse smooths (in MSE terms) than the function that was finally settled on. Investigating the good performance of the 3-D projection model goes some way to explaining why the penalty adjustment doesn't offer much improvement.

Looking at the plots of the prediction points in MDS space for the peninsulae domain (figure 4-12 and figure 4-13) it is easy to see that in two dimensions, the first peninsula has been squashed to a line.

By truncating $\tilde{\mathbf{X}}^*$ in the MDS procedure (section 4.2) the information in $\mathbf{D}$ relating to the width of the peninsula has been lost. Projecting into higher dimensions reduces the truncation (smaller eigenvalues of $\mathbf{D}$ and their corresponding eigenvectors are use to construct $\tilde{\mathbf{X}}^*$) and therefore more information about the relative positions of the points is included (this can be seen for the peninsula domain in figure 4-13). The adjusted penalty attempts to account for this squashing by allowing a more flexible model to be fit in areas where the point density is

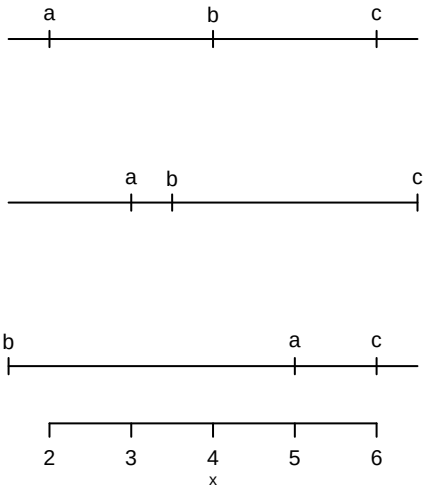

Figure 4-23: An illustration of how spatial mappings can squash points (middle line) and reorder them (bottom line) from their original configuration (top line).

higher. However, what is not taken into account is that some of the points in the peninsula are projected on top of (or on the wrong side of) one another. In other words, the 2-dimensional MDS projection makes the points lose their ordering.

As a simple, unidimensional example, take three points, $a, b,$ and c in the top line of figure 4-23. The projection could squash them in the way shown on the second line and then the penalty adjustments as described in Wood (2000) could be used to correct the squashing. However, with MDS the situation shown in the bottom line of figure 4-23 can occur (changing the order of $a, b,$ and c). This phenomena is mentioned in Hastie et al. (2001, pp. 572-573).

In the MDS projection we can see this happening for the peninsulae domain in figure 4-13. The projection takes a side-on view of the peninsula, making the points lose their ordering. In this case, the penalty adjustment can't save the model. If the ordering of the points is not guaranteed, then the smoother's job is potentially impossible, especially moving into two dimensions.

4.7.2 Why moving to higher dimensions is tricky

In the two cases above, moving into three dimensions allowed MDS+RS to more accurately reproduce the true function. The extra dimension allows for more information to be included in the projection giving "width" to parts of the domain that appear extremely thin in the 2-D projection.

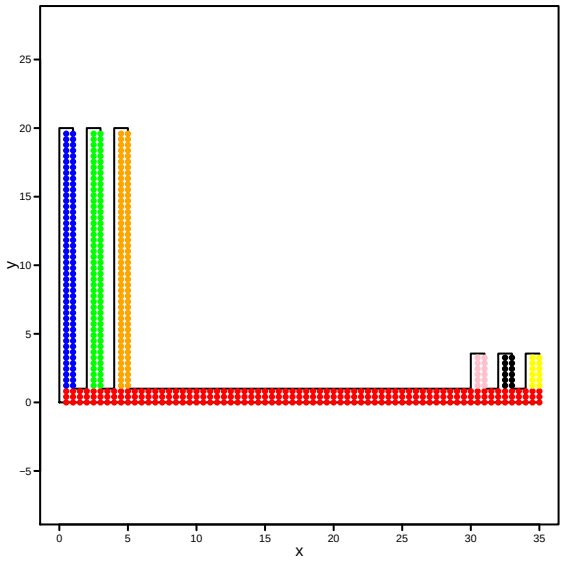

Figure 4-24: The "comb" domain from section 4.7.2.

It seems then that there is some milage in taking a 3-dimensional projection to solve this problem without the need for penalty adjustments. However, unfortunately, this is not the case in general. Figure 4-24 shows a long domain with three peninsulae at each end. The MDS projection of the domain in two dimensions gives the points in figure 4-25. There are only four peninsulae in this figure, not the six which were in the original. Using the colours in figure 4-24 and figure 4-25, one can see the separation of the larger peninsulae and that the smaller peninsulae have been positioned end-to-end. There could be interesting features in the response in these smaller peninsulae and thus leakage would be undesirable. This behaviour may well be worse than simply incurring leakage by using a standard smoother.

Adding an extra dimension could help. Taking a 3-dimensional projection, figure 4-26 is produced; however there is still no separation of the smaller peninsulae. Moving into four dimensions (figure 4-27) we begin to see separation in the smaller peninsulae. However, even in four dimensions the separation is not particularly large and leakage could still occur.

Although these plots are illustrative, a quantitative measure of how well the within-area distances are being approximated by the MDS projection is desirable. Given the eigen-decomposition attempts to minimize the spectral norm, this is the logical metric to use. The spectral norm may be calculated as the the square root of the largest eigenvalue of $\mathbf{D} - \mathbf{D}_{\mathrm{E}}$ where $\mathbf{D}$ is the matrix of within-area

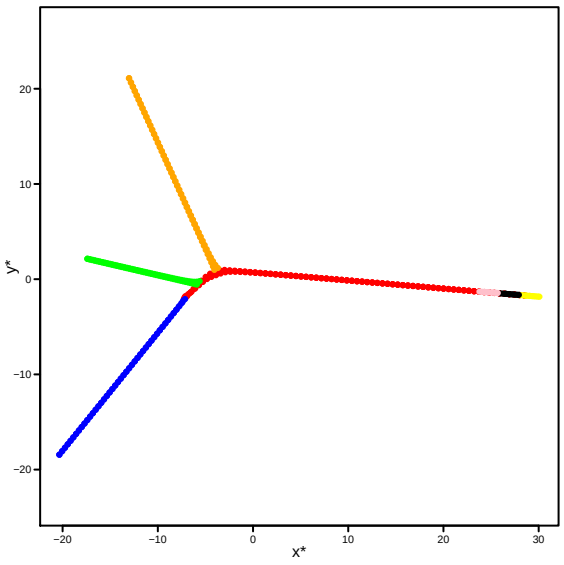

Figure 4-25: Two-dimensional MDS projection of the domain in figure 4-24, note that there are only four "legs" here not the six that should be there, as in figure 4-24.

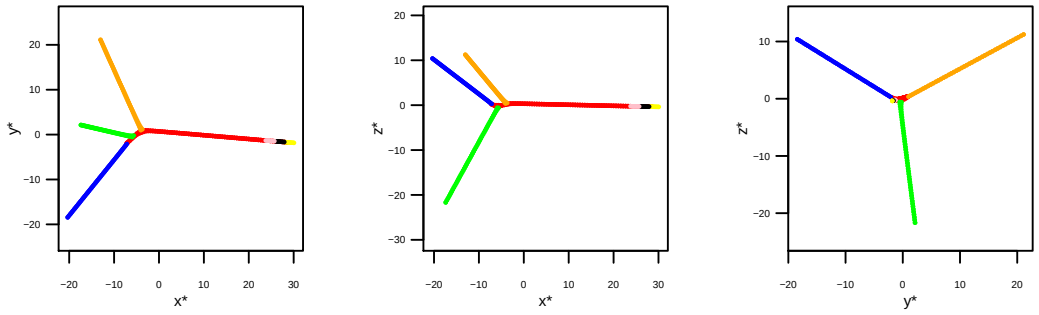

Figure 4-26: The MDS projection of the domain in figure 4-24 into three dimensions. Note that there is still no separation in for the smaller peninsulae.

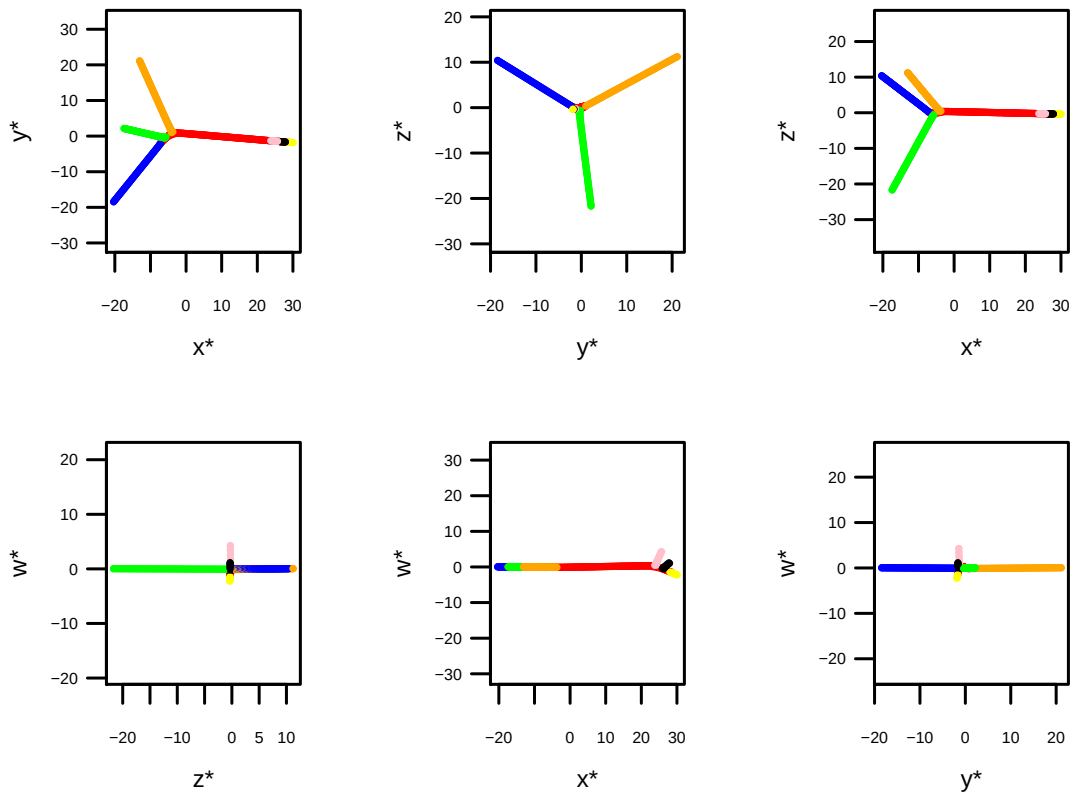

Figure 4-27: The MDS projection of the domain in figure 4-24 in four dimensions.

distances (as above) and $\mathbf{D}_E$ is the matrix of Euclidean distances in MDS space.

Looking at this measure for the domain considered in this section (the "comb"), peninsulae domain (from section 4.4.2) and the Aral sea as dimension of projection is increased yields some interesting results. These are summarized in figure 4-28. The plot indicates that there is some optimum number of dimensions to project into, such that adding a further dimension gives only a negligible decrease in the spectral norm.

The "optimal" projection dimension (in a spectral norm sense) is not common to all of the domains. For the peninsulae domain there is a large decrease (roughly halving each time) up to four dimensions, but the Aral sea appears to settle down after three. However, it's clear that the spectral norm is not the best guide for this given that for the "comb" domain, four dimensions appears to be optimal but figure 4-26 shows that this doesn't offer much separation in the peninsulae. The spectral norm doesn't offer a direct solution to the issue of dimension selection but it does at least offer the insight that there is some dimensional beyond which an increase in dimension only offers marginal returns, even if the best separation is after this point. Considering dimension selection separately from smoothing is sure to cause difficulties, since a point set that approximates the distances in $\mathbf{D}$ well does not give an guarantees about the quality of the resulting smooth. These dimension selection techniques only take into account variation in space, completely ignoring the effect that this has on the response. A method which takes into account how the projection affects the response will surely perform better than one which does not.

Moving into continually higher dimensions is appealing, but practically it is rather more taxing. As the dimension of the problem is increased, the order of the derivative in the spline penalty increases too (see section 1.1.2 and section 5.2.1). As this happens, the dimension of the nullspace of the penalty increases, meaning that both the number and complexity of the unpenalized functions in the model increases. An increasingly large space of unpenalized functions is certainly unappealing, but the next chapter will investigate how to work around such issues.

4.8 Conclusion

This chapter has investigated the utility of using a combination of multidimensional scaling and penalized regression splines to combat the phenomenon of

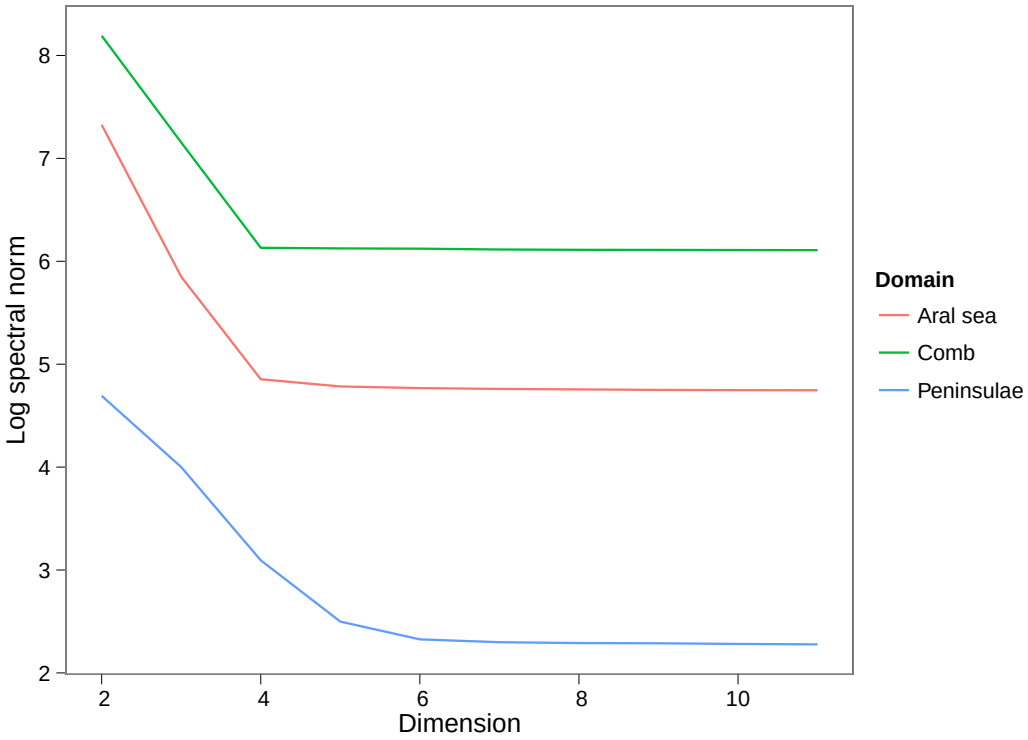

Figure 4-28: Logarithm of the spectral norm of $\mathbf{D} - \mathbf{D}_E$ versus dimension for each of the domains detailed in section 4.7.2.

leakage in spatial smoothing. This was with a view to MDS+RS being a less complex, faster, alternative to soap film smoothing, with an equally interesting motivating physical model.

The model certainly seems less complex. Provided that one knows about spline smoothing (which one would have to know to use the soap film smoother) and about multidimensional scaling (which is commonly taught at an undergraduate level outside of statistics, in biology, ecology, computer science, etc.), the method is relatively easy to get to grips with. No understanding of differential equations is required, nor is there any need to specify knots (unlike soap film smoothing). The physical model is still quite close to that which was outlined in section 1.4.3 part 4: the idea of morphing the domain into a shape which does not suffer from leakage.

In comparison to using the Schwarz-Christoffel transform, MDS+RS has the disadvantage of not being a functional mapping. Having a fixed functional form makes the Schwarz-Christoffel transform extremely fast since only a single function evaluation is needed to map a single point. In comparison, having to find the within-area distances is rather taxing computationally. However, the speed-ups presented in section 4.5 greatly enhance the utility of the method from a practical viewpoint. MDS+RS and soap film smoothing differ in how their computational time is divided. The main computational burden of the soap film smoother is in solving the PDEs needed to form the basis and from that calculating the penalty matrix (see table 4.2 and section 2.2.3), prediction is merely a case of evaluation. MDS+RS on the other hand, spends the bulk of its computational time on finding the within-area distances for both fitting and prediction (comparing the "thin plate" and "MDS+RS(pp)" columns of table 4.2). The advantage of this is that since the distance calculation is effectively a black box from the perspective of the smoothing, if a faster routine for distance calculation were to be used there would be no difference in the results (provided that the routine calculated the distances exactly) but the computational time to calculate the final smooth would be significantly reduced. This approach also opens up the possibility of other distance metrics being used to form the distance matrix (as will be seen in section 5.4).

Figures 4-13 and 4-20 show that MDS does achieve the kind of domain morphing that was sought in section 3.4. Unlike the Schwarz-Christoffel transform, the projection does not force the points to move into a fixed transformation domain and as such avoids some of the issues with point density.

Using MDS to re-arrange the points does not entirely alleviate the problem of squashing. Although there is no numerical crowding (see section 3.2.5), points may still have an uneven spatial distribution, which causes problems for isotropic smoothers like the thin plate spline. Section 4.6 sought to avoid the problems that occur when the space in which smoothing is to be performed by adjusting the penalty based on the density of the points. This did not work because of the confounding issue of point ordering (as discussed in section 4.7.1). The varying point density and ordering problems were due entirely to using a low-dimensional MDS projection. Ensuring that point ordering is maintained and that the point density remains roughly even can be controlled by the projection dimension. Higher dimensional projections will be investigated further in the next chapter.

As we have seen over the last two chapters, although domain transformation methods are appealing from a mathematical and physical point of view, in practice they are tricky to apply and can produce artefacts in the resulting smooths. These artefacts can be avoided by projecting into higher dimensions where the ordering of the points is not disrupted. The potential problem of using higher dimensional projections is that model parsimony is jeopardized by an increasingly complex nullspace. In the next chapter the use of high dimensional MDS projections will be investigated when an appropriate spline basis can be used to perform smoothing. If reliable high dimensional smoothing can be performed, then the only remaining issue is to resolve the issue of projection dimension selection, the next chapter will deal with this too.

Chapter 5

Generalized distance smoothing

5.1 Introduction

In the previous chapter multidimensional scaling was used to project the points within some geographical area into a new space in which smoothing can be performed without the problem of leakage. However, it has become clear that there is a trade-off involved in using this method: if the ordering of the points in space is to be maintained (section 4.7.1), a sufficiently high dimensional projection of the data must be obtained. Increasing the projection dimension causes the nullspace of the thin plate spline penalty (section 1.1.3) to become large, causing a large space of wiggly functions to be used without being penalized, leading to unreliable smoothing results (section 4.7.2). To resolve this issue, a smoother which can deal with high dimensional data must be used.

High dimensional smoothing has been approached in several different ways in the literature to date, some of these are detailed below.

1. Without straying too far from thin plate regression splines, one could consider using tensor products of splines of any basis (section 1.1.3). For example, marginal smooths of P-splines or cubic splines could be combined using a tensor product to create a multidimensional smooth in the requisite number of dimensions. In practice several problems arise with this approach. First is the problem of knot placement, which in high dimensions can become a massive computational burden, requiring some kind of knot selection. Second is that tensor product smooths are anisotropic so one smoothing parameter is used per dimension, aside from the additional computational burden. The point of the high dimensional projections used

is in part to combat anisotropy. Finally, although the marginal functions in a tensor product smooth can be simple, the combination of the functions can turn out to be rather complex (section 2.2.2).

2. Rather than using a basis function decomposition of the smoother, local regression could be used. Two popular local regression techniques are LOESS which smooths the data using a low-degree local polynomial (Cleveland, 1979, Cleveland and Devlin, 1988) or kernel smoothing by taking weighted averages of the response with weightings based on distance (e.g. Hastie et al., 2001, pp. 194-200). Local methods are, of course, local and as such rely on the data being relatively dense, which cannot be guaranteed, especially in high dimensions (Hastie et al., 2001, p. 200).

   Integrating these methods into a GAM (which is desirable so models such as the one for the Italian data described in chapter 2 can be specified) requires that backfitting be used for the fitting prodecure. Backfitting consists of smoothing the residuals with respect to that component's covariate. Estimation of smoothing parameters along with the model coefficients is hard to integrate into a backfitting procedure (Wood, 2006, p. 213), making it less appealing than the approaches investigated so far.

3. Kriging could be used to smooth in high dimensions, however there are technical limitations on the number of dimensions that are possible. Ensuring that the semivariogram remains positive definite can be problematic (Boisvert et al., 2009). There have been several investigations into using MDS to obtain isotropic distances for semivariogram estimation (Curriero, 2006; Løland and Høst, 2003; Jensen et al., 2006) however there are still a number of outstanding issues with these approaches. Comparisons between kriging and the methods proposed here are covered in more detail in section 6.1.

All of the methods listed above have problems that thin plate regression splines do not suffer from. Thin plate splines do have the problem that the number of functions in nullspace of the penalty becomes far too large when smoothing is performed in high dimensions (see figure 5-1). Limiting the number of functions in the nullspace of the penalty would avoid the side effects of having complicated, unpenalized functions in the resulting smooths.

When originally proposing thin plate splines in his seminal 1977 paper, Duchon actually describes a much more general set of interpolation methods; thin plate

splines are just a particular example of these. This chapter illustrates how a more general version of the thin plate spline (henceforth referred to as *Duchon splines*) can be used for high dimensional smoothing whilst avoiding a large and complex penalty nullspace.

Duchon splines have been largely neglected in statistical smoothing literature with the exception of Girosi et al. (1995) which discusses the connections between neural networks and GAMs. Hastie et al. (2001, p. 168) also discuss Girosi's work briefly.

The next section goes into the technical detail of how Duchon splines work and how they can be used in the finite area smoothing case. Section 5.3 shows how this can be useful in the within-area distance case discussed in chapters 3 and 4. Section 5.4 expands the methodology to look at any problems where distances can be considered, taking the MDS projection of these distances and then smoothing in the projected space. Finally, section 5.5 concludes the chapter.

5.2 Using Duchon splines for reliable high dimensional smoothing

This section starts from the definition of the thin plate spline penalty and shows how one can motivate the more general Duchon spline penalty. The aim is to show how the penalty works and that the main differences between it and the thin plate spline penalty. A full technical exposition (from a mathematical rather than statistical point of view) is given in Duchon (1977). This section goes on to show how (once reliable high dimensional smoothing can be performed) the size of the MDS projection dimension can be selected.

5.2.1 From thin plate splines to Duchon splines

First re-iterating and expanding on the description of thin plate splines given in section 1.1.3, the thin plate spline penalty (as originally given in (1.5)) in d dimensions with derivative order m is :

$$J_{m,d} = \int \dots \int_{\mathbb{R}^d} \sum_{\nu_1+\dots+\nu_d=m} \frac{m!}{\nu_1! \dots \nu_d!} \left(\frac{\partial^m f(x_1, \dots, x_d)}{\partial x_1^{\nu_1} \dots \partial x_d^{\nu_d}} \right)^2 \mathrm{d}x_1 \dots \mathrm{d}x_d, \quad (5.1)$$

where the summation index generates all of the possible combinations of derivative orders such than their sum is still m (thereby finding all the correct cross-

terms for the derivatives). In order to ensure that f remains continuous, $2m > d$.

It can then be shown that the thin plate spline basis minimizes (5.1) and is given (originally in (1.6)) by:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \delta_i \eta_{m,d}(r_i) + \sum_{j=1}^{M} \alpha_j \phi_j(\mathbf{x}), \tag{5.2}$$

where $r_i = ||\mathbf{x} - \mathbf{x_i}||$.The first summation is a set of radial basis functions (i indexing the n data) and the second summation are a set of linearly independent polynomials of degree less than m. The terms in the second summation are unpenalized (since their m^{th} derivatives are zero). There are M of these polynomials lying in the nullspace of the penalty, where M is given by:

$$M = \binom{m+d-1}{d}. \tag{5.3}$$

In the cases presented so far, d (the MDS projection dimension) is known and m is dictated by d, since $2m > d$. M therefore increases very quickly with the number of dimensions; this is shown by the blue line in figure 5-1. As more basis functions are included in the nullspace, the more wiggly the functions are. A large number of increasing complex, global functions which are unpenalized pose a serious threat to the fitting of parsimonious models.

Starting from (5.1), the first step toward the more general Duchon penalty is to consider taking the Fourier transform of the derivatives before squaring and integrating them. The Fourier transform allows us to consider the derivatives as an infinite sum of frequencies; decomposing functions defined in space into their frequency domain representations. Mathematically, the Fourier transform of g, a function $\mathbf{x}$ (a d-vector), is defined as:

$$\mathfrak{F}g(\boldsymbol{\tau}) = \int \dots \int_{\mathbb{R}^d} e^{2\pi\sqrt{-1}\mathbf{x}^{\mathrm{T}}\boldsymbol{\tau}} g(\mathbf{x}) \mathrm{d}\mathbf{x}.$$

Here $\mathfrak{F}$ is an operator applied to g, so $\mathfrak{F}g$ may be considered as a function of $\boldsymbol{\tau}$ (a d-vector of continuous frequencies). More detail on Fourier transforms can be found in Bracewell (1986), Chu (2008) and Beerends et al. (2003).

Taking the Fourier transform of the derivatives in the penalty allows us to think of how the penalty is calculated in a different way. Rather than integrating the field of derivatives over space, the penalty is calculated from measuring the

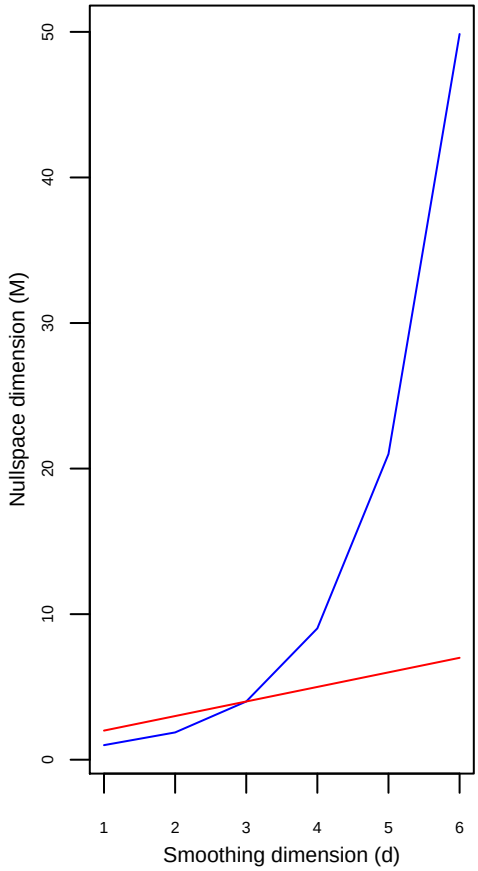

Figure 5-1: Relationship between smoothing dimension (d) and the nullspace dimension (M) when m (the derivative penalty order) is set to 2 for thin plate regression splines (blue) and Duchon splines (red). Note that as the nullspace dimension increases, the complexity of those functions in the nullspace increases too. For the thin plate splines a combination of the continuity condition that $2m > d$ and the form of M (see (5.3)) makes the size of the nullspace increase very quickly with smoothing dimension.

intensity of the different frequencies of the derivatives over the whole domain. Intuitively, the low frequency components of the derivatives of f are likely performing a similar task to those functions in the nullspace of the penalty, where as the more complicated, high frequency components are more complicated parts of the function, likely to be the parts of f that attempt to interpolate the data.

Taking the Fourier transform of the derivative terms in (5.1) yields the following penalty:

$$J_{m,d} = \int \dots \int_{\mathbb{R}^d} \sum_{\nu_1+\dots+\nu_d=m} \frac{m!}{\nu_1! \dots \nu_d!} \left(\mathfrak{F} \frac{\partial^m f}{\partial x_1^{\nu_1} \dots \partial x_d^{\nu_d}} (\boldsymbol{\tau}) \right)^2 \mathrm{d}\boldsymbol{\tau}. \tag{5.4}$$

The penalties (5.1) and (5.4) are in fact equivalent by Plancherel's theorem (Vretblad, 2003, p. 180) in the sense that they evaluate to the same numerical value.

Since taking the Fourier transform of the derivatives has allowed us to think of the derivatives as made up of frequencies, it then follows to exploit this interpretation by introducing a weighting into the penalty:

$$\int \dots \int_{\mathbb{R}^d} w(\boldsymbol{\tau}) \sum_{\nu_1+\dots+\nu_d=m} \frac{m!}{\nu_1! \dots \nu_d!} \left(\mathfrak{F} \frac{\partial^m f}{\partial x_1^{\nu_1} \dots \partial x_d^{\nu_d}} (\boldsymbol{\tau}) \right)^2 \mathrm{d}\boldsymbol{\tau}, \tag{5.5}$$

the function w can then be used to pick out particularly high frequencies and penalize those more than the lower frequency ones. Setting $w(\boldsymbol{\tau}) = 1, \forall \boldsymbol{\tau}$ recovers the usual thin plate spline penalty in (5.1).

Duchon suggests the use of $w(\boldsymbol{\tau}) = |\boldsymbol{\tau}|^{2s}$ for some choice of s:

$$\breve{J}_{m,d} = \int \dots \int_{\mathbb{R}^d} |\boldsymbol{\tau}|^{2s} \sum_{\nu_1+\dots+\nu_d=m} \frac{m!}{\nu_1! \dots \nu_d!} \left(\mathfrak{F} \frac{\partial^m f}{\partial x_1^{\nu_1} \dots \partial x_d^{\nu_d}} (\boldsymbol{\tau}) \right)^2 \mathrm{d}\boldsymbol{\tau}. \tag{5.6}$$

s takes an integer value divided by two; increasing the value of s will penalize higher frequencies more (and setting $s = 0$ will give (5.1), where all frequencies are penalized equally). This allows some of the frequencies of the radial basis functions to do the job of the M linearly independent polynomials which were not included due to reduced nullspace size. This will still give a minimizer of broadly the same form as the thin plate spline functions in (5.2) but M will change, giving a reduced nullspace (in both size and complexity terms) while not sacrificing the continuity of f.

When $s > 0$ higher frequencies are penalized more than lower ones. In order to obtain smooth functions it is required that $m + s > d/2$ (this replaces the

condition $2m > d$). Using a value of $s > 0$ allows for high dimensional smoothing while still using lower-order penalties without yielding discontinuous functions. One can therefore think of s as a kind of "fudge factor" that allows the conditions on m and d to be relaxed. Given some fixed combination of m and d, an s can be found by simply calculating:

$$s > d/2 - m. \tag{5.7}$$

For the examples below, the smallest s which satisfies (5.7) is used with $m = 2$, so:

$$s = d/2 - 1. \tag{5.8}$$

The red line in figure 5-1 gives the number of functions that lie in the nullspace of penalty (5.6), i.e. the result of using (5.8) with (5.3). For plotting $m = 2$ so the derivative order is constant as the dimensionality increases, leading to a linear increase in nullspace size with the smoothing dimension.

Note that the eigen-decomposition technique for thin plate regression splines shown in section 1.1.3 can also be used for Duchon splines. This low-rank approximation is used throughout the rest of the chapter.

5.2.2 Duchon splines with MDS+RS : MDS+DS

With the addition of Duchon splines to the MDS+RS we are now in a position to project data into higher dimensions and smooth over the response without having to worry about the the size and complexity of the nullspace causing problems. The use of Duchon splines along with MDS+RS and the projection dimension selection techniques below will be known as MDS+DS (MultiDimensional Scaling with Duchon Splines) from here on.

5.2.3 Choosing MDS projection dimension

With Duchon's basis, it is now possible to smooth over any number of dimensions whilst using second order derivatives in the penalty, simply by picking s according to (5.7). This seems reasonable since, for the within-area distance cases considered here, the resulting MDS configurations look like 2-dimensional manifolds. Given some point, $\mathbf{x}$ say, points nearby $\mathbf{x}$ in the MDS projection will have had distances to $\mathbf{x}$ calculated using the Euclidean metric rather than the within-area

distance algorithm. This locally Euclidean property is what defines a manifold (Munkres, 2000, p. 225).

Picking the dimension of the MDS projection is now a concern since there is no reason to believe that simply going to higher and higher dimensions will yield better results. For reasons of model parsimony, it is preferable to use as low a projection dimension as possible.

Using the spectral norm

As discussed in section 4.7.2, as higher dimensional projections are used, a larger proportion of the variation in the distance matrix is explained. It therefore seems reasonable to base the choice of dimension on the proportion of the variation explained in the initial grid. However, what proportion should be used? 80%? 90%? 99%? There is no reason *a priori* to choose any one of these over the others. Setting the proportion of variation to be explained without thought of the domain in question is surely a bad idea, since what works for one domain may well be a disaster for another (see figure 4-28). Such an approach also does not take into account the response values, thus discarding useful information.

Using scores

Since fitting a smooth of Duchon splines and using MDS to project the distance matrix is relatively cheap compared to the cost of finding the within-area distances, it is not problematic to fit many models (varying the projection dimension) provided no new distances need to be calculated. Using a criteria that is based on how well a model fits at a particular dimension (and fitting models at many different projection dimensions) removes the need for the arbitrary decisions described above.

When a GAM is fitted using GCV for smoothing parameter selection, the GCV score is calculated as part of the fitting process. So for each model we have the GCV score for the optimal (set of) smoothing parameter(s). Models can be fitted to a series of projections (increasing in dimension), their GCV scores compared, and the best selected as the projection to use for the model. For each projection dimension, the model has the GCV-optimal degree of smoothing; models are then discriminated between using the GCV scores.

Starting from a 2-dimensional projection, the dimensionality is increased and models fitted. The upper bound is the number of dimensions that explain 95%

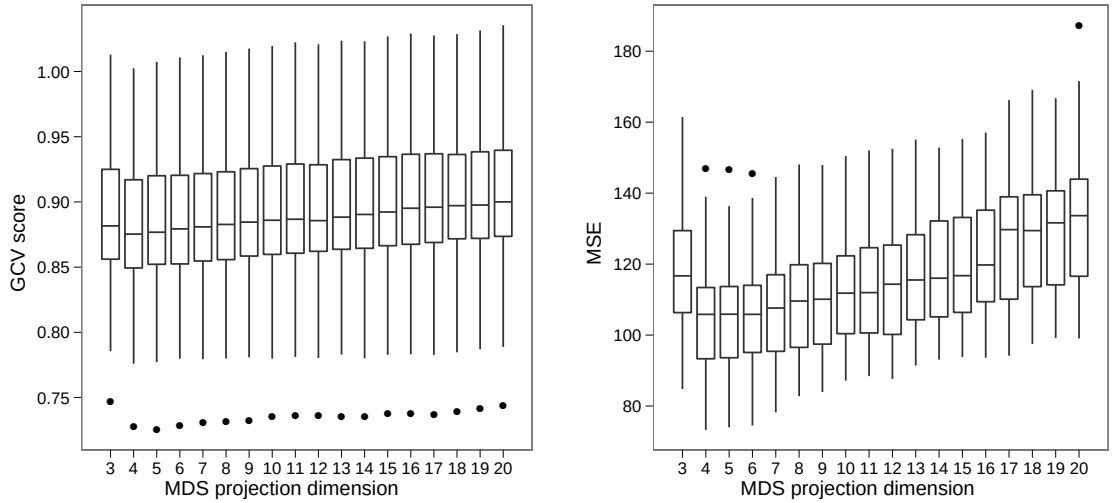

Figure 5-2: GCV score and MSE for the peninsula domain when different dimensional projections are used. Boxplots show the results for 60 simulations from the domain. Here a 4-dimensional projection minimizes both the GCV score (left) and the MSE (right).

(say) of the variation in the distance matrix of the initial grid (see section 4.2.3). Although a complete search on all dimensions could be performed, this could become extremely time consuming and it is likely that higher dimensions offer very little difference in the point configuration given that the associated eigenvalues will become smaller and smaller (as was observed in section 4.7.2). It is also possible to proceed step-wise and stop once the GCV score starts to increase, however there is no particular reason to believe that the GCV score would be unimodal in the number of dimensions. Looking at plots such as those in figure 5-2 gives guidance on how to proceed. Throughout the following analyses all dimensions between 2 and that which explains 95% of the variation in the distances are tested.

Simulations show that the minima in the GCV score and MSE are in agreement. For example figure 5-2 shows a plot of GCV score and mean squared error for 60 simulations from the peninsula domain, for each of the 60 realisations MDS+DS was fitted using a 2 through 20 dimensional projection. The boxplots are grouped according to the dimension of the MDS projection used. The graph shows that there is a minima in the score when the dimension is four, this corresponds with the minimum MSE. This simulation shows that there is a clear minima in the GCV score as dimension increases, however this may not always be the case.

If smoothing parameter selection is to be performed via ML (section 1.2.3), the ML score can be adapted into an AIC-like score to be used in place of the GCV score, provided there is appropriate penalisation to take into account the increase in the dimension of the projection space. Note that the REML score cannot be used as REML scores cannot be compared when their fixed effects are changed (Wood, 2011). The (AIC-like) score (denoted ML_P) used was:

$$\mathrm{ML}_P = -2\hat{l} + 2P$$

where $\hat{l}$ is the log-likelihood at the MLE and P is a penalty. Setting the penalty to be the nullspace dimension (i.e. M from above with $m = 2$ and d set to the MDS projection dimension):

$$P = \binom{2+d-1}{d} = d + 1.$$

So the score used to select dimension is:

$$\mathrm{ML}_P = -2\hat{l} + 2(d+1)$$

The extra penalty can be justified in the following way: it is possible that an increase in dimension (which adds much more complexity into the model) will give only a slight increase in the likelihood. This model would be taken as "best" even though it only offers a small improvement on a much simpler model. To avoid such overly complicated models being selected, the penalty above is added.

There is, of course, no guarantee that there will always be a clear minimum to find or even that either score will be unimodal in projection dimension. As with all automated methods, diagnostics (for example the plots of score against dimension) should be used.

5.3 Within-area distance examples

5.3.1 Simulations

To test the utility of MDS+DS, the simulation study on the peninsulae domain in section 4.6.4 was re-run including MDS+DS as a possible fitting method. MDS projection dimension selection based on both GCV and ML_P scores was used

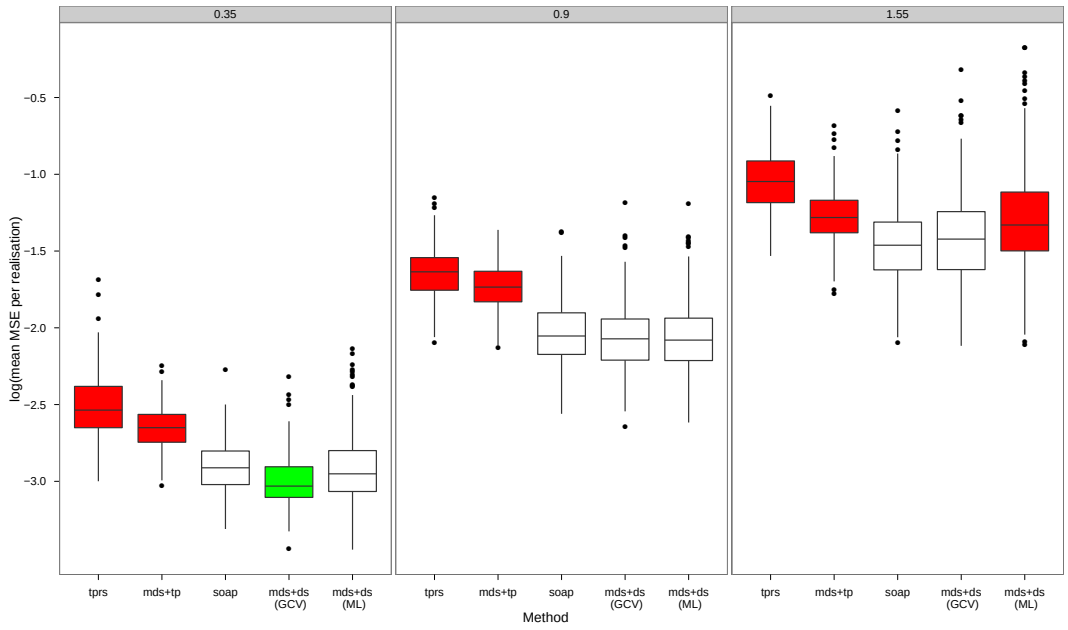

Figure 5-3: Boxplots of logarithm of per realisation average mean squared error from simulations on the peninsula domain. The boxplots are in groups of five for each error level (0.35, 0.9, 1.55). Colours indicate the result of a Wilcoxon paired signed rank test of whether the MSE was significantly ($p < 10^{-2}$) different from the soap film smoother. Red indicates different and worse, green different and better.

and the maximum spline basis dimension was set at 100. The results are shown for the original simulations (for thin plate regression splines and the soap film smoother) along side those for the new method in figure 5-3. MDS+DS with GCV dimension selection obtained a lower MSE than the soap film smoother when the noise is low and was indistinguishable (via a paired Wilcoxon signed rank test) from the soap film smoother at higher noise levels. Using ML_P generally had poorer performance but this only became significant at the highest noise level.

Using GCV does give a lower MSE than using the soap film smoother (although perhaps not statistically significantly at higher noise levels). This is a good indication that using GCV to control the MDS projection dimension is working well. Visual inspection of the smooths produced also show that the GCV score works well as a method for selecting the projection dimension.

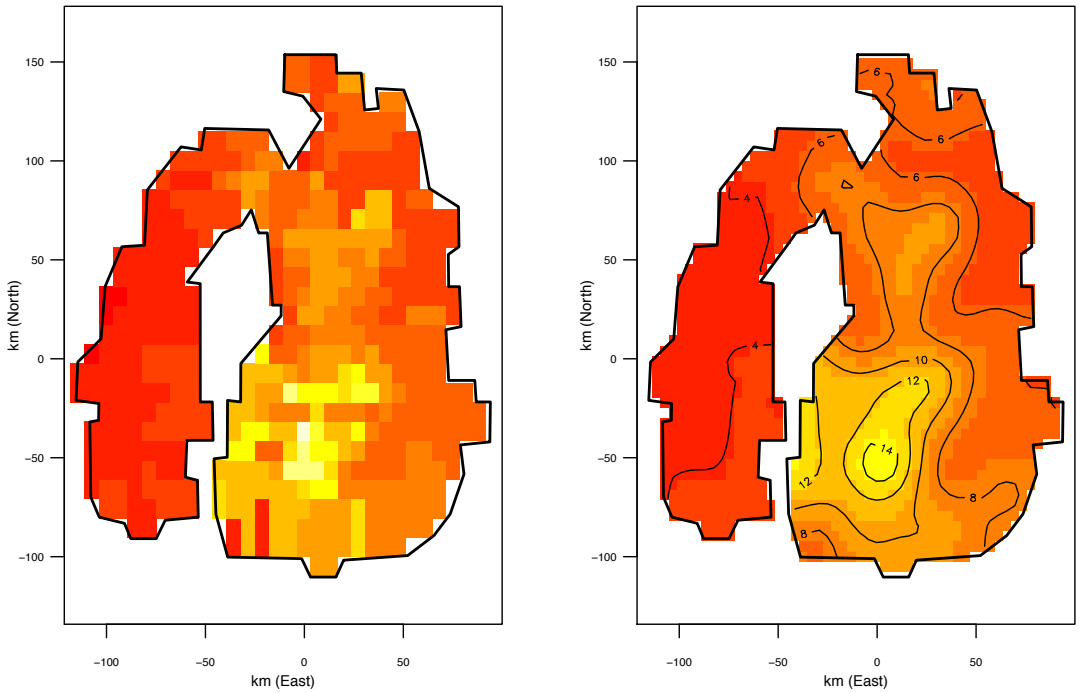

Figure 5-4: Left: the raw Aral sea chlorophyl data. Right: the data smoothed using MDS+DS, when a 5-dimensional MDS projection is employed. Note the lack of artefacts in comparison to previous MDS+RS models, e.g. figure 4-22.

5.3.2 Revisiting the Aral sea

Returning to the Aral sea example from section 4.6.4, MDS+DS can be used to fit a model using the optimal dimension (in GCV/ML$_P$ terms). Figure 5-4 shows the raw data and a smoothed version, using a 5-dimensional projection (given by minimising the GCV score). The plot does not contain any of the artefacts that were present in the previous smooths of the data in high dimensions (see figure 4-22).

Using the ML$_P$ statistic to select the MDS projection dimension resulted in a 19-dimensional smooth, significantly greater than the dimension selected by GCV. The image plot in figure 5-5 does not look particularly different from the GCV selected one in figure 5-4, although perhaps there is some overfitting (for example in the $(-50, 100)$ and $(80, -20)$ areas). Figure 5-6 shows plots of score (both GCV and ML$_P$) against MDS projection dimension. The GCV plot shows a clear minima where as ML$_P$ does not. Given this plot and the marginally worse performance in the peninsulae domain simulations, it seems that GCV is preferable for projection dimension selection when using within-area distances.

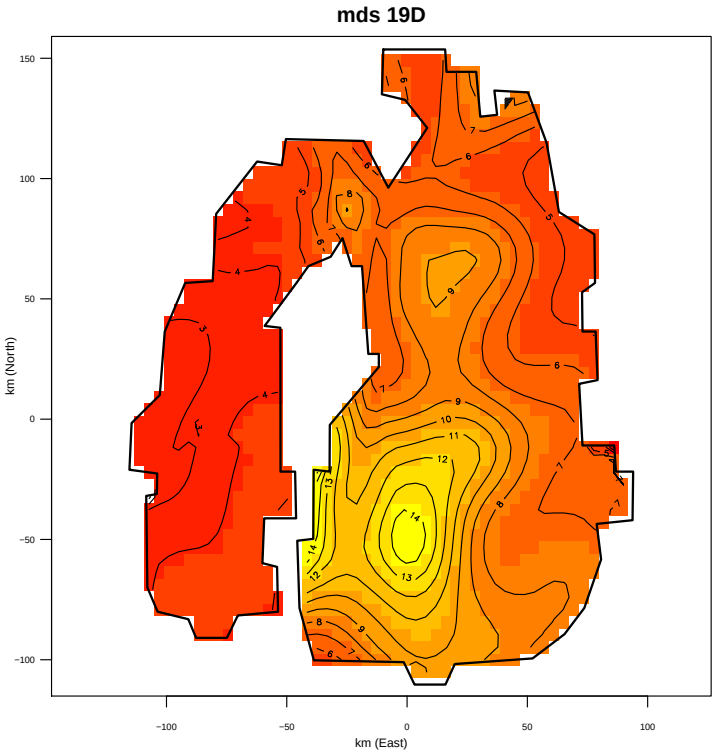

Figure 5-5: Image plot of the smoothed surface fitted by MDS+DS for the Aral sea when ML_P is used to select the MDS projection dimension.

5.4 Generalized distance smoothing

Since Duchon splines give reliable results when smoothing in high dimensions and multidimensional scaling allows the projection of any arbitrary distance matrix into Euclidean space, why not explore how this combination of techniques can be used with more general data? This section investigates the utility of performing MDS on a general set of distances and then using Duchon splines to smooth over that projection in (potentially high-dimensional) space.

Data are often collected on scales that are not necessarily physically meaningful (for example in psychological studies or attitude surveys) but the data are used as if the scale was absolute in some sense (Cox, 2007, Torgerson, 1952). In such cases the distances between the observations may be meaningful but the actual observed values may not be.

Taking data which are either already distances or from which distances can be calculated, the MDS projection can be found and then a smooth over those data can be used to model some response. Situations where MDS+DS might be useful fall into three classes:

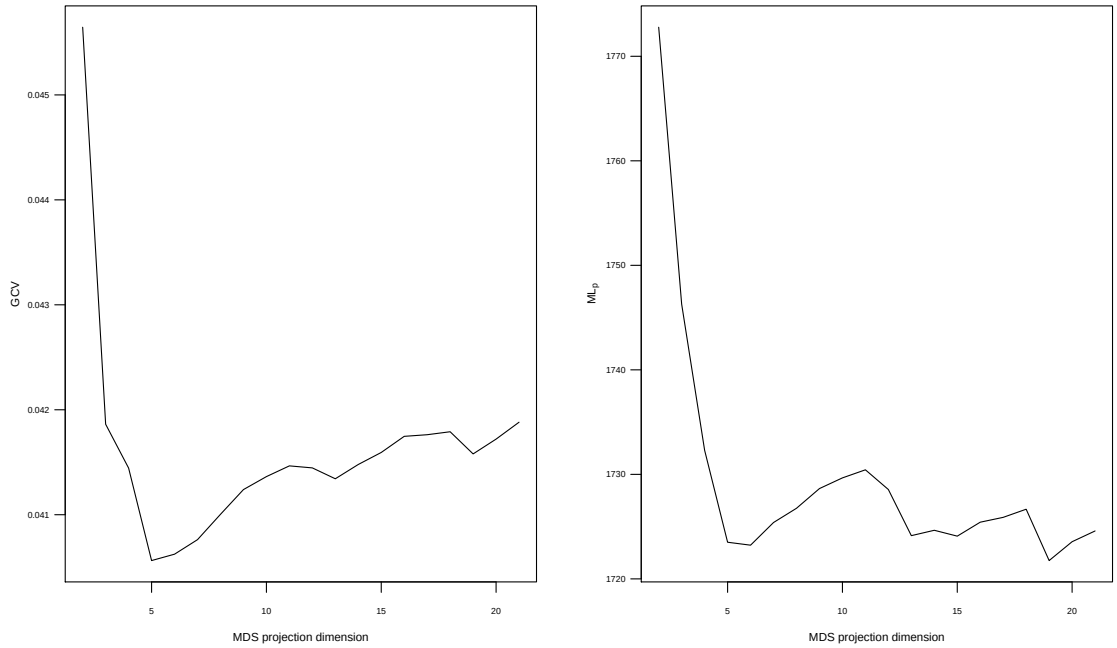

Figure 5-6: Plots of score against MDS projection dimension for the Aral sea data when GCV (left) and ML_P (right) are used for dimension selection.

1. Those in which distances are intrinsically meaningful, where distances between the subjects in a study represent some obviously meaningful physical quantity. For example, in the within-area distance situations that have been seen so far.

2. Those in which the combinations of variables in the MDS configuration are not meaningful physically but come together to give a measure of dissimilarity between subjects. In this case all of the variables could be of the same "kind" , such as a politician's voting record. Alternatively, it could be combinations of measurements of different phenomena similar to the indices constructed by econometricians from socio-economic variables like household income, education level or state benefit eligibility.

3. Finally, similar to the above, situations where there are too many variables to be reasonably used in a conventional additive model and therefore using MDS could be used as a variable "reduction" technique similar to principal components regression (Hastie et al., 2001, p. 79–80).

The latter two situations pose two interrelated problems. First is that of measurement error: if there are large errors in the measured covariates which are

used in the MDS projection, these could unduly influence the result (this has not been a problem so far since, usually geographical locations are accurate). If there is one very large (erroneous) observation in one of the variables, this could dominate the eigenvalues and cause that covariate to have undue prominence in the MDS projection. A thorough treatment of measurement error in non-linear models is given in Carroll et al. (2006). Second is the issue of variable selection; since all variables are used in finding the distances, we have made the assumption that they all affect the response in a way which is related to their variation (or rather, their contribution to the eigenvalues). There is no reason to believe that this is the case.

Below, the hope is that a combination of appropriate distance metric, projection dimension selection and usual model checking will work around the above issues.

5.4.1 Examples

Data where distances between observations are meaningful can come from many different disciplines. Here three examples are given, one from political science and two from medicine. The choice of data here does not indicate any limitation of fields of study to which MDS+DS can be applied, any discipline in which distances can be measured (or calculated) may well benefit from this approach.

Predicting party allegiance using free votes

The website Public Whip (`http://www.publicwhip.org.uk/`) provides data from the Hansard on the votes of the both UK houses of parliament. Divisions of the 676 MPs in the 1997–2001 parliament are considered here. During this time the House took 1273 divisions. Each vote is coded according to table 5.1. Also available from Public Whip are the party allegiances of each MP.

Of the 1273 divisions in the 1997–2001 parliament, 17 of them were declared as "free votes" (House of Commons Library Department of Information Services, 2011), where MPs were not "whipped" (pressured to take the party line). Predicting affiliation based on whipped votes is relatively easy since MPs are likely to vote along party lines. Using free votes makes the classification much more difficult, not only because there are significantly less data. The free votes are summarized in table 5.2, most of which are "conscience" votes.

Using the free vote data, MPs were sampled, MDS+DS fitted to the data and

Value	Description	Code
Missing	MP did not vote in this division	0
Tell aye	MP voted for the motion and was a teller	1
Aye	MP voted for the motion	1
Both	MP voted both for and against the motion	0
No	MP voted against the motion	-1
Tell no	MP voted against the motion and was a teller	-1

Table 5.1: Coding of UK MP voting data. For the purposes of the analysis here the teller's votes are counted as if they voted since we are interested in how voting can be used to predict party affiliation. Note that "both" is perfectly possible, and occurs when the MP walks through both the "Aye" and "No" gates, this can correspond to the MP abstaining (as with "Missing") or to nullify a mis-cast vote.

Date	Bill name
22 March 2001	Election of a Speaker
17 January 2001	Hunting Bill
17 January 2001	Hunting Bill
17 January 2001	Hunting Bill
20 December 2000	Hunting Bill
19 December 2000	Human Fertilisation and Embryology
31 October 2000	Stem Cell Research
14 April 2000	Medical Treatment (Prevention of Euthanasia) Bill
28 February 2000	Sexual Offences (Amendment) Bill
10 February 2000	Sexual Offences (Amendment) Bill
28 January 2000	Medical Treatment (Prevention of Euthanasia) Bill
25 January 1999	Sexual Offences (Amendment) Bill
22 June 1998	Crime and Disorder Bill
22 June 1998	Crime and Disorder Bill
28 November 1997	Wild Mammals (Hunting with Dogs) Bill

Table 5.2: Free votes in the 1997-2001 parliament (see House of Commons Library Department of Information Services, 2011).

a prediction of party affiliation (simplified to Labour party versus not Labour party) made for those MPs not in the sample. A logit link function was used. Euclidean distances between the MPs were found and used to form the distance matrix. Multidimensional scaling was then used to project these distances into MDS space (dimension selection was performed by optimizing the GCV or ML_P score). This was repeated for 200 realisations with sample sizes of 200, 300, 400 and 500.

For comparison, the usual approach for such problems would be to use either (*i*) a linear regression with subset selection (e.g. step-wise selection of model terms using AIC) or (*ii*) the lasso (Hastie et al., 2001 pp. 68–69).

The idea behind the lasso is that when performing a linear regression with a large number of covariates, many of these covariates may not be useful, while some may only be partially useful. Subset selection will remove those covariates that are completely non-informative, however, those which are partially informative may only be removed or left in the model. The lasso penalizes the sum of the absolute value of the coefficients and therefore allows the coefficients to shrink towards zero. This will mean that those covariates that are completely uninformative will be removed (since their coefficients will be set to zero) but those which are partially informative will be allowed to remain (but with reduced influence). As with smoothing, a parameter must be estimated in order to find the optimal level of shrinkage; this is usually found via k-fold cross validation (like LOOCV seen in section 1.3.3, this fits models to partial subsets of the data, but rather than leaving only one observation out, the data are split into k parts and the model fitted to all but one subset, this is repeated for all subsets).

The built-in procedure `glm()` is suitable for (*i*) and the R package `glmnet` provides a lasso implementation for (*ii*).

The four models used in the simulation were:

1. MDS+DS (GCV): MDS projection of the data, selected by minimum GCV score over the full range of 2 to the number of dimensions that account for 85% of the variation in the sample. Maximum spline basis size was 100.

2. MDS+DS (ML_P): as above but using ML_P score to select the MDS projection dimension.

3. Lasso: as implemented in `glmnet`, 10-fold cross validation was used to select the amount of shrinkage per realisation.

4. GLM: as implemented in `glm()` with `step()` providing step-wise variable selection based on AIC.

For all models the response distribution was binomial and a logit link function was used. To compare the results the MSE and Brier score were calculated (see section 1.3.1 and section 1.3.2).

Figures 5-7 and 5-8 show boxplots of the MSE and Brier scores respectively at the varying sample sizes. The picture painted by the results is unambiguous and consistent across sample sizes: the lasso out-performs all other methods. Interestingly, using the ML_P score rather than the GCV score for the MDS projection dimension yields both a lower median MSE and a smaller standard error. Looking at the plots of score against dimension per simulation (figure 5-9), we can see that per simulation (the black lines) the GCV score appears to be much more volatile than the ML_P criterion. The selected projection dimensions (red dots) are spread across the whole range, showing no clear preference for one over another. Smooths (blue lines, green confidence bands) through the full set of scores do not show that there is any particular, definite minima in the scores. Figure 5-10 shows histograms of the EDFs (section 1.3.4) for the GCV and ML_P selected models at the varying sample sizes. These plots clearly show that the EDFs for GCV are bimodal (becoming more so at higher sample sizes) and that ML_P selects models with significantly lower EDFs.

MDS+DS's performance in predicting MPs allegiance using the free vote data is disappointing. This poor performance may be due to there not being enough information in the distances to predict the party; since the data were ternary (votes were coded only -1, 0 or 1) there would be many distances that were similar. This theory is supported by looking at distance matrix for all MPs for the free votes, in that matrix there are only 63 unique values. Also worth noting is potential confounding between Labour MPs (which made up 429 out of the 676) and other ideologically similar parties (such as Plaid Cymru, SDLP and the Liberal Democrats (at that time)) which might cause potential problems for classification. This second theory is less likely (given the performance of the lasso and GLM) but in combination with the first seems plausible. There is also something interesting happening in the differences between the GCV and ML_P results. GCV seems to prefer fitting models with higher EDFs than the ML_P, this may be an manifestation of the phenomena reported in section 1.2.3.

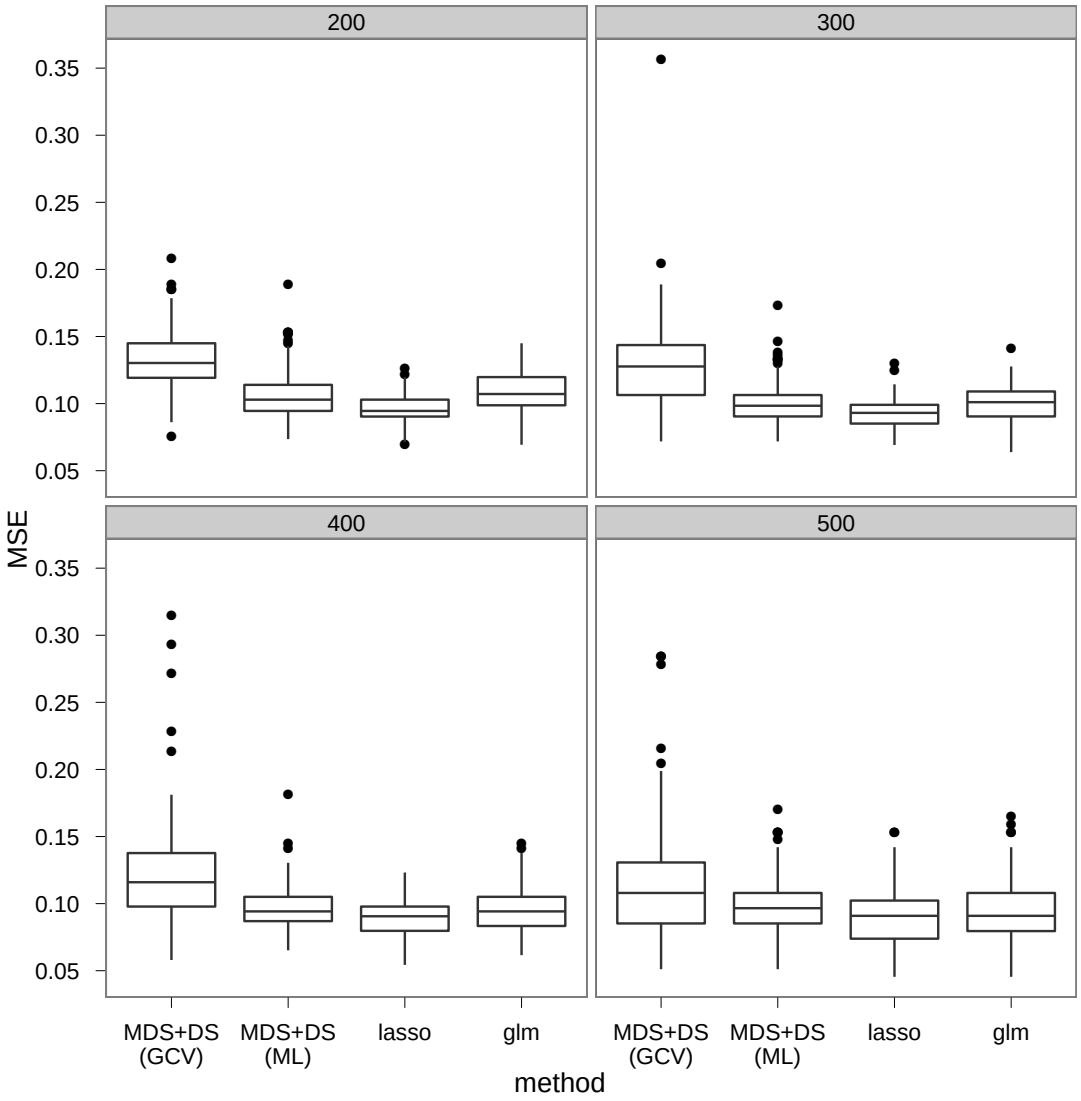

Figure 5-7: Boxpot of MSE per model for the MP free vote data set at varying sample sizes. The MSE for the lasso was significantly different (and smaller) than all of the other models by a pairwise Wilcoxon signed rank test (at the 0.01 level).

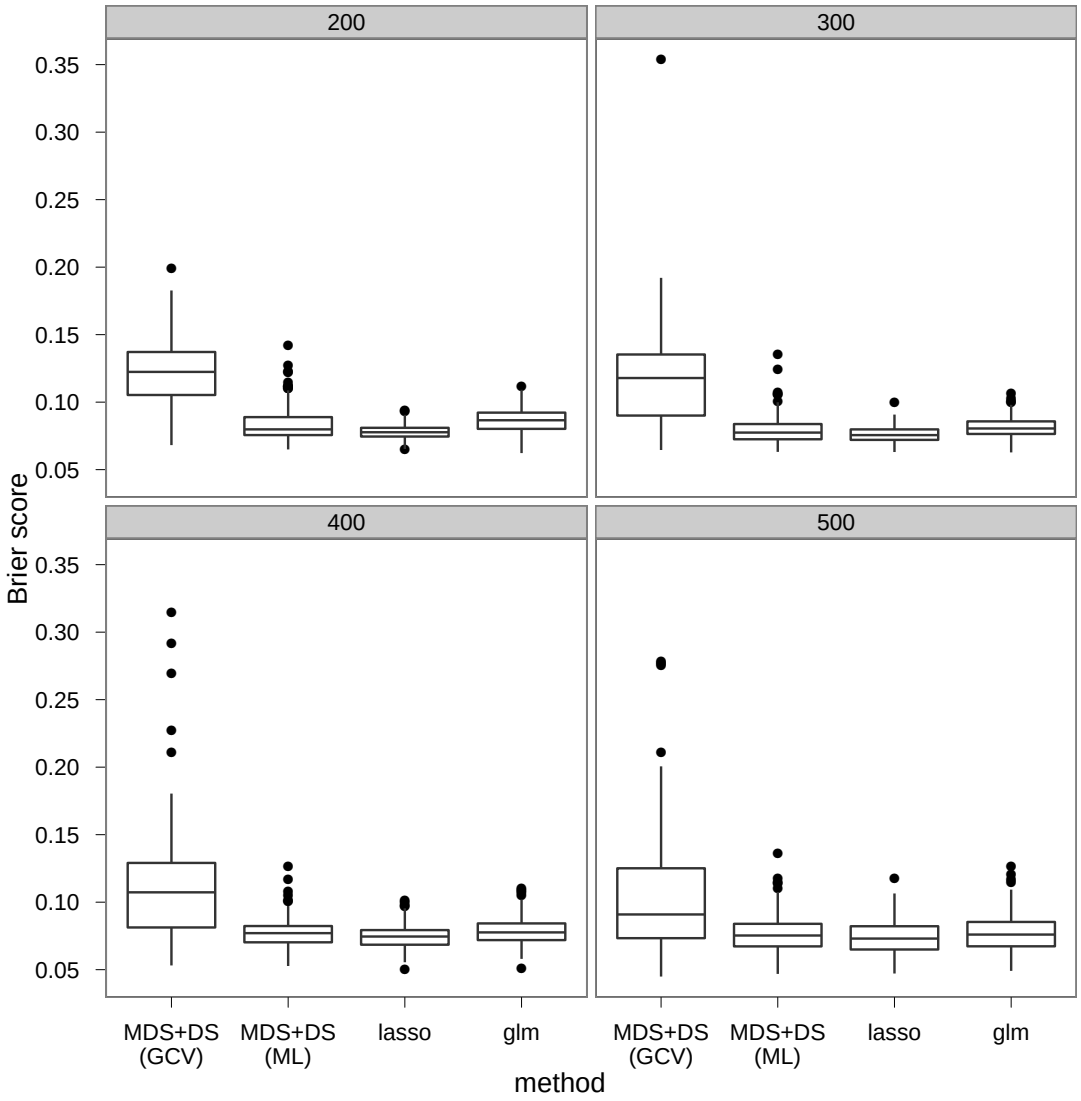

Figure 5-8: Boxplot of Brier score per model for the MP free vote data set at varying sample sizes. The Brier score for the lasso was significantly different (and smaller) than all of the other models by a pairwise Wilcoxon signed rank test (at the 0.01 level).

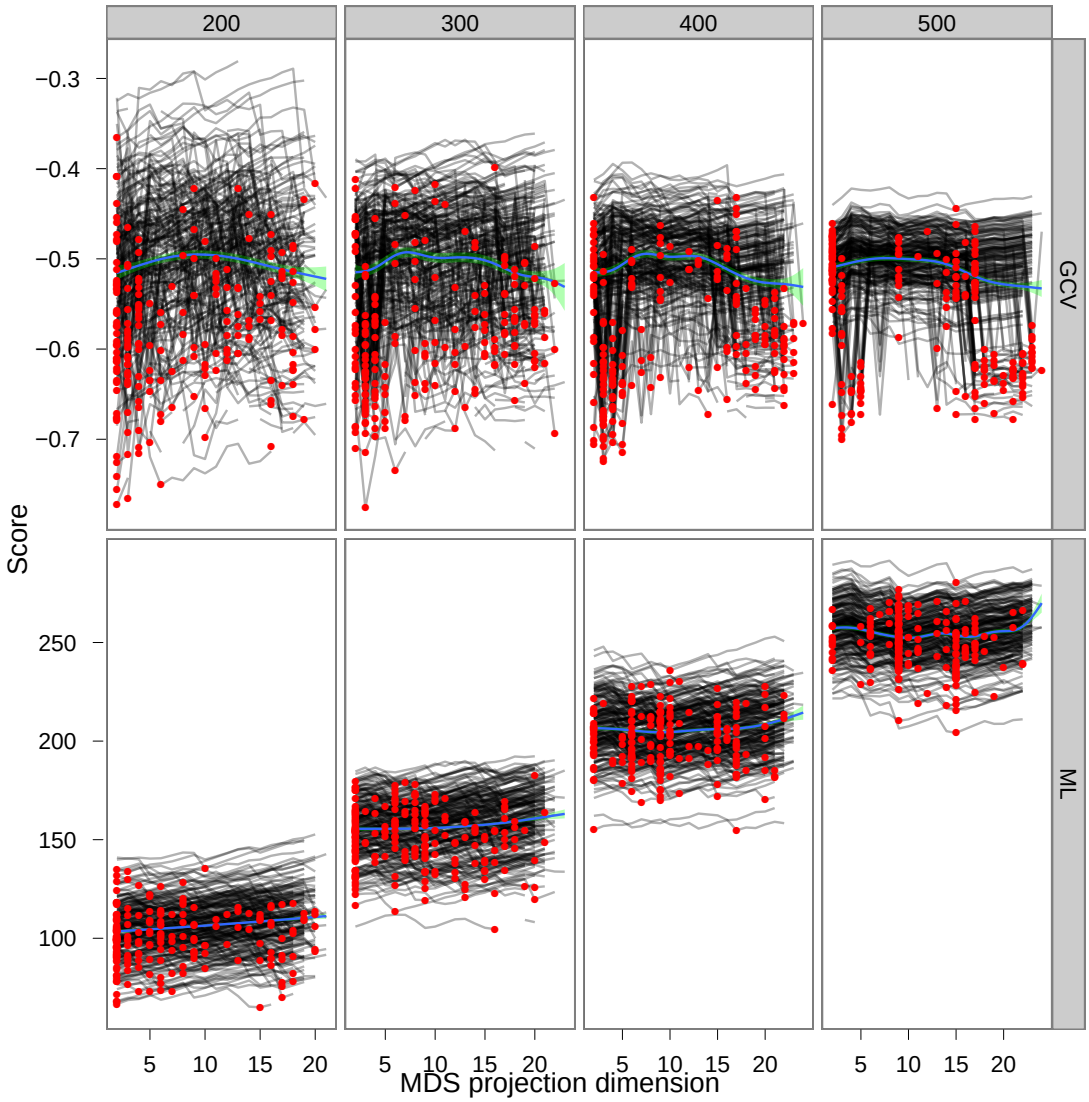

Figure 5-9: Plots of MDS projection dimension against score (GCV and ML_P) for the MP voting simulation per sample size. Each line represents one round of cross validation (some lines are broken due to convergence failure in the GAM), red dots indicate the selected projection dimensions (score minima) per simulation, blue lines are (thin plate regression spline) smooths through the full data set and the green bands are 95% confidence bands.

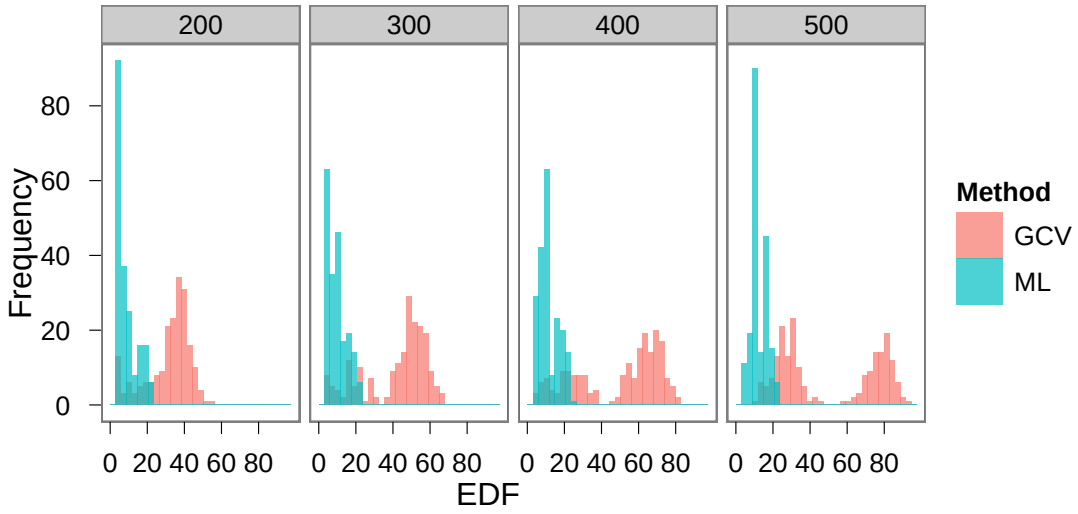

Figure 5-10: Histogram of EDFs of the selected models for the free vote simulation. When dimension selection is performed via GCV the EDFs are bimodal. In contrast the histogram of EDFs for the models selected by ML_P shows a clear mode much lower than for GCV.

Breast cancer microarrays

Microarrays typically consist of small silicone or glass chips on which thousands of strands of RNA (which can be thought of as carrying instructions from DNA about how to create proteins) are stuck. These strands are known as *targets* or *probes*. When RNA from the sample comes into contact with that on the chip *hybridisation* occurs (hydrogen bonds form between matching pairs). The targets have a fluorescent die applied to them before hybridisation and can be scanned afterward to quantify the number of strands from the sample which have attached themselves to the chip, this is known as the *expression level*.

Wit and McClure (2004, pp. 7-9) describe a data set where both microarray and non-genetic data were collected on 62 patients with breast cancer. Rather than using RNA on the microarray, the experiment used DNA from cancer tissue and measured the differences between the genomic DNA of patients with breast cancer as compared to controls; the theory predicting that cancer causes loss of genetic material or additonal copies of genes to be obtained. The microarray contains expression data on 59 genes. Among the non-genetic data collected was the Nottingham Prognostic Index (NPI) (Haybittle et al., 1982 and Todd et al., 1987), thought to be a good general measure of prognosis for patients with primary breast cancer (cancer that has not spread beyond the breast). The NPI

combines three pieces of information in a simple equation:

NPI = 0.2×(size of index lesion in cm)+number of lymph nodes+tumour grade.

Further information can be found in the references above; it suffices to say that high values of NPI identify patients with very poor prognoses.

Rather than attempt to predict survival based on microarray data while controlling for other factors (as was the case in Wit and McClure (2004, pp. 240-245)), here the NPI is predicted based on the microarray data. This is not an unreasonable proposal since if one believes that there is some genetic mechanism behind breast cancer (or at least an individuals vulnerability/resistance to it) then the factors making up the NPI could be considered proxies for susceptibility.

Spang et al. (2002) propose that rather that considering a large number individual genes, combinations are used. Spang et al. (2002) use a singular value decomposition of the microarray data to perform dimension reduction. The quantities resulting being referred to as "super-genes". In a similar way, using distances between patients and then taking the MDS projection, we can consider "eigen-genes".

Section 5.4 noted that both errors and non-standardized columns in the data matrix can cause issues with MDS since those variables with the greatest degree of variation do not necessarily contain the most information. One can easily imagine the case in which a completely unrelated gene was measured with huge error and then made up a huge proportion of the first eigenvalue in the decomposition of the distance matrix, this would dominate the projection but contain no information about the prediction (see also Wit and McClure, 2004, pp. 220-221). To get around this problem, rather than use the Euclidean distance between patients, the *Mahalanobis distance* (Mahalanobis, 1936) can be used.

The Mahalanobis distance is easily calculated in the following way. Let $\mathbf{m}_i$ be a single row from the microarray matrix, $\mathbf{M}$ say, so that $\mathbf{m}_i$ is the vector gene expressions for a single patient. Under the assumption that all of the subjects are drawn from the same multivariate distribution some mean and covariance matrix $\mathbf{\Sigma}$, the Mahalanobis distance d_{ij}^{M}, between subjects i and j is then defined as:

$$d_{ij}^{\mathrm{M}} = (\mathbf{m}_i - \mathbf{m}_j)^{\mathrm{T}} \mathbf{\Sigma}^{-1} (\mathbf{m}_i - \mathbf{m}_j), \tag{5.9}$$

where $\mathbf{\Sigma}^{-1}$ is replaced with the inverse of the sample covariance matrix of $\mathbf{M}$. The

Model	Mean	Median	Standard error
lasso	1.67	1.021	1.837
MDS+DS (GCV) - normal	1.41	0.695	1.759
MDS+DS (ML) - normal	1.426	0.629	1.873
MDS+DS (GCV) - quasi	1.427	0.654	1.739
MDS+DS (ML) - quasi	1.419	0.575	1.857

Table 5.3: Summary of the results for the breast cancer cross validation. Summary statistics are over 45 rounds of cross validation.

calculated distance takes into account that the data may be more variable in some directions than in others. Calculating the Mahalanobis distance for each pair of patients and putting this into a distance matrix, we can then obtain the MDS projection in the same way as we would with a set of within-area or Euclidean distances. Gentleman et al. (2005) suggest using the Mahalanobis distance in a microarray setting.

Of the 62 patients in the study, 45 had non-missing NPIs and out of the 59 genes in the microarray, 27 did not have missing values. In order to keep the analysis simple the the non-missing data (NPI measurements of 45 patients using distance from 27 genes) were used. For the MDS projection a lower bound of 2 and an upper bound of 85% of the variation in the distance matrix was used (this equated to 19 or 20 dimensions usually). A number of different MDS+DS models were fitted, with various error distributions. Using standard checks (eg. `gam.check()` in `mgcv`, fitting to residuals etc) two were deemed most promising. Those were a model with normal errors and one using a quasi-likelihood (McCullagh, 1983, Wood, 2008) with a square root link function and variance proportional to the square of the mean. For comparison the lasso was (again) used. The implementation of the lasso does not allow the use of quasi-likelihood, so the normal model for MDS+DS is a fairer comparison.

Since the sample size is rather small, it was not possible to reasonably split the data into training and validation sets as with the MP data. Instead, leave-one-out cross validation (LOOCV) (section 1.3.3) was used to assess the sensitivity of the models to changes in the data, as well as overall prediction. Table 5.3 and figure 5-11 show the results. Both the table and plot show that although MDS+DS has a slightly lower LOOCV score than the lasso across the board and that the variability in the models is about the same. Performing a paired Wilcoxon signed rank test showed that each of the MDS+DS models were not significantly different from the lasso.

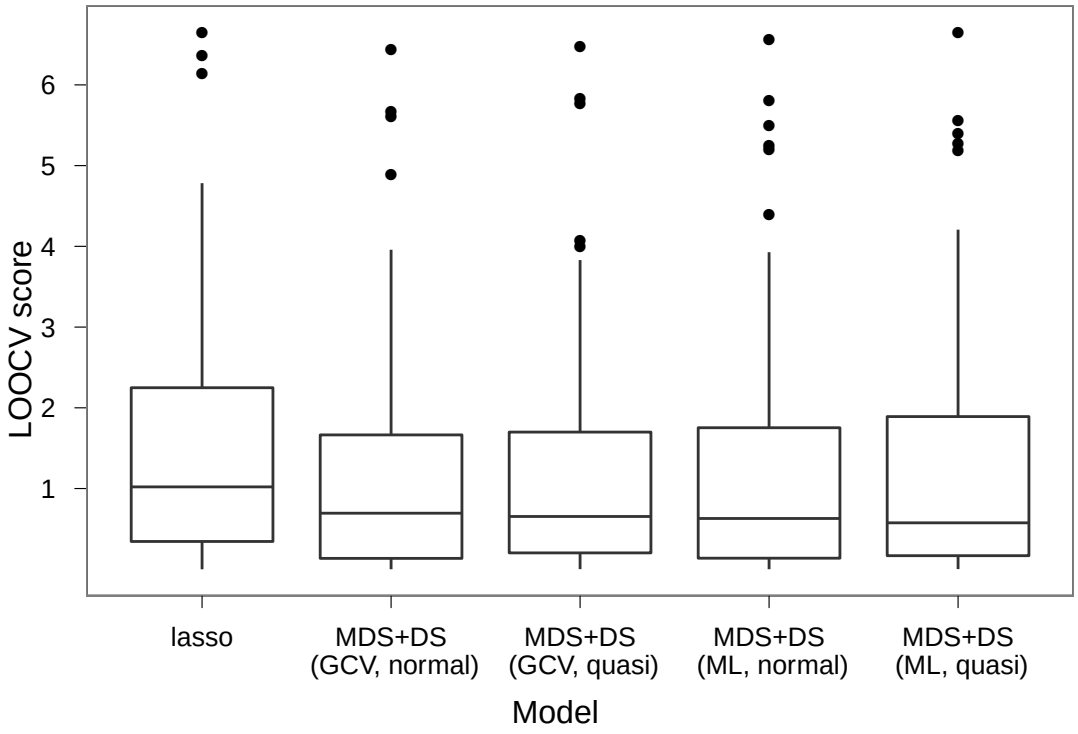

Figure 5-11: Boxplots of the LOOCV score per model for the breast cancer cross validation. A Wilcoxon (paired) signed rank test did not show that there was a significant difference between the MDS+DS models and the lasso (in fact $p > 0.5$ in every case).

Figure 5-12 shows the plots of MDS projection dimension against score, each line represents one round of cross validation (some lines are broken due to convergence failure in the fitting routine), red dots indicate minima in the score per simulation (the selected dimension) and the blue lines are (thin plate regression spline) smooths through the full data set to give a general idea of what is going on (with green confidence bands). Both error distributions show a similar relationship between dimension and score.

The GCV score appears to have a minima somewhere between 5 and 11 dimensions, which is fairly well defined given the size of the data set. ML_P, on the other hand, appears to select high dimensional solutions (almost all models were either 19 or 20 dimensions). Upon first inspection one might think that the optima was at 19 or 20 and that this was a true minima. Further analysis shows that this is not the case. Increasing from 85% of the variation to 95% and 99% of the variation only pushes the ML_P "minima" to higher dimensions. This behaviour may indicate that the penalization used for ML_P is not strong enough in

the generalized distance case. Comparison of penalized versus unpenalized scores shows that the penalty is having some effect (and local minima are introduced) but this is not enough to introduce a global minima that is not at the highest dimension.

This analysis shows that although MDS+DS can be used for generalized distance smoothing in a microarray setting, it is far from conclusive that the methods out performs the lasso. Choice of the metric used to calculate the distances has a large influence on the results (preliminary results using Euclidean distance lead to much worse LOOCV scores). It would be interesting to investigate the behaviour of the ML_P score further on larger data sets to see why such high dimensional models are chosen. The study here is rather small and specialized, therefore it is hard to draw a conclusion about how MDS+DS will perform in other situations, although its performance here against the lasso (technique that has been in development for considerably longer) is encouraging. The final analysis again looks at microarray data in an attempt to further investigate the utility of MDS+DS in such a situation.

Leukaemia microarrays

Yeoh et al. (2002) investigate expression data from 327 patients with acute lymphoblastic leukaemia (ALL), collected on 12,626 genes. The original purpose of the study was to classify patients into one of ten prognostically important ALL subtypes (since treatment is dependent on subtype), using expression data. Yeoh et al. (2002) used heirarchical clustering to find groupings of genes in the data and then found those groups corresponded to particular ALL subtypes. Genes were ranked per subtype according a χ^2 statistic (based on the expected value per cluster) to find the most relevant genes for each subtype. The "true" diagnoses had been found by other methods. More information on the study may be found at `http://www.stjuderesearch.org/data/ALL1`.

To simplify the analysis here, a binary response was used (of a particular subtype versus not that subtype). In all four simulations settings were used, varying the model and data. Two simulations used whether the patient had the TEL-AML1 subtype as the response (TEL-AML1 was the largest group, with 79 patients) and two simulations used T-ALL (43 patients were diagnosed with T-ALL). For each subtype two different simulations were run: (*i*) the 40 genes selected by χ^2 in Yeoh et al. (2002) as the best indicators of that subtype were used as the data and (*ii*) those 40 genes chosen by χ^2 with an additional 100

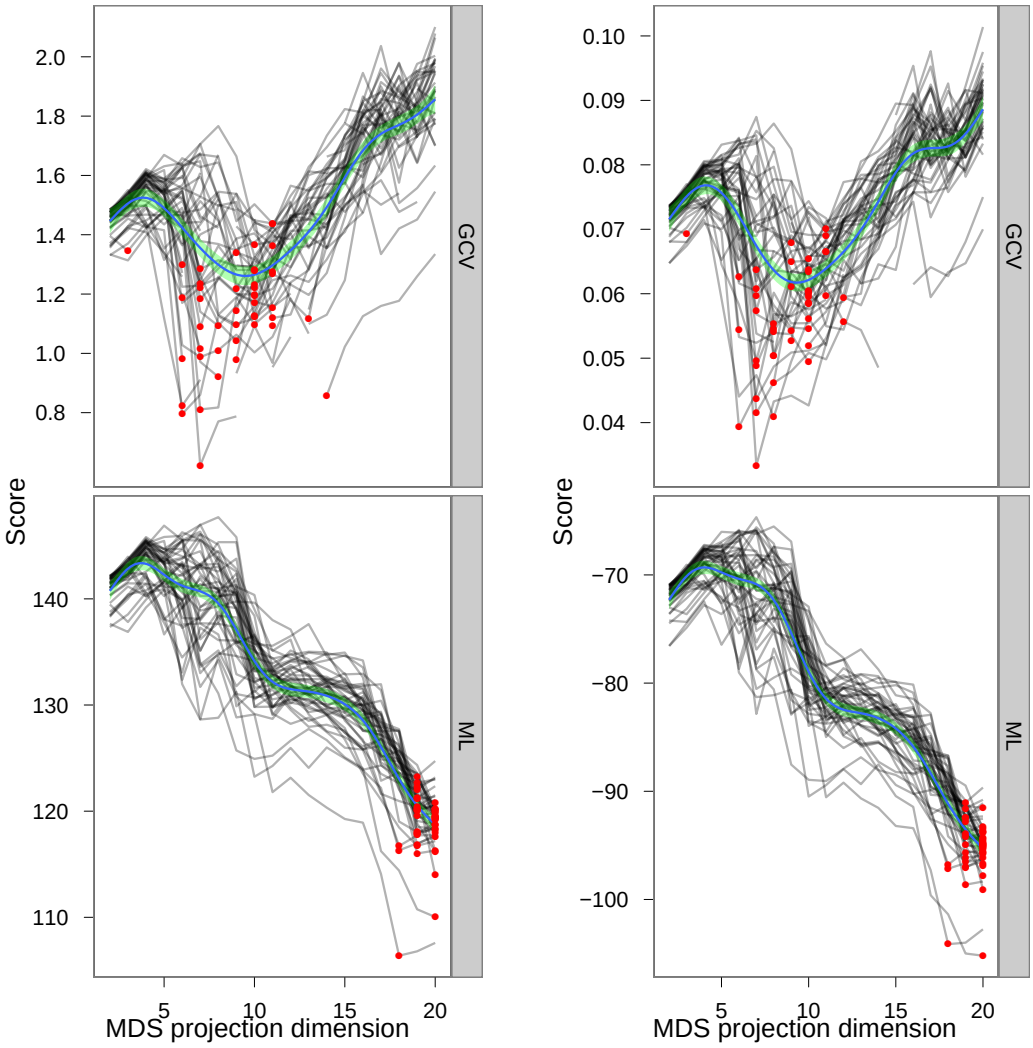

Figure 5-12: Plots of MDS projection dimension against score (GCV and ML_P) for both normal (left) and quasi-likelihood (right) models for the breast cancer microarray LOOCV. Each line represents one round of cross validation (some lines are broken due to convergence failure in the optimization), red dots indicate the selected projection dimensions (score minima) per simulation, blue lines are (thin plate regression spline) smooths through the full data set and the green bands are 95% confidence bands.

Model	MSE Mean	MSE Median	MSE Standard error	Brier Mean	Brier Median	Brier Standard error
T-ALL						
lasso	0.26	0	0.06	0.19	0.02	0.04
MDS+DS (GCV)	14.81	15	0.25	14.8	15	0.25
MDS+DS (ML_P)	14.79	15	0.26	14.74	14.99	0.26
TEL-AML1						
lasso	1.59	1.5	0.12	1.29	1.19	0.08
MDS+DS (GCV)	31.44	20.5	2.3	29.11	18.29	2.24
MDS+DS (ML_P)	14.59	15	0.28	12.97	12.98	0.18

Table 5.4: Summary of the results for the leukaemia simulation when the 40 genes selected by Yeoh et al. (2002) using χ^2 were used. Note the huge difference between the lasso and MDS+DS results.

genes chosen at random (but kept the same over realisations). The two scenarios show the difference between an idealized situation where the best predictors have already been found versus a more realistic situation in which there is a lot of noise from non-relevant genes.

In each of the simulations, 100 realisations were generated and in each of these 215 patients were selected as samples and models fit using their data (as was the case in Yeoh et al., 2002). The models were then used to predict the classes of the remaining 112 patients. Brier scores and MSEs were recorded. Models using both ML and GCV dimension selection were used and as above, the lasso was used for comparison using the same settings as in the MP voting data example. Distances were again calculated using the Mahalanobis distance (as in the breast cancer example).

Unfortunately the results are not encouraging at all. For both the T-ALL and TEL-AML1 data, both with and without the additional confounding genes, the lasso outperformed MDS+DS by a large margin. Boxplots of MSE and Brier scores are shown in figure 5-13. The boxplots clearly show that the lasso is much better suited to this type of problem.

Tables 5.4 and 5.5 show that MDS+DS does not even begin to approach the predictive power of the lasso on this data set. One might expect that the lasso would perform well on the dataset consisting of only the 40 genes selected by Yeoh et al. (2002) (and indeed that MDS+DS would perform better), however it is extremely impressive that the lasso has such a low MSE and Brier score for the data with the confounding genes. This may, however, give some insights into why MDS+DS has such poor performance.

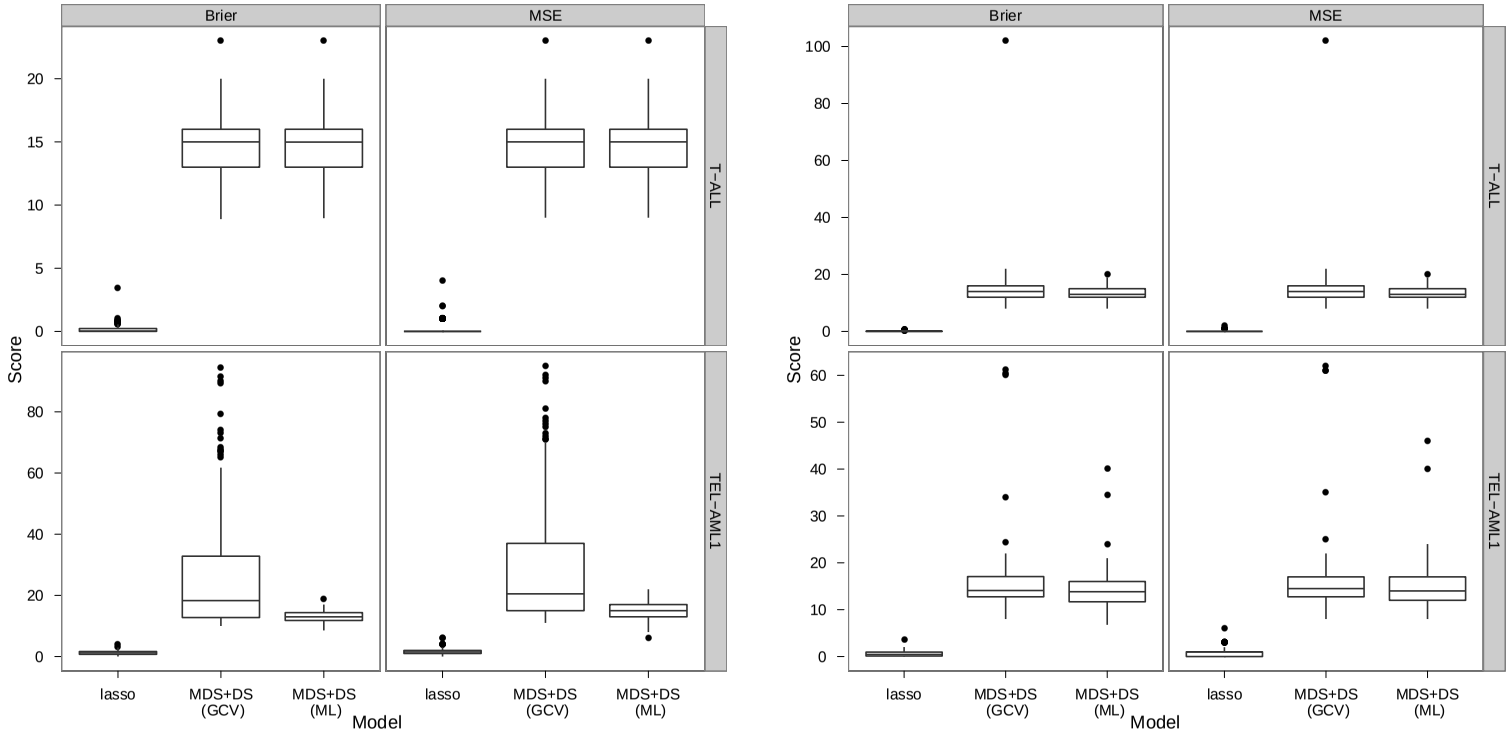

Figure 5-13: Boxplot of Brier and MSE scores (columns) for the T-ALL and TEL-AML1 data (rows) when left: the 40 genes selected by χ^2 were used and right: 100 extra genes were used for each of the models fit (lasso, MDS+DS with ML_P and MDS+DS with GCV dimension selection. The lasso vastly outperforms MDS+DS.

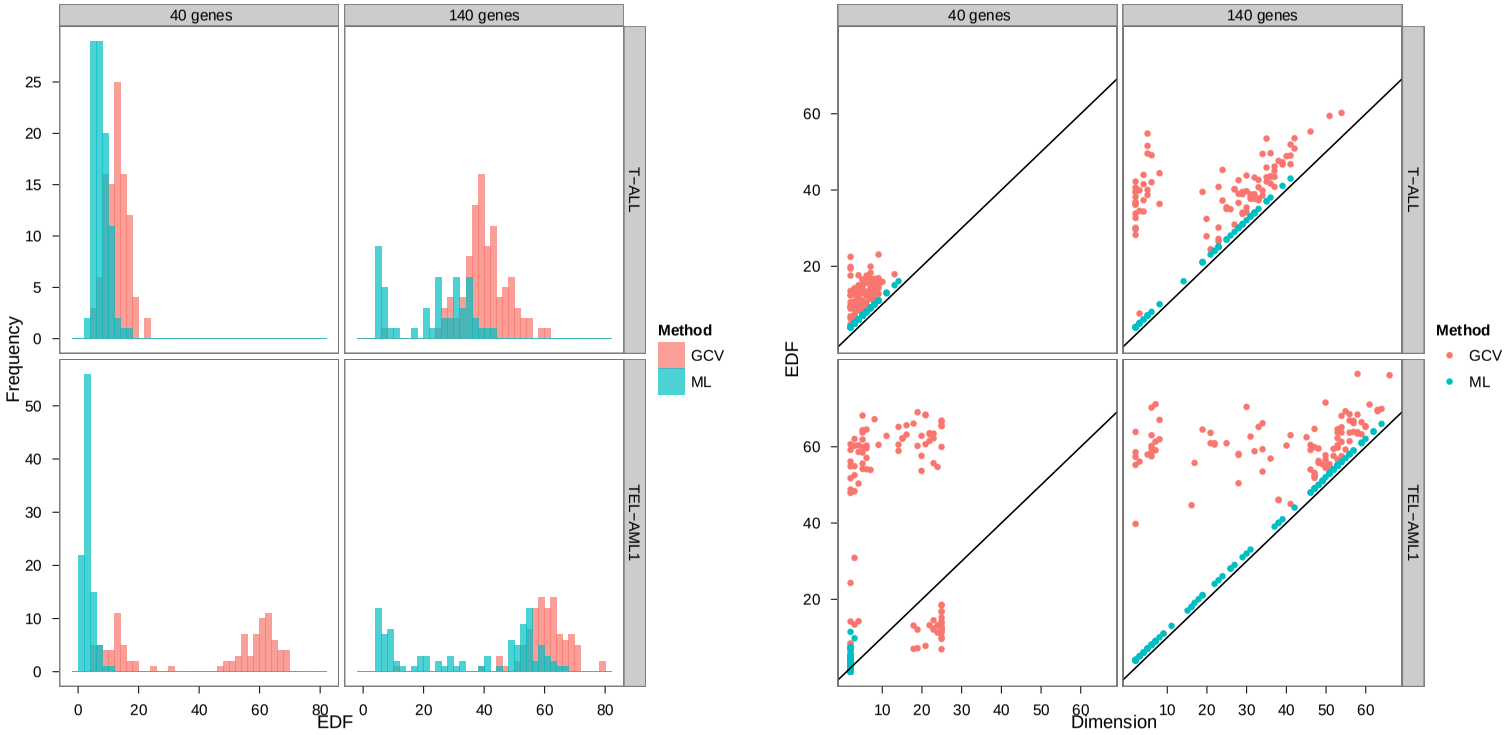

Figure 5-14: EDF and MDS projection dimension selection investigation for the leukaemia data. Left: histogram of EDFs of the models selected by GCV (red) and ML_P (blue). Right: plot of EDF against selected MDS projection dimension (with the same colour coding), lines of $x = y$ are included to aid comparison. The histograms on the left of the figure are the marginals of the plots on the right.

Model	MSE Mean	MSE Median	MSE Standard error	Brier Mean	Brier Median	Brier Standard error
T-ALL						
lasso	0.14	0	0.06	0.1	0.02	0.02
MDS+DS (GCV)	15.14	14	0.96	15.13	14	0.96
MDS+DS (ML_P)	13.75	13	0.36	13.74	13	0.36
TEL-AML1						
lasso	0.94	1	0.11	0.63	0.45	0.06
MDS+DS (GCV)	16.1	14.5	0.89	16.01	14.09	0.88
MDS+DS (ML_P)	14.81	14	0.54	14.31	13.84	0.48

Table 5.5: Summary of the results for the leukaemia simulation when 100 extra confounding genes were added to the 40 selected using χ^2. The lasso performs extremely well, even with 100 confounding genes.

Concentrating on the MDS+DS results, some information can be gleaned, aside from the fact that the performance of MDS+DS is not as good as that of the lasso. The left panel of figure 5-14 shows histograms of the EDFs of the models selected by GCV and ML. Similarly to the models fitted to the MP voting data, above, it appears that ML_P selects models that in general have lower EDFs than those selected by GCV. There is, again, bimodality in the histograms of the EDFs (or at least, no clear single mode).

Generally GCV selected much more complex models than ML_P, in all but one simulation setting (40 genes with TEL-AML1) ML_P selected models that had EDFs only slightly bigger than the dimension. This corresponds to ML_P fitting (hyper)planes to the data. The results for ML_P are in general slightly better than for GCV selection (tables 5.4 and 5.5), except for the 40 genes case for TEL-AML1 where the ML_P results are much better. These results may well be a manifestation of the behaviour described in section 1.2.3: that GCV is prone to overfitting.

The performance of ML_P, combined with the EDF/dimension selection behaviour indicates that perhaps the assumption of a non-linear response is not appropriate. Given the performance of the lasso, this seems likely. It is rather difficult to test for this graphically in high dimensions, but the EDFs do seem to indicate that this is the case.

Another potential source of poor performance is the metric used to find the distances. Although the Mahalanobis distance may be an obvious choice, there may be other measures that provide better results.

5.5 Conclusion

The first part of this chapter described the Duchon spline basis, a generalization of the thin plate regression spline basis seen in section 1.1.3. Duchon splines, although largely ignored in the statistical literature to date, can be used to smooth in high dimensions while not succumbing to the large nullspaces that prove problematic for thin plate splines.

Combining Duchon splines with high dimensional MDS projections allowed points in the domain to be separated based on their within-area distances, while maintaining their ordering. This avoided both leakage and the artefacts seen in section 4.6.4. MDS+DS also outperformed the soap film smoother in MSE terms and provided comparable maps on real data.

Although in the spatial setting MDS+DS appears to perform very well, the results for general distance smoothing are disappointing. Between examples the results are quite variable, with a lack of conclusive behaviour; within each example the results are also variable and it is difficult to draw a solid conclusions.

What is clear is that the metric used to calculate the distances plays a key role in the performance of MDS+DS, as one would expect. In the examples above, the best metrics were presented to keep the presentation compact; many other metrics were evaluated. For example with the microarray data sets, when the Euclidean metric was used the performance dropped significantly (in MSE and Brier score terms). Another issue with gene expression data is that different genes will express at different levels and there may be measurement errors in the data. Using Mahalanobis distance allows the procedure to account for the variability in the data, however it does not account for outliers. Outliers can cause problems with MDS, changing the resulting point configuration in an extreme way. As will be expanded on in section 6.4, choosing a metric that aids the modelling process, rather than one that just happens to generate distances is important.

Another factor may be that the response may not have been non-linear enough to warrant smoothing in the examples above. This is partially captured in the plots of the EDFs against selected dimension (figure 5-14) and in the good performance of the lasso in all situations. There was a hint in the leukaemia data that the results from GCV were more variable than those of ML (as discussed in section 1.2.3), however this does not appear to manifest itself as a general problem with MDS+DS since, looking at figure 5-15 the EDF-dimension relationship exhibits the opposite relationship: ML_P appears much more variable.

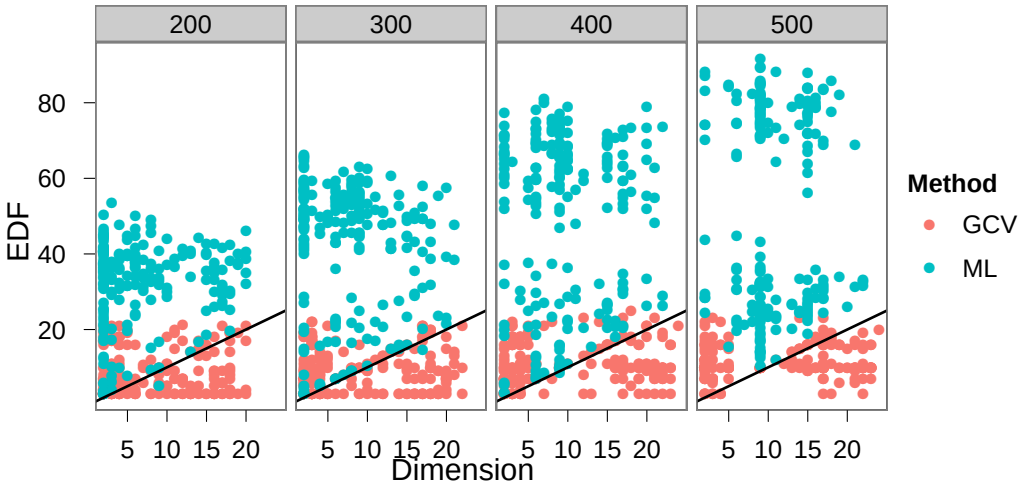

Figure 5-15: Relationship between EDF and selected dimension for the MP voting data set. Diagonal line shows where EDF would equal dimension. In this situation ML_P is much more variable.

An additional consideration in the poor performance in the MP voting data and the leukaemia data is that there response was binary, which makes model checking rather difficult. For the breast cancer data, using NPI, which is a combination of variables may also have complicated matters. Finding data which have a continuous response might yield interesting results (it is possible that in the breast cancer data the sample size was a key issue, even if MDS+DS did marginally outperform the lasso).

Although MDS+DS does not clearly outperform the lasso in any of the examples shown here, there is certainly promise in the method. With a suitable metric (or a way to select the most appropriate metric objectively), general distance smoothing could become very useful. The model has a strong motivation in biological studies, since there is evidence that those who are genetically similar have a greater propensity to contract certain conditions (one only needs to think of hereditary conditions to see this). Having some way of relating the genetic distances between patients to the conditions that they present seems like a logical and useful development, provided that a suitable distance metric can be found.

Chapter 6

Conclusion

This chapter draws to a close the smoothing part of the thesis. It first compares the methods set forth here with the kriging methods alluded to in section 1.4.3, it then goes on to give information about the software implementation of the models discussed so far. Finally, the work from the last five chapters is summarized and further work proposed.

6.1 Comparison with MDS-based kriging methods

Kriging is focused on the explicit modelling of the correlations between points in space as a function of the distance between them via the estimation of the semivariogram. It is therefore logical that in, say, a river system the distances between points are calculated along the river's course rather than the Euclidean distance.

For the semivariogram to be a valid covariance function, it must be positive definite or conditionally negative definite (see Diggle and Ribeiro (2007, p. 47) for more information). However when non-Euclidean distances are used the semivariogram may no-longer fulfil either of these conditions (Curriero, 2006). The work of Løland and Høst (2003) attempts to solve this problem by using multidimensional scaling to project water distances into Euclidean space. Distances are not found exactly, a series of approximations are used rather than directly calculating the distances between the data. First the domain in question is triangulated, then the "river distance" between all of the nodes in the triangulation are calculated via Dijkstra's algorithm. The river distances are then projected using

MDS. Finally, the data locations are mapped into the MDS space by interpolating between the grid points from the triangulation in MDS space. The Euclidean distances in MDS space are then used in the estimation of the semivariogram. There are a number of issues with this approach.

Most prominently, the authors only consider ordinary kriging (where the mean process is treated as constant). In this case spatial variation only enters the model through the semivariogram.The effect of using the MDS projected points for a spatially varying mean process (in addition to the estimation of the semivariogram) has not been investigated. Prevailing opinion is that only polynomial trends should be used for the mean process (Diggle and Ribeiro, 2007, p. 57), how such an approach would perform in higher dimensions is not clear.

Although the approximations used undoubtedly decrease the computational time, the validity of the approximations is not tested, especially on the fitting of the semivariogram (Jensen et al., 2006). The discretisation of the domain necessary to compute the graph distance via Dijsktra's algorithm has similar pitfalls to Wang and Ranalli (2007). Jensen et al. (2006) suggest using the proportion of variation explained or the Bayesian criterion of Oh and Raftery (2001) as possible metrics to perform projection dimension selection but do not full address the issue, resorting to 2-dimensional projections. Neither of the proposed selection methods take into account the effect that the dimension of the MDS projection has on the overall model (as discussed in section 5.2.3). Boisvert et al. (2009) suggest that to best approximate the distances, an $n-1$ dimensional projection of the distance matrix (if there are n data, or triangulation nodes) be used (which is of course true) however they go on to point out that the use of such a high-dimensional projection could lead to numerical problems. Interpolating to find the distances in higher dimensions may also have its own issues and so the approximations may run into further problems. In all of these works the MDS projection is being used to approximate the within-area distances by a set of distances obeying the rules of a Euclidean metric (the criterion given by Curriero (2006) to ensure valid semivariograms). Unlike in the material presented here, the MDS point configuration itself is not being used except to obtain a Euclidean approximation to the distances matrix so that the semivariogram can be estimated.

In general kriging methods suffer from having developed as an *ad hoc* set of tools used in the mining industry (Diggle and Ribeiro, 2007, preface). Although much work has been done to improve the mathematical basis of kriging, models

are not as flexible as GAMs, in particular the incorporation of other covariates, temporal effects and random effects is not straight forward as it is for additive and generalized additive models (in both theory and practice).

6.2 Software implementation - `msg`

The methods detailed in this first part of the thesis: the combination Duchon splines and MDS to perform smoothing (along with the within-area distance algorithm) are provided in the R package `msg` (MultiDimensional Scaling for GAMs) which is available at `http://www.github.com/dill/msg`. Documentation is provided in the package and has been designed to be familiar to users of `mgcv` (`msg` implements the methods detailed here as an extra basis for `mgcv` so minimal code changes are needed to try MDS+DS on existing problems).

6.3 Finite area smoothing conclusion

This part of the thesis started by introducing additive and generalized additive models and the problem which arises when smoothing in a finite area when the boundary is a complex shape: leakage. Current approaches to the problem were then reviewed. Chapter 2 then illustrated the methods of the first chapter in practice. Based on the work in Marra et al. (2011), the first application of the soap film smoother to model spatiotemporal data (via a tensor product formulation) was presented.

Chapters 3, 4 and 5 developed two transformation-based methods to combat leakage. At each stage of the work presented in this thesis the models became more refined and a more nuanced view of how the problem should be addressed was developed. Moving from a strictly functional mapping based on the boundary (the Schwarz-Christoffel transform) to one that preserves within-area distances (MDS) was key to finding a reliable transformation that avoided the artefacts caused by the squashing of space. The discovery that the projections produced by MDS can cause the ordering of the points to be lost in 2-dimensions explained the poor performance of the model up to that point. The final breakthrough was understanding that by projecting into higher dimensions, the ordering problem can be avoided and that by using Duchon splines reliable high dimensional can take place. This final model, MDS+DS, performed very well in simulation, rivalling the soap film smoother.

As well as developing a method which is competitive with the current "best" (the soap film smoother), the investigations in the previous chapters have also revealed a set of essential and a set of desirable properties for transformation methods if they are to be used to perform spatial smoothing. The essential properties are:

1. The mapping of points must be smooth, there should be no sudden jumps or gaps. Points that are near one-another in the original space must be near one-another in the transformed space (section 3.4).

2. The transformation must not squash space too much. Squashing points so they are numerically indistinguishable (crowding) must be absolutely avoided, but less severe compressions of space can also cause problems (section 3.2.5 and section 4.6).

3. Ordering of points must be maintained. If the response values are misordered then modelling becomes impossible (section 4.7.1).

As well as the above, there are other non-essential but desirable properties:

1. To make the method competitive in terms of computational time (chapter 2), the mapping of points from the domain of interest into the transformed space must be fast. This can be achieved by using some kind of functional mapping (chapter 3) or by a sufficiently optimizing the procedure (section 4.5).

2. Being able to integrate the spatial smooth into a larger model incorporating covariates, temporal interactions and random effects is extremely useful in practice (chapter 2 and section 6.1).

3. Creating a method which appears to be familiar to the practitioner, will make the model building process much more streamlined. Combining this with a software implementation in a standard environment with a well-known paradigm can only help users (section 2.4).

The methods proposed in chapters 3, 4 and 5 do not fully achieve all of these goals, however they achieve enough to be viable in practice. The lists above may be useful to those wishing to develop new methods based on transformations of space or further develop the methods presented here.

Moving on to further work, the speed of the algorithm for finding the within-area distances is still an issue, although there are several relatively simple tweaks that could help.

Sticking with the current algorithm, as seen in section 4.3, finding schemes for the layout of starting grids and perhaps adapting the methods described in Løland and Høst (2003) to approximate the distances using a triangulation, would certainly increase performance (although perhaps at the price of accuracy).

As it stands distance generation is seen as a black box procedure to the model. This means that any procedure that can generate a distance matrix can be used. Aside from the discrete space approximation algorithms mentioned in section 4.3, other measures can be used while still keeping in a roughly spatial context. One interesting approach would be to use distances in three dimensions, finding the shortest path over say a mountain range, which would include minimizing changes in altitude as well as avoiding obstactles. Alternatively, a cost based distance approach that takes into account fuel cost or taking into account difficult conditions (e.g. a bog or ford that can be crossed but at additional cost in terms of effort or time). One issue with such general cost-distance approach might be that the "distance" measure could turn out to be non-metric. That is, that the distance from A to B is not the same as the distance from B to A (for example going against versus going with the flow of a river). In this case non-metric MDS must be used, this relies only on the rankings of the data and discards other information, which is probably undesirable.

The examples presented here only consider smoothing inside of simple polygons since the within-area distance algorithm given in section 4.3 can only find shortest paths inside such shapes. This excludes domains with islands in them, which can occur in ship-board studies. This could be worked-around using other shortest path algorithms, however the behaviour of MDS (especially in higher dimensions) in such situations is unknown.

Even given its limitations, MDS+DS does still show an improvement over the soap film smoother in MSE terms, as well as producing reasonable-looking maps in situations with real data. Only further testing on more data sets will show the limitations and the strengths of MDS+DS, for now the method appears to be a useful addition to a practitioner's toolkit.

6.4 Generalized distance smoothing conclusion

The last chapter discussed smoothing in a more general setting, using MDS to project a matrix of dissimilarities that were not spatial in nature. Using general distances is appealing since most measurements have an arbitrary zero point (with the exception of some physical quantities like temperature). What is really of interest in most situations is the differences between the observations, and using these quantities directly makes sense. This is especially true in medical studies since there is evidence that those who are genetically similar are at risk of the same diseases, even if it is not known which genes in particular are indicators of the disease.

Despite the appeal of such a modelling strategy, problems arose. These centred on the choice of distance measure to be used. Choosing certain metrics gave better results than others, so the selection of the metric was down to trying many different options and seeing which was best (in the cases considered here, in terms of MSE, LOOCV or Brier score). The lack of any kind of continuum for the various possible distance measures means that a combination subject specific knowledge and trial and error must be used (rather than automation) to find the appropriate distance measure.

Even if a "correct" distance measure can be found, there is no guarantee that the resulting model will capture the key features of the data. This is due to the nature of MDS, as touched on in section 5.4. MDS is based on taking the eigen-decomposition of the distance matrix, then using those eigenvectors with the largest eigenvalues to represent the points. Those directions with the largest eigenvalues are not necessarily those with the best predictive power. Using scores to select the number of dimensions overcomes this to some degree but because of the hierarchical nature of the projection only the number of dimensions can be selected. Of course, one could imagine the situation where the full MDS projection was found (in, say $n-1$ dimensions) and then variable selection could be performed on all of the possible combinations. This is not appealing if only because of the computational burden of performing the necessary subset selection. Using a full projection also rather subverts the point of the MDS, it would surely be easier to use a more traditional variable selection technique in that case.

In the examples in the previous chapter, the aim of using general distance smoothing has been to perform dimension reduction. However, in the finite area case the idea is to embed further information (about the boundary) into the

distances, these two approaches are rather different. In the finite area case the MDS projected coordinates not only contain information about the position of the points in space in relation to one another, but also their position with respect to the boundary: the within-area distance algorithm and MDS procedure have imbued the projected coordinates with extra information. However, in the general distance case the idea is to discard the data which is not useful, a rather different objective and one that MDS+DS does not appear to excel at.

6.5 Conclusion

This first part of the thesis has developed a transformation-based method for dealing with the problem of leakage in finite area smoothing. The physical model behind MDS+DS is appealing since it does what one would intuitively want to do with a domain with a complex boundary: pull apart those areas of the domain that unduly influence one another. The final model presented in chapter 5 performs at least as well in a spatial setting as the soap film smoother.

The key developments in this parts of the thesis are:

1. First application of the soap film smoother as part of a spatiotemporal model (chapter 2).

2. Rejection of the Schwarz-Christoffel transform as a method for domain transformation, due to its propensity to overly squash together points in the resulting domain (chapter 3).

3. A new algorithm for finding distances between points in simple polygons (chapter 4).

4. Development of a domain transformation method to avoid leakage in finite area smoothing based on preserving within-area distances using multidimensional scaling. Subsequently that when using multidimensional scaling in this context that low-dimensional projections of points can cause a loss of order which is detrimental to smoothing (chapter 4).

5. Application of Duchon splines to avoid the problems associated with thin plate regression splines when performing high dimensional smoothing and use of GCV and REML scores to determine the necessary multidimensional scaling projection dimension in a spatial setting (chapter 5).

6. A general method for smoothing dissimilarities using multidimensional scaling to project the data (chapter 5).

Further work includes adapting MDS+DS to work with more complex domains (like non-simple polygons), applications to a wider set of domains and an investigation of utility of the method in larger models. The further development of the generalized distance smoothing ideas in the last chapter may prove extremely useful, provided that the issues surrounding the choice of metric can be addressed (particularly in a medical setting). For now it is hoped that MDS+DS becomes a useful tool for those performing spatial modelling in complex domains.

Part II

Distance sampling

Chapter 7

Introduction to distance sampling

Distance sampling (Buckland et al., 2001, Buckland et al., 2004) is a popular method for estimating the abundance of biological populations. It has been used by researchers across the globe to assess the abundance of everything from birds nests to marine mammals. Surveys are cheap to run since they do not require many observers (unlike a census) or multiple site visits (unlike mark-recapture). Distance sampling is also rather different from methods like mark-recapture (King et al., 2011) as it does not explicitly include the abundance in the likelihood, as shall be seen below. The popularity of distance sampling is in part due to the software Distance (Thomas et al., 2010) which makes it easy to record and analyse distance sampling data.

7.1 From quadrat sampling to distance sampling

One can think of distance sampling as the logical extension of quadrat and strip transect sampling. In quadrat sampling a series of squares (quadrats) are laid out at random over the sample area and the number of objects of interest within each is counted. It is assumed that within each quadrat a census is performed. From the per-quadrat abundance the density is estimated and multiplied-up to find the total abundance. For quadrat sampling to be efficient the quadrats need to be large and hence it is almost impossible to ensure that all objects in the quadrat are seen, this can be further hindered by animals moving between the quadrats during the survey (Buckland et al., 2001, p. 2).

To make the task of counting the objects within the quadrat easier, one could modify the square design to be a long strip, so that the observer could walk down the centreline of the strip, observing those objects within the strip. Mathemati-

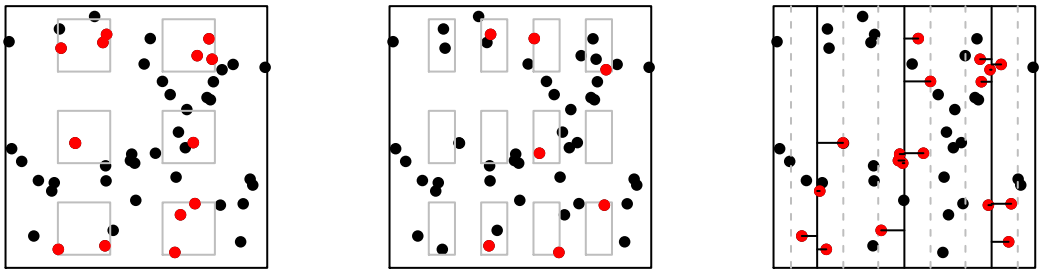

Figure 7-1: An example of quadrat sampling (left), strip transect sampling (middle), and distance sampling (right). Dots indicate individuals, red dots are observed individuals, black those missed. In the first two cases, the grey boxes represent the sampling units. Note that there are many observations just outside of the boxes, which cannot be recorded by survey staff. In the distance sampling case, the solid vertical lines represent the transects and the dashed line gives the effective strip width. Distances are shown by the solid horizontal lines.

cally, if we let the each strip be of width $2w$ (w either side of the line the observer walks down) and the sum of all strip lengths be L, if n objects are observed we have a simple estimator of the density, D:

$$\hat{D} = \frac{n}{2wL}. \tag{7.1}$$

The problem with both quadrat and strip sampling is that there may well be many objects just outside of the covered area. Clearly this is a waste of survey effort, since observers must ignore objects that they have seen but that are not within the strip. It would be preferable to include as many observations as possible and leverage the maximum amount of data that can be collected to assess the abundance of the population.

Distance sampling is based on this principle; if the objects of interest are seen, then their presence should be recorded. Instead of using fixed-area sampling units, distance sampling requires that only centrelines are specified. The observers should walk (or swim, ride, drive, sail, etc) down the centrelines recording the distances (x_i) to the observed objects as they go. Once the survey is complete, the distances are used to estimated the effective area that was sampled. Figure 7-1 shows the evolution from quadrat to strip to distance sampling.

In equation (7.1) one can think of replacing w with an estimate of μ, the *effective strip (half-)width*. This is the distance at which as many animals were

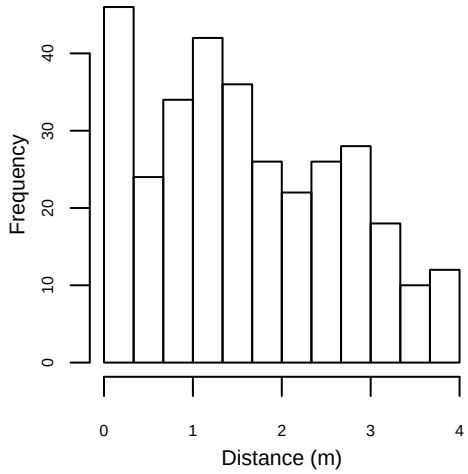

Figure 7-2: A histogram of line transect data. In this case from an experiment conducted at the University of St Andrews. 760 golf tees were randomly distributed over a 1680m^2 area, then observed in 11 transects by 8 independent surveys. Further detail may be found in Buckland et al. (2004, p. 140) and Borchers et al. (2002).

detected beyond as there were missed inside. Further explanation of μ is given in section 7.4, but it serves for now to say that by replacing w with μ an estimate of the area that was effectively surveyed can be found. (7.1) can then be modified to:

$$\hat{D} = \frac{n}{2\hat{\mu}L}. \tag{7.2}$$

We could also consider that a certain proportion ($\hat{p}$, the probability of detection) of the objects in a fixed-area ($2wL$) were sampled, so the above can also be expressed as:

$$\hat{D} = \frac{n}{2wL\hat{p}}.$$

Here w is the point after which observations are discarded and is referred to as the *truncation* distance. Truncation is used to discard outliers that make the estimation process tricky (Buckland et al., 2001, pp. 15-16). From these two expressions we can see that the relationship between p and μ is $p = \mu/w$, these quantities will be investigated further below.

A typical example of line transect data is shown in figure 7-2. The figure shows a histogram of perpendicular distances. Note how, as distance increases the number of detections decreases. This characteristic will be exploited later.

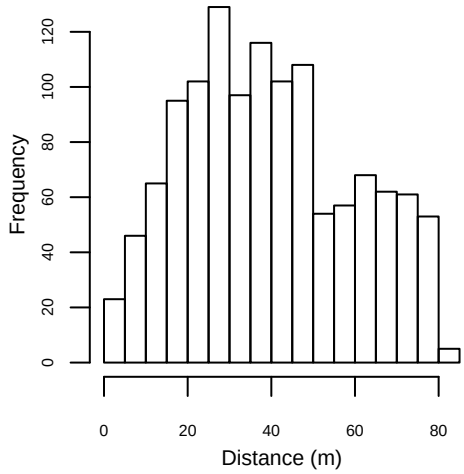

Figure 7-3: A histogram of point transect data of Hawaiian amakihi (*Hemignathus virens*) taken from Marques et al. (2007).

7.1.1 Point transects

Line transects are not the only way of collecting data for a distance sampling analysis; point transects may also be used. When using point transects the observer stands at one of a series (m, say) of points and observes the objects surrounding him/her. Again, distances to the objects (r_i) are recorded. An *effective radius* (ρ) is then calculated (analogously to μ) and then object density can be estimated by:

$$\hat{D} = \frac{n}{m\pi\hat{\rho}^2} = \frac{n}{m\pi w^2 \hat{p}}.$$

The relation between these quantities will be explained below in section 7.4.

An example of point transect data is given in figure 7-3. In contrast to the line transect case, there are very few observations near 0, they increase to a point and then fall off beyond that. Note that as the distance, r, from the observation point increases the area surveyed increases as r^2. Rescaling this histogram by the distance to the midpoints of each bin will give a histogram that has a similar shape to that of figure 7-2 (i.e. the rescaling accounts for the increasing area available to the observer as the distance from the point increases).

7.2 Assumptions

In order to ensure that estimation is unbiased several assumptions are made. It is first assumed that the objects are distributed throughout the survey region according to some stochastic process. Line or point placement must be random with respect to the distribution of objects; given that this is done, it can be safely assumed that the objects are distributed uniformly (i.e. the process is stationary; see Buckland et al. (2004, p. 49)) from the point or line. (Buckland et al., 2001, p. 29).

In order to obtain reliable estimates of the density from the point or line, the following three assumptions must hold:

- Objects on the line (or point) are detected with probability 1.
- Objects are observed in their initial location, not after movement in response to the observer.
- The recorded distances are accurate.

Further assumptions regarding field procedure may be found in Buckland et al. (2001), chapter 2.

7.3 The detection function

One would expect that the probability of observing an object would decrease as the distance from the observer increased (as in figure 7-2). This is the relationship captured by the detection function ($g(x)$) which is defined as (Buckland et al., 2001, p. 10):

$$g(x) = \mathbb{P}(\text{object detected}|\text{object was at distance } x). \tag{7.3}$$

The goal in distance sampling is to accurately model the detection function and through this estimate the effective strip width (or effective radius), see section 7.4 below. Before talking about models for $g(x)$, we look at the desirable properties.

First, as stated in the assumptions above, objects on the line (or point) are detected with certainty, so $g(0) = 1$. Second, it is preferable to have a model where detection is almost certain near zero distance, i.e. that the function has a *shoulder*. This is physically realistic since the observer should see most things close to him/her (not just those directly under/in front of him/her). Third, it is

also desirable that the model for the detection function is robust, in the sense that it is a general, flexible model that can take many plausible shapes (see Buckland et al., 2001, p. 41). Finally, it is desirable to have a model that is efficient, in the sense that estimates have a relatively small variance, however this is only of use when the other criteria are met.

Buckland (1992) gives a "key function plus adjustment terms" formulation for the detection function. In this formulation the key function (denoted k) is used as a starting point for the basic shape of the detection function (it has certain detection at zero distance and has a shoulder). The adjustment terms (each denoted s_j) consist of a series expansion that improve the fit of the model. The model is written as:

$$g(x;\boldsymbol{\theta}) = \frac{k(x;\boldsymbol{\theta}_k)\{1+\sum_j s_j(x;\boldsymbol{\theta}_s)\}}{k(0;\boldsymbol{\theta}_k)\{1+\sum_j s_j(0;\boldsymbol{\theta}_s)\}},$$

where $\boldsymbol{\theta}$ is a vector of all of the parameters ($\boldsymbol{\theta} = (\boldsymbol{\theta}_k, \boldsymbol{\theta}_s)$) and the index of summation depends on the form of adjustments. The denominator ensures the detection function is 1 at zero distance.

The key function, k, is usually selected as one of:

$$k(x,\boldsymbol{\theta}_k) = \begin{cases} 1/w & \text{(uniform)} \\ \exp\left(-\frac{x^2}{2\sigma^2}\right) & \text{(half-normal)} \\ 1-\exp\left(-\left[\frac{x}{\sigma}\right]^b\right) & \text{(hazard-rate).} \end{cases}$$

In each of the above cases $\boldsymbol{\theta}$, the parameter(s) to estimate is/are: non-existant (no parameters are estimated for uniform key), σ (known as the *scale parameter* and (σ, b) respectively (b is known as the *shape parameter*). The adjustment terms are then one of:

$$s_j(x,\boldsymbol{\theta}) = \begin{cases} 0 & \forall j \text{ (no adjustments)} \\ a_j\left(\frac{x}{w}\right)^{2j} & j=1,\ldots,J \text{ (simple polynomial)} \\ a_j\cos\left(\frac{j\pi x}{w}\right) & j=2,\ldots,J \text{ (cosine)} \\ a_j H_{2j}(x) & j=2,\ldots,J \text{ (Hermite polynomial).} \end{cases}$$

In each case we wish to estimate the a_js (so $\boldsymbol{\theta}$ is a vector of a_js). More information on the formulation can be found in Buckland et al. (2001, p. 47–48) and Buckland (1985). Figure 7-4 shows some possible detection functions.

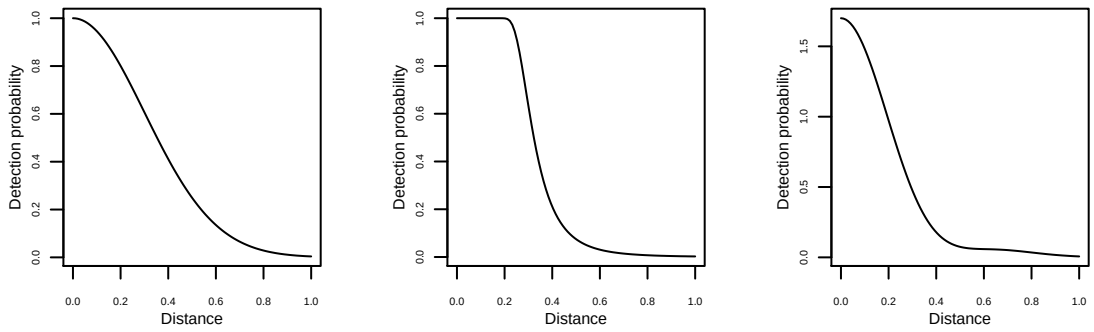

Figure 7-4: Three possible detection functions. The first is a half-normal distribution, the second a hazard-rate function and the third the same half-normal as the first but with a cosine adjustment term. The hazard-rate function has a controllable "shoulder".

This formulation leads to a class of highly flexible models. Data analysis usually consists of running models made up of various combinations of key functions and adjustment terms. Model parameters are found using maximum likelihood (see section 7.4). Model selection is performed via AIC (Buckland et al., 2001, p. 69), that is:

$$\text{AIC} = -2\hat{l} + 2P, \tag{7.4}$$

where $\hat{l}$ is the value of the log-likelihood at the MLE and P is the number of parameters in the model. Models with many parameters are penalized more heavily than those which are more parsimonious.

7.4 From $g(x)$ to D

Equation (7.2), above shows that in order to estimate the density of the population in question, we must find an estimator of μ, the effective strip width (or, equivalently for point transects, ρ). To do this the detection function is used.

7.4.1 Line transects

As mentioned above, μ is defined to be the distance from the lines for which as many objects are detected beyond μ as are missed within μ (Thomas et al., 2002). Looking at figure 7-5, the shaded area above the detection function represents those objects that the observer missed up to a distance μ from the line. The shaded area below the curve represents those objects that were observed beyond

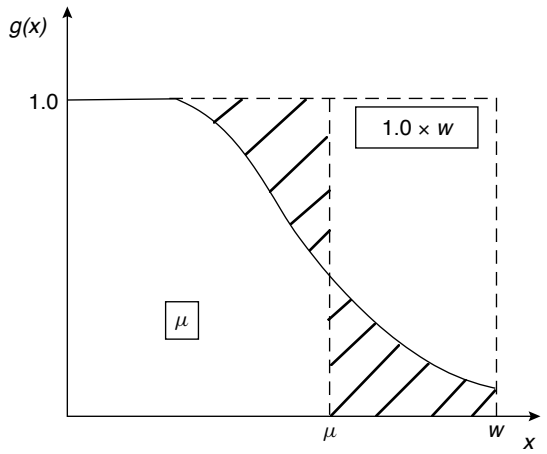

Figure 7-5: A detection function $g(x)$ with the effective strip width μ marked as well as the truncation distance w. The shaded regions have equal area, this means that the area under the curve has the same size as the rectangle with base length μ. Figure taken from Buckland et al. (2001).

a distance μ. By moving μ to the left or to the right, it is possible to find a point at which the two shaded areas are of equal size. This fulfils the criterion for μ.

Now the question is: how is μ calculated and how does this relate to the detection function? Note that the rectangle with side 0 to 1 on the y axis and 0 to μ on the x axis has area μ and that this is the same as the area under the detection function (by the argument above). So μ can be defined as:

$$\mu = \int_0^w g(x)\mathrm{d}x, \tag{7.5}$$

where w is again the truncation distance and ignoring the exact form of $g(x)$. Hence p is defined as:

$$p = \frac{\int_0^w g(x)\mathrm{d}x}{w}.$$

So, an estimator of density may be written as:

$$\hat{D} = \frac{n}{2w\hat{p}L} = \frac{n}{2\hat{\mu}L}.$$

That is, the density is estimated by the number of observations divided by the effective area that was surveyed. Since $\hat{\mu}$ gives the distance to which as many objects were observed as missed, the area can be treated like a strip transect where everything was observed out to $\hat{\mu}$, the strip is $2\hat{\mu}$ wide (since we are only estimating the half-width), and L long.

The detection function is one of the functions described in section 7.3 and so only its parameter(s) must be estimated. However to first form the likelihood, the probability density function of the distances must be defined. Note that the expected number of objects at a distance x from the transect line (including those not observed) is independent of x. This then implies (for line transects) that the shape of the density functions is the same as that of the detection function and can therefore be obtained by rescaling (Buckland et al., 2001, p. 38). So, the PDF of the perpendicular distance data, conditional on the object being observed is then:

$$f(x;\boldsymbol{\theta}) = \frac{g(x;\boldsymbol{\theta})}{\int_0^w g(x;\boldsymbol{\theta})\mathrm{d}x} = \frac{g(x;\boldsymbol{\theta})}{\mu},$$

where $\boldsymbol{\theta}$ is a vector of all of the parameters of g.

Now we have obtained an expression for the PDF, we can form a likelihood:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta};\boldsymbol{x}) &= \prod_{i=1}^{n} f(x_i;\boldsymbol{\theta}),\\ &= \prod_{i=1}^{n} \frac{g(x_i;\boldsymbol{\theta})}{\mu},\end{aligned} \tag{7.6}$$

where $\boldsymbol{x}$ is an n-vector of the observed distances.

The log-likelihood can then be used as an objective function in an optimization procedure in order to find the maximum likelihood estimators of the parameters.

7.4.2 Point transects

For point transects, instead of effective strip width, we look at effective radius. This effective radius, ρ is defined in the same way as μ: that there are as many objects missed up to ρ as observed beyond. Related to the effective radius is the *effective area of detection*, $\nu = \pi\rho^2$.

For line transects, an infinitesimal strip has area $l\mathrm{d}x$ (the length of the line multiplied by an infinitesimal distance perpendicular to the line) and this value is independent of x. However, in the point transect case an incremental annulus depends on r (the distance from the point), such an annulus has area $2\pi r\mathrm{d}r$. So, the effective area of detection is therefore defined as:

$$\nu = 2\pi \int_0^w r g(r)\mathrm{d}r.$$

Then, following through the arguments above, we can define the PDF as:

$$f(r) = \frac{rg(r)}{\int_0^w rg(r)\mathrm{d}r},$$

and hence:

$$f(r) = \frac{2\pi rg(r)}{\nu}.$$

A more rigorous proof of this is given in Buckland et al. (2001, p. 54).

By analogy to the line transect case, the relation between ν and p in the point transect case is

$$p = \frac{\nu}{\pi w^2},$$

so:

$$f(r) = \frac{2\pi rg(r)}{\pi w^2 p}.$$

Again, a likelihood can be formed:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}; \boldsymbol{r}) &= \prod_{i=1}^{n} f(r_i; \boldsymbol{\theta}), \\ &= \prod_{i=1}^{n} \frac{2\pi r_i g(r_i; \boldsymbol{\theta})}{\nu},\end{aligned}$$

where $\boldsymbol{r}$ is an n-vector of the observed distances.

As above, the log of this expression can then be used in an optimization procedure to find the MLEs of the parameters.

7.5 Multiple covariate distance sampling

One can very easily think of the case in which one or more factors (as well as distance) affect the detectability of objects in the survey. A typical example might be the sex or size of the object or weather conditions at the time of observation. When available this information can be very useful in handling heterogeneity in the detectability (Buckland et al., 2001, p. 88). Covariates are included in detection function models by using a link function (Borchers, 1996; Marques and Buckland, 2003). Figure 7-6 shows the effect of both continuous and factor covariates on a hazard-rate detection function.

Non-covariate distance sampling, as in the previous section, is commonly referred to as CDS (conventional distance sampling) and covariate distance sam-

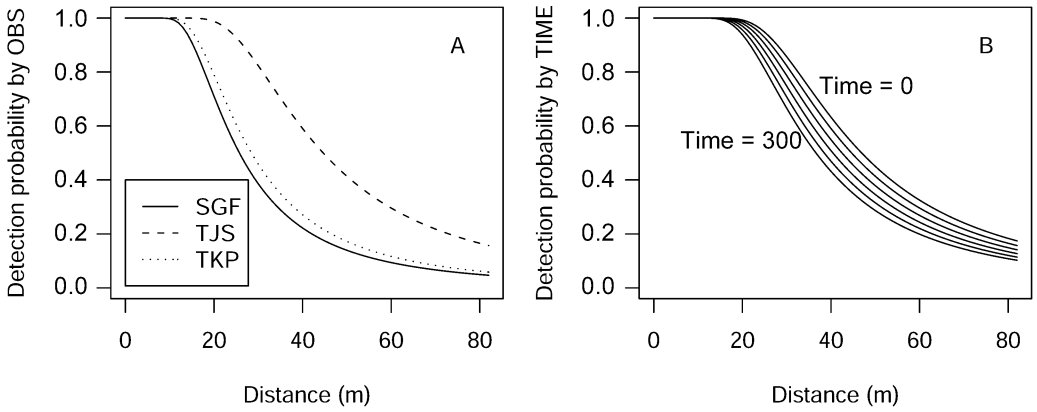

Figure 7-6: The influence of covariates on the detection function (taken from Marques et al., 2007). The detection functions are for Hawaiian amakihi (*Hemignathus virens*). The left panel shows the effect of a factor covariate (observer), while the right shows the effect of a continuous covariate (time). In each case the other covariate is held constant (time at 0900 and observer as "TJS", respectively) while the other is varied.

pling as MCDS (multiple covariate distance sampling).

Referring back to the detection functions given in section 7.3, the covariates affect the scale parameters (σ) of the hazard-rate and half-normal detection functions only. It is also assumed that the distributions of the distances and covariates are independent (Marques and Buckland, 2003). Since the distribution of the covariates is unknown, the conditional distribution of the distances, given the observed covariates is modelled.

Say that for observation i a set of K covariates are collected and denote them store them in the K-vector $\mathbf{z}$ (containing $z_{i1}, \ldots, z_{iK}$). The definition of the detection function in (7.3) is now:

$$g(x, \mathbf{z}) = \mathbb{P}\left(\text{object detected}|\text{object was at distance } x \text{ with covariates } \mathbf{z}\right).$$

So then defining the scale parameter on a per-observation basis, we have:

$$\sigma_i = \exp(\beta_0 + \sum_{k=1}^{K} \beta_k z_{ik}). \tag{7.7}$$

Now, rather than estimating the scale parameter, we estimate the β_ks. This covariate formulation can be thought of as a generalisation of the non-covariate model, the latter simply being the case where there is only an intercept term.

Note that it may be necessary to standardize the distances by dividing them by either the scale parameter or the truncation distance, guidance on when to use which standardization is given in Marques and Buckland (2003).

This formulation fits nicely into the likelihood expressions above (although note that for line transects $f(x_i;\boldsymbol{\theta})$, is replaced by $f(x_i|\mathbf{z}_i;\boldsymbol{\theta})$ and similarly for point transects), the evaluations of the detection function are as above, but the calculation of μ (and therefore p) changes.

The effective strip width, μ_i is now expressed as:

$$\mu_i = \int_0^w g(x;\sigma_i)\mathrm{d}x, \tag{7.8}$$

that is, that the effective strip width depends on the covariate values. So in the covariate case there is no single probability of detection, instead the per observation (equivalently, per unique covariate combination) probability of detection can be calculated for line transects:

$$p_i = \frac{\int_0^w g(x;\sigma_i)\mathrm{d}x}{w}.$$

For point transects, the probability of detection and effective area are given as:

$$p_i = \frac{2}{w^2}\int_0^w rg(r;\sigma_i)\mathrm{d}r, \qquad \nu_i = 2\pi\int_0^w rg(r;\sigma_i)\mathrm{d}r.$$

7.5.1 Estimating population size

Population size can be found either by multiplying D by the survey area or from a Horvitz-Thompson-like estimator (Thompson, 2002, pp. 53-56, Buckland et al., 2004, p. 23).

For the covariate models the population size may be estimated by:

$$\hat{N} = \sum_{i=1}^{n}\frac{1}{c_i\hat{p}_i}, \tag{7.9}$$

where c_i is the *coverage probability* (the probability of an individual lying within the surveyed area), and $\hat{p}_i$ is the probability of the ith observation being detected given it is within the sampled area (Buckland et al., 2004, p. 7 and 38). For the analyses presented here it is assumed that the probability of being in the sampled

area is constant across the observations ($c_i = c$) and that:

$$c = \frac{2wL}{A},$$

where A is the area of the surveyed region. For non-covariate models, this simplifies to:

$$\hat{N} = \frac{n}{2wL\hat{p}}A,$$

for line transects and

$$\hat{N} = \frac{n}{m\pi w^2\hat{p}}A,$$

for point transects.

A standard summary statistic is the *average detection probability* for an animal within the covered region, $\hat{P}_a$, which is given by:

$$\hat{P}_a = n/\hat{N}.$$

7.5.2 Plotting covariate models

In both this and the next chapter it will be necessary to plot the detection function for a covariate analysis. There are (at least) two ways of plotting the detection function in a covariate analysis. The first is as shown in figure 7-6, where all but one of the covariates are held at a particular level and the other is varied (over say its levels or quantiles). The other approach (which will be adopted from here) is to consider a covariate analysis as effectively fitting a detection function to each unique covariate combination, so averaging over the detection functions point-wise should give an indication of the overall shape of the detection function. In this case we evaluate the detection function at each unique covariate combination over distances ranging from 0 to w, then simply average their values at each distance to create an overall average detection function. To plot levels or quantiles, the requisite covariate is fixed to that level if there is only one covariate. If there is more than one covariate, the average is performed as before fixing the covariate to its chosen value. This approach is useful here since the analyses are designed to highlight the possible shapes of the detection function, rather than to discover ecologically useful or informative covariates.

From here on, when the phrase "average detection function" is used, this refers to a point-wise average of the possible detection functions (i.e. all of the unique values of the scale parameter) over the range $(0, w)$. When quantile/level

plots are shown, it is the case that the quantile is fixed and the detection function is averaged over the other values. In each case the detection functions are normalised so that $g(0|\cdot) = 1$.

7.6 Other considerations

7.6.1 Line and point placement

In section 7.2, the placement of the lines and points above is said to be "random" with respect to the distribution of objects. In practice randomly placing and orientating lines can be expensive and time-consuming (in particular in shipboard surveys where one wishes to minimize off-effort time). The solution to this is simply randomly placing and orientating a grid of lines or points (Buckland et al., 2001, p. 2). For shipboard surveys "zigzag" designs can be used to minimize off-effort time (Strindberg and Buckland, 2004). It is also important to ensure that transects to not run parallel to geographical features as doing this will incur bias. For example using roads as transects (as was done in the US breeding bird survey) leads to bias since animals may be compelled to move away from the road and toward neighbouring hedgerows (Buckland et al., 2001, p. 18).

7.6.2 Clusters

If animals are observed in clusters (for example pods for whales or packs for wolves) then it might be more convenient to estimate the abundance of clusters and use them as the fundamental unit to estimate. The cluster size can also be estimated and the abundance of clusters "multiplied up" to give the overall abundance (Buckland et al., 2001, p. 13). It is assumed throughout that individuals rather than clusters are being addressed but the method developed here should also work on clusters.

7.6.3 Goodness of fit testing

Although AIC is a good measure of relative fit of a model, some formal absolute measure of goodness of fit is also useful (the best of a bad lot is still bad). χ^2 testing has been suggested (Buckland et al., 2001, pp. 69-71), however the choice of interval is subjective. As a replacement, Kolmogorv-Smirnov and Cramer-von Mises tests (Buckland et al., 2004, pp. 385-389) are suggested. Both are used

to compare empirical to cumulative distribution functions (EDFs and CDFs, respectively). The Kolmogorov-Smirnov test uses the largest difference between the fitted CDF and the EDF as a test statistic, with the null hypothesis that the functions are the same. The Cramer-von Mises test has the same null hypothesis but the test statistic is instead based on the differences between the CDF and EDF over their entire range. The Kolmogorov-Smirnov test is used in the analyses in the next chapter.

7.7 Monotonicity

One potential pitfall of both CDS and MCDS is that it is possible to formulate models which are not physically realistic. In particular it is possible to create models for the detection function which are non-monotonic functions of distance. Data with a mode away from zero distance may occur when there has been heaping (when observers "eyeball" distances and round to convenient numbers like 5, 10, 20m etc, Buckland et al., 2001, pp. 34-35), when objects move prior to observation. Fitting models to such "bumps" can cause bias in abundance estimates (Buckland et al., 2001, p. 132). To get around this problem the software package Distance (Thomas et al., 2010) constrains the detection function to be monotonic. This is done by taking 10 equally spaced distances from 0 to w and checking that when the detection function is evaluated at each of these points they are less than the last ($g(x_i) \geq g(x_{i+1})$ for distances $x_1 \ldots x_{10}$ where $x_1 = 0$). This is referred to as *strong monotonicity*. Alternatively, *weak monotonicity* may be enforced, where each point is checked only against the value of g at the origin ($g(0) \geq g(x_i)$).

Constraining the shape of the detection function does not necessarily always lead to monotonic detection functions since the constraints can only be applied at a finite number of points. This can lead to constraints missing the non-monotonic points in the function. An example from Williams and Thomas (2007) is shown in the first plot of figure 7-7. Here a half-normal detection function with one second order cosine adjustment term was fitted to humpback whale sightings. In the MCDS case, Thomas et al. (2010) do not constrain the detection function. The second and third panels in figure 7-7 show a detection function when covariate data was included in the model. In this case for long-finned pilot whales (Pike et al., 2003) the Beaufort sea state was added as a covariate, the first plot shows that the detection function, when averaged over the covariate values,

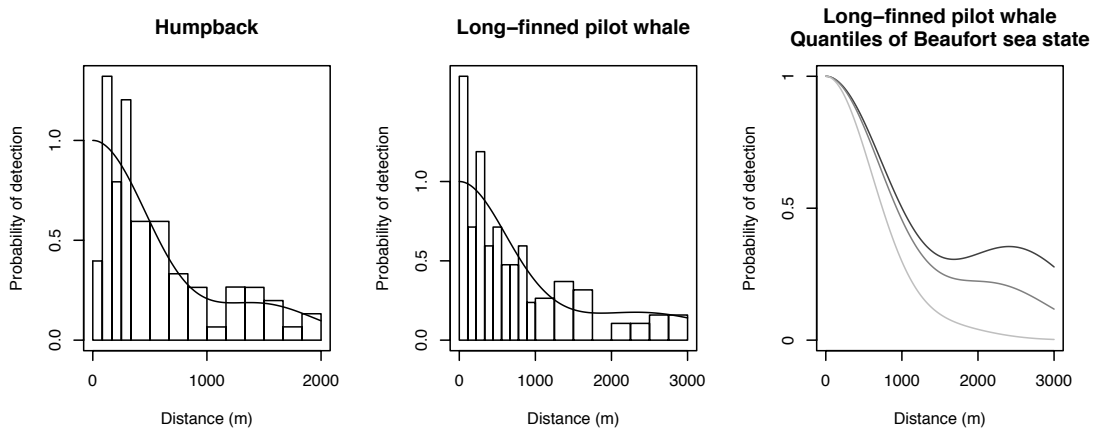

Figure 7-7: Two examples of detection functions which are not monotone. The first panel is data from humpback whale (reproduced from data in Williams and Thomas (2007)), a half-normal detection function with cosine adjustments provided the best fit to the data, even with constraints in place, the detection function is non-monotonic. The second and third panels show plots of a half-normal detection function with cosine adjustments for the long-finned pilot whale data taken from Pike et al. (2003). The second panel shows the average detection function (as described in section 7.5.2) and the third shows the detection function when the covariate values for the Beaufort sea state are set to the values 1.5, 2 and 3 (from light to dark), showing that the non-monotonicity gets worse at higher levels of the covariate.

shows some non-monotonic behaviour. However, in the third panel shows the detection function the covariate is fixed to the values 1.5, 2 and 3 (from bottom to top) and here the non-monotonicity is particularly pronounced.

Although constrained optimization is appealing, it is obviously always preferable to perform unconstrained optimization if possible. Using a class of functions to model the detection function which were both flexible and did not exhibit the undesirable property of non-monotonicity could offer a more convincing and physically realistic alternative to the conventional way of performing distance analyses.

Aside from the monotonicity constraints that can be used in a CDS analysis in the Distance software, the problem of monotonicity has not been directly addressed in the literature to date. In practice, it is often the case that non-monotonicity is ignored in the hope that the resulting bias incurred is not too high. It is also possible that those analysing the data are unable to observe the non-monotonic nature of the detection function (for example in a complex covariate model). Innes et al. (2002) avoid a non-monotonic detection function

by aggressively truncating the data (i.e. reducing w); this approach is rather wasteful, particularly when most surveys are expensive to set up and run. Using all of the data, whilst maintaining monotonicity is certainly preferable.

7.8 Mixture models

Recent developments, particularly in mark-recapture (Pledger, 2000, Dorazio and Royle, 2003, Pledger, 2005 and Morgan and Ridout, 2008), have shown that mixture models can be an extremely useful and flexible method of modelling heterogeneity in biological populations. Their main utility has been in better accounting for between-individual heterogeneity which can cause severe bias if unmodelled (Link, 2003).

In distance sampling bias due to unmodelled heterogeneity is not severe unless the heterogeneity is extreme (Buckland et al., 2004, pp. 389-392), provided that the detection at zero distance is certain and a flexible detection function model is used. So called *pooling robustness* allows abundance estimates of the whole population to have low bias (although abundance estimates of subpopulations, e.g. males or females alone, may be biased).

Mixture models offer the potential for flexible modelling of the detection function. They also have the appealing property that if the individual parts of the mixture model (the *mixture components*) are each monotonic decreasing then a sum of these functions will also be monotonic decreasing. Each constituent component of the mixture model is simple but the combinations yield a great many possible shapes (see figure 8-2). Using such a set of models avoids constrained optimization.

Finally, the approach is also interesting for its own sake. There is no current literature on the use of mixture models as detection functions for distance sampling. Many detection function forms have been proposed (Buckland, 1992 and Becker and Quang, 2009), each having their own merits and pitfalls. Therefore, there may be useful and unexplored properties of using a mixture model for the detection function that have not been previously considered.

A mixture model approach to distance sampling detection functions is developed further in the next chapter.

7.9 Summary

Distance sampling is unlike many statistical methods, in that the quantity which we wish to find, abundance, is not given explicitly in the likelihood we wish to optimize. Instead we wish to find the parameters for the detection function, to then estimate of μ, in order to estimate density (and hence the abundance). Full likelihood methods for distance sampling require that probabilistic models for the animal distribution be specified which may well be very tricky. CDS and MCDS specify probability models only for the distance parts of the data and then assume that the density is log-normally distributed to obtain variances and confidence intervals. Although this means that estimators do not have properties such as asymptotic efficiency, they do avoid the specification of the animal distribution or, for MCDS, the joint distribution of the covariates and distances (see Buckland et al., 2004, p. 6 and pp. 31-33).

Distance sampling benefits from a relatively simple field procedure, a wealth of literature and easy to use software for analysis. Distance sampling has also been adapted for many different scenarios, including analysing data which were not initially part of a survey (incidental data).

Other variants of distance sampling exist, for example mark-recapture distance sampling (MRDS, Borchers et al., 1988) and spatial distance sampling models (Buckland et al., 2004, chapter 4), although these are not addressed here.

In the next chapter, a mixture model approach for distance sampling detection functions will be proposed and applied to both simulated data and to case studies.

Chapter 8

Mixture model distance sampling detection functions

8.1 Introduction

As was shown in chapter 7, distance sampling (Buckland et al., 2001, Buckland et al., 2004) is a suite of methods for estimating the size and/or density of biological populations. The most common approach, referred to as conventional distance sampling (CDS), is based on methods developed by Buckland (1992). A commonly used extension to this methodology allows for the probability of detection to vary with covariates as well as just distance from the observer (multiple covariate distance sampling, MCDS, Buckland et al., 2004, chapter 3).

Estimating the detection function is key to distance sampling. Standard distance sampling methods use the "key function plus adjustment series" formulation for the detection function. This can lead to unrealistic functions being fitted to the data, in particular non-monotone detection functions (as seen in section 7.7), in particular in figure 7-7.

In this chapter, a new class of distance sampling detection function models based on mixtures of simple parametric key functions is introduced. In the next section the models are described and following from that section 8.3 gives details of the optimization procedure and parametrization. Sections 8.4 and 8.5 show the results of a simulation study and then analysis of real data, respectively. The final section gives some concluding remarks.

8.2 Finite mixture model detection functions

Following on from the definitions of the detection functions given in section 7.3, this section shows how from the definition of a mixture model detection function, the likelihood can be found for covariate and non-covariate models of both line and point transect data.

8.2.1 Formulation

Denoting the detection function g, as before, consider a sum of J mixture (detection function) components g_j, scaled by some mixture proportions ϕ_j:

$$g(x, \mathbf{z}; \boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{j=1}^{J} \phi_j g_j(x, \mathbf{z}; \boldsymbol{\theta}_j), \tag{8.1}$$

where $\sum_{j=1}^{J} \phi_j = 1$, $\boldsymbol{\phi}$ is a J-vector of the ϕ_js. As in chapter 7, the distance is denoted x, the $\boldsymbol{\theta}_j$s are vectors of parameters for function g_j, $\boldsymbol{\theta}$ is a vector of all of the $\boldsymbol{\theta}_j$s and $\mathbf{z}$ is a K-vector of covariates (more detail on the covariates is provided in the next section).

Here all of the g_js are chosen to be half-normal functions, although other monotonic functions such as hazard-rate could be chosen (and the g_js need not all have the same form). A non-covariate, half-normal mixture model with J components (henceforth, a J-point mixture) would then be written as:

$$g(x; \boldsymbol{\theta}, \boldsymbol{\phi}) = \sum_{j=1}^{J} \phi_j \exp\Big(- \frac{x^2}{2\sigma_j^2}\Big).$$

So here $\boldsymbol{\theta} = (\sigma_1, \ldots, \sigma_J)$ and there are no covariates so $\mathbf{z}_i$ is removed.

Covariates

Covariates can be included in the same way as MCDS (section 7.5, Buckland et al., 2004, Chapter 3): it is assumed that each mixture component has a different scale parameter, but that the covariates affect all of the scale parameters in the same way. This means that the "intercept" term in the covariate model is different for each mixture component but the other covariates are estimated jointly for all components. Again, the detection function is modelled as conditional on a given set of covariates. Other more complex models with different covariates in each

mixture component may be possible, but only the simplest model is considered here.

Analogously to section 7.5, using i to subscript each observation, the formulation for the scale parameter σ_{ij}, is therefore:

$$\sigma_{ij} = \exp(\beta_{0j} + \sum_{k=1}^{K} \beta_k z_{ik}),$$

where z_{ik} is the k^{th} covariate for the i^{th} observation. It is therefore possible to separate the "individual" and "mixture" parts of the scale parameter since $\sigma_{ij} = \sigma_i \sigma_j$.

As an example, for a 2-point mixture with 1 covariate, the scale parameters then have the form:

$$\sigma_{i1} = \exp(\beta_{01} + \beta_1 z_{i1}) \quad \text{and} \quad \sigma_{i2} = \exp(\beta_{02} + \beta_1 z_{i1}) \text{ for } i = 1, \ldots, n,$$

and then $\boldsymbol{\theta}_j = (\beta_{0j}, \beta_1)$ and $\boldsymbol{\theta} = (\beta_{01}, \beta_{02}, \beta_1)$. Note the similarity to (7.7).

8.2.2 Likelihood

Line transects

For line transects, given an n-vector of perpendicular distances $\mathbf{x} = (x_1, \ldots, x_n)$ and associated covariate vectors $\boldsymbol{z}_i$ (the rows of $\mathbf{Z}$), the conditional likelihood is given by:

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{Z}) &= \prod_{i=1}^{n} f(x_i | \boldsymbol{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi}) \\ &= \prod_{i=1}^{n} \frac{g(x_i, \boldsymbol{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi})}{\mu_i} \\ &= \prod_{i=1}^{n} \frac{\sum_{j=1}^{J} \phi_j g_j(x_i, \boldsymbol{z}_i; \boldsymbol{\theta}_j)}{\mu_i} \end{aligned} \tag{8.2}$$

where μ_i, the effective strip width for an observation with the same covariates as observation i, is given by substituting (8.1) into (7.8):

$$\begin{aligned}\mu_i &= \int_0^w g(x, \boldsymbol{z}_i; \boldsymbol{\theta}) \mathrm{d}x, \\ &= \int_0^w \sum_{j=1}^J \phi_j g_j(x, \boldsymbol{z}_i; \boldsymbol{\theta}_j) \mathrm{d}x, \\ &= \sum_{j=1}^J \phi_j \int_0^w g_j(x, \boldsymbol{z}_i; \boldsymbol{\theta}_j) \mathrm{d}x.\end{aligned}$$

Point transects

For point transects, with radial distances $\mathbf{r} = (r_1, \ldots, r_n)$ and associated covariate vectors $\boldsymbol{z}_i$, the likelihood is:

$$\begin{aligned}\mathcal{L}(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{r}, \mathbf{Z}) &= \prod_{i=1}^n f(r_i, \boldsymbol{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi}) \\ &= \prod_{i=1}^n \frac{2\pi r_i g(r_i, \boldsymbol{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi})}{\nu_i} \\ &= \prod_{i=1}^n \frac{2\pi r_i \sum_{j=1}^J \phi_j g_j(x_i, \boldsymbol{z}_i; \boldsymbol{\theta}_j)}{\nu_i}\end{aligned} \tag{8.3}$$

where the effective area of detection for an observation with covariate vector $\mathbf{z}_i$, ν_i is defined as:

$$\begin{aligned}\nu_i &= 2\pi \int_0^w r g(r|\boldsymbol{z}_i; \boldsymbol{\theta}) \mathrm{d}r, \\ &= 2\pi \int_0^w \sum_{j=1}^J \phi_j r g_j(r, \boldsymbol{z}_i; \boldsymbol{\theta}_j) \mathrm{d}r, \\ &= 2\pi \sum_{j=1}^J \phi_j \int_0^w r g_j(r, \boldsymbol{z}_i; \boldsymbol{\theta}_j) \mathrm{d}r.\end{aligned}$$

Parameters are estimated using maximum likelihood. In practice maximization is performed on the log-likelihood (see section 8.3) with analytic gradients (given in appendix A).

8.2.3 Estimating population size

As in section 7.5.1, estimates of population size can be found from a Horvitz-Thompson-like estimator, all that is needed is the p_is, then (7.9) can be used. For line transects the p_is are given by:

$$p_i = \frac{1}{w} \sum_{j=1}^{J} \phi_j \int_0^w g_j(x, \mathbf{z}_i; \boldsymbol{\theta}_j) \mathrm{d}x,$$

and for point transects:

$$p_i = \frac{2\pi}{w^2} \sum_{j=1}^{J} \phi_j \int_0^w r g_j(r, \mathbf{z}_i; \boldsymbol{\theta}_j) \mathrm{d}r.$$

Again, average detection probability for an animal within the covered region, $P_a = n/N$ can be used as a summary statistic.

8.3 Optimization

As noted in the literature (for example, Gelman et al., 2004, 463-480, Marin et al., 2005), mixture model likelihoods are notoriously multimodal. This multimodality can cause serious problems when finding MLEs of the parameters. To combat this a mix of optimization routines was to attempt to explore the parameter space as much as possible.

First, simulated annealing (Press et al., 2007, pp. 549-554) was used to explore the parameter space (for 500 iterations) then after that a quasi-Newton method (BFGS, Byrd et al., 1994) was used to find the maxima (the implementations in the R function `optim()` were used). These two steps were run 5 times and the model with the lowest AIC was selected as the final model. This two step approach appears to be satisfactory in most cases. To aid the optimization, analytic derivatives were used by BFGS; these can be found in appendix A.

8.3.1 Parametrization of the mixture proportions

When using 2-point mixtures, the constraint that the mixture proportions must sum to unity is enforced by definition (since $\phi_2 = 1 - \phi_1$). However, in J-point mixtures in general, it is not guaranteed that the proportions sum to 1. The obvious way to get around this would be to add a penalty to the likelihood,

should the optimization procedure propose values for the ϕ_js that are not in accordance with this condition. This approach is not appealing since the point of using mixtures here is to avoid penalizing or constraining the likelihood if possible. Given the problems inherent in optimizing mixture likelihoods in general, adding penalization only further complicates matters. Instead, a parametrization is used for the mixture proportions which yields ϕ_js that comply.

Rather than estimating the ϕ_js, estimate α_js, where the relationship between the two is:

$$\phi_j = F(\sum_{p=1}^{j} e^{\alpha_p}) - F(\sum_{p=1}^{j-1} e^{\alpha_p}) \qquad \text{for } 1 \leq j \leq J-1$$

and

$$\phi_J = 1 - \sum_{j=1}^{J-1} \phi_j$$

where F is any continuous CDF on $(0, \infty]$. Exponentiation ensures that $e^{\alpha_p} \geq 0$, so α_p may lie anywhere on the real line, allowing unconstrained optimisation. Summing these orders the ϕ_js, since only offsets are estimated. Finally, using the cumulative density function ensures that the ϕ_js sum to 1. In practice the Gamma$(3, 2)$ CDF is (somewhat arbitrarily) used. Figure 8-1 illustrates the relationship.

To transform from the ϕ_js back to the α_js by simply re-arranging the above expression.

$$\alpha_j = \log_e \Big(F^{-1}\Big(\phi_j + F(\sum_{p=1}^{j-1} e^{\alpha_p})\Big) - \sum_{p=1}^{j-1} e^{\alpha_p} \Big).$$

Note that only as many α_js are needed as ϕ_js, so no additional parameters are required.

8.3.2 Starting values

Beavers and Ramsey (1998) give a method for estimating starting values for the scale parameter of a half-normal detection function. In the non-covariate case, the estimate is given as the intercept parameter from intercept only regression on $\log(x + \frac{w}{1000})$ (where w denotes the truncation distance, as above). For covariate models, the equation used for the scale parameters is used in the regression and the estimated parameters from the linear regression are used as the starting values for the βs.

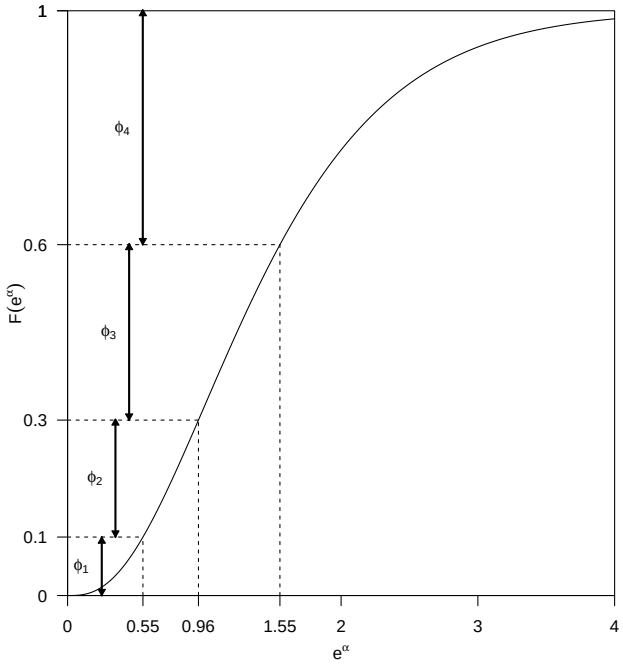

Figure 8-1: Illustration of the relationship between the mixture proportions, ϕ_j and the quantites estimated in the optimization procedure α_j.

A similar approach can be use in the mixture case by first sorting the distances from smallest to largest, then dividing the sorted distances into J (approximately) equal groups. For each of these groups a Beavers and Ramsay-type estimate is used for the β_{0j}s. The standard Beavers and Ramsay estimate using the full data was used on the other parameters. The ϕ_j had starting values of $1/J$ since there is no reason *a priori* to believe anything else.

8.4 Simulations

Extensive simulations were carried out to ensure that both parameters could be recovered and that abundance estimates were unbiased. The latter is more important than the former since our primary interest is estimating abundance. Since (as has been illustrated in section 7.5.1) the abundance depends only on the probability of detection, this was used as a target quantity to estimate and compare the results. Each simulation involved generating 200 replicate datasets from a specified mixture model using rejection sampling, fitting each dataset with 1-, 2-, and 3-point mixture models for the detection function, and in each case

recording estimated (average) probability ($\hat{p}$ for non-covariate situations, $\hat{P}_a$ for covariates) of detection from the model with the lowest AIC. Rejection sampling was used to generate the observed distances (although direct simulation is also possible by rescaling the ϕ_js).

8.4.1 Simulation settings

Clearly there are many possible simulation settings that could be used. Four sets of simulations were run, which give a fairly broad range of possible scenarios.

1. *Non-covariate 2-point detection functions for line transect data.* Four different detection functions were tested, and are shown in the first row of figure 8-2. The first two are deliberately fairly easy to fit. The third should be harder, testing the behaviour of the model when one of the parameters is hard to estimate (the scale parameter of one of the mixture components is very large relative to the truncation distance). Finally, the fourth detection function has a large spike, which is similar to that in some of the data analysed in section 8.5.

2. *Non-covariate 2-point detection functions for point transect data.* The detection functions were as above, with data generated as if they came from point transects. The PDFs are given in the second row of figure 8-2; the dotted lines indicate the component PDFs scaled such that the area under each curve is one.

3. *Non-covariate 3-point detection functions for line transect data.* Two different models for the detection functions were used. They are shown in the third row of figure 8-2. The first is much like the second line transect detection function, enabling us to investigate the efficacy of model selection. The second is a more complex shape that could only be created using a 3-point mixture; it has the added complication (as with the third line transect simulation) that one of the components may be non-identifiable.

4. *Covariate 2-point detection functions for line transect data.* Two different models were tested, the first of which has a binary factor covariate: half of the observations had covariate value 1 and half had covariate value 0. The second model had a continuous covariate, whose values were generated from a standard normal distribution function. Detection functions are shown in

the fourth row of figure 8-2, along with the marginal detection functions for the levels/quantiles of the covariates. One goal was to test the properties of the estimator in the presence of covariates, and for these runs covariates were included in all models. However, it is also interesting to evaluate the performance of the mixture model formulation in cases where a covariate affects detectability but the covariate is not available to the observer. Therefore the same simulated datasets were fit with mixture models that did not include covariates, and compared the resulting estimates to their with-covariate counterparts.

8.4.2 Results

Results from the simulation study are shown in figure 8-3, the boxplots therein are of the estimated detection probability ($\hat{p}$) for non-covariate models and average detection probability ($\hat{P}_a$) for covariate models. The number under each boxplot gives the proportion of AIC best models that were of the same form as the true model (i.e. same number of mixture components and the same covariates). For the first row (line transects with no covariates) we can see that in scenarios 1 and 3, both the true model and true probability of detection were recovered, even at low sample sizes. Whereas in scenarios 2 and 4 there was a positive bias in the estimates of the probability of detection as well as a lower proportion of "correct" models being selected by AIC. In scenario 2 and 4 most of the weight (0.7 and 0.6, respectively) is given to the larger mixture component, thus the smaller component is swamped; this would make the simulated data look as if it were generated from a 1-point mixture.

This effect can also be seen in the point transect simulations, but more severely. In this case one can see from the second row of figure 8-2 that the PDFs of the distances look very much like only one of the components of the mixture. Fitting only a 1-point mixture to data from a 2-point mixture has lead to a positive bias in the detection probabilities, however this is only to expected given that data generated from such models would look like a 1-point mixture.

The 3-point mixtures show that detection function specified in scenario 1 can be easily represented using a smaller number of mixture terms. At sample sizes of 120 and lower, a 1-point mixture model dominated (around 150 of the AIC selected models were 1-point mixtures) then at the two higher sample sizes, 2-point mixtures dominated. Scenario 2 seemed more like a "true" 3-point data set,

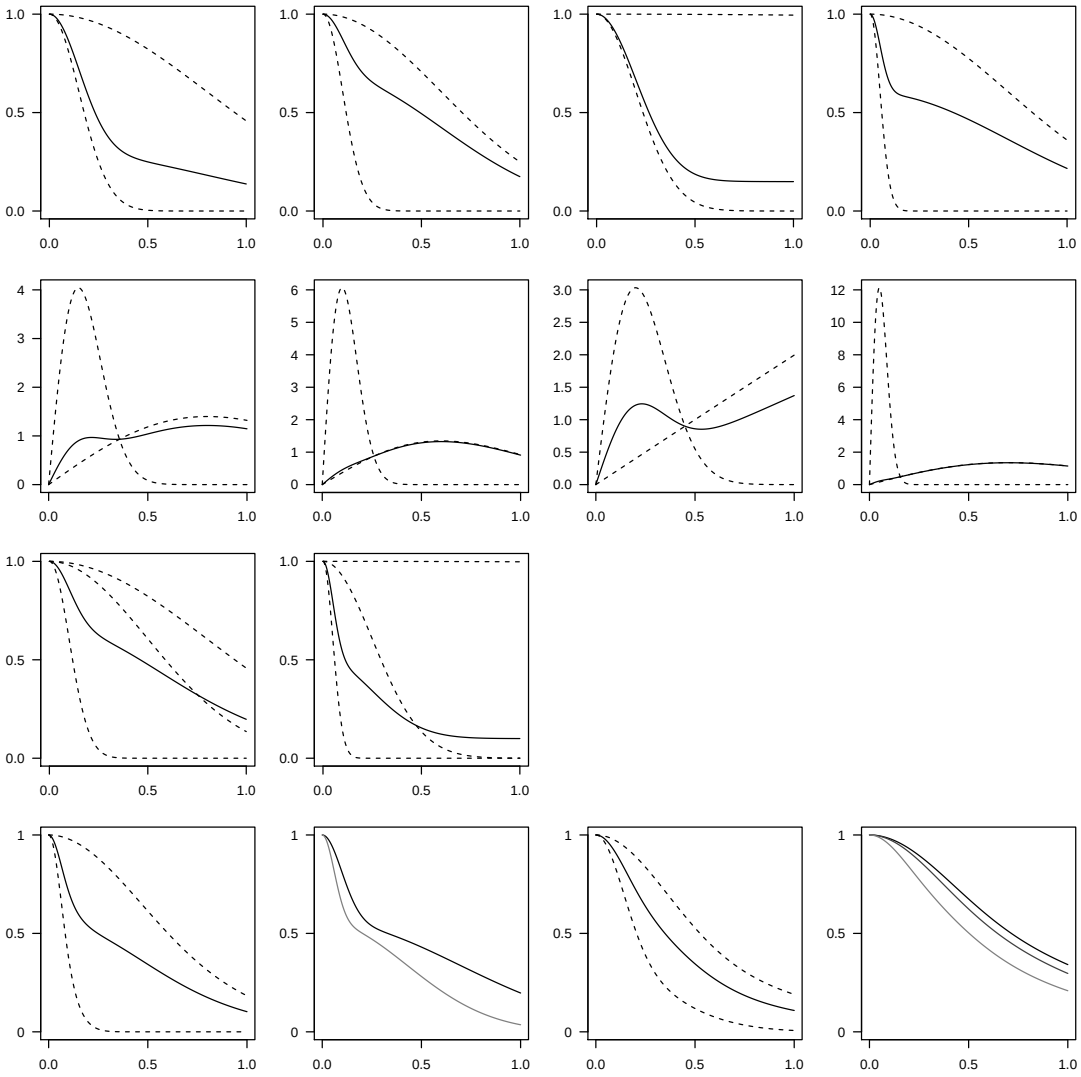

Figure 8-2: Plots of the models used in the simulation study. Top row: detection functions for the line transect simulations with no covariates (solid lines) and their constituent mixture components (dashed lines). Second row: PDFs for the point transect simulations with no covariates (bold lines), with associated component PDFs (dashed lines) rescaled so the area under each curve is one; the detection functions are as in the top row. Third row: two 3-point mixtures for non-covariate line transect data, again with components (dashed lines). Fourth row, two covariate models, the first two panels are for a binary covariate, the second two for a continuous covariate; the first panel in each pair shows the detection function averaged over the covariates (along with the mixture components, similarly averaged; dashed lines) and the second panels show marginal detection functions with the levels (dashed) or quantiles (grey lines) of the detection function.

although only at the two larger sample sizes was a 3-point model selected more than half of the time. Interestingly at the lower sample sizes, a 2-point mixture was selected more often than 1-point (137, 173 and 182 times for the lowest three sample sizes). This effect may be due to the large, potentially non-identifiable, scale parameter being confounded with the other parameters, although this did not appear to be problematic in the scenario 2 of the line transect non-covariate models, above.

The covariate models again show an initial positive bias in $\hat{P}_a$ but do still converge to the true value at higher sample sizes. The numbers beneath the boxplots mask rather more than in the other models, since model selection was performed on not only the number of mixtures but also as to whether covariates were included in the model. Further analysis shows that at the larger two sample sizes we see that the "true" model is selected almost all of the time. However, at smaller sample sizes we see more interesting behaviour. For scenario 1, at the two lowest sample sizes a 1-point non-covariate model dominates, followed by 1-point with covariates, then 2-point covariate and 2-point non-covariate. So, as one would expect, the simplest two models are selected most, but the added information from the covariates means that the 2-point covariate is selected ahead of the 2-point non-covariate. At sample size 120, the 1- and 2-point covariate models are both selected about 70 times, and the non-covariate 1- and 2-point models about 30 times. Contrary to expectations the 3-point mixtures are barely selected at all (with or without covariates) leading to the conclusion that the covariates cannot simply be substituted for further mixture terms, even in the binary covariate case. Scenario 2 shows a slightly different picture; the non-covariate models are selected less, even at lower sample sizes (the 1-point covariate model was selected 90 times at the lowest sample size) which might be expected given that scenario 2 included a continuous covariate. As was the case for scenario 1, the 3-point models were barely selected at all. In both scenarios there seems to be a bite point between a sample size of 120 and 480 where the correct model is selected almost every time.

8.5 Real data

Given the performance of the method in simulation, four data sets were analysed using the method. They were chosen because either they exhibited non-monotonicity (Williams and Thomas, 2007, Pike et al., 2003), were particularly

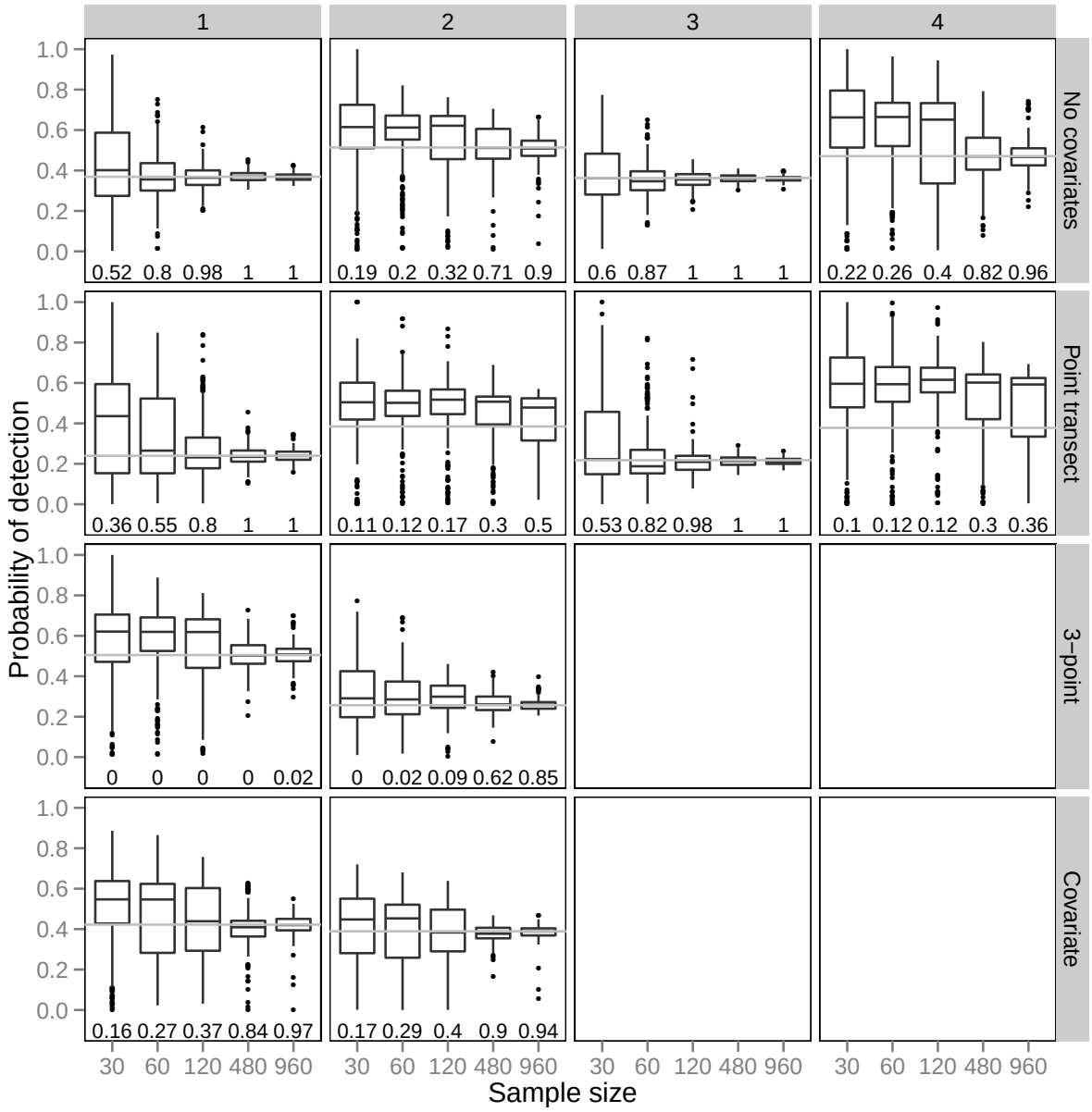

Figure 8-3: Simulation results: boxplots of the estimated detection probabilities for the best model (by AIC score). Layout is as in figure 8-2. Grey lines indicate the true value of the average detection probability. Numbers underneath each boxplot give the proportion of AIC best models that were of the same form as the model that the data was simulated from (e.g. in covariate case 1 the proportion of AIC best models that were two point mixtures that included the covariate in the model).

Species	Model	AIC	$\hat{P}_a$	K-S p
Harbour seal	Hn+cos(2) (W&T)	2771.05	0.425	0.515
(in water)	Hn 2-pt	**2769.86**	0.335	0.945
Harbour porpoise	Hr (W&T)	**690.66**	0.212	0.99
	Hn 2-pt	692.09	0.254	0.99
Humpback	Hn+cos(2) (W&T)	**1033.06**	0.386	0.672
	Hn 2-pt	1035.94	0.381	0.649

Table 8.1: Comparison of the results for the Williams and Thomas (2007) data. AIC and average detectability ($\hat{P}_a$) are given for each model (bold indicates lowest AIC). Kolmogorov-Smirnov test p-values (KS p) are also given. W&T indicates the results reported in Williams and Thomas (2007), other results are from mixture models where the number of mixture components was selected by AIC. cos(x) indicates a cosine adjustment of order x.

spiked (Borkin et al., 2011) or were sufficiently big that it was though they might support a mixture with many components (Marques et al., 2007). Between them they cover both covariate and non-covariate data from line and point transect surveys on boats and on foot. In each case the original analyses (i.e. the non-mixture models for the data) were fitted using the `mrds` library for R (available at `http://github.com/jlaake/mrds`). Mixture model detection functions were fitted using the `mmds` library for R written by the author (see section 8.6, below).

8.5.1 Williams and Thomas (2007) cetacean survey

Williams and Thomas (2007) study several species of cetaceans off the coast of British Columbia. Here data for three of the species are re-analysed: harbour seal (*Phoca vitulina*) in water, harbour porpoise (*Phocoena phocoena*) and humpback whale (*Megaptera novaeangliae*) (with truncation at 500m, 500m and 2000m respectively). Results are summarised in table 8.1 and detection functions for the models selected by AIC are shown in figure 8-4.

In each case a 2-point mixture was selected by AIC to be the best model. For the harbour seal data, the mixture model had a better AIC than the half-normal with cosine adjustments of order 2 which was selected in the paper. The significant increase in Kolmogorov-Smirnov p-value is due to the fact that the data are quite spiked: a half-normal detection function cannot fit such a spike, but the mixture model detection function can (see the spiked, dotted line in figure 8-4), this then also leads to the increase in $\hat{P}_a$. The relative lack of change in AIC is down to the fact that the mixture uses one more parameter than the

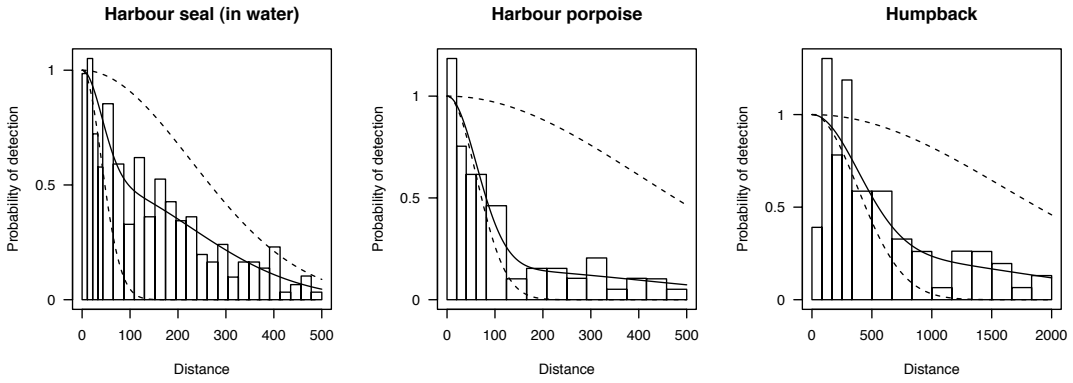

Figure 8-4: Plots of the detection functions fit to the Williams and Thomas (2007) data. In each case the model selected by AIC was a 2-point mixture. Dashed lines show the mixture components.

half-normal with cosine adjustment, despite this the AIC is still lower for the mixture model, providing some encouragement that the mixture formulation is worthwhile pursuing.

The mixture models for the harbour porpoise and the humpback both did not perform as well as the models selected in Williams and Thomas (2007). However, for the harbour porpoise the AIC is only different by 2, the detectability is very similar so it does not appear that much is lost by using the mixture model; both models fit the data very well (especially the spike near zero distance), as indicated by the Kolmogorov-Smirnov p-values. However (as mentioned in Williams and Thomas (2007)), the spike in the data at zero distance may be due to the porpoise being attracted to the boat so the Kolmogorov-Smirnov p-values should be treated rather skeptically in this case. In the case of the humpback, the fitted detection function is monotone, which was not the case in Williams and Thomas (2007); this added to the fact that there is again not a huge difference in results, leads us to believe that mixture models for the detection function can be useful.

8.5.2 Wood ants in the Abernethy forest

Borkin et al. (2011) analyse data on two species of wood ant (*Formica aquilonia* and *Formica lugubris*) in the Abernethy Forest in Strathspey Scotland over the period of August–September 2003. The number of nests sighted was 150 out to a distance of 72.04m from the track line, although 45% of the nest sightings lay within 4m of the line. As part of their analysis several different truncation

distances were used and larger truncation distances led to large variance in the encounter rate estimates and hence in overall abundance estimates. This is due to the spike caused by the large number of detections close to the line.

As well as distances, three covariates were recorded: habitat type (`habitat`, a four level factor, the size (calculated as half-width of the nest multiplied by its height) of each nest (`nest.size`, continuous variable) and whether *Formica aquilonia* or *Formica lugubris* were observed (`species`, a two level factor). In order to avoid numerical problems due to large values of the nest size, the variable was standardised. The habitat type is defined in Borkin et al. (2011) and is a proxy for the amount of disturbance of the area (1 indicates new pre-thicket and 4 indicates a wooded bog/old open woodland).

As can be seen from table 8.2, the best model by AIC was a 2-point mixture with nest size and habitat as covariates. This is not that surprising given the best CDS model (selected by AIC) in Distance was a hazard-rate with the same covariates. What is rather surprising is that the best mixture model had a better AIC than the hazard-rate, even though it has 1 more parameter (hazard-rate models have two parameters, shape and scale). This is particularly encouraging since one might expect that mixtures would give good fits but would not achieve lower AIC scores due to the number of parameters required.

8.5.3 Long-finned pilot whales

Pike et al. (2003) analyse data on long-finned pilot whales (*Globicephala melas*). There were 84 pods sighted as part of the NASS-2001 survey (of which the pilot whale was not a target). Here the data are analysed as if there were 84 individuals. The Beaufort sea state was recorded as a covariate during the survey and enter the model as either a continuous variable (`BSS`), 6 level factor (`BSS`, as a factor), a 2 level factor (`BSS3`) or a 3 level factor (`BSS2`).

A model was fit with each of these covariates, as well as a model with no covariates. A summary of the results is given in table 8.2 and the detection function for the best model is shown in figure 8-6.

The best model by AIC score was a 2-point mixture with `BSS2` as a covariate. Figure 8-6 shows the average detection function and the marginal detection function with the levels of `BSS2`. The plot looks significantly better than the one shown in figure 7-7.

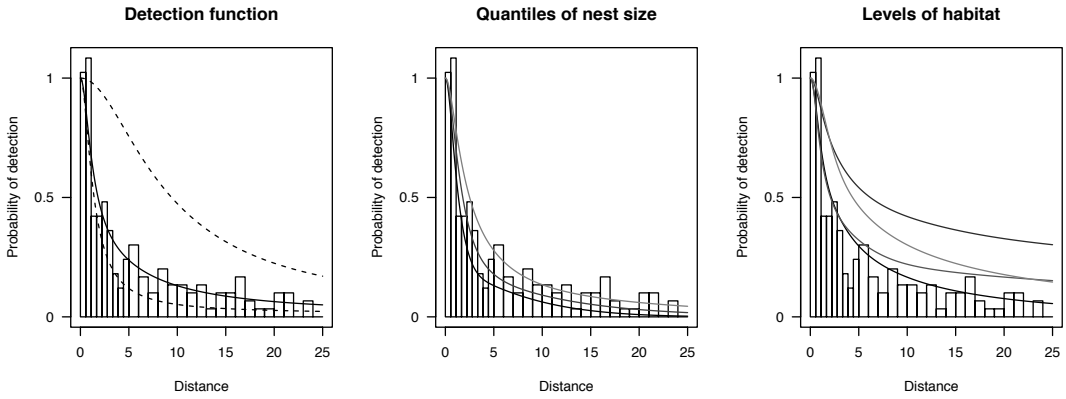

Figure 8-5: Plot of the detection functions for the AIC best model for the ants data set (2-point mixture with nest size and habitat as covariates). The first panel shows the average detection function, in the sense of section 7.5.2 (dashed lines are the two mixture components, calculated in the same way but setting one of the mixture components to zero). The second and third panels show the quantiles (25%, 50% and 75% from bottom to top) of nest size and the levels of habitat type (1 to 4 from dark to lightest line, see main description for more information) respectively. (Again, these are calculated as above but holding one covariate fixed to the appropriate quantile/level and averaging over the other.)

8.5.4 Amakihi

Marques et al. (2007) analyse data on the Hawaii Amakihi (*Hemignathus virens*). The data consist of 1485 observations on the bird ($n = 1243$ after truncation at 82.5m), collected at 41 point transects from July 1992 to April 1995. Data was also collected on three covariates: the observer (`obs`, a three level factor), minutes after sunrise (`mas`, continuous) and hours after sunrise (`has`, a six level factor).

AIC selected a two point mixture with observer and minutes after sunrise as covariates (shown in figure 8-7), closely followed by the model with only observer as a covariate (see table 8.2). In this case a CDS analysis with a hazard-rate detection function using observer and minutes after sunrise as covariates beat the mixtures in AIC terms. This is not too surprising given that the mixtures use one more parameter than a hazard-rate in this situation. It is encouraging that there is such a small difference in AIC, and that covariate mixture models were selected despite the large number of parameters that such models entail.

Note that all of the models for the amakihi data in table 8.2 have rather low Kolmogorov-Smirnov p-values (including those reported in Marques et al. (2007). This is because the data set is rather large and very complex; given that

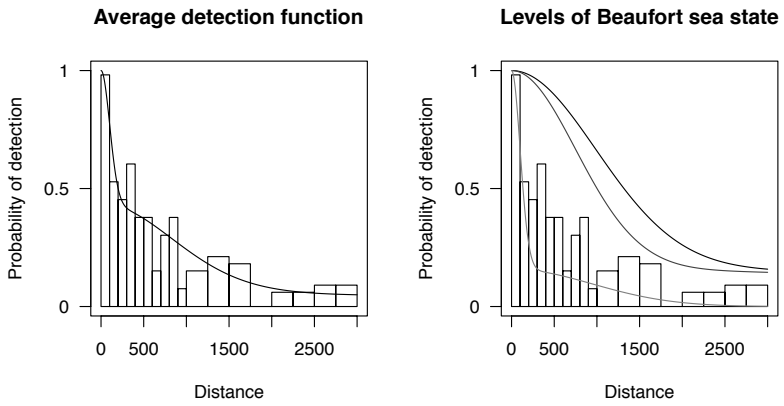

Figure 8-6: The (AIC) best model for the long-finned pilot whale data – a 2-point mixture model detection function with Beaufort sea state. Left: the average detection function, right: the detection function for the levels of `BSS2` (1.5, 2 and 3 working from light to dark).

the models fitted are simple (three covariates at most, two of which (`has` and `mas` are highly correlated), the null hypothesis is rejected relatively easily.

8.6 Software implementation - `mmds`

All the models discussed in this article are available as an R package, `mmds` (Mixture Model Distance Sampling) which is available from `http://github.com/dill/mmds` along with documentation. Mixture model detection functions will be available in the next version of the software package Distance (Thomas et al., 2010) as an additional option for modelling the detection function.

8.7 Conclusion

This chapter has investigated and demonstrated the utility of detection functions constructed from mixtures of half-normal functions for both line and point transect distance sampling. The method was also extended to include covariates. The formulation presented here for mixture detection functions can be simply "dropped in" to the existing theory and make a handy additional approach when there are issues with non-monotonicity.

It has been shown that the method performs well on both simulated and real data. In many cases the proposed model outperforms the existing CDS and

Model	Covariates	AIC	$\hat{P}_a$	K-S p
	(*i*) *Ants*			
Hn 2-pt	None	754.61	0.183	0.96
Hn 2-pt	`habitat`	751.27	0.150	0.97
Hn 2-pt	`species`	756.59	0.184	0.94
Hn 2-pt	`nest.size`	741.64	0.214	0.76
Hn 2-pt	`habitat` + `species`	749.05	0.152	0.94
Hn 2-pt	`habitat` + `nest.size`	**737.27**	0.179	0.72
Hn 2-pt	`nest.size` + `species`	741.92	0.210	0.77
Hn 2-pt	`nest.size` + `species` + `habitat`	739.77	0.178	0.68
Hr	`habitat` + `nest.size`	745.2	0.194	0.95
	(*ii*) *Long-finned pilot whales*			
Hn 2-pt	None	1298.42	0.295	0.95
Hn 2-pt	`BSS`	1286.6	0.209	0.82
Hn 2-pt	`BSS2`	**1284.99**	0.211	0.95
Hn 2-pt	`BSS3`	1296.27	0.27	0.99
Hn 2-pt	`BSS` (cont.)	1286.26	0.152	0.84
Hn 1-pt + cos(2)	`BSS` (cont.)	1296	0.37	0.319
	(*iii*) *Amakihi*			
Hn 2-pt	None	10805.48	0.283	0.12
Hn 2-pt	`obs`	10778.69	0.289	0.04
Hn 2-pt	`has`	10807.19	0.282	0.33
Hn 2-pt	`mas`	10805.11	0.284	0.31
Hn 2-pt	`obs` + `has`	10782.53	0.283	0.23
Hn 2-pt	`obs` + `mas`	10778.07	0.289	0.14
Hn 2-pt	`has` + `mas`	10809.17	0.282	0.43
Hn 2-pt	`obs` + `has` + `mas`	10784.5	0.282	0.35
Hr	`obs` + `mas`	**10777.72**	0.30	0.036

Table 8.2: Results for the three data sets with covariates. Bold indicates lowest AIC for each set. In each set the final model is the lowest AIC model reported in the original analysis (in Distance). Kolmogorov-Smirnov test p-values (KS p) are also given. Results are from (*i*) the wood ant data from Borkin et al. (2011) with truncation at 25m; (*ii*) the long-finned pilot whales Pike et al. (2003) with truncation at 3000m, (cont.) denotes that the covariate was included in the model as continuous, otherwise covariates entered the model as factors; (*iii*) the amakihi data from Marques et al. (2007) with truncation at 82.5m. "Hn j-pt" indicated an j-point mixture was used. "Hr" indicates a hazard-rate models was used.

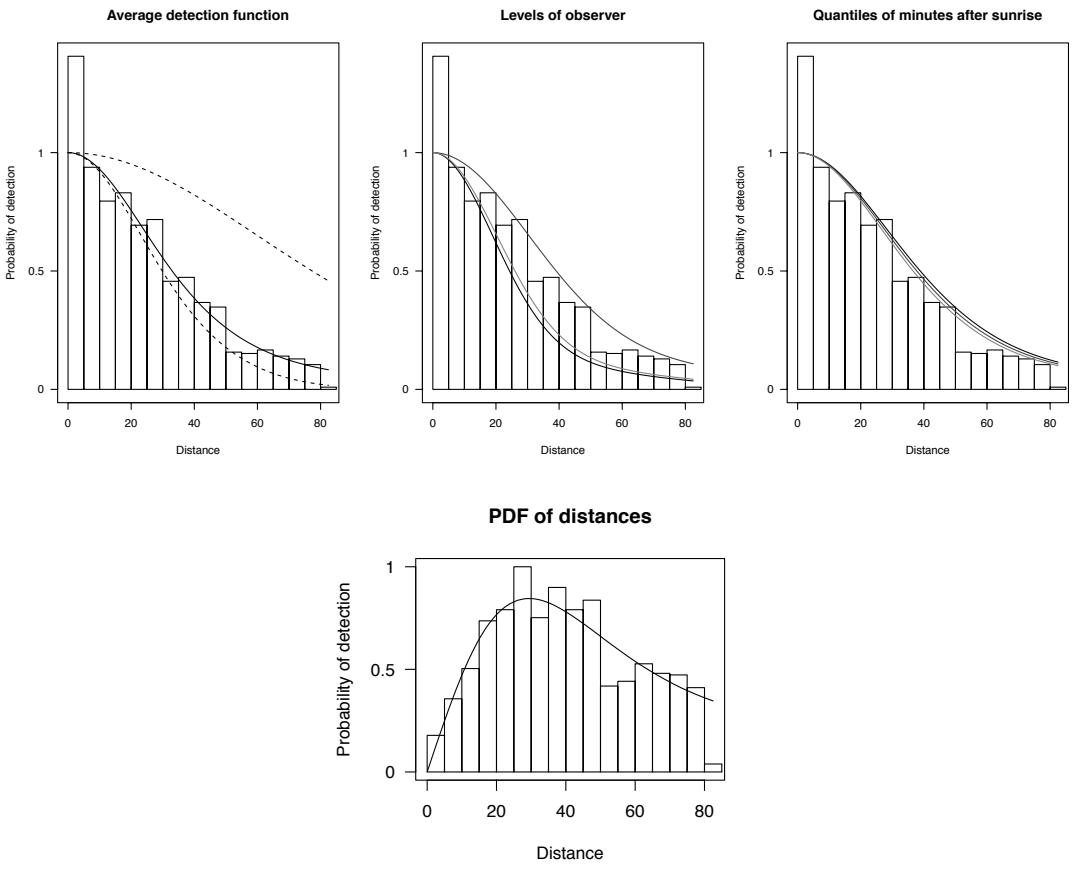

Figure 8-7: Plots of the (AIC) best model for the amakihi data. Top row: detection function averaged over covariates (dashed lines are each mixture component averaged over covariates), marginal detection function showing the levels of observer (averaged over the values of minutes after sunrise) and marginal detection function for the quantiles minutes after sunrise (averaged over the levels of observer). Bottom row: PDF of distances averaged over the covariate values.

MCDS models, which is surprising given that the mixture models in question often had more parameters than the CDS/MCDS models. It should be noted however, that models do not tend fit data particularly well at small sample sizes (as was found via simulation). This problem can be avoided in practice by using AIC forward selection for the number of mixtures, ensuring the simplest model is found.

In simulation 3-point mixtures did not appear to be good surrogates for missing covariate structure in the model. In general 2-point mixtures were chosen by AIC as good models. In the examples in section 8.5, 2-point mixtures consistently provided the best fit, even in the case of the amakihi (which was a very large data set). Only examination of further data will show whether 3-point and

higher mixtures can be supported in real data, however in CDS analyses when the key function with adjustment series formulation is used, detection functions with 5 or more parameters are rarely selected by AIC and a 3-point mixture with no covariates requires 5 parameters.

A possible extension the methodology would be to allow continuous mixtures. In that case the detection function can be modelled as:

$$g(x, \mathbf{z}; \boldsymbol{\theta}, \kappa) = \int_{\mathbb{R}} \varphi(\kappa) g_\kappa(x, \mathbf{z}; \boldsymbol{\theta}, \kappa) \mathrm{d}\kappa$$

where $\varphi(\kappa)$ is some function which controls the mixing of g_κ. Such an approach may present significant computation issues. However, the benefits for model fitting may be considerable. Provided that an appropriate function can be chosen for φ, more flexible models could be used whilst keeping the number of parameters low. In addition, a finite mixture of both finite and continuous mixtures could be used, echoing the work of Morgan and Ridout (2008) in capture-recapture.

Mixture model detection functions present an appealing alternative to the adjustment term approach of Buckland (1992) that are currently popular in the distance sampling literature. The nested nature of the models is much more elegant than adjustment terms, and using only half-normal functions as the components of the mixtures makes model choice an easier process for the investigator. As has been seen in the analysis of the long-finned pilot whales in figure 7-7, combining covariates and adjustment terms in the current MCDS formulation does not yield monotonic detection functions in the current setup. Mixture model detection functions present a way of having both flexible functional forms and including parametric effects without these problems.

In all, the methodology set out here presents a useful addition to the family of possible functions that can be used for distance sampling detection functions. The fact that the method includes the standard 1-point half-normal model as a sub-case only makes it more appealing to practitioners. Additionally, its seamless integration into the current framework (allowing goodness of fit and other statistics to be calculated relatively easily) makes it easy to include it in analyses. Hopefully, because of these factors, mixture model detection functions will be seen as one of the standard tools for those analysing distance sampling data.

8.8 Acknowledgement

This chapter of the thesis is based on the paper "Mixture models for distance sampling detection functions" by the author and Len Thomas of the Centre for Ecological and Environmental Modelling at the University of St Andrews and is in preparation to submit to *Biometrics*. A mixture model approach to line transect detection functions without covariates was developed as part of the author's MMath thesis at the University of St Andrews under the supervision of Len Thomas. The material detailed here is a significant development from the methods developed in that thesis.

The author would like to thank David Borchers in particular for the suggestion of the parametrization in section 8.3.1.

Appendix A

Derivatives of the likelihood for mixture model detection functions

This appendix gives the derivations of the derivatives that were used to numerically maximize the likelihood for the mixture model detection functions in chapter 8.

A.1 Line transects

Starting from the likelihood given in (8.2), the derivatives with respect to the optimisation parameters are found.

A.1.1 With respect to β_{0j*}

For the intercept terms (also considering in the non-covariate case, these are just the parameters), the parameters have no effect outside of their mixture (ie. β_{0j*} only has an influence on mixture component $j*$), so we can write:

$$\frac{\partial l\left(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x},\mathbf{Z}\right)}{\partial\beta_{0j*}}=\sum_{i=1}^{n}\frac{1}{g\left(x_i,\mathbf{z}_i;\boldsymbol{\theta},\boldsymbol{\phi}\right)}\phi_{j*}\frac{\partial}{\partial\beta_{0j*}}g_{j*}\left(x_i,\mathbf{z}_i;\boldsymbol{\theta}_{j*}\right)-\frac{\phi_{j*}}{\mu_i}\frac{\partial}{\partial\beta_{0j*}}\mu_{ij*}.$$

Now, to first find $\frac{\partial}{\partial\beta_{0j*}}g_{j*}\left(x_i,\mathbf{z}_i;\boldsymbol{\theta}_{j*}\right)$:

$$\frac{\partial g_{j*}\left(x_i,\mathbf{z}_i;\boldsymbol{\theta}_{j*}\right)}{\partial\beta_{0j*}}=\frac{\partial}{\partial\beta_{0j*}}\exp\left(-\frac{x_i^2}{2\sigma_{j*}^2}\right),$$

applying the chain rule and remembering that σ_{j*} is a (trivial) function of the β_{0j}s:

$$\frac{\partial g_{j*}\left(x_i, \mathbf{z}_i; \boldsymbol{\theta}_{j*}\right)}{\partial \beta_{0j*}} = \left(\frac{x_i}{\sigma_{j*}}\right)^2 \exp\left(-\frac{x_i^2}{2\sigma_{j*}^2}\right).$$

Expressing μ_{ij*} in terms of the error function:

$$\begin{aligned}\frac{\partial \mu_{ij*}}{\partial \beta_{0j*}} &= \frac{\partial}{\partial \beta_{0j*}}\left\{\sqrt{\frac{\pi}{2}}\sigma_{j*}\mathrm{Erf}\left(\frac{w}{\sqrt{2\sigma_{j*}^2}}\right)\right\}, \\ &= \mathrm{Erf}\left(\frac{w}{\sqrt{2\sigma_{j*}^2}}\right)\frac{\partial}{\partial \beta_{0j*}}\left(\sqrt{\frac{\pi}{2}}\sigma_{j*}\right) + \sqrt{\frac{\pi}{2}}\sigma_{j*}\frac{\partial}{\partial \beta_{0j*}}\left\{\mathrm{Erf}\left(\frac{w}{\sqrt{2\sigma_{j*}^2}}\right)\right\}.\end{aligned} \tag{A.1}$$

To find $\frac{\partial}{\partial \beta_{0j*}}\mathrm{Erf}\left(\frac{w}{\sqrt{2\sigma_{j*}^2}}\right)$, note that we can write and then apply the chain rule:

$$\begin{aligned}\frac{\partial}{\partial \beta_{0j*}}\mathrm{Erf}\left(\frac{w}{\sqrt{2\sigma_{j*}^2}}\right) &= \frac{\partial}{\partial \beta_{0j*}} S\left\{u(\sigma_{j*})\right\}, \\ &= \frac{\partial S(u)}{\partial u}\frac{\partial u(\sigma_{j*})}{\partial \sigma_{j*}}\frac{\partial \sigma_{j*}}{\partial \beta_{0j*}},\end{aligned}$$

where

$$S(u) = \int_0^u \exp\left(-t^2\right)\mathrm{d}t \quad \text{and} \quad u(\sigma_{j*}) = \frac{w}{\sqrt{2\sigma_{j*}^2}}.$$

Their derivatives being

$$\frac{\partial S(u)}{\partial u} = \frac{2}{\sqrt{\pi}}\exp(-u^2), \quad \frac{\partial u(\sigma_{j*})}{\partial \sigma_{j*}} = -\frac{w}{\sqrt{2}}\sigma_{j*}^{-2}.$$

Given these terms, it's just a case of multiplying them:

$$\frac{\partial S(u)}{\partial u}\frac{\partial u(\sigma_{j*})}{\partial \sigma_{j*}}\frac{\partial \sigma_{j*}}{\partial \beta_{0j*}} = -\sqrt{\frac{2}{\pi}}\frac{w}{\sigma_{j*}}\exp\left(-\frac{w^2}{2\sigma_{j*}^2}\right).$$

Substituting into (A.1):

$$\frac{\partial \mu_{ij*}}{\partial \beta_{0j*}} = \mu_{ij*} - w\exp\left(-\frac{w^2}{2\sigma_{j*}^2}\right).$$

Finally, the derivative is:

$$\frac{\partial l(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{Z})}{\partial \beta_{0j*}} = \sum_{i=1}^{n} \left(\frac{x_i}{\sigma_{j*}} \right)^2 \phi_{j*} \frac{g_{j*}(x_i, \mathbf{z}_i; \boldsymbol{\theta}_{j*})}{g(x_i, \mathbf{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi})} - \frac{\phi_{j*}}{\mu_i} \left\{ \mu_{ij*} - w g_{j*}(w, \mathbf{z}_i; \boldsymbol{\theta}_{j*}) \right\}.$$

A.1.2 With respect to β_{k*}

Derivatives with respect to the common covariate parameters are found in a similar way to above. The expressions are slightly more complicated since the β_ks affect all of the mixture components.

$$\frac{\partial l(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{Z})}{\partial \beta_{k*}} = \sum_{i=1}^{n} \left\{ \frac{1}{g(x_i, \mathbf{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi})} \sum_{j=1}^{J} \phi_j \frac{\partial}{\partial \beta_{k*}} g_j(x_i, \mathbf{z}_i; \boldsymbol{\theta}_j) - \frac{1}{\mu_i} \sum_{j=1}^{J} \phi_j \frac{\partial}{\partial \beta_{k*}} \mu_{ij} \right\}.$$

Every σ_j is a function of the β_ks, so:

$$\begin{aligned} \frac{\partial \sigma_j}{\partial \beta_{k*}} &= \frac{\partial}{\partial \beta_{0j*}} \exp\left(\beta_{0j} + \sum_{k=1}^{K} z_{ik} \beta_k \right), \\ &= z_{ik*} \exp\left(\beta_{0j} + \sum_{k=1}^{K} z_{ik} \beta_k \right), \\ &= z_{ik*} \sigma_j. \end{aligned}$$

Hence:

$$\frac{\partial}{\partial \beta_{k*}} \exp\left(-\frac{x_i^2}{2\sigma_j^2} \right) = z_{k*} \left(\frac{x_i}{\sigma_j} \right)^2 \exp\left(-\frac{x_i^2}{2\sigma_j^2} \right) = z_{k*} \left(\frac{x_i}{\sigma_j} \right)^2 g_j(x_i, \mathbf{z}_i; \boldsymbol{\theta}_j).$$

And so for the μ_{ij}s:

$$\frac{\partial \mu_{ij}}{\partial \beta_{k*}} = z_{ik*} \left\{ \mu_{ij} - w \exp\left(-\frac{w^2}{2\sigma_j^2} \right) \right\}.$$

The derivative is then:

$$\begin{aligned} \frac{\partial l(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{Z})}{\partial \beta_{k*}} = \sum_{i=1}^{n} \Bigg[& \frac{1}{g(x_i, \mathbf{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi})} \sum_{j=1}^{J} \phi_j z_{k*} \left(\frac{x_i}{\sigma_j} \right)^2 g_j(x_i, \mathbf{z}_i; \boldsymbol{\theta}_j) \\ & - \frac{1}{\mu_i} \sum_{j=1}^{J} \phi_j z_{ik*} \left\{ \mu_{ij} - w g_j(x_i, \mathbf{z}_i; \boldsymbol{\theta}_j) \right\} \Bigg]. \end{aligned}$$

A.1.3 With respect to α_{j*}

First note that we can write the likelihood (8.2) as:

$$l(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{Z}) = \sum_{i=1}^{n} \left[\log \left\{ \sum_{j=1}^{J-1} \phi_j g_j(x_i, \mathbf{z}_i; \boldsymbol{\theta}_j) + \left(1 - \sum_{j=1}^{J-1} \phi_j \right) g_J(x_i, \mathbf{z}_i; \boldsymbol{\theta}_J) \right\} - \log \left\{ \sum_{j=1}^{J-1} \phi_j \mu_{ij} + \left(1 - \sum_{j=1}^{J-1} \phi_j \right) \mu_{ij} \right\} \right].$$

The derivatives with respect to the α_{j*} of this expression are then:

$$\frac{\partial l(\boldsymbol{\theta}, \boldsymbol{\phi}; \mathbf{x}, \mathbf{Z})}{\partial \alpha_{j*}} = \left[\sum_{i=1}^{n} \frac{1}{g(x_i, \mathbf{z}_i; \boldsymbol{\theta}, \boldsymbol{\phi})} \left\{ \sum_{j=1}^{J-1} g_j(x_i, \mathbf{z}_i; \boldsymbol{\theta}_j) \frac{\partial \phi_j}{\partial \alpha_{j*}} - g_J(x_i, \mathbf{z}_i; \boldsymbol{\theta}_J) \sum_{j=1}^{J-1} \frac{\partial \phi_j}{\partial \alpha_{j*}} \right\} - \frac{1}{\mu_i} \left(\sum_{j=1}^{J-1} \mu_{ij} \frac{\partial \phi_j}{\partial \alpha_{j*}} - \mu_{iJ} \sum_{j=1}^{J-1} \frac{\partial \phi_j}{\partial \alpha_{j*}} \right) \right]. \tag{A.2}$$

Finding the derivatives is then simply a matter of finding the derivatives of ϕ_j with respect to α_{j*} and substituting them back into (A.2).

$$\frac{\partial \phi_j}{\partial \alpha_{j*}} = \frac{\partial}{\partial \alpha_{j*}} F \left(\sum_{p=1}^{j} e^{\alpha_p} \right) - \frac{\partial}{\partial \alpha_{j*}} F \left(\sum_{p=1}^{j-1} e^{\alpha_p} \right).$$

Looking at each of the terms:

$$\frac{\partial}{\partial \alpha_{j*}} F \left(\sum_{p=1}^{j} e^{\alpha_p} \right) = A_j = \begin{cases} e^{\alpha_{j*}} f \left(\sum_{p=1}^{j} e^{\alpha_p} \right) & \text{for } j \geq j*, \\ 0 & \text{for } j < j*. \end{cases}$$

and

$$\frac{\partial}{\partial \alpha_{j*}} F \left(\sum_{p=1}^{j-1} e^{\alpha_p} \right) = A_{(j-1)} = \begin{cases} e^{\alpha_{j*}} f \left(\sum_{p=1}^{j-1} e^{\alpha_p} \right) & \text{for } j-1 \geq j*, \\ 0 & \text{for } j-1 < j*. \end{cases}$$

So

$$\frac{\partial \phi_j}{\partial \alpha_{j*}} = A_j - A_{j-1}.$$

Substituting these back into (A.2) and re-arranging gives:

$$\frac{\partial l(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{x},\mathbf{Z})}{\partial \alpha_{j*}} = \sum_{i=1}^{n}\left[\frac{1}{g(x_i,\mathbf{z}_i;\boldsymbol{\theta},\boldsymbol{\phi})}\sum_{j=1}^{J-1}(A_j - A_{j-1})\left\{g_j(x,\mathbf{z}_i;\boldsymbol{\theta}_j) - g_J(x,\mathbf{z}_i;\boldsymbol{\theta}_J)\right\} - \frac{1}{\mu_i}\sum_{j=1}^{J-1}(A_j - A_{j-1})(\mu_{ij} - \mu_{iJ})\right].$$

A.2 Point transects

A.2.1 With respect to β_{0j}

Starting with the likelihood in (8.3), one can see that we obtain:

$$\begin{aligned}\frac{\partial l(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{r},\mathbf{Z})}{\partial \beta_{0j*}} &= \sum_{i=1}^{n}\left\{\frac{\partial}{\partial\beta_{0j*}}\log\sum_{j=1}^{J}\phi_j g_j(r_i,\mathbf{z}_i;\boldsymbol{\theta}_j) - \frac{\partial}{\partial\beta_{0j*}}\log\sum_{j=1}^{J}\phi_j\nu_{ij}\right\},\\ &= \sum_{i=1}^{n}\left\{\frac{\phi_{j*}\frac{\partial}{\partial\beta_{0j*}}g_{j*}(r_i,\mathbf{z}_i;\boldsymbol{\theta}_j)}{g(r_i,\mathbf{z}_i;\boldsymbol{\theta},\boldsymbol{\phi})} - \frac{\phi_{j*}\frac{\partial}{\partial\beta_{0j*}}\nu_{ij*}}{\sum_{j=1}^{J}\phi_j\nu_{ij}}\right\},\end{aligned}$$

the first part of which (the derivatives of the detection function) are as in the line transect case. The derivatives of ν_{ij} are simpler in the point transect case, since there is an easy analytic expression for ν_{ij} when g_j is half-normal :

$$\nu_{ij} = 2\pi\sigma_{ij}^2\left\{1 - \exp(-w^2/2\sigma_{ij}^2)\right\},$$

then simply applying the product rule yields:

$$\frac{\partial\nu_{ij}}{\partial\beta_{0j*}} = 2\left\{\nu_{ij*} + \pi w^2 g_{j*}(w)\right\}.$$

Substituting this into the above expression:

$$\frac{\partial l(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{r},\mathbf{Z})}{\partial\beta_{0j*}} = \sum_{i=1}^{n}\left[\frac{\phi_{j*}(r_i/\sigma_{j*})^2 g_{j*}(r_i,\mathbf{z}_i;\boldsymbol{\theta}_{j*})}{g(r_i,\mathbf{z}_i;\boldsymbol{\theta},\boldsymbol{\phi})} - \frac{\phi_{j*}2\left\{\nu_{j*} + \pi w g_{j*}(w)\right\}}{\sum_{j=1}^{J}\phi_j\nu_{ij}}\right].$$

A.2.2 With respect to β_{k*}

Again working from (8.3), we obtain:

$$\begin{aligned}\frac{\partial l(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{r},\mathbf{Z})}{\partial \beta_{k*}} &= \sum_{i=1}^{n}\left\{\frac{\partial}{\partial \beta_{k*}}\log\sum_{j=1}^{J}\phi_j g_j(r_i,\mathbf{z}_i;\boldsymbol{\theta}_j) - \frac{\partial}{\partial \beta_{k*}}\log\sum_{j=1}^{J}\phi_j\nu_{ij}\right\}\\ &= \sum_{i=1}^{n}\left\{\frac{\sum_{j=1}^{J}\phi_j\frac{\partial}{\partial\beta_{k*}}g_j(r_i,\mathbf{z}_i;\boldsymbol{\theta}_j)}{g(r_i,\mathbf{z}_i;\boldsymbol{\theta},\boldsymbol{\phi})} - \frac{\sum_{j=1}^{J}\phi_j\frac{\partial}{\partial\beta_{k*}}\nu_{ij}}{\sum_{j=1}^{J}\phi_j\nu_{ij}}\right\}.\end{aligned}$$

The derivatives of g_j are as in (A.1.2). For ν_{ij}:

$$\frac{\partial \nu_{ij}}{\partial \beta_{k*}} = 2z_{ik*}\left\{\nu_{ij} - \pi w^2 g_j(w)\right\}.$$

Putting that together:

$$\frac{\partial l(\boldsymbol{\theta},\boldsymbol{\phi};\mathbf{r},\mathbf{Z})}{\partial \beta_{k*}} = \sum_{i=1}^{n}\left[\frac{\sum_{j=1}^{J}\phi_j z_{k*}\left(\frac{x_i}{\sigma_j}\right)^2 g_j(x_i,\mathbf{z}_i;\boldsymbol{\theta}_j)}{g(r_i,\mathbf{z}_i;\boldsymbol{\theta},\boldsymbol{\phi})} - \frac{\sum_{j=1}^{J}\phi_j 2z_{ik*}\left\{\nu_{ij}-\pi w^2 g_j(w)\right\}}{\sum_{j=1}^{J}\phi_j\nu_{ij}}\right].$$

Bibliography

Abramowitz, M. and I. A. Stegun (1972). *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover.

Augustin, N. H., M. Musio, K. von Wilpert, E. Kublin, S. N. Wood, and M. Schumacher (2009). Modeling spatiotemporal forest health monitoring data. *Journal of the American Statistical Association 104*(487), 899–911.

Banerjee, S., A. E. Gelfand, and B. P. Carlin (2003). *Hierarchical Modeling and Analysis for Spatial Data*. Monographs on statistics and applied probability. CRC Press.

Beavers, S. C. and F. L. Ramsey (1998). Detectability analysis in transect surveys. *Journal of Wildlife Management 62*(3), 948–957.

Becker, E. F. and P. X. Quang (2009). A gamma-shaped detection function for line-transect surveys with mark-recapture and covariate data. *Journal of Agricultural, Biological, and Environmental Statistics 14*(2), 207–223.

Beerends, R. J., H. G. ter Morsche, J. C. van den Berg, and E. M. van de Vire (2003). *Fourier and Laplace Transforms*. Cambridge University Press.

Bernstein, M., V. de Silva, J. C. Langford, and J. B. Tenenbaum (2000). Graph approximations to geodesics on embedded manifolds. Technical report.

Boisvert, J. B., J. G. Manchuk, and C. V. Deutsch (2009). Kriging in the presence of locally varying anisotropy using non-Euclidean distances. *Mathematical Geosciences 41*, 585601.

Borchers, D. L. (1996). *Line transect estimation with undertain detection on the trackline*. Ph. D. thesis, University of Cape Town.

Borchers, D. L., S. T. Buckland, and W. Zucchini (2002). *Estimating animal abundance: closed populations*. Statistics for biology and health. Springer.

Borchers, D. L., W. Zucchini, and R. M. Fewster (1988). Mark-recapture models for line transect surveys. *Biometrics 54*, 1207–1220.

Borkin, K., R. Summers, and L. Thomas (2011). Surveying abundance and stand type associations of wood ant Formica spp.(Hymenoptera: Formicidae) nest mounds over an extensive area: trialing a novel method. *European Journal of Entomology*. In press.

Bracewell, R. N. (1986). *The Fourier transform and its applications*. McGraw-Hill.

Brier, G. W. (1950). Verification of forecasts expressed in terms of probability. *Monthly weather review 78*, 1–3.

Bronshtein, I. N., K. A. Semendyayev, G. Musiol, and H. Muehlig (2003). *Handbook of Mathematics*. Springer.

Buckland, S. T. (1985). Perpendicular distance models for line transect sampling. *Biometrics 41*, 177–195.

Buckland, S. T. (1992). Fitting density functions using polynomials. *Applied Statistics 41*, 63–76.

Buckland, S. T., D. R. Anderson, K. P. Burnham, J. L. Laake, D. L. Borchers, and L. Thomas (2001). *Distance Sampling*. Oxford University Press.

Buckland, S. T., D. R. Anderson, K. P. Burnham, J. L. Laake, D. L. Borchers, and L. Thomas (2004). *Advanced Distance Sampling*. Oxford University Press.

Burnham, K. P. and D. R. Anderson (2002). *Model selection and multimodel inference: a practical information-theoretic approach*. Springer.

Byrd, R. H., P. Lu, J. Nocedal, and C. Zhu (1994). A limited memory algorithm for bound constrained optimization. Technical Report NAM-08, Northwestern University Department of Electrical Engineering and Computer Science.

Cameron, C. (2009). The politics of supreme court nominations. In A. Gelman and J. Cortina (Eds.), *A Quantitative Tour of the Social Sciences*. Cambridge University Press.

Candy, S. G. (2004). Modelling catch and effort data using generalized linear models, the Tweedie distribution, random vessel effects and random stratum-by-year effects. *CCAMLR Science 11*, 59–80.

Carroll, R. J., D. Ruppert, L. A. Stefanski, and C. Crainiceanu (2006). *Measurement error in nonlinear models: a modern perspective*. Monographs on statistics and applied probability. Chapman & Hall/CRC.

Chandler, R. (2005). On the use of generalized linear models for interpreting climate variability. *Environmetrics 16*, 699–715.

Chatfield, C. and A. J. Collins (1980). *Introduction to multivariate analysis*. CRC Press.

Chu, E. (2008). *Discrete and Continuous Fourier Transforms: Analysis, Applications and Fast Algorithms*. CRC Press.

Cleveland, W. S. (1979). Robust locally weighted regression and smoothing scatterplots. *Journal of the American Statistical Association 74*(368), 829–836.

Cleveland, W. S. and S. J. Devlin (1988). Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting. *Journal of the American Statistical Association 83*(403), 596–610.

Cox, D. R. (2007). Applied Statistics: A Review. *The Annals of Applied Statistics 1*(1), 1–16.

Craven, P. and G. Wahba (1979). Smoothing noisy data with spline functions. *Numerische Mathematik 31*, 377–403.

Curriero, F. (2006). On the use of non-Euclidean distance measures in geostatistics. *Mathematical Geology 38*(8), 907–926.

de Silva, V. and J. B. Tenenbaum (2004). Sparse multidimensional scaling using landmark points. Technical report, Stanford University.

Diaconis, P., S. Goel, and S. Holmes (2008). Horseshoes in multidimensional scaling and local kernel methods. *Annals of Applied Statistics 2*(3), 777–807.

Diggle, P. J. and P. J. Ribeiro (2007). *Model-based Geostatistics*. Springer.

Dijkstra, E. W. (1959). A note on two problems in connexion with graphs. *Numerische Mathematik 1*(1), 269–271.

Dorazio, R. M. and J. A. Royle (2003). Mixture models for estimating the size of a closed population when capture rates vary among individuals. *Biometrics 59*(2), 351–364.

Driscoll, T. A. (1996). Algorithm 756: A MATLAB toolbox for Schwarz-Christoffel mapping. *ACM Trans. Math. Soft 22*(2), 168–186.

Driscoll, T. A. (2005). Algorithm 843: Improvements to the Schwarz-Christoffel toolbox for MATLAB. *ACM Trans. Math. Soft 31*(2), 239–251.

Driscoll, T. A. and L. N. Trefethen (2002). *Schwarz-Christoffel Mapping*. Cambridge University Press.

Driscoll, T. A. and S. A. Vavasis (1996). Numerical conformal mapping using cross-ratios and Delaunay triangulation. Technical Report CTC96TR233, Cornell University.

Duchon, J. (1977). Splines minimizing rotation-invariant semi-norms in sobolev spaces. In *Constructive theory of functions of several variables*, pp. 85–100. Springer.

Dunn, P. and G. Smyth (2005). Series evaluation of tweedie exponential dispersion models densities. *Statistics and Computing 15*, 267–280.

Eilers, P. H. C. (2006). P-spline smoothing on difficult domains. Seminar at the University of Munich.

Eilers, P. H. C. and B. D. Marx (1996). Flexible smoothing with B-splines and penalties. *Statistical Science 11*(2), 89–102.

Fahrmeir, L., T. Kneib, and S. Lang (2004). Penalized structured additive regression for space-time data: a Bayesian perspective. *Statistica Sinica 14*, 715–745.

Floyd, R. W. (1962). Algorithm 97: Shortest path. *Communications of the ACM 5*(6), 345.

Fonseca, M. (2008). New waves of immigration to small towns and rural areas in portugal. *Population, Space and Place 14*, 525–535.

Gelman, A., J. B. Carlin, H. S. Stern, and D. B. Rubin (2004). *Bayesian Data Analysis*. CRC Press.

Gentleman, R., B. Ding, S. Dudoit, and J. Ibrahim (2005). Distance measures in DNA microarray data analysis. In R. Gentleman, V. Carey, W. Huber, R. Irizarry, and S. Dudoit (Eds.), *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*, pp. 189–208. Springer.

Girosi, F., M. Jones, and T. Poggio (1995). Regularization theory and neural networks architectures. *Neural computation 7*, 219–269.

Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika 53*(3 and 4), 325–338.

Gower, J. C. (1968). Adding a point to vector diagrams in multivariate analysis. *Biometrika 55*(3), 582–585.

Hardin, J. W. and J. M. Hilbe (2002). *Generalized Estimating Equations*. Chapman Hall/CRC.

Hart, P. E., N. J. Nilsson, and B. Raphael (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Transactions on Systems Science and Cybernetics SSC4 4*(2), 100–107.

Hastie, T., R. Tibshirani, and J. Friedman (2001). *Elements of Statistical Learning*. Springer.

Hastie, T. J. and R. J. Tibshirani (1990). *Generalized Additive Models*. Monographs on statistics and applied probability. CRC Press.

Haybittle, J. L., R. W. Blamey, and C. W. Elston (1982). A prognostic index in primary breast cancer. *British Journal of Cancer 45*, 361–366.

House of Commons Library Department of Information Services (2011). Free votes – parliamentry information list. Standard Note SN/PC/04793, House of Commons Library.

Howell, L. H. and L. N. Trefethen (1990). A modified Schwarz-Christoffel transformation for elongated regions. *Siam J. Sci. Stat. Comput. 11*(5), 928–949.

Innes, S., M. P. Heide-Jø rgensen, J. L. Laake, K. L. Laidre, H. J. Cleator, P. Richard, and R. E. A. Stewart (2002). Surveys of belugas and narwhals in the Canadian High Arctic in 1996. *NAMMCO Scientific Publications 4*, 169–190.

Jensen, O. P., M. C. Christman, and T. J. Miller (2006). Landscape-based geostatistics: a case study of the distribution of blue crab in Chesapeake Bay. *Environmetrics 17*(6), 605–621.

Jørgensen, B. (1987). Exponential dispersion models. *Journal of the Royal Statistical Society: Series B 49*, 127–162.

Kahanec, M. and K. F. Zimmermann (2009). Migration in an enlarged EU: A challenging solution? Economic paper, European Comission.

King, R., B. J. T. Morgan, O. Gimenez, and S. P. Brooks (2011). *Bayesian Analysis for Population Ecology*. CRC Press.

Krzanowski, W. J. (1990). *Principles of Multivariate Analysis*. Oxford University Press.

Link, W. A. (2003). Nonidentifiability of population size from capture-recapture data with heterogeneous detection probabilities. *Biometrics 59*(4), 1123–1130.

Løland, A. and G. Høst (2003). Spatial covariance modelling in a complex coastal domain by multidimensional scaling. *Environmetrics 14*, 307–321.

Longhi, S., P. Nijkamp, and J. Poot (2010). Joint impacts of immigration on wages and employment: review and meta-analysis. *Journal of Geographical Systems 12*, 355–387.

Lowell, L. B. (2007). Trends in international migration flows and stocks, 1975-2005. OECD social, employment and migration working paper 58, OECD.

Mahalanobis, P. C. (1936). On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India 2*(1), 49–55.

Manning, A. (2010). Feature: the integration of immigrants and their children in Europe: introduction. *The Economic Journal 120*, F1–F3.

Marin, J. M., K. Mengersen, and C. P. Robert (2005). Bayesian modelling and inference on mixtures of distributions. In D. Dey and C. R. Rao (Eds.), *Handbook of Statistics 25*. Elsevier-Sciences.

Marques, F. F. C. and S. T. Buckland (2003). Incorporating covariates into standard line transect analyses. *Biometrics 59*, 924–935.

Marques, T. A., L. Thomas, S. G. Fancy, and S. T. Buckland (2007). Improving estimate of bird density using multiple-covariate distance sampling. *The Auk 124*(4), 1229–1243.

Marra, G., D. L. Miller, and L. Zanin (2011). Modelling the spatiotemporal distribution of the incidence of resident foreign population. *Statistica Neerlandica*. In press.

Marra, G. and S. N. Wood (2011). Coverage properties of confidence intervals for generalized additive model components. *Scandinavian Journal of Statistics*. In press.

McCullagh, P. (1983). Quasi-likelihood functions. *The Annals of Statistics 11*(1), 59–67.

Morgan, B. J. T. and M. Ridout (2008). A new mixture model for capture heterogeneity. *Journal of the Royal Statistical Society: Series C 57*(4), 433–446.

Munkres, J. R. (2000). *Topology*. Prentice-Hall.

Oh, M.-S. and A. E. Raftery (2001). Bayesian multidimensional scaling and choice of dimension. *Journal of the American Statistical Association 96*(455), 1031–1044.

Pike, D. G., T. Gunnlaugsson, G. A. Víkingsson, G. Desportes, and B. Mikkelson (2003). An estimate of the abundance of long-finned pilot whales (*globicephala melas*) from the NASS-2001 shipboard survey. Technical Report Paper SC/11/AE/10.

Pledger, S. (2000). Unified maximum likelihood estimates for closed capture-recapture models using mixtures. *Biometrics 56*(2), 434–442.

Pledger, S. (2005). The performance of mixture models in heterogeneous closed population capture-recapture. *Biometrics 61*(3), 868–873.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (1992). *Numerical recipes in FORTRAN: the art of scientific computing*. Cambridge University Press.

Press, W. H., S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery (2007). *Numerical recipes: the art of scientific computing*. Cambridge University Press.

R Development Core Team (2011). *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. ISBN 3-900051-07-0.

Ramsay, T. (2002). Spline smoothing over difficult regions. *Journal of the Royal Statistical Society: Series B 64*(2), 307 – 319.

Reiss, P. T. and R. T. Ogden (2009). Smoothing parameter selection for a class of semiparametric linear models. *Journal of the Royal Statistical Society: Series B 71*, 505–523.

Rue, H., S. Martino, and N. Chopin (2009). Approximate bayesian inference for latent gaussian models using integrated nested laplace approximations (with discussion). *Journal of the Royal Statistical Society: Series B 71*, 319–392.

Ruppert, D., M. P. Wand, and R. J. Carroll (2003). *Semiparametric Regression*. Cambridge University Press.

Ruppert, D., M. P. Wand, and R. J. Carroll (2009). Semiparametric regression during 2003–2007. *Electronic Journal of Statistics 3*, 1193–1256.

Saff, E. B. and A. D. Snider (1993). *Fundamentals of Complex Analysis for Mathematics, Science And Engineering*. Prentice Hall.

Schoenberg, I. J. (1935). Remarks to Maurice Frechet's article "Sur la definition axiomatique d'une classe d'espace distances vectoriellement applicable sur l'espace de Hilbert". *The Annals of Mathematics 36*(3), 724–732.

Shabenberger, O. and C. A. Gotway (2005). *Statistical methods for spatial data analysis*. CRC Press.

Spang, R., H. Zuzan, M. West, J. Nevins, C. Blanchette, and J. R. Marks (2002). Prediction and uncertainty in the analysis of gene expression profiles. *In Silico Biology 2*(3), 369–381.

Strindberg, S. and S. T. Buckland (2004). Zigzag survey designs in line transect sampling. *Journal of Agricultural, Biological, and Environmental Statistics 9*(4), 443–461.

Strozza, S. (2004). Estimates of the illegal foreigners in italy: a review of the literature. *International Migration Review 38*, 309–331.

ter Braak, C. J. F. (1986). Canonical correspondence analysis: A new eigenvector technique for multivariate direct gradient analysis. *Ecology 67*(5), 69–77.

Thomas, L., S. T. Buckland, K. P. Burnham, D. R. Anderson, J. L. Laake, D. L. Borchers, and S. Strindberg (2002). Distance sampling. In A. H. El-Shaarawi and W. W. Piegorsch (Eds.), *Encyclopedia of Environmetrics*, pp. 544–552. John Wiley & Sons.

Thomas, L., S. T. Buckland, E. A. Rexstad, J. L. Laake, S. Strindberg, S. L. Hedley, J. R. B. Bishop, T. A. Marques, and K. P. Burnham (2010). Distance software: design and analysis of distance sampling surveys for estimating population size. *Journal of Applied Ecology 45*, 5–14.

Thompson, S. K. (2002). *Sampling*. John Wiley & Sons.

Todd, J. H., C. Dowle, M. R. Williams, C. W. Elston, I. O. Ellis, C. P. Hinton, R. W. Blamey, and J. L. Haybittle (1987). Confirmation of a prognostic index in primary breast cancer. *British Journal of Cancer 56*(4), 489–492.

Torgerson, W. S. (1952). Multidimensional scaling: I. theory and applications. *Pyschometrika 17*(4), 401–419.

Venables, W. N. and B. D. Ripley (2002). *Modern applied statistics with S*. Springer.

Vretblad, A. (2003). *Fourier Analysis and Its Applications*. Springer.

Wahba, G. (1990). *Spline Models for Observational Data*. Society for Industrial and Applied Mathematics.

Wand, M. P. and M. C. Jones (1994). *Kernel Smoothing*. Monographs on statistics and applied probability. CRC Press.

Wang, H. and M. G. Ranalli (2007). Low-rank smoothing splines on complicated domains. *Biometrics 63*, 209–217.

Wetherill, G. B. (1982). *Elementary statistical methods*. Chapman Hall.

Williams, R., S. L. Hedley, T. A. Branch, M. V. Bravington, A. N. Zerbini, and K. P. Findlay (2011). Chilean Blue Whales as a case study to illustrate methods to estimate abundance and evaluate conservation status of rare species. *Conservation Biology 25*(3).

Williams, R. and L. Thomas (2007). Distribution and abundance of marine mammals in the coastal waters of British Columbia, Canada. *Journal of Cetacean Research and Management 9*(1), 15–38.

Wit, E. and J. McClure (2004). *Statistics for microarrays*. Wiley.

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

Wood, S. N. (2003). Thin plate regression splines. *Journal of the Royal Statistical Society: Series B 65*(1), 95–114.

Wood, S. N. (2006). *Generalized Additive Models: An Introduction with R*. Chapman & Hall.

Wood, S. N. (2008). Fast stable direct fitting and smoothness selection for generalized additive models. *Journal of the Royal Statistical Society: Series B 70*(3), 495–518.

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

Wood, S. N., M. V. Bravington, and S. L. Hedley (2008). Soap film smoothing. *Journal of the Royal Statistical Society: Series B 70*(5), 931–955.

Yeoh, E.-J., M. E. Ross, S. A. Shurtleff, W. K. Williams, D. Patel, R. Mahfouz, F. G. Behm, S. C. Raimondi, M. V. Relling, A. Patel, C. Cheng, D. Campana, D. Wilkins, X. Zhou, J. Li, H. Liu, C.-H. Pui, W. E. Evans, C. Naeve, L. Wong, and J. R. Downing (2002). Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer cell 1*(2), 133–143.

Corrections

A few points:

- **Bold** indicates a query.
- Page numbers are updated ones.
- Page numbers are for the start of the correction, so they may flow onto other pages.

A.3 From e-mail 30th January 2012.

The points raised in the e-mail have been addressed as follows:

- Corrections to "minor mathematical and grammatical errors throughout the thesis" according to the two lists sent are itemized in sections A.4 and A.5, below.
- Rewrote the thin plate (p. 30) and cubic spline (p. 36) explanations.
- p. 49 Elaborated on soap film from a intuitive point of view, saving the main technical details for chapter 2.
- Regarding the software used:
  - p. 63 Explained that `soap` and `mgcv` were used.
  - p. 69 Clarified that all the models were fitted using existing libraries.
  - p. 77, p. 79 Added mention of the *SC Toolbox* and its author.
  - p. 85 Explains that the MATLAB package *SC Toolbox* is used for the Schwarz-Christoffel mapping and `mgcv` and `soap` are used for smoothing.

  - p. 114 Clarified that I wrote the code for finding the distances and performing Gower's interpolation. `soap` and `mgcv` were used for smoothing. R's built-in MDS routine was used.
  - p. 226 Clarified that `mrds` and `mmds` were used to fit the case studies for the mixture models.

- **In section 4.6.3, the opening paragraph now makes clear that the method used for finding $\mathcal{L}^*$ is heuristic. I don't mention the Jacobian as when I looked back over my notes I found that I'd started to look at it but then switched to the "engineering" solution as it was more logical (to me). I must have been thinking of something else in the examination (as you could probably tell, I was not dealing very well with the pressure). I can look into the Jacobian, but as becomes clear later on (section 4.7.1), the real problem is with the ordering of points, so it still won't solve the overall problem.**

- Chapter 7 has been restructured, this is detailed below.

A.4 Minor corrections from Richard Chandler

1. p. 26 Changed "this data" to "these data". Also: p. 54, p. 166, p. 168, p. 170 "is" → "are"; p. 177, p. 213 "was" → "were"; p. 221 "it" → "they".

2. p. 26 Changed to talk about trend surface, rather than process.

3. p. 28 "Complementary"

4. p. 29, p. 30, p. 69, p. 237 "effect" → "affect".

5. p. 30, changed caption to say that the function begins to interpolate the data.

6. p. 30, difference between thin plate splines and thin plate regression splines given from the outset.

7. p. 38 "datum" → "data"

8. p. 40. now reads as requested.

9. p. 41 "PIRLS" → "IRLS", included definition of $\mathcal{V}_g$ in the GAM case. p. 39 Added brief discussion of other smoothing parameter selection methods. The "hierarchical" method is always used by `mgcv` by default (and this was not changed by me for any of the work here), except in the Gaussian/identity link case. This case is covered in section 1.1.4 and section 1.1.2.

10. Section 1.3 title "practice" → "practise", same p. 183

11. p. 42 corrected two typos

12. (1.10) corrected Brier definition to have the data in.

13. p. 44 explained EDF–complexity link better.

14. p. 45 corrected two typos. I have expanded on the "flawed" nature of existing models briefly in the abstract (p. 1), the whole of section 1.4.1 explains why conventional models are flawed.

15. p. 49 corrected typo

16. p. 50 unsure what I meant by "leakage as a breakdown of stationarity", I've elaborated further on how these methods work as an introduction to the MDS stuff that comes later on.

17. p. 50 corrected typo

18. *Chapter 2* p. 54 corrected typo.

19. p. 55 Added comment about missing data, changed plot to highlight those cells with no data to be blue.

20. p. 57 Expanded section on Tweedie distribution.

21. p. 56 Explained why lat/long lead to anisotropy.

22. p. 63 Said that `soap` and `mgcv` were used to construct the model.

23. p. 62 Added an extra line explaining that λ_{int} and λ_{bnd} are actually estimated rather than λ_{space}. I'm inclined to leave (2.3) as having only λ_{space} in as it's true in general for tensor products and it makes the explanation of the tensor product simpler. What do you think?

24. p. 63 Added assumptions for posterior.

25. p. 65 Added Richard's paper to the bibliography.

26. p. 66 Added figure showing spatial groups and changed caption accordingly. Caption also claimed that the grid was 10x10, it was 5x5. p. 65Added comment on the line in the scale-location plot, which does indeed correspond to zeros in the data, coloured these data grey.

27. p. 68 Added comment about how tight the intervals in figure 2-5 are.

28. p. 69 Added timing for model fitting.

29. *Chapter 3* p. 78 Much of this is covered in the re-write of the rectangle section (see corresponding question from SS).

    - p. 75 Tidied up and reordered these paragraphs so that we know we're talking about complex numbers before we start.
    - p. 78 I've removed the equation for the Jacobi elliptic sine function; I don't think that laying out the definition here is actually very useful to the understanding of the Schwarz-Christoffel transform itself and distracts from the rest of the section. The full explanation would probably take up about another page, which seemed like a major diversion.
    - p. 79 "can be computed".
    - p. 78 Removed definition of J_e, see two above.
    - p. 78 More detail on the strip mapping added.
    - p. 80 Removed sentence, as advised by SS, below.

30. *Chapter 4* p. 102, p. 167 "principle" $\rightarrow$ "principal".

31. p. 105 Clarified that we're pretending that we don't know the x_is.

    - p. 106"The first two terms in on" $\rightarrow$ "The contributions from the first two terms"
    - p. 106 **Not clear what Richard means here by "why does the eigen-decomposition have this form" is there any way you can clarify?** I've removed the bit about the Choleski decomposition because I don't think it's actually relevant to the definition of MDS.
    - p. 107 Removed sentence as suggested.

- p. 107 Removed paragraph as suggested.
- p. 108 Corrected matrix dimensions, $(\mathbf{X}^*)^{\mathrm{T}} \rightarrow (\tilde{\mathbf{X}}^*)^{\mathrm{T}}$. p. 108 $\mathrm{diag}(\mathbb{S})_{ii} \rightarrow \mathrm{diag}(\mathbb{S})_i$. Further clarification.
- p. 109 **The fact that you can perform MDS on a reduced set of points, insert the other points and end up with the same configuration as if you'd done the lot from the start means that it is not sensitive to starting values. I've clarified that this is only in the Euclidean case (according to Gower's paper anyway). Unsure whether this is what Richard meant or not.**
- p. 109 "in a similar to check" → "in a similar way to check".

32. p. 109 "is show" → "is shown".
33. p. 111 "novel" → "a novel".
34. p. 114 "the" → "that".
35. p. 114 Spelled out EDF, clarified the noise levels, changed "models" to "fitting methods" (also on p. 163).
36. p. 117 Added explanation of the change in surface from the earlier peninsula domain.
37. p. 124 "sever" → "severe"
38. p. 126 Changed axis labels to be correct.
39. p. 131, p. 137 Clarified what I mean by "squashing".
40. p. 133 Noted that numerical calculation of the penalty wasn't necessary before.
41. p. 135 Removed †, relic of a previous document.
42. p. 137 Clarified that the factor by which the quadrants were stretched were known for one of the cases, clarified that $\mathcal{L}^*$ was calculated from the grid transformed into MDS space.
43. p. 141 Referenced back to section 1.1.
44. p. 141 Response is assumed Gamma distributed.

45. p. 147 Corrected grammar.

46. p. 151 Bug in plotting code caused the dimension to be shifted, corrected.

47. p. 152 "an as such" → "and as such".

48. p. 154 *Chapter 5* "massive" → "a massive".

49. p. 155 Removed incorrect statement about backfitting, put in argument about integrating smoothing parameter selection. Toned down kriging criticism, linked forward to full discussion in chapter 6.

50. p. 159 "fequencies" → "frequencies".

51. p. 161 (Assuming that GCM is a typo here and Richard means GCV). Clarified that at each stage we have GCV-optimal smoothness and that then we pick between the models using the GCV score.

52. p. 164 Boxplots are in groups of 5!

53. p. 168 Made this sentence make more sense.

54. **I'm very much aware of red-green colourblindness as this is a condition that I suffer from! As far as I can tell, all of the figures in the thesis are visible to someone with red-green colourblindness (at least for the subtype which I have). I've tested the figure in question (5-10) and it looks fine on the website suggested.**

55. p. 184 By "smooth" I meant non-linear here, corrected.

56. p. 188 *Chapter 6* "it's" → "its".

57. p. 190 "disirable" → "desirable".

58. p. 193 Removed paragraph.

59. p. 203 *Chapter 7* Added some further discussion here. p. 204 It's the pdf of the distances. Stationarity is covered in the assumptions (p. 200), see also below.

60. p. 206 Typo removed by (SS) correction, below.

61. p. 207 Added hats to N, p and P_a as necessary. Re-write of assumptions section (see below) covers the part about the random sampling beforehand now.

62. p. 205 Mentioned CDS/MCDS earlier.

63. p. 213 Clarified that I'm talking about the log-normality of $\hat{D}$.

64. p. 211 Typo corrected, see modifications below regarding plotting of covariates.

65. p. 229 Plotting of covariates is covered below. Added legend to caption.

A.5 Minor corrections from Simon Shaw

A.5.1 Chapter 1

- p. 29 Added ... (twice).
- p. 29 Reference corrected.
- p. 30 Added ..., removed emboldening.
- p. 32 Changed "Say we put" to jump straight into talking about the matrix.
- p. 34 "practise" $\rightarrow$"practice", throughout.
- p. 34,35 B-spline definition now uses x throughout. Summation corrected and i terms removed.
- p. 35 Objective function corrected.
- p. 36 Removed self-reference.
- p. 36 Corrected definition of cubic splines.
- **p. 36 "p36 β_j and δ_j should be defined in relation to the form of $f(x)$". Can you elaborate on what you mean by this?**
- p. 36 Added discussion of cyclic splines.
- p. 36 Changed definition to $\delta_j = \frac{\partial^2 f(x)}{\partial x^2}|_{x=x_j^*}$
- p. 37 Tensor product definition now has two different size bases.

- p. 38,39 defined $\hat{\mathbf{f}}$.
- p. 39 "leasst" $\rightarrow$ "least".
- p. 40 $\boldsymbol{\beta}^{[k]} \rightarrow \hat{\boldsymbol{\beta}}^{[k]}$
- p. 40 Added ...
- p. 41 Smoothing parameter does need to be a vector, changed later references
- p. 41 Corrected reference
- p. 42 Added brackets to references
- p. 45 Cleared up explanation of a complex boundary with an example
- p. 45 Fixed typo
- p. 47 Fixed typo
- p. 48 Removed
- p. 49 Fixed typo

A.5.2 Chapter 2

- p. 67 Corrected.
- p. 69 Removed.
- p. 69 Clarified that I'm talking about the speed of the fitting for the models. Section 4.5 deals with ensuring that the model we're using is fast enough not to cause frustration to users. Should I reference this here?

A.5.3 Chapter 3

- I've made a number of changes to this opening explanation of the Schwarz-Christoffel transform, hopefully it is now much more consistent. p. 77 Changed Γ to W. p. 73 stated that W is the inside of Γ.
- p. 50 Removed references to φ at this point.

- p. 72 Expanded opening paragraph(s) to discuss the Ramsay horseshoe in a bit more depth.

- p. 73 Replaced figure 3-1 with 3-5; corrected x to w and w^* in caption; reversed diagram to be same ordering as others, later. p. 73 Added some elaboration here.

- p. 74 Cleared up caption and figure. Corrected (and added) line label. Caption: $w_6^* \to w_6$.

- p. 73 Corrected.

- p. 74 Cleared up caption and figure. Changed line label. "upper half-plane" $\to$ "unit disc"

- p. 74 It should be a polygon, that shouldn't have implied that wasn't the case – corrected. p. 75 Clarified φ and φ^{-1} are found. p. 75 $\varphi \to \varphi^{-1}$. p. 75 Cleared up transforming back to the data domain.

- p. 76 Corrected angles in diagram. Increased shading.

- p. 75 Corrected as suggested.

- p. 76 Final φ should be φ^{-1}.

- p. 76 Corrected "may" $\to$ "many".

- p. 76 Attempted to clarify introduction to the section. p. 77 Clarified that the prevertices lie on the real line for the upper half plane case. p. 77 Prevertices are complex in the unit disc case. p. 78 Prevertices for the rectangle case are covered in re-write of that section, see below.

- p. 77 Changed Γ to W. p. 77 First sentence the right way around. p. 77 $w_0 \to w_0^*$. p. 77 Talking about calculation numerically, put into separate paragraph. Added some more information about the base point of the integration.

- p. 78 Added further explanation to the rectangle map, including a figure illustrating how the series of maps work.

- p. 78 Put this computational bit in a separate paragraph, cleared up who solved the numerical issue.

- p. 79 Introduced the section a bit more.
- p. 80 Removed ref to (3.3), twice. p. 80 I've reordered this paragraph, putting the rectangle first and referencing where that comes from. I've left the unit disc but in since I do show examples of the unit disc later.
- p. 80 Removed sentence, don't really need to know this.
- p. 81 Added initial values in step 1 of algorithm.

A.5.4 Chapter 4

- p. 106 I had already mentioned that $\tilde{\mathbf{X}}^{*\mathrm{T}}$ was centered, but added clarification afterwards.
- p. 108 Emboldened.
- p. 116 Added "the".
- p. 117 Re-ordered figures.
- p. 122 Corrected caption.
- p. 121 Cleared up contradiction.
- p. 125 Corrected value in table.
- p. 124 Grammar correction.
- p. 137 Changed equation as suggested.
- p. 139 Corrected line break.

A.5.5 Chapter 5

- p. 154 Added full stop.
- p. 155 Added "that".
- p. 157 Corrected equation.
- p. 156 Changed reference to be to equation in this chapter rather than in chapter 1, did the same for the basis function.

- p. 159 Added "for some choice of s". Added some description of the role of s here and re-ordered the section.
- p. 160 Clarified that I mean the smallest s satisfying (5.7).
- p. 160 Replaced hyphen with colon.
- p. 202 "l" $\rightarrow$ "$\hat{l}$".
- p. 168 Removed "the".
- p. 170 Removed reference to section 1.2.
- p. 170 Corrected grammar.
- p. 176 Corrected reference.
- p. 176 $\mathbf{x}_i \rightarrow \mathbf{m}_i$
- p. 181 "MP example" $\rightarrow$ "MP voting data example"
- p. 184 "fit" $\rightarrow$ "fitted"

A.5.6 Chapter 6

- p. 187 Corrected reference.
- p. 188 Corrected reference.
- p. 192 Added "on".
- p. 192 Added "of".

A.5.7 Chapter 7

- p. 199 "can be written as" $\rightarrow$ "can be estimated by".
- p. 200 Re-ordered section. Removed the field procedure-based assumptions, since we're not really interested in that here. Covered the uniform density wrt to line assumption (random line placement) to begin with, as suggested later.
- p. 200 "animal" $\rightarrow$ "object".

- p. 200 “figures” → “figure’.
- p. 201 Added explanation of the “key function plus adjustment terms” formulation.
- p. 203 Removed parentheses, corrected reference.
- p. 204 Put in x_i.
- p. 204 “Instead” → “instead”.
- p. 206 Removed specific example and re-ordered.
- p. 206 Pointed out that the covariates only affect the scale but not shape, referred back to the detection function definitions. Added further discussion of assumptions. p. 205 Added reference to Borchers (1996).
- p. 207 Corrected definition of ν and ν_i.
- p. 200 Added that random line/point placement is critical into “Assumptions”.
- p. 213 Put the summary at the end.
- p. 211 Added a paragraph on how people deal with monotonicity. p. 210 explicitly mentioned the Distance software, then cited the standard reference for the software.
- p. 234 Moved acknowledgement.

A.5.8 Chapter 8

- p. 214 “being fit to data” → “fitted to the data”.
- p. 215 Referenced the added detection functions.
- p. 215 Corrected equation (8.1). p. 206 Added further discussion in previous chapter about covariates. p. 205 Added reference to Marques and Buckland, 2003. p. 215 Explicitly said that I’m modelling the detection function as conditional on the covariates, as advised.
- p. 216 explicitly stated $\sigma_{ij} = \sigma_i \sigma_j$.

- p. 216 I do mean the conditional likelihood, I've cleared up the section inline with the corrections detailed above.
- p. 217 "$\mathbf{Z}$" $\rightarrow$ "$\mathbf{z}_i$". (Also corrected throughout Appendix 1.)
- p. 217 "$\mathbf{Z}$" $\rightarrow$ "$\mathbf{z}_i$".
- p. 219 Cleared up subscripts.
- p. 219 Clarified what I mean when sorting the distances then splitting them into groups to obtain starting values.
- p. 221 Detection functions now defined in the previous chapter. p. 215 Cleared up what I mean by a "a J-point mixture".
- p. 226 Expanded discussion of the results for the Williams and Thomas (2007) data. p. 226 Expanded caption in figure 8.1. p. 226 Added some discussion of the harbour seal fits.
- p. 208 Added a section about how the plotting is done for covariates. p. 229 Added some more clarification here.
- p. 229 Added discussion of p-values for the amakihi data in the main body.

A.6 Things I found as I went through...

- p. 36, added limits to the penalty for cubic splines.
- p. 26 " (in the sense that they take the value one at one knots and zero at all others)" $\rightarrow$ (in the sense that they take the value one at one knot and zero at all others)
- p. 36 Noted that the b_js that result are implicit.
- Chapter 3 "disk" $\rightarrow$ "disc".
- p. 78 Full stop should have been a colon.
- p. 77 "near the other vertices" $\rightarrow$ "near any other vertex".
- p. 80 Clarified the Schwarz-Christoffel algorithm.
- p. 107 Added some brackets to "$n-1$-dimensional" for clarity.

- p. 107 Corrected bracketing for Gower reference.
- p. 235 “fine” → “find”.
- p. 224 “a” → “at”.
- p. 211 “sate” → “state”.
- p. 211 Corrected Pike reference.
- p. 211 Caption claimed the Pike model was half-normal, when it in fact has a cosine adjustment.
- p. 228 Added some more information on habitat type.
- p. 230 Added level legend for Pike detection function plot.