An investigation into the application of the Finite Element Method in counting process models

by

Boyan Bejanov

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs in partial fulfillment of the requirements for the degree of

Master of Science

in

Probability and Statistics

Carleton University
Ottawa, Ontario

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

A man with one watch knows what time it is;
a man with two watches is never quite sure.

Proverb

To Maria and Yulia.

Acknowledgement

It is my pleasure to offer my deepest gratitude to my supervisor, Jason Nielsen, who has made his support available in many ways, and whose patience, guidance and encouragement have made this thesis possible.

I extend my warmest love and regards to my family, who also supported and encouraged me during the completion of this project. Most of all, I am eternally thankful to my daughter, Maria, for understanding.

Boyan Bejanov

Abstract

The intensity function of a nonhomogeneous point process is estimated using the semi-parametric penalized maximum likelihood approach. We consider the independent Poisson and mixed Poisson models in one dimensional (temporal) and two dimensional (spatial) problems. We develop a methodology based on the Finite Element Method, which can be applied on an arbitrary underlying geometry. The details of using the method are presented for the case where the exact event times/locations are known as well as for the case where panel count data are available. The optimal value of the penalty parameter is selected using two criteria: restricted maximum likelihood and generalized cross validation. The methodology is validated through analyses of simulated and real data.

Table of Contents

List of Tables

List of Figures

Introduction

In this work we develop a numerical technique for the analysis of event counts. This type of data commonly arise in studies of chronic medical conditions, such as asthma, migraine and epilepsy. The Poisson and mixed Poisson models have become the standard in this line of empirical research, since they naturally account for the nonnegative and discrete nature of the data. Often the rate at which the counts are generated changes substantially over time and cannot be approximated reasonably by a constant, rather it is modelled as a function of time. With the continuous increase of computational power and the availability of ever larger data samples, the functional approach to estimation of the underlying intensity function is of growing practical significance.

If the underlying intensity is assumed to be a smooth function a common method would be to model it using splines. However, spline interpolation is only one of many numerical methods that can be used to solve for an unknown smooth function. Here we consider the possibility of applying the method of Finite Elements instead. It is an extremely successful general purpose numerical technique that is widely used in engineering applications, applied mathematics and other areas where computational methods are essential. The Finite Element Method is used in large scale simulations on the world's largest supercomputers and cutting edge methods and technologies are continuously being developed to extend and improve it. The possibility to tap into such immense computational resource and apply it to the analysis of event counts is well worth investigating.

In chapter 1 we recall the basic ideas about point processes and models of event locations/panel counts. We also present the fundamental ideas of the Finite Element Method and describe in some detail the construction of the interpolation basis associated with the specific type of Finite Elements that we use. Chapter 2 fully

develops the models for event locations and panel counts. It also describes all aspects of the numerical procedures that we use for the estimation of the intensity and for the selection of the penalty parameter. Some of the more technical details from this section are outsourced to appendices A and B. In chapter 3, we validate the proposed methodology with simulation studies for each of the models considered in chapter 2. An illustration using a real-world dataset is also given at the end of chapter 3.

1 Background

1.1 Modelling counts

We consider a collection of M random variables $\{N_i(s)\}_{i=1}^{M}$. Each N_i is a stochastic process counting the number of events experienced by the i-th experimental unit. Here s is a deterministic variable which indexes location within the domain of interest Ω. We say that our model is of dimension d, if Ω is a subset of $\mathbb{R}^d$ with non-empty interior. An example of 1-dimensional model is to consider a study that follows M asthma patients over a period of T years and records the number of asthma incidents each patient experiences. In this case, we interpret s as time and set $\Omega = [0, T]$.

Another study, that records the occurrences of a certain type of cancer in the province of BC over a number of years, we analyze using a 3-dimensional model. In this case, Ω is a right cylinder having the territory of the province as its base and s is interpreted as the time and geographic location.

It is possible to record count data in two ways - event time/location data and panel data. In the first case, the exact values of s for all events of all subjects are known. This data may be reported as a collection, $\{s_{ij}\}$, of event locations, where $i = 1, \ldots, M$ and $j = 1, \ldots, N_i$, with N_i being the total count for subject i.

The panel data approach aggregates the data over L observation regions and re-

ports the number of events experienced by each of the M subjects in each observation region. In this case, the data is an $M \times L$ matrix N, with N_{ij} being the count of subject i observed in region j.

The event time/location data is the limiting form of the panel data case, if we let the number of regions, L, increase to infinity, while the measures of the regions shrink down to zero. It is clear that aggregating the data to produce panel data results in loss of information. However, the reality is that panel data are much more common, especially in the medical field outside of critical care situations, since continuous monitoring of patients is expensive and intrusive. Moreover, even when event times/locations are available, they are often recorded imprecisely, and aggregation helps to eliminate problems due to excessive noise in the event times/locations data.

1.1.1 Nonhomogeneous Poisson model

The usual notion of a Poisson process is one of a process counting the number of events as they appear randomly in time, i.e. on a one-dimensional semiinfinite real line (see for example section 2.1 in [24], or [22]). Here we prefer the following equivalent definition of a Poisson process as a random point process, because it generalizes straightforwardly to the multidimensional case (see section 2.5 in [24], or section 2.1 in [18]).

Consider a stochastic process $N(s)$, which is defined for $s \in \Omega \subset \mathbb{R}^d$. We say that $N(s)$ is a Poisson process with intensity $\lambda > 0$, if the following two conditions are met.

1. For any $A \subset \Omega$, the number, $N(A)$, of events occurring in A is a Poisson random variable with mean $\mu_A = \lambda |A|$, where $|A|$ is the measure of A.

2. For any two disjoint subsets $A, B \subset \Omega$, $A \cap B = \emptyset$, the random variables $N(A)$ and $N(B)$ are independent.

The Poisson process is called nonhomogeneous, if the intensity λ is not constant, but a function of s. In this case, we require that $\lambda(s) > 0$ for all $s \in \Omega$, and replace the mean in the definition above with

$$\mu_A = \int_A \lambda(s)ds.$$

Event times/locations data. Let $N_i = N_i(\Omega)$ be the total number of events recorded by the i-th counting process and let s_{ij} for $j = 1, \ldots, N_i$ be the exact locations of these events. Let i and j be arbitrary but fixed and consider a ball $\mathbf{B}_{ij,\epsilon}$ with some small radius $\epsilon > 0$ and centre s_{ij}. Assuming that the Poisson postulates hold, we have

$$\lim_{\epsilon \to 0} \frac{P(N_i(\mathbf{B}_{ij,\epsilon}) = 1)}{|\mathbf{B}_{ij,\epsilon}|} = \lambda(s_{ij}),$$

which can be used to derive the likelihood of the even times/locations data (see [2])

$$\mathcal{L} = \prod_{i=1}^{M} \left\{ \left[\prod_{j=1}^{N_i} \lambda(s_{ij}) \right] \exp\left(- \int_\Omega \lambda(s)ds \right) \right\}.$$

Panel data. For brevity of our exposition we assume that the observation regions are the same for all experimental units, however such assumption is not essential. Let $R_1, \ldots, R_L$ be a partitioning of Ω such that $R_j \cap R_k = \emptyset$, whenever $j \neq k$ and $\bigcup_{j=1}^{L} R_j = \Omega$. The count of process $N_i(s)$ over region R_j is denoted $N_{ij} = N_i(R_j)$, and has expected value

$$\mu_{ij} = \int_{R_j} \lambda(s)ds.$$

The likelihood of the panel data is given by

$$\mathcal{L} = \prod_{i=1}^{M} \prod_{j=1}^{L} \frac{\mu_{ij}^{N_{ij}} \mathrm{e}^{-\mu_{ij}}}{N_{ij}!};$$

this is known as independent Poisson model.

1.1.2 Proportional intensity assumption

So far we have modelled the M subjects identically in their stochastic behaviour. It is more realistic to allow each subject to have their own intensity function $\lambda_i(s)$. The *proportional intensity assumption* of Cox, [9], stipulates that the individual intensities are pairwise proportional, i.e. their ratios do not vary with s. This leads to the following representation of $\lambda_i(s)$

$$\lambda_i(s) = \lambda_o(s)\, \mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}},$$

where $\mathbf{x}_i$ is a vector of covariates associated with subject i, $\boldsymbol{\beta}$ is a vector of covariate effects and $\lambda_o(s)$ is called the *baseline intensity* function.

1.1.3 Overdispersion

One commonly encountered problem when working with the Poisson distribution is *overdispersion*, where the variability of the data is larger than the variability suggested by the model. Since for a Poisson model the mean is equal to the variance, this indicates that the Poisson model is not adequate in general situations.

One well established technique to remedy this situation in the analysis of count data, as suggested in [20], is to introduce a subject-specific random effect ν_i like so

$$\lambda_i(s) = \nu_i \lambda_o(s)\, \mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}}. \tag{1.1}$$

Thus, $N_i(s)$ becomes a mixed Poisson process, and in order to be well defined, the random effect ν_i must be positive. Without loss of generality, we also assume that $\mathrm{E}\big[\nu_i\big] = 1$, so that the unconditional mean of N_i remains unchanged: $\mathrm{E}\big[N_i\big] = \mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_\Omega \lambda_o(s) ds$.

If the distribution of ν_i has a variance parameter τ, it allows the model the added flexibility required to explain the extra observed variability.

The conditional distribution of our data, given ν_i is Poisson

$$N_{ij} | \nu_i \overset{\text{indep}}{\sim} \text{Poisson}(\mu_{ij}), \tag{1.2}$$

with mean

$$\mu_{ij} = \nu_i e^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_{R_j} \lambda_o(s) ds \, . \tag{1.3}$$

However, now the marginal distribution will depend on the distribution of ν_i. In particular, assuming that the ν_i are independent, the likelihood of the data becomes

$$\mathcal{L} = \prod_{i=1}^{M} \int_0^\infty \mathcal{L}_{\{\text{data}|\nu_i=\nu\}} p_i(\nu) d\nu,$$

where $p_i(\nu)$ is the p.d.f. of ν_i.

The usual choice of distribution for ν_i is $\Gamma\left(\frac{1}{\tau}, \tau\right)$ with p.d.f.

$$f(\nu) = \frac{\nu^{\frac{1}{\tau}-1}}{\Gamma\left(\frac{1}{\tau}\right) \tau^{\frac{1}{\tau}}} \exp\left(-\frac{\nu}{\tau}\right), \quad \nu, \tau > 0, \tag{1.4}$$

which has mean 1 and variance τ. In this case the unconditional distribution of N_{ij} will be negative binomial with mean μ_{ij} and variance $\mu_{ij} + \tau \mu_{ij}^2$.

1.1.4 Estimation

The fact that our model fully specifies the distributions of all random quantities, allows the use of Maximum Likelihood estimation. We choose a nonparametric approach to the estimation of the baseline intensity, because of its generality and greater flexibility.

One caveat that is good to address from the start, is the fact that $\lambda_o(s)$ must be

positive. A simple solution, one that allows us to avoid constrained optimization, is to use a transformation of the form

$$\lambda_o(s) = \mathrm{e}^{\alpha(s)}.$$

Now for any function $\alpha(s)$ we get a positive function $\lambda_o(s)$ of the same smoothness.

We employ a penalized maximum likelihood approach, where we seek a smooth function α and a set of parameters $(\tau, \boldsymbol{\beta})$ that maximize the following functional

$$J(\tau, \beta, \alpha) = \mathcal{L}(\tau, \boldsymbol{\beta}, \alpha) - \delta \mathcal{P}(\alpha). \tag{1.5}$$

Here $\mathcal{P}(\alpha)$ is a penalty term, which controls the "smoothness" of α, while δ is a given penalty parameter, which controls the trade-off between fidelity to the observed data (at $\delta = 0$) and smoothness of the solution (as $\delta \to \infty$).

The most commonly used penalty is the square of the L^2 norm of the second derivative of alpha, e.g. in the 1-d case

$$\mathcal{P}(\alpha) = \int_\Omega [\alpha''(s)]^2 ds. \tag{1.6}$$

In general, the penalty term may be any functional defined on the space of admissible α, satisfying $\mathcal{P}(\alpha) \geq 0$. As the penalty parameter, δ, increases, giving more weight to the penalty term in (1.5), the optimal α approaches the function in the null space of $\mathcal{P}$, $\mathrm{Null}(\mathcal{P}) = \left\{\alpha \middle| \mathcal{P}(\alpha) = 0\right\}$, that maximizes the likelihood. In this sense, the choice of penalty functional is in fact a choice of the space of models that are favoured by the maximizer of (1.5). Therefore, the form of the penalty term should be selected in each case, by taking into account the nature of the underlying processes and phenomena.

As suggested in [14], one way to select a penalty term is to consider

$$\mathcal{P}(\alpha) = \int_{\Omega} [L\alpha]^2 ds, \tag{1.7}$$

where L is any linear differential operator. The favoured class of models in this case is the space of solutions of the differential equation $L\alpha = 0$

In the examples considered hereafter, we use the penalty (1.6) for simplicity. However the developed methodology is general and, at least in principle, can be applied with (1.7) with any choice of the operator L.

The use of splines to represent α in the numerical maximization of (1.5) is commonplace, due to their popularity and simplicity of implementation. Furthermore, as shown in [23], a natural cubic spline is the optimal solution in the 1-dimensional case. In this study, we suggest the use of Finite Elements discretization as an alternative to splines.

1.2 The Finite Element Method

The Finite Element method (hereafter abbreviated FEM) has its origins in the mid 1950's, when it was developed as a method for solving elasticity and structural mechanics problems in civil and aeronautical engineering [3, 8]. Since then the number of applications using this method has steadily grown and they nowadays include heat transfer, solid and fluid mechanics, wave scattering, mass transfer, data smoothing and option pricing, just to name a few [7, 28, 4, 25, 21, 1]. The method has been so successful in solving Partial Differential Equations (PDE), that today the term "Finite Element Method" is practically synonymous with the numerical solutions of PDEs.

The FEM derives its immense success from two crucial properties: the superior interpolation properties of the finite element discretization and a flexible solution

framework that allows proper mathematical formulation of a wide range of problems. The former is of practical significance, while the latter is essential, since failure of the approximate solution to achieve acceptable accuracy is often linked to deviation from the mathematical foundations.

It is difficult to give a precise definition of what FEM is since the term is used very loosely to refer to a vast collection of approximation techniques for the solutions of differential equations. Here, we will use only the essence of the method, namely the Finite Element interpolation, and apply it to the analysis of count data instead of splines.

Splines in one dimension are usually represented with a B-spline basis, which is used due to a number of desirable properties [10, 23]. The B-splines, by definition, are the splines of minimal support for the given degree and smoothness, which means that evaluation of a spline at given point requires evaluation of only part of the B-spline basis functions. In addition, B-splines can be computed by the familiar de Boor algorithm, which is numerically very stable and efficient [10]. Similarly, the FE basis functions have a very local support, spanning at most two neighbouring elements, and their computation is straight forward, since explicit formulae are always known.

In two (or more) dimensions, spline bases are often constructed as Cartesian product of two (or more) univariate B-spline bases, one for each dimension; the so-called tensor product B-splines. This approach clearly requires a rectangular domain, which may not be appropriate. The FE interpolation, on the other hand, is famously flexible and can be used on any regular domain. All that is needed is to generate a triangular or quadrilateral grid covering the domain, for which there are many methods, see for example [12, 5], and a rich collection of commercial and open-source libraries is available. In addition, the local support of the FE basis functions means that the matrices involved in the discretization of the problem are sparse. This becomes very impor-

tant from the point of view of implementation, since adequate interpolation in higher dimensions requires ever larger numbers of degrees of freedom, and full matrices of such sizes soon become too large to fit in the computer memory.

In one dimension, the natural cubic spline is a natural choice for the numerical solution of a penalized maximum likelihood estimation problem, due to the following famous property. Among all smooth functions interpolating a set of n values $\{y_i\}$ at n distinct points $\{x_i\}$, the one with minimal second derivative, in the sense of its L_2 norm, $\int_{x_1}^{x_n}[f''(x)]^2dx$, is the natural cubic spline. However, this remarkable property does not generalize to higher dimensions, since the solutions of such minimization problems in two or more dimensions are not piecewise polynomials, see remark 5.7 in [19]. In fact, these minimization problems are solved via variational formulations, which are the back bone of the FE framework, making FEM a natural choice for the multivariate extension of penalized maximum likelihood estimation.

1.2.1 Fundamentals of FE interpolation

General idea of interpolation

Most generally, an interpolation problem is one of finding a function, belonging to a specified space of functions, which satisfies specified interpolation conditions. A necessary condition for the uniqueness of the solution is that the number of interpolation conditions equals the dimension of the functional space.

Let $\mathbb{X}$ be a linear space and let $\mathbb{X}_h$ be a linear sub-space of $\mathbb{X}$, with finite dimension n. Suppose that the following n interpolation conditions are given

$$\gamma_i(f) = f_i, \qquad i = 1, \ldots, n, \tag{1.8}$$

where $f \in \mathbb{X}$ is a function, $\gamma_i : \mathbb{X} \to \mathbb{R}$ are linear functionals on $\mathbb{X}$ and $f_i \in \mathbb{R}$ are real numbers. A function $f_h \in \mathbb{X}_h$ which satisfies the interpolation conditions (1.8) is

called th ıe *interpolant* of f in $\mathbb{X}_h$. The subspace $\mathbb{X}_h$ is called the *interpolation space* or *appro ximation space.*

Let $\{\varphi_j, j = 1, \ldots, n\}$ be a basis of $\mathbb{X}_h$. Then each function in $\mathbb{X}_h$ is a linear combina tion of the φ_j's, in particular, there is a set of real numbers $\{a_j, j = 1, \ldots, n\}$, such tha t

$$f_h = \sum_{j=1}^{n} a_j \varphi_j.$$

Applying each of the conditions (1.8) on f_h and using the linearity of the γ_i's, we get the following system of equations

$$\gamma_i (f_h) = \sum_{j=1}^{n} a_j \gamma_i (\varphi_j) = f_i, \qquad i = 1, \ldots, n. \tag{1.9}$$

This is a linear system of n equations with n unknowns. Written in matrix-vector notation, it can be expressed as

$$\mathbf{Sa} = \mathbf{f} \tag{1.10}$$

where $\mathbf{S}$ is an $n \times n$ matrix with entries $\mathbf{S}_{ij} = \gamma_i(\varphi_j)$, $\mathbf{a} = (a_1, \ldots, a_n)^T$ is a vector of the unknown coefficients and $\mathbf{f} = (f_1, \ldots, f_n)^T$. The matrix $\mathbf{S}$ is called the *interpolation matrix* and depends on the choice of the basis $\{\varphi_i\}$. The interpolation problem has a unique solution provided that $\mathbf{S}$ is nonsingular.

A special choice of the basis $\{\varphi_j\}$, which satisfies

$$\gamma_i(\varphi_j) = \delta_{ij}, \tag{1.11}$$

where δ_{ij} is the Kronecker symbol, is called the *interpolation basis.* In this case the interpolation matrix is the $n \times n$ identity and the solution of (1.10) is trivially

$$a_i = f_i, \qquad i = 1, \ldots, n.$$

The mapping $\mathcal{I} : \mathbb{X} \to \mathbb{X}_h$, which associates functions f with their interpolants $f_h = \mathcal{I}f$ is called the *interpolation operator*. In the interpolation basis, the interpolation operator is

$$\mathcal{I}f = \sum_{i=1}^{n} \gamma_i(f)\varphi_i, \qquad \forall f \in \mathbb{X}.$$

The linear functionals γ_i are often called *degrees of freedom*. The functions φ_i that form the interpolation basis for given degrees of freedom, are called *shape functions*. The interpolation operator is specified by the interpolation space and the degrees of freedom.

Example 1. Find the straight line passing through two points (x_1, y_1) and (x_2, y_2) with $x_1 \neq x_2$.

For this problem, the interpolation space is $\mathbb{X}_h = \mathbb{P}_1$, the space of polynomials of degree not exceeding 1. We may consider $\mathbb{X}$, most generally, to be the space of all functions that are defined at x_1 and x_2. Since the interpolation conditions are

$$f_h(x_1) = y_1, \qquad f_h(x_2) = y_2,$$

we have the following degrees of freedom

$$\gamma_1(f) = f(x_1), \qquad \gamma_2(f) = f(x_2).$$

It is easy to verify that the shape functions are

$$\varphi_1(x) = \frac{x_2 - x}{x_2 - x_1}, \qquad \varphi_2(x) = \frac{x - x_1}{x_2 - x_1}.$$

Thus, the solution of our interpolation problem is

$$y_h(x) = y_1\varphi_1(x) + y_2\varphi_2(x) = y_1\frac{x_2 - x}{x_2 - x_1} + y_2\frac{x - x_1}{x_2 - x_1},$$

and in general, the interpolation operator for this problem is

$$\mathcal{I}f(x) = f(x_1)\frac{x_2 - x}{x_2 - x_1} + f(x_2)\frac{x - x_1}{x_2 - x_1}.$$

Interpolation using finite elements

There are a large number of introductory texts on FEM. While writing what follows, we have referred to the introductory sections of more advanced FEM books, namely chapter 1 in [11] and chapter 2 in [7].

The Finite Element Method is first and foremost a systematic technique for building finite-dimensional interpolation spaces. The first step is to *discretize* the domain, Ω, by partitioning it into small and simple subdomains called *finite elements*. In 1-d the finite elements are always intervals, while in 2-d they are usually triangles or quadrilaterals. Usually the partitioning is characterized by a *discretization parameter* h, which measures the maximum size of the finite elements. We denote the set of all finite elements by $\mathcal{T}_h$. The finite element interpolation space $\mathbb{X}_h$ is defined through a specific process, which we will first describe in general and later illustrate with examples.

The most basic aspect of the FEM, is that the interpolation problem is solved locally, within each finite element, and then the local interpolants are patched together to form a global solution defined on the whole Ω.

In order to define a type of finite elements, one must specify a geometric shape and formulate an interpolation problem on it. Let $K \in \mathcal{T}_h$ be a finite element. The interpolation problem on K is specified by choosing an interpolation space, $\mathcal{P}_K$, and degrees of freedom $\Gamma_K = \left\{\gamma_i^K, i = 1, \ldots, k\right\}$. From these, one can derive the corresponding shape functions, which we denote $\left\{\varphi_i^K, i = 1, \ldots, k\right\}$. Here k equals the dimension of $\mathcal{P}_K$.

The second basic aspect of the FEM is that the spaces $\mathcal{P}_K$ contain polynomi-

als. Although, the methodology can be extended to more general local interpolation spaces, all convergence results and convenient implementation techniques of FEM rely on $\mathcal{P}_K$ being polynomial spaces.

The global interpolation space contains functions, whose restrictions to the finite elements belong to their local interpolation spaces, but usually not all such functions. In addition, there may be a requirement that the functions in $\mathbb{X}_h$ be of a certain smoothness. For example, if we require m-times differentiable functions, then we may have

$$\mathbb{X}_h = \left\{ v \in C^m(\Omega) \,\middle|\, v|_K \in \mathcal{P}_K, \forall K \in \mathcal{T}_h \right\}. \tag{1.12}$$

In the simplest form of FEM, the local interpolation problem is the standard Lagrangian interpolation, where one seeks a polynomial that matches given function values at given points, also called *interpolation nodes*. When this is the case, we say that the finite element is of *Lagrangian* type. Thus, each element is equipped with k local interpolation nodes.

When two (or more) neighbouring finite elements share a common node, naturally they also share the associated degree of freedom. All degrees of freedom of all finite elements in the discretization of Ω make up the set of *global* degrees of freedom, which we denote Γ.

The third basic aspect of the FEM is that the interpolation basis in $\mathbb{X}_h$ for the degrees of freedom in Γ contains shape functions that are easy to describe and whose supports are localized to a small number of finite elements.

Consider a global degree of freedom $\gamma_j \in \Gamma$. The global shape function φ_j corresponding to γ_j is defined piecewise on each element $K \in \mathcal{T}_h$ through the local shape

functions as follows

$$\varphi_j : \Omega \to \mathbb{R} \quad \text{s.t.} \quad \varphi_j\Big|_K = \begin{cases} \varphi_i^K & \text{if } \gamma_j = \gamma_i^K \in \Gamma_K, \\ 0 & \text{if } \gamma_j \notin \Gamma_K. \end{cases} \tag{1.13}$$

It is easy to see that the functions defined this way form an interpolation basis for the degrees of freedom in Γ on the space they span. However, it is not necessarily true that the functions defined in (1.13) span $\mathbb{X}_h$ as defined in (1.12), i.e. the φ_j's may not have the required smoothness. For now, we will just say that the smoothness of the shape functions must be insured by clever choice of the local nodes and degrees of freedom and later we will discuss further this issue.

To illustrate the process of constructing finite element interpolation spaces we will consider the simplest cases of 1-d and 2-d piecewise linear interpolation.

Example 2. *1-d linear Lagrangian Finite Element.* See section 1.1.2 in [11] for a detailed treatment of this FE type.

Let $\Omega = [a, b]$. For $h = \frac{b-a}{n}$ we have the following partitioning of Ω into n finite elements

$$\mathcal{T}_h = \Big\{ K_i = [x_{i-1}, x_i], i = 1, ..., n \Big\}, \tag{1.14}$$

where

$$x_i = a + ih, \qquad i = 0, \ldots, n. \tag{1.15}$$

On each element we have a linear interpolation problem, the same one we considered in example 1. The local interpolation space is $\mathcal{P}_{K_i} = \mathbb{P}_1 = \text{span}\{1, x\}$, the same for all i. We choose the local interpolation nodes to be the endpoints of the finite element, namely x_{i-1} and x_i. The local degrees of freedom are the function values at the nodes

$$\gamma_1^{K_i}(f) = f(x_{i-1}), \qquad \gamma_2^{K_i}(f) = f(x_i),$$

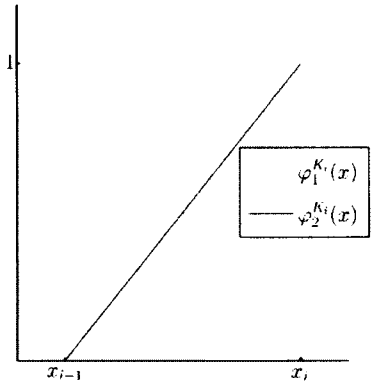

Figure 1: Local basis of 1d linear Lagrangian Finite Element

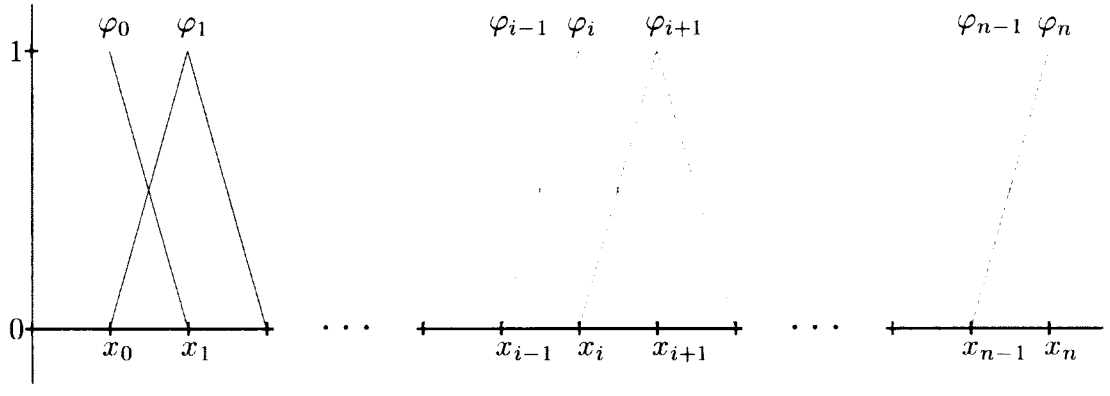

Figure 2: Global basis of 1d linear Lagrangian Finite Element

and the local shape functions are (see figure 1):

$$\varphi_1^{K_i}(x) = \frac{x_i - x}{h}, \qquad \varphi_2^{K_i}(x) = \frac{x - x_{i-1}}{h}. \tag{1.16}$$

All interpolation nodes of all finite elements are $\{x_i, i = 0, ..., n\}$ and the function values at these points are the global degrees of freedom

$$\gamma_i(f) = f(x_i), \qquad i = 0, ..., n.$$

Clearly, each interior node, x_i for $i = 1, ..., n-1$, is common to two elements: it is local node 2 in K_i and local node 1 in K_{i+1}. So, the global shape functions are (see

figure 2):

$$i = 0 \qquad \varphi_0(x) = \begin{cases} \varphi_1^{K_1}(x) = \dfrac{x_1 - x}{h}, & x \in K_1 \\ 0 & \text{otherwise} \end{cases}$$

$$i = 1, ..., n-1 \qquad \varphi_i(x) = \begin{cases} \varphi_2^{K_i}(x) = \dfrac{x - x_{i-1}}{h}, & x \in K_i \\ \varphi_1^{K_{i+1}}(x) = \dfrac{x_{i+1} - x}{h}, & x \in K_{i+1} \\ 0 & \text{otherwise} \end{cases}$$

$$i = n \qquad \varphi_n(x) = \begin{cases} \varphi_2^{K_n}(x) = \dfrac{x - x_{n-1}}{h}, & x \in K_n \\ 0 & \text{otherwise} \end{cases}$$

It is not difficult to verify that $\varphi_i(x_j) = \delta_{ij}$. In addition, the continuity of the shape functions is achieved by placing the interpolation nodes at the points where the different pieces of $\varphi_i(x)$ must connect, i.e. at the endpoints of the finite elements. Thus, it is insured that at the endpoints of the finite elements, x_j, the value from the left and the value from the right both match the value of the associated degree of freedom, γ_j. Therefore, the global interpolation space, $\text{span}\{\varphi_0, \ldots, \varphi_n\}$, is (see proposition 1.1 in [11]):

$$\mathbb{X}_h = \left\{ v \in C([a,b]) \,\middle|\, v|_K \in \mathbb{P}_1, \forall K \in \mathcal{T}_h \right\}.$$

Example 3. *2-d linear Lagrangian triangular finite element.* (section 1.2.3 in [11]).

Let $\Omega \subset \mathbb{R}^2$ be closed, bounded and connected and let $\mathcal{T}_h$ be a triangulation of Ω. Each finite element, $K \in \mathcal{T}_h$ is a triangle (see figure 3). The local interpolation space is the space of bivariate polynomials of degree one, $\mathcal{P}_K = \mathbb{P}_1 = \text{span}\{1, x, y\}$, which has dimension 3. We choose as interpolation nodes the three vertices of K, which we denote by $\mathbf{x}_i^K, i = 1, 2, 3$. Here the bold x denotes a point in the xy-plane,

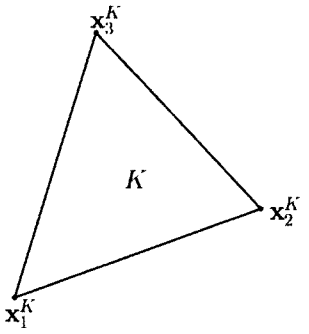

Figure 3: Triangular 2d finite element

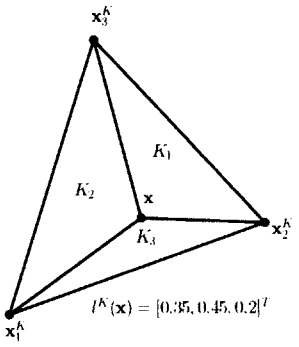

Figure 4: Barycentric coordinates

i.e. $\mathbf{x}_i^K = (x_i^K, y_i^K)$; we will also use the notation $f(\mathbf{x})$ to denote $f(x, y)$. The degrees of freedom are the function values at the nodes

$$\gamma_i^K(f) = f(\mathbf{x}_i^K), \quad i = 1, 2, 3.$$

It is customary and convenient to work with the so called *barycentric coordinates* (see section 2.1 in [19]). With each point, $\mathbf{x}$, in the interior of K we can associate a triple of real numbers $(\ell_1^K, \ell_2^K, \ell_3^K)$, which determine uniquely the position of $\mathbf{x}$ with respect to the triangle. The point $\mathbf{x}$ splits K into three triangles, K_1, K_2, K_3, as shown in figure 4. By definition, the barycentric coordinates sum up to one and are in the same proportion as the areas of the three triangles. One convenient formula for the area of a triangle using the Cartesian coordinates of its three vertices is

$$A_K = \frac{1}{2} \begin{vmatrix} 1 & 1 & 1 \\ x_1^K & x_2^K & x_3^K \\ y_1^K & y_2^K & y_3^K \end{vmatrix}.$$

This formula gives the signed area of K, with positive sign if the numbering of the three vertices increase in counter-clockwise direction about K, as shown in figure 3,

and the sign is negative otherwise. Similarly, the areas of the three subtriangles are

$$A_{K_1}(\mathbf{x}) = \frac{1}{2}\begin{vmatrix} 1 & 1 & 1 \\ x & x_2^K & x_3^K \\ y & y_2^K & y_3^K \end{vmatrix}, A_{K_2}(\mathbf{x}) = \frac{1}{2}\begin{vmatrix} 1 & 1 & 1 \\ x_1^K & x & x_3^K \\ y_1^K & y & y_3^K \end{vmatrix}, A_{K_3}(\mathbf{x}) = \frac{1}{2}\begin{vmatrix} 1 & 1 & 1 \\ x_1^K & x_2^K & x \\ y_1^K & y_2^K & y \end{vmatrix} \tag{1.17}$$

Since the areas of the three subtriangles add up to the area of K, we have the following formula for the barycentric coordinates

$$\ell_1^K(\mathbf{x}) = \frac{A_{K_1}(\mathbf{x})}{A_K}, \quad \ell_2^K(\mathbf{x}) = \frac{A_{K_2}(\mathbf{x})}{A_K} \quad \text{and } \ell_3^K(\mathbf{x}) = \frac{A_{K_3}(\mathbf{x})}{A_K}. \tag{1.18}$$

The definition of barycentric coordinates can be extended to points outside K by the above formulas. The points inside K are characterized by the fact that all three barycentric coordinates are positive (and less than one), while for points outside K there is always at least one ℓ that is negative.

Notice that the barycentric coordinate $\ell_i^K(\mathbf{x})$ is one when $\mathbf{x} = \mathbf{x}_i^K$ and zero at the other two vertices, i.e.

$$\ell_i^K(\mathbf{x}_j^K) = \delta_{ij}, \quad \forall i, j = 1, 2, 3.$$

This follows from the definition and can be verified directly from (1.18). It is also clear from (1.17) that the barycentric coordinates are affine functions of x and y. Therefore, they are the local interpolation basis for the linear Lagrangian triangular FE (see figure 5)

$$\varphi_i^K(\mathbf{x}) = \ell_i^K(\mathbf{x}), \quad i = 1, 2, 3. \tag{1.19}$$

The global degrees of freedom are once again the function values at the nodes and the global shape functions are constructed piecewise from the local shape functions according to formula (1.13). A typical shape function is illustrated in figure 6. The continuity of the global shape functions in all of Ω is a special case of Theorem 2.2.3

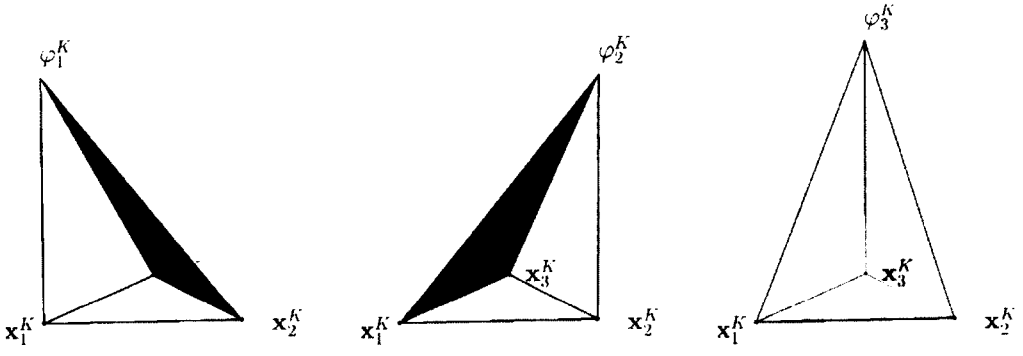

Figure 5: Local basis of 2d linear Lagrangian Finite Element

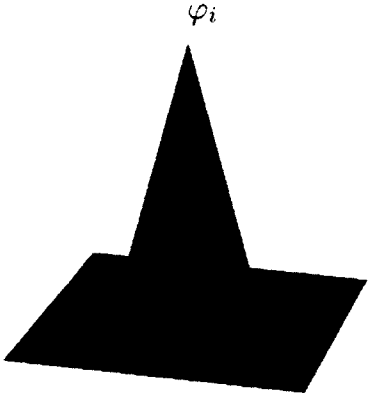

Figure 6: Global basis of 2d linear Lagrangian Finite Element

in [7]. Consider a function $v \in \text{span}\{\varphi_i\}$ and an edge e of the grid, which is the common side of two adjacent triangles, K_i and K_j. The two pieces of v from the two triangles, $v|_{K_i}$ and $v|_{K_j}$, restricted to the common edge are lines that match at the two nodes of e. Therefore, they match at every point of e, making v a continuous function. Thus the global interpolation space is

$$\mathbb{X}_h = \left\{ v \in C^0(\Omega) \,\middle|\, v|_K \in \mathbb{P}_1, \forall K \in \mathcal{T}_h \right\}.$$

1.2.2 The C^1 cubic finite element

The motivation behind using C^1 finite elements is that this is the lowest smoothness that is sufficient to penalize the second derivative. In order for (1.6) to be well defined, we need functions that have square integrable second derivatives. To see that this holds, consider a finite element interpolation space (1.12) for $m = 1$. If $v_h \in \mathbb{X}_h$, then v_h and its first derivatives are continuous, so $v_h \in L^2(\Omega)$ and $\nabla v_h \in L^2(\Omega)$. In addition, the second derivatives of v_h are piecewise polynomial functions, which may have jump discontinuities at the interfaces between elements. Nevertheless, the second derivatives are also square integrable. Therefore $v_h \in H^2(\Omega)$ (proposition 1.81 in [11]).

We saw in the previous section that C^0 continuity of the global FE interpolants is achieved by placing interpolation nodes at the vertices of the finite elements and having the function values as the degrees of freedom. In a similar way, in this section we obtain C^1 finite element interpolation by using the same nodes and having as degrees of freedom the values of the first derivatives in addition to the function values. This is known as *Hermitian interpolation* and so finite elements of this type are called *Hermitian.*

The 1-d cubic Hermitian finite element (section 1.4.6 in [11]).

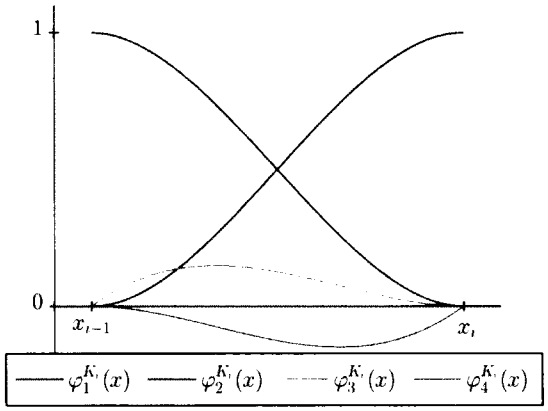

Figure 7: Local basis of 1d cubic Hermitian Finite Element

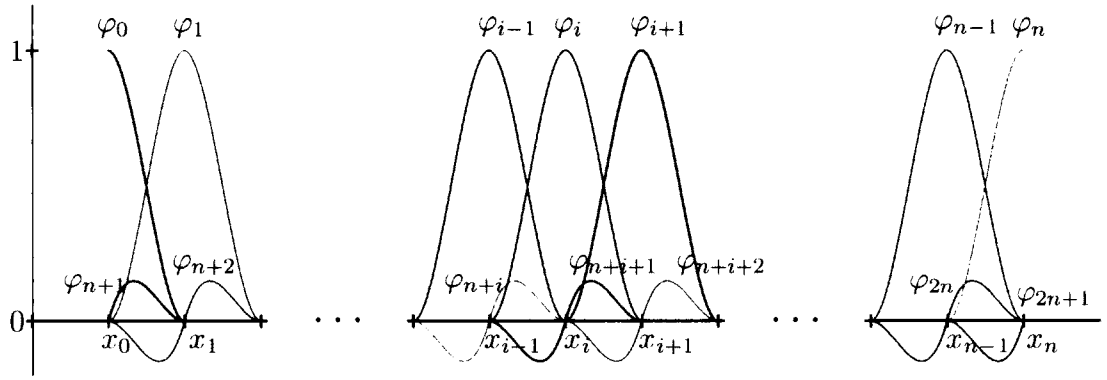

Figure 8: Global basis of 1d cubic Hermitian Finite Element

The partitioning of Ω into finite elements and interpolation nodes is the same here as in the linear Lagrangian FE discussed above in (1.14) and (1.15).

Consider a finite element K_i with nodes x_{i-1} and x_i. We have two degrees of freedom at each node, namely

$$\begin{aligned} \gamma_1^{K_i}(f) &= f(x_{i-1}), & \gamma_2^{K_i}(f) &= f(x_i), \\ \gamma_3^{K_i}(f) &= f'(x_{i-1}) \quad \text{and} & \gamma_4^{K_i}(f) &= f'(x_i). \end{aligned} \tag{1.20}$$

Because we have four interpolation conditions, the natural choice of local approximation space is the four dimensional $\mathbb{P}_3 = \text{span}\{1, x, x^2, x^3\}$. The shape functions associated with this Hermitian interpolation problem can be written in terms of the

one-dimensional analogue of barycentric coordinates. Specifically, if we denote

$$\ell_1^{K_i}(x) = \frac{x_i - x}{x_i - x_{i-1}} \quad \text{and} \quad \ell_2^{K_i}(x) = \frac{x - x_{i-1}}{x_i - x_{i-1}}, \tag{1.21}$$

we have the following formulas for the shape functions (see figure 7).

$$\begin{aligned}
\varphi_1^{K_i}(x) &= \ell_1^{K_i}(x) + \ell_1^{K_i}(x)\ell_2^{K_i}(x)\Big[1 - 2\ell_2^{K_i}(x)\Big] \\
\varphi_2^{K_i}(x) &= \ell_2^{K_i}(x) + \ell_1^{K_i}(x)\ell_2^{K_i}(x)\Big[1 - 2\ell_1^{K_i}(x)\Big] \\
\varphi_3^{K_i}(x) &= (x_i - x_{i-1})\Big[\ell_1^{K_i}(x)\Big]^2 \ell_2^{K_i}(x) \\
\varphi_4^{K_i}(x) &= -(x_i - x_{i-1})\ell_1^{K_i}(x)\Big[\ell_2^{K_i}(x)\Big]^2
\end{aligned} \tag{1.22}$$

The global degrees of freedom are $2n + 2$, and just to fix notation we number them so that degrees of freedom from 0 to n are the function values, while the ones from $n + 1$ to $2n + 2$ are the values of the first derivative. More specifically, the four local degrees of freedom of element K_i are the same as global degrees of freedom $i-1, i, n+i, n+i+1$ in this order. The standard formula (1.13) for the global shape functions applies here. Some global shape functions are illustrated in figure 8.

The 2-d cubic Hermitian triangular finite element (section 2.2 in [7]).

Once again we start with the same triangulation $\mathcal{T}_h$ of Ω as in the linear Lagrangian case and consider a single element $K \in \mathcal{T}_h$. The local approximation space is

$$\mathbb{P}_3 = \text{span}\left\{1, x, y, x^2, xy, y^2, x^3, x^2y, xy^2, y^3\right\},$$

which has dimension ten. There are nine degrees of freedom for the values of the function and its two partial first derivatives at the three vertices of K. In order to

close the system, we add an interpolation node at the centroid of the triangle

$$\mathbf{x}_4^K = \frac{1}{3}\sum_{i=1}^{3}\mathbf{x}_i^K$$

and the function value at $\mathbf{x}_4^K$ becomes the tenth degree of freedom. We have

$$\begin{aligned}
&\gamma_1^K(f) = f(\mathbf{x}_1^K), \quad \gamma_2^K(f) = f(\mathbf{x}_2^K), \quad \gamma_3^K(f) = f(\mathbf{x}_3^K),\\
&\gamma_4^K(f) = \frac{\partial f(\mathbf{x}_1^K)}{\partial x}, \quad \gamma_6^K(f) = \frac{\partial f(\mathbf{x}_2^K)}{\partial x}, \quad \gamma_8^K(f) = \frac{\partial f(\mathbf{x}_3^K)}{\partial x},\\
&\gamma_5^K(f) = \frac{\partial f(\mathbf{x}_1^K)}{\partial y}, \quad \gamma_7^K(f) = \frac{\partial f(\mathbf{x}_2^K)}{\partial y}, \quad \gamma_9^K(f) = \frac{\partial f(\mathbf{x}_3^K)}{\partial y},\\
&\gamma_{10}^K = f(\mathbf{x}_4^K).
\end{aligned} \tag{1.23}$$

This is a well defined two-dimensional interpolation problem and has a unique solution, as long as the triangle K is not degenerate (Theorem 2.2.8 in [7]).

In order to simplify our notation, until the end of this paragraph, we will drop the super-script K indicating that local quantities are associated with finite element K. We will reintroduce this notation, where it is necessary to avoid ambiguity.

The cubic Hermite triangle. We consider the following basis of $\mathbb{P}_3$, using the barycentric coordinates defined in (1.18) (see equation 2.2.37 in [7]).

$$\begin{aligned}
\tilde{b}_1(\mathbf{x}) &= -2\Big[\ell_1(\mathbf{x})\Big]^3 + 3\Big[\ell_1(\mathbf{x})\Big]^2 - 7\ell_1(\mathbf{x})\ell_2(\mathbf{x})\ell_3(\mathbf{x})\\
\tilde{b}_2(\mathbf{x}) &= -2\Big[\ell_2(\mathbf{x})\Big]^3 + 3\Big[\ell_2(\mathbf{x})\Big]^2 - 7\ell_1(\mathbf{x})\ell_2(\mathbf{x})\ell_3(\mathbf{x})\\
\tilde{b}_3(\mathbf{x}) &= -2\Big[\ell_3(\mathbf{x})\Big]^3 + 3\Big[\ell_3(\mathbf{x})\Big]^2 - 7\ell_1(\mathbf{x})\ell_2(\mathbf{x})\ell_3(\mathbf{x})
\end{aligned} \tag{1.24}$$

$$\begin{aligned}
\tilde{b}_4(\mathbf{x}) &= \ell_1(\mathbf{x})\ell_2(\mathbf{x})\Big[2\ell_1(\mathbf{x}) + \ell_2(\mathbf{x}) - 1\Big]\\
\tilde{b}_5(\mathbf{x}) &= \ell_1(\mathbf{x})\ell_3(\mathbf{x})\Big[2\ell_1(\mathbf{x}) + \ell_3(\mathbf{x}) - 1\Big]
\end{aligned} \tag{1.25}$$

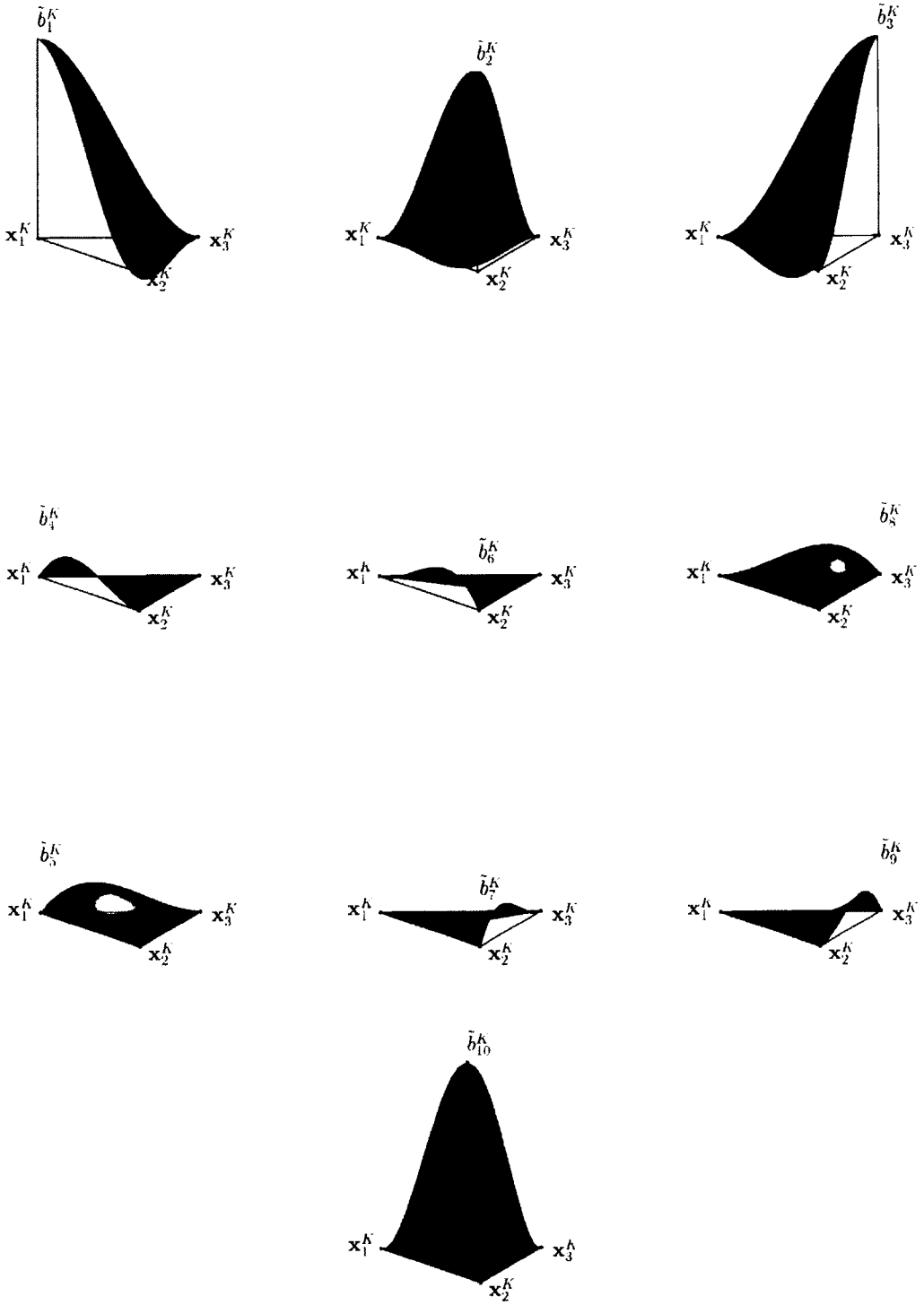

Figure 9: Local basis of cubic Hermitian triangle

$$\tilde{b}_6(\mathbf{x}) = \ell_2(\mathbf{x})\ell_1(\mathbf{x})\Big[2\ell_2(\mathbf{x}) + \ell_1(\mathbf{x}) - 1\Big]$$
$$\tilde{b}_7(\mathbf{x}) = \ell_2(\mathbf{x})\ell_3(\mathbf{x})\Big[2\ell_2(\mathbf{x}) + \ell_3(\mathbf{x}) - 1\Big] \tag{1.26}$$

$$\tilde{b}_8(\mathbf{x}) = \ell_3(\mathbf{x})\ell_1(\mathbf{x})\Big[2\ell_3(\mathbf{x}) + \ell_1(\mathbf{x}) - 1\Big]$$
$$\tilde{b}_9(\mathbf{x}) = \ell_3(\mathbf{x})\ell_2(\mathbf{x})\Big[2\ell_3(\mathbf{x}) + \ell_2(\mathbf{x}) - 1\Big] \tag{1.27}$$

$$\tilde{b}_{10}(\mathbf{x}) = 27\ell_1(\mathbf{x})\ell_2(\mathbf{x})\ell_3(\mathbf{x}) \tag{1.28}$$

These local basis functions are illustrated in figure 9.

The first three functions, (1.24), are associated with the function values at the vertices of K. Functions four and five, (1.25), are associated with two derivatives at vertex $\mathbf{x}_1$, functions six and seven, (1.26), are associated with a pair of derivatives at $\mathbf{x}_2$, functions eight and nine, (1.27) with derivatives at $\mathbf{x}_3$, and finally function ten, (1.28), takes care of the function value at the centroid. However, this is not an interpolation basis, since (1.11) does not hold.

Direct verification confirms that the derivatives of $\tilde{b}_4$ and $\tilde{b}_5$ at $\mathbf{x}_1$ are

$$\nabla\tilde{b}_4(\mathbf{x}_1) = \nabla\ell_2, \qquad \nabla\tilde{b}_5(\mathbf{x}_1) = \nabla\ell_3. \tag{1.29}$$

Notice that the gradients of the barycentric coordinates are constant. Also, the graph of $z = \ell_i(x, y)$ is a plane that takes value 1 at $\mathbf{x}_i$ and crosses the xy-plane along the side of K that is opposite vertex $\mathbf{x}_i$ (see figure 5). Therefore, $\nabla\ell_i$ is a vector perpendicular to that side and pointing inwards.

We see that $\tilde{b}_4$ and $\tilde{b}_5$ are shape functions for the directional derivatives at $\mathbf{x}_1$ in directions perpendicular to the sides of K that meet at vertex $\mathbf{x}_1$. All we need to do is rotate the two functions so that their gradients point in the x- and y- coordinate directions instead. In particular, we seek two functions b_4 and b_5 as linear combinations

of $\tilde{b}_4$ and $\tilde{b}_5$,

$$\begin{pmatrix} b_4(\mathbf{x}) \\ b_5(\mathbf{x}) \end{pmatrix} = A \begin{pmatrix} \tilde{b}_4(\mathbf{x}) \\ \tilde{b}_5(\mathbf{x}) \end{pmatrix}, \tag{1.30}$$

such that

$$\nabla b_4(\mathbf{x}_1) = (1,0)^T, \qquad \nabla b_5(\mathbf{x}_1) = (0,1)^T \tag{1.31}$$

From (1.29), (1.30), and (1.31) it follows that A is the matrix solution of

$$\begin{pmatrix} \nabla \ell_2 & \nabla \ell_3 \end{pmatrix} A^T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \quad \Longrightarrow \quad A = \begin{pmatrix} \nabla \ell_2 & \nabla \ell_3 \end{pmatrix}^{-T},$$

where $(.)^{-T}$ means the transposed inverse matrix.

A new basis for $\mathbb{P}_3$ is

$$\begin{gathered} b_1(\mathbf{x}) = \tilde{b}_1(\mathbf{x}) \quad b_2(\mathbf{x}) = \tilde{b}_2(\mathbf{x}) \quad b_3(\mathbf{x}) = \tilde{b}_3(\mathbf{x}) \\ \begin{pmatrix} b_4(\mathbf{x}) \\ b_5(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \nabla \ell_2 & \nabla \ell_3 \end{pmatrix}^{-T} \begin{pmatrix} \tilde{b}_4(\mathbf{x}) \\ \tilde{b}_5(\mathbf{x}) \end{pmatrix} \\ \begin{pmatrix} b_6(\mathbf{x}) \\ b_7(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \nabla \ell_1 & \nabla \ell_3 \end{pmatrix}^{-T} \begin{pmatrix} \tilde{b}_6(\mathbf{x}) \\ \tilde{b}_7(\mathbf{x}) \end{pmatrix} \\ \begin{pmatrix} b_8(\mathbf{x}) \\ b_9(\mathbf{x}) \end{pmatrix} = \begin{pmatrix} \nabla \ell_1 & \nabla \ell_2 \end{pmatrix}^{-T} \begin{pmatrix} \tilde{b}_8(\mathbf{x}) \\ \tilde{b}_9(\mathbf{x}) \end{pmatrix} \\ b_{10}(\mathbf{x}) = \tilde{b}_{10}(\mathbf{x}) \end{gathered} \tag{1.32}$$

This basis is the one we will work with, although it is still not an interpolation basis. We have $\gamma_i(b_j) = \delta_{ij}$ for $i = 1, \ldots, 9$ and $j = 1, \ldots, 10$. However $\gamma_{10}(b_j) \neq 0$ for $j = 1, \ldots, 9$, i.e. the first nine basis functions do not vanish at the centroid of the triangle. It is possible to modify the first nine functions so that the basis would satisfy (1.11), but we won't do this, because it is not necessary for our purposes. This

is because the cubic Hermitian triangle does not yield C^1 smooth interpolation and we will not use it as our interpolation space. Nevertheless, we will use this Hermitian basis to compute the basis of a truly C^1 interpolation basis, which we present later in this section.

To see that this basis is not C^1, let us discuss the smoothness of a global interpolant, v_h, assembled piecewise from the solutions of local interpolation problems. Consider an interior edge, separating two adjacent finite elements. Along the edge, both pieces of v_h are (univariate) cubic polynomials. They match values at the two vertices of the edge. Also, their derivatives in direction tangential to the common edge match at the vertices. We have two cubic polynomials satisfying the same four Hermitian interpolation conditions, therefore they match everywhere along the edge. Lastly, the derivatives of both pieces of v_h in direction perpendicular to the edge are quadratic polynomials that match at the two vertices. But two conditions are not sufficient to uniquely determine a quadratic polynomial. It is clear from these considerations that this finite element is C^0 (theorem 2.2.10 in [7]), however, it is not C^1, since the global interpolation space contains functions that do not belong in $C^1(\Omega)$.

The discontinuity of the normal derivatives along interior edges of the finite element grid can be seen in figure 10, showing typical global basis functions associated with the function value and the function's x- and y- partial derivatives at node $\mathbf{x}_i$. There are a total of $3n + k$ degrees of freedom, where n denotes the number of nodes and k denotes the number of triangles in the grid. In addition to these $3n$ functions, we have one global basis function associated with the function value at the centroid of each triangle. Global basis function b_{3n+i} equals local basis function $b_{10}^{K_i}$ within element K_i and 0 outside K_i and can be seen at the bottom of figure 9.

The Hsieh-Clough-Toucher triangle. (section 6.2 in [19], section 6.1 in [7]).

The HCT triangle is a *macro element* in the sense that one HCT triangle is in fact

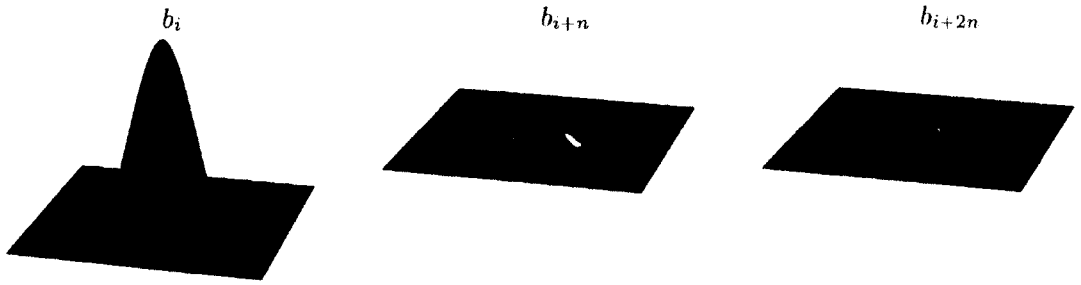

Figure 10: Global basis of cubic Hermitian triangle

a *cluster* of three cubic Hermitian triangles (see figure 11). In this context we refer to the three sub-elements as *micro elements.* The clustering allows us to introduce the normal derivatives at the midpoints of the sides of the triangle as degrees of freedom, insuring C^1 continuity of the global interpolant, without increasing the degree of the underlying polynomial pieces.

We split K into three triangles, K_1, K_2, K_3, by connecting the centroid, $\mathbf{x}_4$, to each of the three vertices. The naming convention here is that micro element K_i is the one not containing vertex $\mathbf{x}_i$. Each micro element is equipped with the degrees of freedom as in (1.23) and basis (1.32). We define the local interpolation space for K as follows

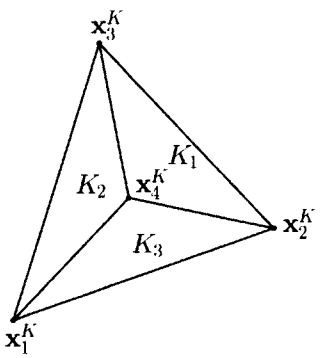

Figure 11: Hsieh-Clough-Toucher C^1-macro element

$$\mathcal{P}_K = \left\{ v \in C^1(K) \,\middle|\, v|_{K_i} \in \mathbb{P}_3, \text{for } i = 1,2,3 \right\}. \tag{1.33}$$

This patch of cubic Hermitian triangles has 15 degrees of freedom: three at each of the four nodes (function value and values of two derivatives) and one at each centroid of the three micro elements. On the other hand, for C^1 continuity, as we saw above, it is enough to additionally match normal derivatives at the midpoints of the internal edges, which gives three conditions. Therefore, $\dim \mathcal{P}_K = 12$.

Denote $\mathbf{m}_i$, the midpoint of the side of K opposite node $\mathbf{x}_i$, i.e.

$$\mathbf{m}_i = \frac{1}{2}\big(\mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3 - \mathbf{x}_i\big), \quad i = 1, 2, 3.$$

Also, let $\boldsymbol{\nu}_i$ denote the unit vector that is normal to the side of K opposite $\mathbf{x}_i$ and pointing inwards, i.e.

$$\boldsymbol{\nu}_i = \frac{\nabla \ell_i}{\|\nabla \ell_i\|}, \quad i = 1, 2, 3.$$

With this notation, we have the following local degrees of freedom

$$\begin{aligned}
&\gamma_1^K(f) = f(\mathbf{x}_1^K), && \gamma_2^K(f) = f(\mathbf{x}_2^K), && \gamma_3^K(f) = f(\mathbf{x}_3^K),\\
&\gamma_4^K(f) = \frac{\partial f(\mathbf{x}_1^K)}{\partial x}, && \gamma_6^K(f) = \frac{\partial f(\mathbf{x}_2^K)}{\partial x}, && \gamma_8^K(f) = \frac{\partial f(\mathbf{x}_3^K)}{\partial x},\\
&\gamma_5^K(f) = \frac{\partial f(\mathbf{x}_1^K)}{\partial y}, && \gamma_7^K(f) = \frac{\partial f(\mathbf{x}_2^K)}{\partial y}, && \gamma_9^K(f) = \frac{\partial f(\mathbf{x}_3^K)}{\partial y},\\
&\gamma_{10}^K(f) = \frac{\partial f(\mathbf{m}_1^K)}{\partial \boldsymbol{\nu}_1}, && \gamma_{11}^K(f) = \frac{\partial f(\mathbf{m}_2^K)}{\partial \boldsymbol{\nu}_2}, && \gamma_{12}^K(f) = \frac{\partial f(\mathbf{m}_3^K)}{\partial \boldsymbol{\nu}_3}.
\end{aligned} \tag{1.34}$$

The interpolation problem in $\mathcal{P}_K$ as defined in (1.33) with the above degrees of freedom (1.34) has a unique solution (Theorem 6.1.2 in [7]). The 12 local shape functions are illustrated in figure 12.

One way to compute the local shape functions is the following. We consider the interpolation space of piecewise cubic polynomials, without any continuity requirements across the boundaries of the micro elements

$$\mathcal{P}_{-K} = \Big\{ v \Big| \, v|_{K_i} \in \mathbb{P}_3, \text{for } i = 1, 2, 3 \Big\}. \tag{1.35}$$

The minus is standard notation in the finite element literature for interpolation with totally discontinuous functions. Clearly $\dim \mathcal{P}_{-K} = 30$, ten basis functions, (1.32), for each micro element in the patch. We extend the three local micro-bases into the

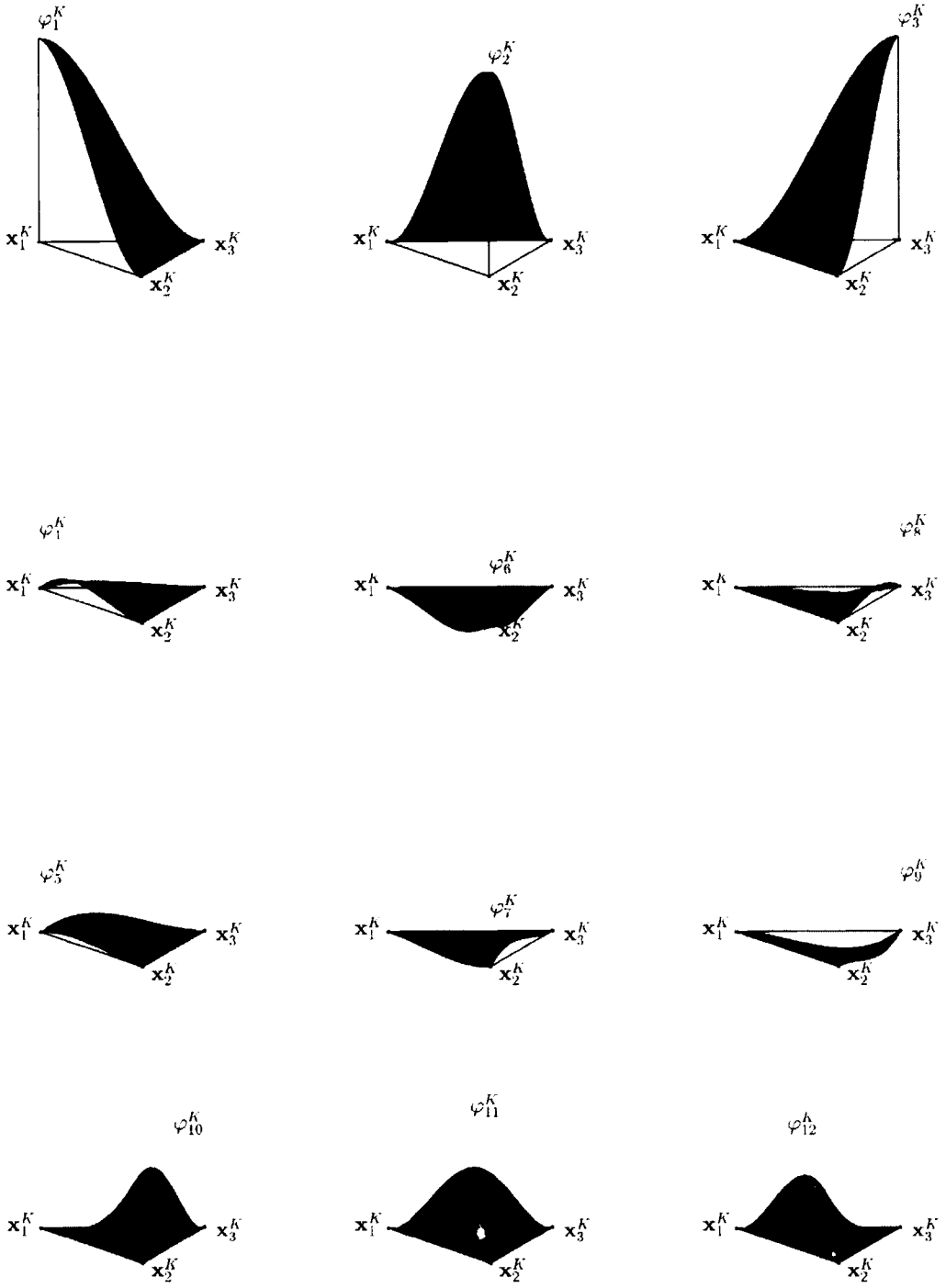

Figure 12: Local basis of the HCT cubic triangle

following "global for the patch" basis of $\mathcal{P}_{-K}$

$$
\begin{aligned}
b_i^K(\mathbf{x}) &= \begin{cases} b_i^{K_1}(\mathbf{x}), & \mathbf{x} \in K_1, \\ 0, & \text{otherwise}, \end{cases} \\
b_{10+i}^K(\mathbf{x}) &= \begin{cases} b_i^{K_2}(\mathbf{x}), & \mathbf{x} \in K_2, \\ 0, & \text{otherwise}, \end{cases} \\
b_{20+i}^K(\mathbf{x}) &= \begin{cases} b_i^{K_3}(\mathbf{x}), & \mathbf{x} \in K_3, \\ 0, & \text{otherwise}, \end{cases} \\
i &= 1, \ldots, 10.
\end{aligned} \tag{1.36}
$$

Since $\mathcal{P}_K \subset \mathcal{P}_{-K}$, we can write the shape functions as linear combinations of the functions in (1.36). To determine the coefficients, we construct a 30 by 30 linear system.

Each vertex of K is shared by two micro elements, so each of the first nine degrees of freedom in (1.34) give us two equations. In addition, the last three degrees of freedom give us one equation each. So we have 21 equations to satisfy the interpolation conditions. We use the remaining 9 equations to insure C^1 continuity in the interior of K. The barycentre $\mathbf{x}_4$ is shared among three micro elements. Requiring that the values of the three pieces match at $\mathbf{x}_4$ gives us two linearly independent equations. Similarly, matching the derivatives at $\mathbf{x}_4$ gives us another four equations. Finally, the last three equations match the values of the normal derivatives at the midpoints of the interior edges.

We solve this linear system once for each of the 12 shape functions. The matrix of the system is the same, while the right-hand-sides differ in the first 21 equations accordingly. The resulting coefficients can be stored in a 30-by-12 matrix, so that the linear systems are only solved once. The 12 local shape functions obtained this way

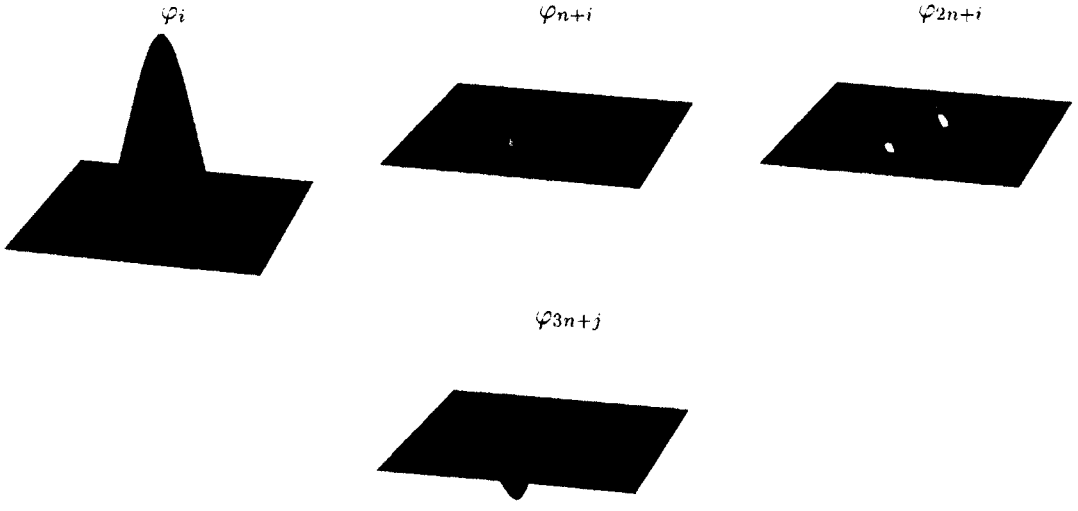

Figure 13: Global basis of cubic Hermitian triangle

form an interpolation basis, so we will denote them $\varphi_1^K, \ldots, \varphi_{12}^K$.

The global shape functions, assembled in the standard FEM fashion (1.13), are of class C^1 (Theorem 6.1.2 in [7]). Typical shape functions are shown in figure 13. We have $3n + e$ basis functions, where e denotes the number of edges in the finite element mesh. There are three shape functions associated with each node and one associated with each edge. The latter, denoted φ_{3n+j} in the figure, is supported on the two triangles containing edge j.

2 Semiparametric Nonhomogeneous Poisson Process Model

2.1 The likelihood function

In this section we add more details to the basic definitions and notation introduced in section 1.1 to precisely specify the model we are considering. We start by presenting a common description of both event locations data and panel data cases, and later

we consider the specifics of each.

We have a collection of M independent random processes $N_1(s), \ldots, N_M(s)$, with s belonging to a bounded domain $\Omega \subset \mathbb{R}^d$ and $N_i(A)$ taking a nonnegative integral value for each $A \subseteq \Omega$. The value of $N_i(A)$ represents the random number of events of subject i occurring in A. We also assume that we have a random sample, which consists of either:

1. event locations data: the total count for each process, $\{N_i = N_i(\Omega)\}_{i=1}^{M}$ together with $\sum_{i=1}^{M} N_i$ points, $\bigcup_{i=1}^{M} \{s_{ij}\}_{j=1}^{N_i}$, where $s_{ij} \in \Omega$ is the location where the j-th event of the i-th subject occurred; or

2. panel data: a partitioning of Ω into L mutually exclusive regions, $R_1, \ldots, R_L$, with $\bigcup R_j = \Omega$, and a table of counts, $\left(N_{ij}\right)_{M \times L}$, where $N_{ij} = N_i(R_j)$ is the number of events for the i-th subject in the j-th region. In this case, we also denote $N_i = \sum_{j=1}^{L} N_{ij} = N_i(\Omega)$, the total number of events for the i-th individual.

The individual intensities, as discussed in section 1.1.2, are assumed to take the form

$$\lambda_i(s) = \nu_i \lambda_o(s) \mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}}, \tag{2.1}$$

where $\lambda_o(s)$ is the baseline intensity, $\mathbf{x}_i$ is a vector of covariates for subject i, $\boldsymbol{\beta}$ is a vector of regression parameters, and ν_i is a positive random variable with mean $\mathrm{E}[\nu_i] = 1$, variance $\mathrm{Var}[\nu_i] = \tau$, and p.d.f. $p(\nu) = p(\nu; \tau)$. With these assumptions on ν_i, we readily find the mean and the variance of the total counts N_i, using the conditional expectation and conditional variance/covariance theorems.

$$\mathrm{E}[N_i] = \mathrm{E}[\mathrm{E}[N_i | \nu_i]] = \mathrm{E}[\nu_i \mu_i] = \mu_i \tag{2.2}$$

$$\mathrm{Var}[N_i] = \mathrm{E}[\mathrm{Var}[N_i | \nu_i]] + \mathrm{Var}[\mathrm{E}[N_i | \nu_i]] \tag{2.3}$$

$$
\begin{aligned}
&= \mathrm{E}\big[\nu_i\mu_i\big] + \mathrm{Var}\big[\nu_i\mu_i\big] = \mu_i + \tau\mu_i^2 \\
\mathrm{Cov}\big[N_i, N_k\big] &= \mathrm{E}\big[\mathrm{Cov}\big[N_i, N_k|\,\nu_i, \nu_k\big]\big] + \mathrm{Cov}\big[\mathrm{E}\big[N_i|\,\nu_i\big], \mathrm{E}\big[N_k|\,\nu_k\big]\big] \\
&= \mathrm{E}\big[0\big] + \mathrm{Cov}\big[\nu_i, \nu_k\big]\,\mu_i\mu_k = 0
\end{aligned}
\tag{2.4}
$$

Here, and in what follows, we denote $\mu_i = \mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}} \int_\Omega \lambda_o(s)ds$. Additionally, in the case of panel data we denote $\mu_{ij} = \mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}} \int_{R_j} \lambda_o(s)ds$. Derivations identical to (2.2), (2.3) and (2.4) give the moments of the panel counts as well.

$$\mathrm{E}\big[N_{ij}\big] = \mu_{ij} \tag{2.5}$$

$$\mathrm{Var}\big[N_{ij}\big] = \mu_{ij} + \tau\mu_{ij}^2 \tag{2.6}$$

$$\mathrm{Cov}\big[N_{ij}, N_{kl}\big] = \delta_{ik}\tau\mu_{ij}\mu_{kl} \tag{2.7}$$

2.1.1 Factorization of the likelihood functions

Assuming independence of the M counting processes $N_i(s)$ and the M frailties ν_i, the likelihood function, $\mathcal{L}$, for both event locations and panel data, can be written by conditioning on the total counts, N_i, as follows

$$\mathcal{L} = \prod_{i=1}^{M} \mathcal{L}_{1i}\mathcal{L}_{2i}, \tag{2.8}$$

where $\mathcal{L}_{1i}$ is the conditional likelihood of the event locations or panel counts of the i-th individual given the total number of events N_i, and $\mathcal{L}_{2i}$ is the likelihood of the total number of events of the i-th individual.

The probability of N_i can be written also by conditioning, this time on the value of ν_i. We have

$$\mathcal{L}_{2i} = \mathrm{P}\big[N_i\big] = \int_0^\infty \mathrm{P}\big[N_i|\,\nu_i\big]\,p(\nu_i)d\nu_i = \frac{1}{N_i!}\int_0^\infty \big(\nu_i\mu_i\big)^{N_i}\mathrm{e}^{-\nu_i\mu_i}p(\nu_i)d\nu_i. \tag{2.9}$$

Under the assumption that $\nu_i \sim \Gamma\left(\frac{1}{\tau}, \tau\right)$, with p.d.f. (1.4), it is well known that the gamma mixture of a Poisson random variable has a negative binomial distribution, [6], with p.d.f.

$$\mathrm{P}[N_i] = \frac{\Gamma\left(N_i + \frac{1}{\tau}\right)}{N_i!\Gamma\left(\frac{1}{\tau}\right)} \left(\frac{1}{1+\tau\mu_i}\right)^{\frac{1}{\tau}} \left(\frac{\tau\mu_i}{1+\tau\mu_i}\right)^{N_i}. \tag{2.10}$$

2.1.2 Conditional likelihood in the event locations case

The conditional distribution of the event locations given N_i is the same as that of the order statistic of N_i independent and identically distributed random variables with p.d.f. proportional to the individual intensity function (see [24, section 2.3.3])

$$f(S) = \frac{\lambda_i(S)}{\int_\Omega \lambda_i(s)ds} = \frac{\nu_i e^{\mathbf{x}_i^T\boldsymbol{\beta}}\lambda_o(S)}{\nu_i e^{\mathbf{x}_i^T\boldsymbol{\beta}}\int_\Omega \lambda_o(s)ds} = \frac{\lambda_o(S)}{\Lambda_o(\Omega)}, \quad \text{for } S \in \Omega. \tag{2.11}$$

Here we have used the notation Λ_o for the cumulative baseline intensity functions, which we define as

$$\Lambda_o(A) = \int_A \lambda_o(s)ds, \quad \forall A \subseteq \Omega. \tag{2.12}$$

If we consider a collection, $S_1, \ldots, S_{N_i}$, of N_i unordered and independent random variables with density (2.11), their joint density would be $\prod_{j=1}^{N_i} f(S_j)$. However, any arbitrary relabelling would produce the same sample of event locations. Therefore, the joint distribution of the event locations $\{s_{ij}\}$ for process i, conditional on the total count, N_i, is

$$\mathcal{L}_{1i} = N_i! \prod_{j=1}^{N_i} \left(\frac{\lambda_o(s_{ij})}{\Lambda_o(\Omega)}\right) = \frac{N_i!}{\Lambda_o(\Omega)^{N_i}} \prod_{j=1}^{N_i} \lambda_o(s_{ij}). \tag{2.13}$$

2.1.3 Conditional likelihood in the panel data case

In this case, the regional counts, N_{ij}, conditional on the total count, N_i, have a multinomial distribution with L cells and N_i trials. To find the cell probabilities, we recognize that they are proportional to the individual cumulative intensity over the regions. In particular, for the j-th region, we have the cell probability

$$p_j = \frac{\int_{R_j}\lambda_i(s)ds}{\int_\Omega \lambda_i(s)ds} = \frac{\nu_i e^{\mathbf{x}_i^T\boldsymbol{\beta}}\int_{R_j}\lambda_o(s)ds}{\nu_i e^{\mathbf{x}_i^T\boldsymbol{\beta}}\int_\Omega \lambda_o(s)ds} = \frac{\Lambda_o(R_j)}{\Lambda_o(\Omega)}. \tag{2.14}$$

Finally, the conditional likelihood is

$$\mathcal{L}_{1i} = \frac{N_i!}{N_{i1}!\cdots N_{iL}!}\prod_{j=1}^{L}\left(\frac{\Lambda_o(R_j)}{\Lambda_o(\Omega)}\right)^{N_{ij}} = \frac{N_i!}{\Lambda_o(\Omega)^{N_i}}\prod_{j=1}^{L}\frac{\Lambda_o(R_j)^{N_{ij}}}{N_{ij}!}. \tag{2.15}$$

2.1.4 The unconditional likelihood

In the case of event locations, the unconditional likelihood is obtained by inserting formulas (2.13) and (2.10) into (2.8) and simplifying.

$$\begin{aligned}\mathcal{L} &= \prod_{i=1}^{M}\left[\frac{N_i!}{\Lambda_o(\Omega)^{N_i}}\prod_{j=1}^{N_i}\lambda_o(s_{ij})\right]\frac{\Gamma\left(N_i+\frac{1}{\tau}\right)}{N_i!\Gamma\left(\frac{1}{\tau}\right)}\left[\tau\mu_i\right]^{N_i}\left[1+\tau\mu_i\right]^{-N_i-\frac{1}{\tau}} \\ &= \prod_{i=1}^{M}e^{N_i\mathbf{x}_i^T\boldsymbol{\beta}}\left[1+\tau e^{\mathbf{x}_i^T\boldsymbol{\beta}}\Lambda_o(\Omega)\right]^{-N_i-\frac{1}{\tau}}\left[\prod_{j=1}^{N_i}\left(1+\tau(N_i-j)\right)\right]\left[\prod_{j=1}^{N_i}\lambda_o(s_{ij})\right]\end{aligned} \tag{2.16}$$

In the panel data case, we insert (2.15) and (2.10) into (2.8) and derive the following likelihood.

$$\begin{aligned}\mathcal{L} &= \prod_{i=1}^{M}\left[\frac{N_i!}{\Lambda_o(\Omega)^{N_i}}\prod_{j=1}^{L}\frac{\Lambda_o(R_j)^{N_{ij}}}{N_{ij}!}\right]\frac{\Gamma\left(N_i+\frac{1}{\tau}\right)}{N_i!\Gamma\left(\frac{1}{\tau}\right)}\left[\tau\mu_i\right]^{N_i}\left[1+\tau\mu_i\right]^{-N_i-\frac{1}{\tau}} \\ &= \prod_{i=1}^{M}e^{N_i\mathbf{x}_i^T\boldsymbol{\beta}}\left[1+\tau e^{\mathbf{x}_i^T\boldsymbol{\beta}}\Lambda_o(\Omega)\right]^{-N_i-\frac{1}{\tau}}\left[\prod_{j=1}^{N_i}\left(1+\tau(N_i-j)\right)\right]\left[\prod_{j=1}^{L}\frac{\Lambda_o(R_j)^{N_{ij}}}{N_{ij}!}\right]\end{aligned} \tag{2.17}$$

Expressions (2.16) and (2.17) differ only in the last product of each. The common part considered as a function of the sample depends only on the total individual counts $\{N_i\}_{i=1}^M$. At the same time, the different parts considered as functions of the parameters do not depend on τ or $\boldsymbol{\beta}$ and so $\{N_i\}_{i=1}^M$ is sufficient for the parameters $\left[\tau, \boldsymbol{\beta}, \Lambda_o(\Omega)\right]^T$.

2.2 The penalized maximum likelihood problem

To estimate the model we apply the penalized maximum likelihood method. Here we formally pose this problem and define the penalty term. We also consider the discrete formulation using the finite elements.

2.2.1 Continuous formulation

Since the baseline intensity must be positive, we set

$$\lambda_o(s) = \mathrm{e}^{\alpha(s)} \tag{2.18}$$

and consider the likelihood as a functional of $\alpha(s)$. We formulate the penalized maximum likelihood estimator as the solution of the following maximization problem. *For a given value of the penalty parameter $\delta > 0$, we seek a real number $\hat{\tau} > 0$, a p-dimensional vector $\hat{\boldsymbol{\beta}}$, and a function $\hat{\alpha} \in H^2(\Omega)$ at which the functional*

$$J(\tau, \boldsymbol{\beta}, \alpha) = \ln\Big(\mathcal{L}(\tau, \boldsymbol{\beta}, \alpha)\Big) - \delta\mathcal{P}(\alpha) \tag{2.19}$$

attains its maximum.

2.2.2 The penalty term

To formalize the definition of the penalty term, we denote ∂_i the operator of differentiation with respect to the i-th component of s. Also, we define $\boldsymbol{\partial}^m$ as the operator giving the column vector of all partial derivatives of total order m. Furthermore, we denote $\|.\|$ the Euclidean norm. The cases of interest for us are the following:

- $d = 1, m = 2,$ for all $s \in \Omega$

$$\left\|\boldsymbol{\partial}^2 \alpha(s)\right\|^2 = \left|\alpha''(s)\right|^2,$$

- $d = 2, m = 2,$ for all $s \in \Omega$

$$\left\|\boldsymbol{\partial}^2 \alpha(s)\right\|^2 = \left|\partial_1^2 \alpha(s)\right|^2 + \left|\partial_1 \partial_2 \alpha(s)\right|^2 + \left|\partial_2^2 \alpha(s)\right|^2, \qquad \text{etc.}$$

With this notation, the penalty term is defined as

$$\mathcal{P}(\alpha) = \int_\Omega \left\|\boldsymbol{\partial}^2 \alpha(s)\right\|^2 ds. \tag{2.20}$$

2.2.3 Discrete formulation

Discretization of α

At this point we put to work the Finite Element machinery described in section 1.2. We introduce a finite element mesh with parameter h on the domain Ω and consider the C^1 FE interpolation space $\mathbb{X}_h$, described in subsection 1.2.2. We denote $n = \dim \mathbb{X}_h$.

Any $\alpha \in \mathbb{X}_h$ has the representation

$$\alpha(s) = \sum_{k=1}^{n} \alpha_k \varphi_k(s) = \boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}, \quad \text{for all } s \in \Omega, \tag{2.21}$$

where $\mathbf{\Phi}(s) = [\varphi_1(s), \ldots, \varphi_n(s)]^T$ is the vector function of all FE global basis functions which span $\mathbb{X}_h$. Hereafter it is important to distinguish the following three versions of alpha: the plain symbol α denotes the unknown function in $\mathbb{X}_h$; the indexed symbol α_k denotes the k-th coefficient in the representation of α in the basis $\mathbf{\Phi}(s)$ of $\mathbb{X}_h$; the boldface $\boldsymbol{\alpha}$ denotes the vector of coefficients $[\alpha_1, \ldots, \alpha_n]^T$.

Discretization of the penalty term

Since the finite element interpolation space we consider is H^2 conforming, i.e. $\mathbb{X}_h \subset H^2(\Omega)$, the penalty term is well defined for all $\alpha \in \mathbb{X}_h$. This allows us to apply formula (2.20) without the need for any further approximation. We have

$$\mathcal{P}(\alpha) = \mathcal{P}(\mathbf{\Phi}^T(s)\boldsymbol{\alpha}) = \int_\Omega \left\| D^2\left(\mathbf{\Phi}^T(s)\boldsymbol{\alpha}\right)\right\|^2 ds = \boldsymbol{\alpha}^T\mathbf{P}\boldsymbol{\alpha}, \tag{2.22}$$

where $\mathbf{P}$ is the *penalty matrix* in $\mathbb{X}_h$ with respect to the interpolation basis $\mathbf{\Phi}(s)$. The individual entries of $\mathbf{P}$ are

$$\mathbf{P}_{ij} = \int_\Omega \left(\boldsymbol{\partial}^2\varphi_i(s)\right)^T\left(\boldsymbol{\partial}^2\varphi_j(s)\right)ds$$

for $i, j \in \{1, \ldots, n\}$.

The discrete formulation of the penalized maximum likelihood estimation is the following. *For given $\delta > 0$, we seek $\hat{\tau} > 0$, $\hat{\boldsymbol{\beta}} \in \mathbb{R}^p$, and $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^n$ that maximize*

$$J(\tau, \boldsymbol{\beta}, \boldsymbol{\alpha}) = \ln\left(\mathcal{L}(\tau, \boldsymbol{\beta}, \mathbf{\Phi}(s)^T\boldsymbol{\alpha}\right) - \delta\boldsymbol{\alpha}^T\mathbf{P}_h\boldsymbol{\alpha}. \tag{2.23}$$

It is worthwhile here to write explicitly the discrete objective functions for both

types of data. For event locations data we have

$$J = \sum_{i=1}^{M} \Bigg\{ \sum_{j=1}^{N_i} \ln\Big(1 + \tau(N_i - j)\Big) - \Big(N_i + \frac{1}{\tau}\Big) \ln\Big(1 + \tau e^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_{\Omega} e^{\boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}} ds\Big) + N_i \mathbf{x}_i^T \boldsymbol{\beta} + \sum_{j=1}^{N_i} \boldsymbol{\Phi}^T(s_{ij})\boldsymbol{\alpha} \Bigg\} - \delta \boldsymbol{\alpha}^T \mathbf{P}_h \boldsymbol{\alpha}. \quad (2.24)$$

Similarly, for panel counts data we have

$$J = \sum_{i=1}^{M} \Bigg\{ \sum_{j=1}^{N_i} \ln\Big(1 + \tau(N_i - j)\Big) - \Big(N_i + \frac{1}{\tau}\Big) \ln\Big(1 + \tau e^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_{\Omega} e^{\boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}} ds\Big) + N_i \mathbf{x}_i^T \boldsymbol{\beta} + \sum_{j=1}^{L} N_{ij} \ln\Big(\int_{R_j} e^{\boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}} ds\Big) - \sum_{j=1}^{L} \ln(N_{ij}!) \Bigg\} - \delta \boldsymbol{\alpha}^T \mathbf{P}_h \boldsymbol{\alpha}. \quad (2.25)$$

2.3 Estimation

2.3.1 First derivatives of the objective function

We denote ∂_τ the operator of partial differentiation with respect to τ, while the operators $\nabla_{\boldsymbol{\beta}}$ and $\nabla_{\boldsymbol{\alpha}}$ will denote the column vectors of partial derivatives with respect to the components of $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ respectively. The details of the derivations of the following formulas are shown in Appendix A.

The derivatives of J with respect to τ and $\boldsymbol{\beta}$ are the same in both types of data and are given by the following expressions.

$$\partial_\tau J = \sum_{i=1}^{M} \Bigg[\sum_{j=1}^{N_i} \frac{N_i - j}{1 + \tau(N_i - j)} + \frac{1}{\tau^2} \ln\big(1 + \tau\mu_i\big) - \frac{1}{\tau} \frac{\big(1 + \tau N_i\big)\mu_i}{1 + \tau\mu_i} \Bigg] \quad (2.26)$$

$$\nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M} \frac{N_i - \mu_i}{1 + \tau\mu_i} \mathbf{x}_i \quad (2.27)$$

Note that under the model $\mathrm{E}[N_i] = \mu_i$, $\mathrm{Var}[N_i] = \mu_i + \tau\mu_i^2$ and $\nabla_{\boldsymbol{\beta}}\mu_i = \mu_i \mathbf{x}_i$ (see

(A.1)). Consequently, (2.27) can also be interpreted as

$$\nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M} \frac{N_i - \mathrm{E}\big[N_i\big]}{\mathrm{Var}\big[N_i\big]} \nabla_{\boldsymbol{\beta}} \mu_i$$

The gradient of J with respect to $\boldsymbol{\alpha}$ in the case of event locations is given by

$$\nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M} \frac{N_i - \mu_i}{\mu_i + \tau \mu_i^2} \nabla_{\boldsymbol{\alpha}} \mu_i + \sum_{i=1}^{M} \sum_{j=1}^{N_i} \Big[\boldsymbol{\Phi}(s_{ij}) - \frac{\nabla_{\boldsymbol{\alpha}} \mu_i}{\mu_i} \Big] - 2\delta \mathbf{P} \boldsymbol{\alpha}, \tag{2.28}$$

and in the panel data case is given by

$$\nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M} \frac{N_i - \mu_i}{\mu_i + \tau \mu_i^2} \nabla_{\boldsymbol{\alpha}} \mu_i + \sum_{i=1}^{M} \sum_{j=1}^{L} \left[\frac{N_{ij}}{\mu_{ij}} - \frac{N_i}{\mu_i} \right] \nabla_{\boldsymbol{\alpha}} \mu_{ij} - 2\delta \mathbf{P} \boldsymbol{\alpha} \tag{2.29}$$

The second term in (2.28) can be interpreted in terms of the random variable, S with p.d.f. (2.11). As mentioned in section 2.1.2, the unordered set of event locations conditional on their total number, N_i, is distributed as N_i independent random variables with the same distribution as S. Using

$$\nabla_{\boldsymbol{\alpha}} \mu_i = \nabla_{\boldsymbol{\alpha}} \left(\mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_{\Omega} \mathrm{e}^{\boldsymbol{\Phi}^T(s) \boldsymbol{\alpha}} ds \right) = \mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_{\Omega} \mathrm{e}^{\boldsymbol{\Phi}^T(s) \boldsymbol{\alpha}} \boldsymbol{\Phi}(s) ds = \mathrm{e}^{\mathbf{x}_i^T \boldsymbol{\beta}} \int_{\Omega} \lambda_o(s) \boldsymbol{\Phi}(s) ds$$

we have

$$\mathrm{E}\big[\boldsymbol{\Phi}(s_{ij}) \big| \, N_i \big] = \mathrm{E}\big[\boldsymbol{\Phi}(S) \big] = \int_{\Omega} \boldsymbol{\Phi}(s) \frac{\lambda_o(s)}{\Lambda_o(\Omega)} ds = \frac{\nabla_{\boldsymbol{\alpha}} \mu_i}{\mu_i},$$

which allows us to see the second term in (2.28) as

$$\sum_{i=1}^{M} \sum_{j=1}^{N_i} \Big[\boldsymbol{\Phi}(s_{ij}) - \mathrm{E}\big[\boldsymbol{\Phi}(s_{ij}) \big| \, N_i \big] \Big].$$

Similarly, we can interpret the second term in (2.29) in terms of the conditional expectation of N_{ij} given N_i. Based on the discussion in section 2.1.3, for given j the

random variable $\left(N_{ij}|\, N_i\right)$ has binomial distribution with N_i trials and probability of success given in equation (2.14). Thus,

$$\mathrm{E}\big[N_{ij}|\, N_i\big] = N_i \frac{\Lambda_o(R_j)}{\Lambda_o(\Omega)} = \frac{N_i \mu_{ij}}{\mu_i},$$

which allows the second term in (2.29) to be expressed as

$$\sum_{i=1}^{M}\sum_{j=1}^{L}\left[\Big(N_{ij} - \mathrm{E}\big[N_{ij}|\, N_i\big]\Big)\frac{\nabla_{\boldsymbol{\alpha}}\mu_{ij}}{\mu_{ij}}\right].$$

2.3.2 Second derivatives of the objective function

The second derivatives with respect to τ and $\boldsymbol{\beta}$, as well as all mixed second derivatives are the same for the cases of event locations and panel data.

$$\partial_\tau^2 J = \sum_{i=1}^{M}\left[-\sum_{j=1}^{N_i}\frac{\left(N_i - j\right)^2}{\left[1+\tau\left(N_i - j\right)\right]^2} - \frac{2}{\tau^3}\ln\left(1+\tau\mu_i\right) + \frac{2\mu_i}{\tau^2\left(1+\tau\mu_i\right)} + \frac{\mu_i^2\left(1+\tau N_i\right)}{\tau\left(1+\tau\mu_i\right)^2}\right] \tag{2.30}$$

$$\nabla_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\beta}}{}^T J = \sum_{i=1}^{M}\left[-\frac{\mu_i}{1+\tau\mu_i} - \frac{\left(N_i - \mu_i\right)\tau\mu_i}{\left(1+\tau\mu_i\right)^2}\right]\mathbf{x}_i\mathbf{x}_i^T \tag{2.31}$$

$$\partial_\tau \nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M}\left[-\frac{\left(N_i - \mu_i\right)}{\left(1+\tau\mu_i\right)^2}\mu_i\mathbf{x}_i\right] \tag{2.32}$$

$$\partial_\tau \nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M}\left[-\frac{\left(N_i - \mu_i\right)}{\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\mu_i\right] \tag{2.33}$$

$$\nabla_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\alpha}}{}^T J = \sum_{i=1}^{M}\left[-\frac{1+\tau N_i}{\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\alpha}}{}^T\mu_i\right] \tag{2.34}$$

The second derivatives with respect to $\boldsymbol{\alpha}$ differ. In the case of event locations we have

$$\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}J = \sum_{i=1}^{M}\left[\frac{(1+\tau N_i)\tau}{(1+\tau\mu_i)^2}(\nabla_{\boldsymbol{\alpha}}\mu_i)(\nabla_{\boldsymbol{\alpha}}\mu_i)^{T} - \frac{1+\tau N_i}{1+\tau\mu_i}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_i\right] - 2\delta\mathbf{P} \tag{2.35}$$

and in the case of panel data we have

$$\begin{aligned}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}J = &\sum_{i=1}^{M}\left[\frac{(1+\tau N_i)\tau}{(1+\tau\mu_i)^2}(\nabla_{\boldsymbol{\alpha}}\mu_i)(\nabla_{\boldsymbol{\alpha}}\mu_i)^{T} - \frac{1+\tau N_i}{1+\tau\mu_i}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_i\right] \\ &+ \sum_{i=1}^{M}\sum_{j=1}^{L}\left[\frac{N_{ij}}{\mu_{ij}}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_{ij} - \frac{N_{ij}}{\mu_{ij}^2}(\nabla_{\boldsymbol{\alpha}}\mu_{ij})(\nabla_{\boldsymbol{\alpha}}\mu_{ij})^{T}\right] - 2\delta\mathbf{P}.\end{aligned} \tag{2.36}$$

2.3.3 Newton-Raphson method

The Newton-Raphson method is an iterative method for finding approximate solutions of systems of nonlinear simultaneous equations. It is also applied to optimization problems, by setting the gradient of the objective function equal to zero and solving this nonlinear system, [15, 16, 17].

We denote $\mathbf{G}$ the gradient vector and $\mathbf{H}$ the Hessian matrix of J with respect to the $1+p+n$ parameters, $z = \left[\tau, \boldsymbol{\beta}^T, \boldsymbol{\alpha}^T\right]^T$.

Starting with an appropriate initial approximation, z_0, the Newton-Raphson method iteratively applies the formula

$$z_{k+1} = z_k - \left[\mathbf{H}(z_k)\right]^{-1}\mathbf{G}(z_k) \tag{2.37}$$

until a stopping criterion is reached. The stopping criterion usually requires either that two successive approximations, z_k and z_{k+1}, are sufficiently close (e.g. in the Euclidean norm), or that the value of $\mathbf{G}(z_{k+1})$ is sufficiently close to the zero vector

in $\mathbb{R}^{1+p+n}$, or a combination of the two. In optimization problems, a stopping criterion may also consider the changes in the successive values of the objective function, J, and stop the algorithm when these changes become insignificantly small.

The Newton-Raphson is one of the fastest iterative methods for solving nonlinear equations, with quadratic rate of convergence, i.e. if z^* is the exact solution, then $\|z^* - z_{k+1}\| \leq C \|z^* - z_k\|^2$ for some constant C. However, the method is not considered robust, because it often fails to converge or converges at a slower rate, comparable to simpler methods. If the first derivative (second derivative, if the method is being used for optimization) does not exist or is discontinuous at the solution point, then the method usually fails by diverging to infinity. If the derivative is zero at the solution point (i.e. the root is not simple), then the method usually converges, but at a slower than quadratic rate. The most common source of problems with the Newton-Raphson method is the initial guess, z_0. In order for the method to converge, the initial guess must be sufficiently close to the true solution. In a general situation, this is usually achieved by using a slower, but more robust method for the initial few iterations and switching to the faster Newton-Raphson when the estimate is close. In our case, we have a way of computing an adequate starting approximation for the parameters.

2.3.4 Initial approximations

The initial approximations for $\tau, \boldsymbol{\beta}$, and $\boldsymbol{\alpha}$ are of great importance for the successful application of the Newton-Raphson method. This is why it is crucial that we have a reliable procedure for obtaining initial approximations from the available data.

It is clear that the collection of total individual counts, $\{N_i\}$, is a sufficient statistic for $\boldsymbol{\beta}$ and τ, therefore we are able to solve for these two parameters without considering the separate event locations $\{s_{ij}\}$ or panel counts $\{N_{ij}\}$. An initial

guess for $\boldsymbol{\beta}$ can be derived from a simplified version of the model, the approximation $N_i \sim \text{Poisson}(\mu_i)$.

It should be noted here, that the covariates, $\mathbf{x}_i$, should not contain an intercept term. This requirement does not restrict the generality of the model, because the intercept is modelled in λ_o. In fact if we allowed an intercept term in both, the covariates and the baseline intensity, we would loose uniqueness of the solution, i.e. the model would be unidentifiable.

Clearly, the μ_i's depend on both $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$, while here we are only solving for initial $\boldsymbol{\beta}$. However, the dependence of μ_i on $\boldsymbol{\alpha}$ can be interpreted as an intercept term.

$$\mu_i = \mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}} \int_\Omega \mathrm{e}^{\boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}} ds = \mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}} \Lambda_o(\Omega) = \mathrm{e}^{b_0 + \mathbf{x}_i^T\boldsymbol{\beta}} \tag{2.38}$$

The intercept term is $b_0 = \ln\big(\Lambda_o(\Omega)\big)$. After solving a standard Poisson regression for $N_i \sim \text{Poisson}\big(\mathrm{e}^{\beta_0 + \mathbf{x}_i^T\boldsymbol{\beta}}\big)$, we discard β_0 and keep the vector $\boldsymbol{\beta}$ as our initial estimate, $\boldsymbol{\beta}_0$.

Once we obtain the initial $\boldsymbol{\beta}$, we compute initial guess for τ by matching the sample moments of $\left\{\mathrm{e}^{-\mathbf{x}_i^T\boldsymbol{\beta}} N_i\right\}$ to the model based moments from (2.2) and (2.3). If we denote

$$m = \frac{1}{M}\sum_{i=1]}^{M} \mathrm{e}^{-\mathbf{x}_i^T\boldsymbol{\beta}} N_i \quad \text{and} \quad s^2 = \frac{1}{M-1}\sum_{i=1}^{M}\left(\mathrm{e}^{-\mathbf{x}_i^T\boldsymbol{\beta}} N_i - m\right)^2$$

we solve the approximation $s^2 = m + \tau m^2$ for the initial τ:

$$\tau_0 = \frac{s^2 - m}{m^2}.$$

This initial guess works well with simulated data, however it is not as reliable when fitting real world data. We discuss an alternative approach later in this work.

We produce an initial approximation for $\lambda_o(s)$ using the initial values of $\boldsymbol{\beta}_0$ by projecting the *empirical baseline intensity* onto the finite element interpolation space. In the case of event locations, the empirical baseline intensity is

$$\rho(s) = \frac{1}{M} \sum_{i=1}^{M} \mathrm{e}^{-\mathbf{x}_i^T \boldsymbol{\beta}} \sum_{j=1}^{N_i} \delta\big(s - s_{ij}\big), \tag{2.39}$$

where $\delta(s)$ is the Dirac delta distribution. Similarly, in the case of panel counts, we have the empirical baseline intensity

$$\rho(s) = \frac{1}{M} \sum_{i=1}^{M} \mathrm{e}^{-\mathbf{x}_i^T \boldsymbol{\beta}} \sum_{j=1}^{L} \frac{1}{|R_j|} N_{ij} 1_{R_j}(s), \tag{2.40}$$

where $1_A(s)$ denotes the indicator function of set $A \subset \Omega$.

The standard finite element projection is with respect to the inner product in $L^2(\Omega)$, i.e. $(u, v) = \int_\Omega u(s)v(s)ds$. However, if we proceed this way, there is no guarantee that the resulting $\lambda_o(s)$ will be positive everywhere in Ω. To avoid this problem, we may solve the following projection problem: *find* $\alpha \in \mathbb{X}_h$ *such that for all* $v \in \mathbb{X}_h$

$$\int_\Omega \big(\mathrm{e}^{\alpha(s)} - \rho(s)\big) v(s) ds = 0. \tag{2.41}$$

This is a nonlinear problem, and the effort required for solving it is not justified to produce an initial guess. In practise, we solve the following similar problem that is linear in α: *find* $\alpha \in \mathbb{X}_h$, *such that for all* $v \in \mathbb{X}_h$

$$\int_\Omega \Big(\alpha(s) - \ln\big(\rho(s)\big)\Big) v(s) ds = 0. \tag{2.42}$$

In the case of panel data, we have

$$\ln\big(\rho(s)\big) = \sum_{j=1}^{L} \ln \underbrace{\left(\frac{1}{M\,|R_j|} \sum_{i=1}^{M} \mathrm{e}^{-\mathbf{x}_i^T \boldsymbol{\beta}} N_{ij} \right)}_{\text{denote } \rho_j} 1_{R_j}(s) = \sum_{j=1}^{L} \ln(\rho_j) 1_{R_j}(s)$$

Upon substituting the basis expansion (2.21), the above expression for $\ln(\rho(s))$ and each basis function for $v(s)$ into (2.42) we obtain the $n \times n$ linear system

$$\mathbf{M}\boldsymbol{\alpha} = \sum_{j=1}^{L} \ln(\rho_j) \int_{R_j} \boldsymbol{\Phi}(s) ds,$$

where the matrix $\mathbf{M}$ is called the *mass matrix*

$$\mathbf{M} = \int_{\Omega} \boldsymbol{\Phi}(s) \boldsymbol{\Phi}^T(s) ds, \quad \text{with entries} \quad \mathbf{M}_{ij} = \int_{\Omega} \varphi_i(s) \varphi_j(s) ds. \tag{2.43}$$

In the case of event locations, the empirical intensity functions is a linear combination of Dirac delta functions, and taking the logarithm is not well defined. In this case, we first solve for $\lambda_o(s) \in \mathbb{X}_h, \lambda_o(s) = \boldsymbol{\Phi}^T(s)\boldsymbol{\lambda}$, from the linear system

$$\mathbf{M}\boldsymbol{\lambda} = \sum_{i=1}^{M} \sum_{j=1}^{N_i} \frac{\mathrm{e}^{-\mathbf{x}_i^T \boldsymbol{\beta}}}{M} \boldsymbol{\Phi}(s_{ij}),$$

and then compute the coefficients of the initial α using the relations $\alpha(s) = \ln(\lambda(s))$ for the coefficients corresponding to function values and $\partial\alpha(s) = \partial\lambda(s)/\lambda(s)$ for coefficients corresponding to values of derivatives. Any coefficients in the initial $\boldsymbol{\lambda}$ that correspond to function values that are not positive, we truncate to an arbitrary small positive number. In the solutions presented herein we have used 10^{-6}.

Our numerical experience shows that the approximations of (2.41) that we just described provide adequate initial guess for the baseline intensity. This fact, together

with its linearity, make it our method of choice.

2.3.5 Solution algorithm

Here we discuss the solution algorithm for τ, $\boldsymbol{\beta}$, and $\boldsymbol{\alpha}$ when the penalty parameter δ is given.

We have already said that our method of choice for minimizing J is Newton-Raphson and that a good initial guess is crucial for the success of this method. As it turns out in our numerical solutions, the initial approximations for $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ computed as in section 2.3.4 above are good, but the initial τ is not. In addition, it is not enough to guarantee that the final solution for τ is positive (this is required because only positive values of τ make sense in our setting), but it must be kept positive at every step of the iterative solution, because the expressions for J and $\partial_\tau J$ would be undefined. So, instead of solving a constrained optimization problem for $\tau > 0$, we use the substitution

$$\tau = \mathrm{e}^{\varkappa} \tag{2.44}$$

in J and solve an unconstrained optimization problem for $\varkappa \in \mathbb{R}$. Under this substitution the derivatives of an arbitrary function $f(\tau)$ with respect to the new parameter are

$$\begin{aligned}
\partial_\varkappa f &= \partial_\tau f \partial_\varkappa \tau = \mathrm{e}^{\varkappa} \partial_\tau f \\
\partial_\varkappa^2 f &= \partial_\tau^2 f \left(\partial_\varkappa \tau\right)^2 + \partial_\tau f \partial_\varkappa^2 \tau = \mathrm{e}^{2\varkappa} \partial_\tau^2 f + \mathrm{e}^{\varkappa} \partial_\tau f.
\end{aligned}$$

The first formula is used with for J, $\nabla_{\boldsymbol{\beta}} J$ and $\nabla_{\boldsymbol{\alpha}} J$ to compute the derivatives $\partial_\varkappa J$, $\partial_\varkappa \nabla_{\boldsymbol{\beta}} J$, and $\partial_\varkappa \nabla_{\boldsymbol{\alpha}} J$; the second formula is used to compute $\partial_\varkappa^2 J$.

Solving the full system for $\varkappa$, $\boldsymbol{\beta}$, and $\boldsymbol{\alpha}$ simultaneously is not numerically robust and often leads to divergence of the $\varkappa$ parameter. This is not surprising, because τ

is a variance parameter and the likelihood is more sensitive to its changes than to the parameters controlling the mean of the distribution. After obtaining the initial guesses for $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$, we solve for initial $\varkappa$ using a simple one dimensional grid search algorithm. At each step of this grid search, we maximize J with respect to $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ holding $\varkappa$ fixed. We choose as initial $\varkappa$ the point in the grid which gave the greatest value of J. We may also use the $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ computed at that point as initial guesses as well. Note also that the N_i's are sufficient statistics for τ and $\boldsymbol{\beta}$. Therefore, here we may also hold the $\boldsymbol{\alpha}$ parameter fixed at its given initial value and maximize only with respect to $\boldsymbol{\beta}$ at each value of $\varkappa$ in the grid.

After the grid search, we run the Newton-Raphson algorithm on the full system, maximizing J with respect to $\varkappa$, $\boldsymbol{\beta}$, and $\boldsymbol{\alpha}$. If it does not converge, the grid search is repeated on a finer grid zooming in around the best $\varkappa$ found so far and this process is repeated until convergence is obtained.

2.4 Selecting the best penalty parameter

So far we have assumed that the penalty parameter δ is given. Now we address the question of selecting the best value of δ. To simplify our exposition in this section, we will assume that the optimal values of τ and $\boldsymbol{\beta}$ are known and consider the case where the baseline intensity is the only unknown.

The penalty parameter is a *model selection* parameter and controls the trade-off between minimizing the bias versus minimizing the variance. As $\delta \to \infty$, it gives more weight to the penalty term in J, shrinking the amount by which the solution may deviate from the null space of the penalty operator. This leads to a model selected from a narrower class, resulting in smaller variance at the expense of increased bias. Things are reversed as $\delta \to 0^+$, in which case the importance of the penalty term diminishes allowing the model to effectively "interpolate the data" (in some sense)

giving smaller bias at the cost of larger variance. The goal of selecting the best penalty parameter is to find a spot in the middle of these two extremes that is best in some appropriate sense. There are two popular methods of searching for δ, namely the *generalized cross validation*, or GCV, and the *restricted maximum likelihood*, or REML, [13].

2.4.1 Generalized cross validation

The general idea of cross validation is to find the value of δ that minimizes the *mean squared error*, or MSE, of the model. Suppose that $\lambda_o(s)$ is the true baseline intensity and that $\hat{\lambda}_o(s;\delta)$ is the estimator of λ_o for given δ. Then, the MSE of the model is defined as

$$\mathrm{MSE}(\hat{\lambda}_o(s;\delta)) = \mathrm{E}\Big[\big(\lambda_o(s) - \hat{\lambda}_o(s;\delta)\big)^2\Big]. \tag{2.45}$$

The MSE defined this way is *pointwise*, i.e. it is a function of s. The *integrated MSE* is

$$\mathrm{IMSE}(\hat{\lambda}_o(\delta)) = \int_\Omega \mathrm{MSE}(\hat{\lambda}_o(s;\delta))ds.$$

It is well known that the MSE can be decomposed as follows.

$$\mathrm{MSE}(\hat{\lambda}_o(s;\delta)) = \mathrm{Var}\Big[\hat{\lambda}_o(s;\delta)\Big] + \mathrm{bias}\Big[\hat{\lambda}_o(s;\delta)\Big]^2 \tag{2.46}$$

to get

$$\mathrm{IMSE}(\hat{\lambda}_o(\delta)) = \int_\Omega \mathrm{Var}\Big[\hat{\lambda}_o(s;\delta)\Big]\,ds + \int_\Omega \mathrm{bias}\Big[\hat{\lambda}_o(s;\delta)\Big]^2 ds \tag{2.47}$$

The value of MSE in practise is not known, because it involves the true λ_o. Since MSE is unknown, an estimator is used instead. One such method is cross-validation, [26]. To formulate this estimator, suppose that we have a collection of K random

variables $\{Y_k\}_{i=k}^K$, whose expectations are some prescribed functionals of λ_o. We adopt the following notation

$$\mathrm{E}\big[Y_k\big] = m_k[\lambda_o],$$

where m_k is some known functional, e.g. in the case of panel data we may have $Y_k = N_{ij}$ with $m_k[\lambda_o] = \mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}} \int_{R_j} \lambda_o(s)ds$. Note that this is well defined, since here we assume that the exact $\boldsymbol{\beta}$ is known. Now denote $\hat{\lambda}_o^{-k}$ the estimate of λ_o from the sample $\{Y_k\}_{k=1}^K$ with the k-th observation removed. Then

$$\mathrm{CV}(\delta) = \frac{1}{K}\sum_{k=1}^{K}\Big(Y_k - m_k[\hat{\lambda}_o^{-k}(\cdot;\delta)]\Big)^2$$

is a consistent estimator of IMSE [27].

Computing this expression directly is computationally very expensive, because it involves K solutions of the estimation problem. Instead, the so called *generalized cross-validation*, or GCV, provides an approximation of $\mathrm{CV}(\delta)$

$$\mathrm{GCV}(\delta) = \frac{K}{\big(K - \mathrm{edf}(\delta)\big)^2}\sum_{k=1}^{K}\Big(Y_k - m_k[\hat{\lambda}_o(\cdot;\delta)]\Big)^2. \tag{2.48}$$

Here we have denoted $\mathrm{edf}(\delta)$ the *effective degrees of freedom*. One widely accepted definition of $\mathrm{edf}(\delta)$ is the following. Let $\mathbf{Y}$ be the column vector of Y_k's and let $\mathbf{m}[\lambda_o]$ be the column vector of $m_k[\lambda_o]$'s. Then, for each δ we define the estimation operator $S(\delta) : \mathbb{R}^K \to \mathbb{R}^K$ such that $S(\delta)[\mathbf{Y}] = \mathbf{m}[\hat{\lambda}_o(\cdot;\delta)]$. In the case of a linear model, the operator $S(\delta)$ is linear and therefore is represented by a $K \times K$ matrix. The effective degrees of freedom in this case is defined as the trace of this matrix, i.e.

$$\mathrm{edf}(\delta) = \mathrm{tr}\big[S(\delta)\big].$$

When the model is not linear, which is our case, an appropriate linearization of $S(\delta)$

about the solution point can be used instead.

The application of these ideas to the cases of event locations and panel counts differ significantly in the details and will be discussed in later sections.

Numerical method

Once we have a way of computing the GCV as a function of δ we need a robust method of finding the δ that minimizes it. Here we describe a robust algorithm, which we prefer over Newton-Raphson, because we don't have a way of getting a good initial guess. Instead, the following algorithm is capable of finding the nearest local minimum starting from any initial point. Restarting the algorithm with different initial points, at least in principle, should give us all local minima and therefore the global one.

We note here that the minimization algorithm uses information about the derivative of GCV, which is not available to us in closed form. For this reason we use numerical differentiation by the formula

$$\frac{d}{d\delta}\mathrm{GCV}(\delta) \approx \frac{\mathrm{GCV}(\delta + h) - \mathrm{GCV}(\delta)}{h}$$

where h is a small number. The choice of h depends on the machine accuracy and the magnitude of $\mathrm{GCV}(\delta)$. We have found that values of h which are about six orders of magnitude smaller than $\mathrm{GCV}(\delta)$, but not smaller than 10^{-6}, give approximations of the derivative that are sufficiently accurate for our purposes when we use double precision floating point arithmetic. An analysis of the best choice of h can be found in standard textbooks on numerical methods and falls beyond the scope of this work.

In the minimization of GCV we use a combination of two algorithms. In the first algorithm, we maintain two approximations of the minimizer. At each iteration, we compute the cubic Hermitian interpolating polynomial that matches the values and

the derivatives of the objective function at these two points. Every cubic polynomial has either no local extrema, or exactly two of them – one local minimum and one local maximum. In the latter case, the point where the local minimum occurs replaces one of the two original points and the algorithm proceeds with the next iteration. In the former case, the algorithm fails to produce a new approximation and we resort to our second algorithm.

The second algorithm is the so called *backtracking*. In addition to completing a failed iteration of the first algorithm, we use backtracking in the very first iteration, when we have only one initial point and we need a second point in order start the cubic polynomial minimization. The basic idea of backtracking is to search for the next point by making a step from the current point in the direction in which the function decreases, starting with the largest stepsize that we want to try and then trying smaller and smaller stepsizes until a better approximation is found.

The first algorithm converges very fast once we have two points with the derivative at the left point being negative and the derivative at the right point being positive. However, it is unreliable until this condition is satisfied. The backtracking method on the other hand is slower, but guaranteed to succeed provided that the objective function is locally convex, which should be the case in the vicinity of a local minimum.

The computational details of these methods are given in Appendix B.

2.4.2 Restricted maximum likelihood

The idea of the REML estimate for the penalty parameter is to reparametrize α and to partition the representation into a part that belongs to the null space of the penalty operator and another part that doesn't, i.e.

$$\alpha(s) = \boldsymbol{\Psi}_o^T(s)\boldsymbol{\alpha}_o + \boldsymbol{\Psi}_p^T(s)\boldsymbol{\alpha}_p,$$

where $\left[\boldsymbol{\Psi}_o^T(s), \boldsymbol{\Psi}_p^T(s)\right]^T$ is a basis of $\mathbb{X}_h$ with the penalty operator giving zero for each function in $\boldsymbol{\Psi}_o$ and nonzero for each function in $\boldsymbol{\Psi}_p$. The REML criterion then considers the coefficients $\boldsymbol{\alpha}_p$ as random effects. The marginal distribution of the N_{ij}'s depends on the penalty parameter δ and on the fixed effects $\boldsymbol{\alpha}_o$. The optimal value of δ is the maximum likelihood estimator of this mixed model.

It can be shown that the optimal δ can be approximated by the solution of the equation

$$\delta \approx \frac{\mathrm{edf}(\delta)}{\mathcal{P}(\alpha)}.$$

This equation can be solved by a number of numerical methods for nonlinear equations. We have found in practice that the simple fixed point iteration

$$\delta^{m+1} = \frac{\mathrm{edf}(\delta^m)}{\mathcal{P}(\alpha^m)}$$

works well.

3 Numerical Illustrations

3.1 Introduction

In this section we present the results from numerical computations that we performed in order to investigate the properties of the methodology developed herein.

We apply our methodology on simulated random samples and compare the estimated values of the parameters and the estimated baseline intensity to the ones that were used to simulate the random data.

The main goal here is to investigate the quality of the Finite Element approximation of the baseline intensity and its performance as an estimator of the true baseline intensity. It is also important for us to demonstrate the advantages and usefulness of

FEM in the solutions of 2-dimensional problems.

In our simulation studies we did not include any covariates since the main contribution of this work is in the estimation of the baseline intensity.

3.2 Simulations in the case of panel data

Here we discuss the details of the computation of the GCV score in the case of panel data in the most general setting. Following this, we present the results of numerical simulations in four test cases, namely without and with frailties for 1-d and 2-d models.

3.2.1 Computation of GCV

Since we are considering the case of panel data, in this section J is given in (2.25). We will rewrite the index-based expressions of the gradient and Hessian of J using vector and matrix notation. For this we use the following notation: $\mathbf{1}$ will denote the $L \times 1$ column vector with all entries equal to one, $\mathbf{N}_i = \left[N_{i1}, \ldots, N_{iL}\right]^T$, and $\boldsymbol{\mu}_i = \left[\mu_{i1}, \ldots, \mu_{iL}\right]^T$. It is important to distinguish the boldface $\mathbf{N}_i$ and $\boldsymbol{\mu}_i$, denoting $L \times 1$ vectors from the regular N_i and μ_i, which are the sums of the entries of these vectors, i.e.

$$N_i = \sum_{j=1}^{L} N_{ij} = \mathbf{1}^T \mathbf{N}_i \quad \text{and} \quad \mu_i = \sum_{j=1}^{L} \mu_{ij} = \mathbf{1}^T \boldsymbol{\mu}_i.$$

We will also denote $\mathbf{D}_{\beta i} = \left(\nabla_{\boldsymbol{\beta}} \boldsymbol{\mu}_i{}^T\right)^T = \boldsymbol{\mu}_i \mathbf{x}_i^T$ the $L \times p$ matrix whose (j,k)-th entry is the partial derivative of μ_{ij} with respect to β_k. Similarly, we denote the $L \times n$ matrix $\mathbf{D}_{\alpha i} = \left(\nabla_{\boldsymbol{\alpha}} \boldsymbol{\mu}_i{}^T\right)^T$.

The gradient of J can be written as follows

$$\nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M} \frac{N_i - \mu_i}{\mathrm{Var}\left[N_i\right]} \nabla_{\boldsymbol{\beta}} \mu_i = \sum_{i=1}^{M} \mathbf{D}_{\beta i}^T \mathbf{1} \left[\frac{1}{\mathrm{Var}\left[N_i\right]}\right] \mathbf{1}^T \left(\mathbf{N}_i - \boldsymbol{\mu}_i\right)$$

$$= \sum_{i=1}^{M} \mathbf{D}_{\beta i}^T \mathbf{V}_i^{-1} (\mathbf{N}_i - \boldsymbol{\mu}_i) \tag{3.1}$$

$$\nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M} \left[\sum_{j=1}^{L} \left(\frac{N_{ij} - \mu_{ij}}{\mu_{ij}} - \tau \frac{N_i - \mu_i}{1 + \tau \mu_i} \right) \nabla_{\boldsymbol{\alpha}} \mu_{ij} \right] - 2\delta \mathbf{P} \boldsymbol{\alpha}$$

$$= \sum_{i=1}^{M} \mathbf{D}_{\alpha i}^T \left(\operatorname{diag} \left(\frac{1}{\boldsymbol{\mu}_i} \right) - \frac{\tau}{1 + \tau \mu_i} \mathbf{1}\mathbf{1}^T \right) (\mathbf{N}_i - \boldsymbol{\mu}_i) - 2\delta \mathbf{P} \boldsymbol{\alpha}$$

$$= \sum_{i=1}^{M} \mathbf{D}_{\alpha i}^T \mathbf{V}_i^{-1} (\mathbf{N}_i - \boldsymbol{\mu}_i) - 2\delta \mathbf{P} \boldsymbol{\alpha}, \tag{3.2}$$

where $\mathbf{V}_i$ denotes the covariance matrix of $\mathbf{N}_i$, which is

$$\mathbf{V}_i = \operatorname{diag} (\boldsymbol{\mu}_i) + \tau \boldsymbol{\mu}_i \boldsymbol{\mu}_i{}^T,$$

and the validity of (3.2) can be justified by direct verification of the following identity

$$\left(\operatorname{diag} (\boldsymbol{\mu}_i) + \tau \boldsymbol{\mu}_i \boldsymbol{\mu}_i{}^T \right) \left(\operatorname{diag} \left(\frac{1}{\boldsymbol{\mu}_i} \right) - \frac{\tau}{1 + \tau \mu_i} \mathbf{1}\mathbf{1}^T \right) = \mathbf{I}.$$

The last step in the derivation of (3.1) holds, due to

$$\frac{1}{\operatorname{Var}[N_i]} = \frac{1}{\mu_i + \tau \mu_i^2} = \frac{1}{\mu_i} - \frac{\tau}{1 + \tau \mu_i} \qquad \text{and}$$

$$\mathbf{D}_{\beta i}^T \operatorname{diag} \left(\frac{1}{\boldsymbol{\mu}_i} \right) = \mathbf{D}_{\beta i}^T \left[\frac{1}{\mu_i} \mathbf{1}\mathbf{1}^T \right].$$

It is a standard practise in the numerical solutions of maximum likelihood estimation problems to use the expectation of the Hessian of the log-likelihood function instead of the exact Hessian. The advantage is that taking the expectation usually results in a simpler formula and often provides greater numerical stability. The above expression for the gradient of J makes it easy to perform this computation. We denote

$\mathbf{H}$ as the *negative* expectation of the Hessian matrix of J, which is given by

$$\mathbf{H} = \begin{bmatrix} \sum_{i=1}^{M} \mathbf{D}_{\beta i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\beta i} & \sum_{i=1}^{M} \mathbf{D}_{\beta i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\alpha i} \\ \sum_{i=1}^{M} \mathbf{D}_{\alpha i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\beta i} & \sum_{i=1}^{M} \mathbf{D}_{\alpha i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\alpha i} + 2\delta \mathbf{P} \end{bmatrix}. \tag{3.3}$$

The k-th step of the Fisher's scoring iteration is then

$$\begin{bmatrix} \boldsymbol{\beta}^{k+1} \\ \boldsymbol{\alpha}^{k+1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\beta}^{k} \\ \boldsymbol{\alpha}^{k} \end{bmatrix} + \mathbf{H}^{-1} \begin{bmatrix} \sum_{i=1}^{M} \mathbf{D}_{\beta i}{}^T \mathbf{V}_i^{-1} (\mathbf{N}_i - \boldsymbol{\mu}_i) \\ \sum_{i=1}^{M} \mathbf{D}_{\alpha i}{}^T \mathbf{V}_i^{-1} (\mathbf{N}_i - \boldsymbol{\mu}_i) - 2\delta \mathbf{P} \boldsymbol{\alpha}^k \end{bmatrix}, \tag{3.4}$$

which after some rearrangement can be written as follows

$$\mathbf{H} \begin{bmatrix} \boldsymbol{\beta}^{k+1} \\ \boldsymbol{\alpha}^{k+1} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{M} \mathbf{D}_{\beta i}{}^T \mathbf{V}_i^{-1} \mathbf{N}_i^k \\ \sum_{i=1}^{M} \mathbf{D}_{\alpha i}{}^T \mathbf{V}_i^{-1} \mathbf{N}_i^k \end{bmatrix}$$

with $\mathbf{N}_i^k = \mathbf{N}_i - \boldsymbol{\mu}_i + \mathbf{D}_{\beta i} \boldsymbol{\beta}^k + \mathbf{D}_{\alpha i} \boldsymbol{\alpha}^k$. Note that $(\boldsymbol{\beta}^{k+1}, \boldsymbol{\alpha}^{k+1})$ is the solution of the wighted penalized least squares problem

$$\text{minimize} \sum_{i=1}^{M} \left\| \mathbf{V}_i^{-\frac{1}{2}} \left(\mathbf{N}_i^k - \mathbf{D}_{\beta i} \boldsymbol{\beta} - \mathbf{D}_{\alpha i} \boldsymbol{\alpha} \right) \right\|^2 + 2\delta \mathbf{P} \boldsymbol{\alpha} \tag{3.5}$$

and the effective degrees of freedom for this linear problem is then

$$\text{edf}(\delta) = \text{tr} \left\{ \mathbf{H}^{-1} \begin{bmatrix} \sum_{i=1}^{M} \mathbf{D}_{\beta i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\beta i} & \sum_{i=1}^{M} \mathbf{D}_{\beta i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\alpha i} \\ \sum_{i=1}^{M} \mathbf{D}_{\alpha i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\beta i} & \sum_{i=1}^{M} \mathbf{D}_{\alpha i}{}^T \mathbf{V}_i^{-1} \mathbf{D}_{\alpha i} \end{bmatrix} \right\}. \tag{3.6}$$

We use formula (3.6) evaluated at the solution of the Fisher's scoring algorithm as an approximation of the effective degrees of freedom of the fit.

Furthermore, we use the same approximation for GCV. The formula that gives the GCV score for the linear problem (3.5) is

$$\mathrm{GCV}(\delta) = \frac{ML}{\left(ML - \mathrm{edf}(\delta)\right)^2} \sum_{i=1}^{M} \left\| \mathbf{V}_i^{-\frac{1}{2}} \left(\mathbf{N}_i^k - \mathbf{D}_{\beta i}\boldsymbol{\beta}^{k+1} - \mathbf{D}_{\alpha i}\boldsymbol{\alpha}^{k+1} \right) \right\|^2 \tag{3.7}$$

When we evaluate this formula at the last iteration of the Fisher's scoring method, the differences between the last and the second last approximations of $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ are negligible, so the formula simplifies to

$$\mathrm{GCV}(\delta) = \frac{ML}{\left(ML - \mathrm{edf}(\delta)\right)^2} \sum_{i=1}^{M} \left(\mathbf{N}_i - \boldsymbol{\mu}_i\right)^T \mathbf{V}_i^{-1} \left(\mathbf{N}_i - \boldsymbol{\mu}_i\right). \tag{3.8}$$

3.2.2 Independent Poisson Model

As our first numerical example, we consider M independent and identical Poisson processes. The simplicity of this case facilitated the initial development and testing of the estimation code. It also aims to demonstrate the applicability of the FEM to the approximation of the baseline intensity in a simple and clear setting.

Since the processes are assumed identical, there are no covariates, and the expected means μ_{ij} are equal for all $i = 1, \ldots, M$, we will omit the index i to emphasize this fact. In this case the statistic $\sum_{i=1}^{M} \mathbf{N}_i$ is sufficient for $\boldsymbol{\alpha}$. We denote $\mu_j = \mu_{ij} = \int_{R_j} \mathrm{e}^{\boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}} ds$, $\mathbf{V} = \mathbf{V}_i = \mathrm{diag}(\boldsymbol{\mu})$ and $\mathbf{D} = \mathbf{D}_{\alpha i} = \left(\nabla_{\boldsymbol{\alpha}} \boldsymbol{\mu}^T\right)^T$ for all $i = 1, \ldots, M$. The gradient and the negative expected Hessian simplify to

$$\begin{aligned} \nabla_{\boldsymbol{\alpha}} J &= M\mathbf{D}^T\mathbf{V}^{-1} \left(\frac{1}{M} \sum_{i=1}^{M} \mathbf{N}_i - \boldsymbol{\mu} \right) - 2\delta\mathbf{P}\boldsymbol{\alpha} \\ \mathbf{H} &= M\mathbf{D}^T\mathbf{V}^{-1}\mathbf{D} + 2\delta\mathbf{P}. \end{aligned} \tag{3.9}$$

The one dimensional case

The domain we use for all one dimensional simulation tests is $\Omega = [0, 5]$. For simplicity, when dividing the domain into regions, we use intervals of equal length, i.e. for L regions, we have $R_j = [r_{j-1}, r_j]$, where $r_j = \frac{5j}{L}, j = 0, \ldots, L$.

We choose a function $\lambda_o(s)$ and generate a random sample as per the model we are considering. We first compute the expected regional counts, μ_j, by numerical integration of λ_o over the regions and then we draw L independent random variables from Poisson$(\mu_j), j = 1, \ldots, L$ distributions. We repeat this M times to obtain an $M \times L$ table N_{ij}.

Consider a piecewise polynomial $q(x)$ defined on the interval $x \in [0, 1]$ with three pieces separated by two knots, $0 < x_1 < x_2 < 1$. The first piece is an increasing cubic with $q'(0) = 0$, i.e. it has the form $a_0 + a_1 x^3$. The last piece by symmetry has the form $c_0 + c_1(1 - x)^3$. The middle piece is required to connect with C^2 smoothness at the knots and to have a maximum greater than $\max\{q((x_1), q(x_2)\}$ at a point somewhere in (x_1, x_2). It is easy to see that the lowest degree polynomial that allows these conditions to be satisfied is 4. The function $q(x)$ constructed this way belongs to a seven dimensional space of spline functions, since the three polynomial pieces have $4 + 5 + 4 = 13$ degrees of freedom and six of them are used to satisfy the smoothness conditions at x_1 and x_2. Our choice of $x_1 = 0.4, x_2 = 0.7, a_0 = c_0 = 0, \max_{0<x<1} q(x) = \ln(4)$ gives the following

$$q(x) = \begin{cases} 12.30595x^3 & x \in [0, 0.4] \\ q_1(x) & x \in [0.4, 0.7] \\ 23.39650(1 - x)^3 & x \in [0.7, 1] \end{cases} \tag{3.10}$$

where

$$q_1(x) = 27.34656(x-0.7)^4 + 38.99417(x-0.4)^4 +$$
$$4308.603(x-0.7)^3\left(\tfrac{1}{6}(x-0.7) - \tfrac{1}{12}(x-0.4)\right) + 8.860286(x-0.4) + 0.5660738$$

The exact baseline intensity is chosen to be $\lambda_o(s) = e^{4+0.2q(s/5)}$ for $s \in \Omega$. We have $54.49815 < \lambda_o(s) < 72.04269$ and the expected total count is $\Lambda_o(\Omega) \approx 301.0394$. Note that $4 + 0.2q(s/5)$ is not a member of the FE interpolation space $\mathbb{X}_h$, since the middle piece of $q(s)$ is a quartic polynomial. In order to eliminate any bias resulting from the Finite Element approximation, in our simulation tests we use the projection of $4 + 0.2q(s/5)$ onto $\mathbb{X}_h$ as the exact $\alpha(s) = \boldsymbol{\alpha}^T\boldsymbol{\Phi}(s) \in \mathbb{X}_h$, and respectively the exact baseline intensity becomes $\lambda_o(s) = e^{\boldsymbol{\alpha}^T\boldsymbol{\Phi}(s)}$. The graph of this function is shown in figure 14.

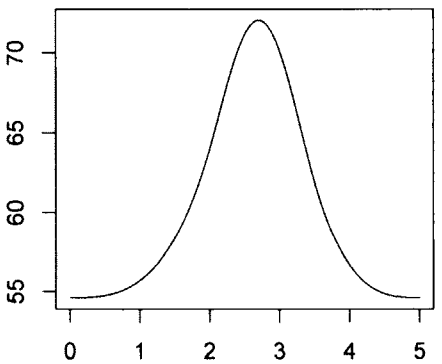

Figure 14: λ_o used in 1-d tests with simulated data

The first numerical test demonstrates the convergence of the solution of the estimation problem, when refining the finite element grid. We generated a single random sample with $M = 100, L = 50$ and estimated λ_o on a series of grids with $1, 3, 7, 15, 31, 63$ and 127 finite elements, respectively giving $n = 4, 8, 16, 32, 64, 128$ and 256 degrees of freedom in the finite element interpolation space. For each grid we ran the estimation twice, once selecting the optimal δ using GCV and again using REML. The results are presented in table 1 and figure 15.

The norm used to measure the differences between functions is the standard $L_2(\Omega)$ norm, i.e. $\|f\|^2 = \int_\Omega f(s)^2 ds$. We see from the table that the solutions become practically indistinguishable for larger n. The conclusion is that using a very fine grid is not justified, as long as the interpolation space has dimension that is sufficiently large

to provide ample flexibility. Through our many numerical simulations we have seen that for this λ_o GCV gives on average edf of about 8 and REML always gives lower edf on the same sample.

In the last test we present in this section, we solved three cases with $(M, L) = (25, 20), (50, 50)$, and $(100, 150)$ on a grid with 31 finite elements giving $n = 64$ degrees of freedom in the interpolation basis. In each case, we simulated $B = 1000$ samples and estimated $\alpha(s)$ using GCV and REML. The plots in figure 16 show the averages over 1000 simulations of the estimated coefficients of $\alpha(s)$, the bias and standard errors. If we denote $\hat{\boldsymbol{\alpha}}_b$ the coefficients estimated from the b-th sample, then we have

$$\overline{\boldsymbol{\alpha}} = \frac{1}{B}\sum_{b=1}^{B}\hat{\boldsymbol{\alpha}}_b \qquad \text{and} \qquad \text{bias} = \overline{\boldsymbol{\alpha}} - \boldsymbol{\alpha} \tag{3.11}$$

We compute two kinds of standard errors, a *simulated* and a *model-based* one. The former is simply the sample standard error

$$\text{simulated s.e.} = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(\hat{\boldsymbol{\alpha}}_b - \overline{\boldsymbol{\alpha}})^2}.$$

For the latter, we compute the model-based variance for each estimate, $\mathbf{V}(\hat{\boldsymbol{\alpha}}_b)$, by inverting the Fisher information matrix and taking the diagonal elements. What we are calling the model-based standard error is as follows

$$\text{model-based s.e.} = \sqrt{\frac{1}{B}\sum_{b=1}^{B}\mathbf{V}(\hat{\boldsymbol{\alpha}}_b)}.$$

We see from the plots in figure 16 that on average we get a better estimator of $\boldsymbol{\alpha}$ as the sample size increases, in the sense that both the bias and the standard error decrease. We also compare the performance of REML and GCV. The plots for the case of

		REML			GCV		
k	n	$\|\hat{\lambda}_k - \lambda_o\|$	edf	$\|\hat{\lambda}_k - \hat{\lambda}_{k-1}\|$	$\|\hat{\lambda}_k - \lambda_o\|$	edf	$\|\hat{\lambda}_k - \hat{\lambda}_{k-1}\|$
1	4	7.11216	3.7	0.00000	7.11176	3.7	0.00000
2	8	2.47490	5.6	4.91814	1.99479	6.1	5.45768
3	16	2.37947	6.0	0.37694	1.91718	6.7	0.46070
4	32	2.37513	6.2	0.05864	2.28789	15.0	2.27895
5	64	2.37529	6.2	0.00556	2.34408	15.7	0.16180
6	128	2.37541	6.2	0.00039	2.34608	15.7	0.01827
7	256	2.37542	6.2	0.00003	2.34581	15.7	0.00124

Table 1: See caption in figure 15 for details.

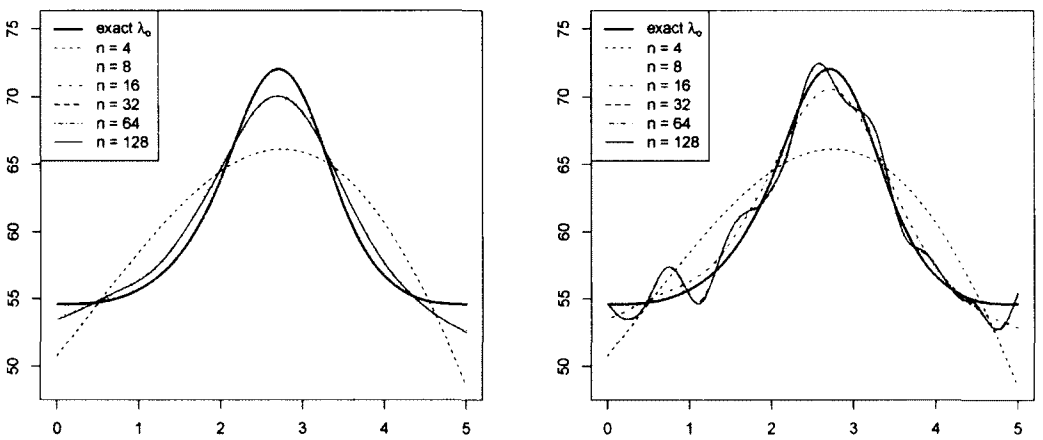

Figure 15: Convergence of the estimation of the same sample on different FE grids. $n = 2(1 + \#\text{nodes})$ is the dimension of the FE interpolation space. Sample contains $M = 100$ individuals, domain $\Omega = [0, 5]$ is divided into $L = 50$ regions. The step function shown is the empirical baseline intensity. Optimal penalty parameter obtained by REML (left) and by GCV (right).

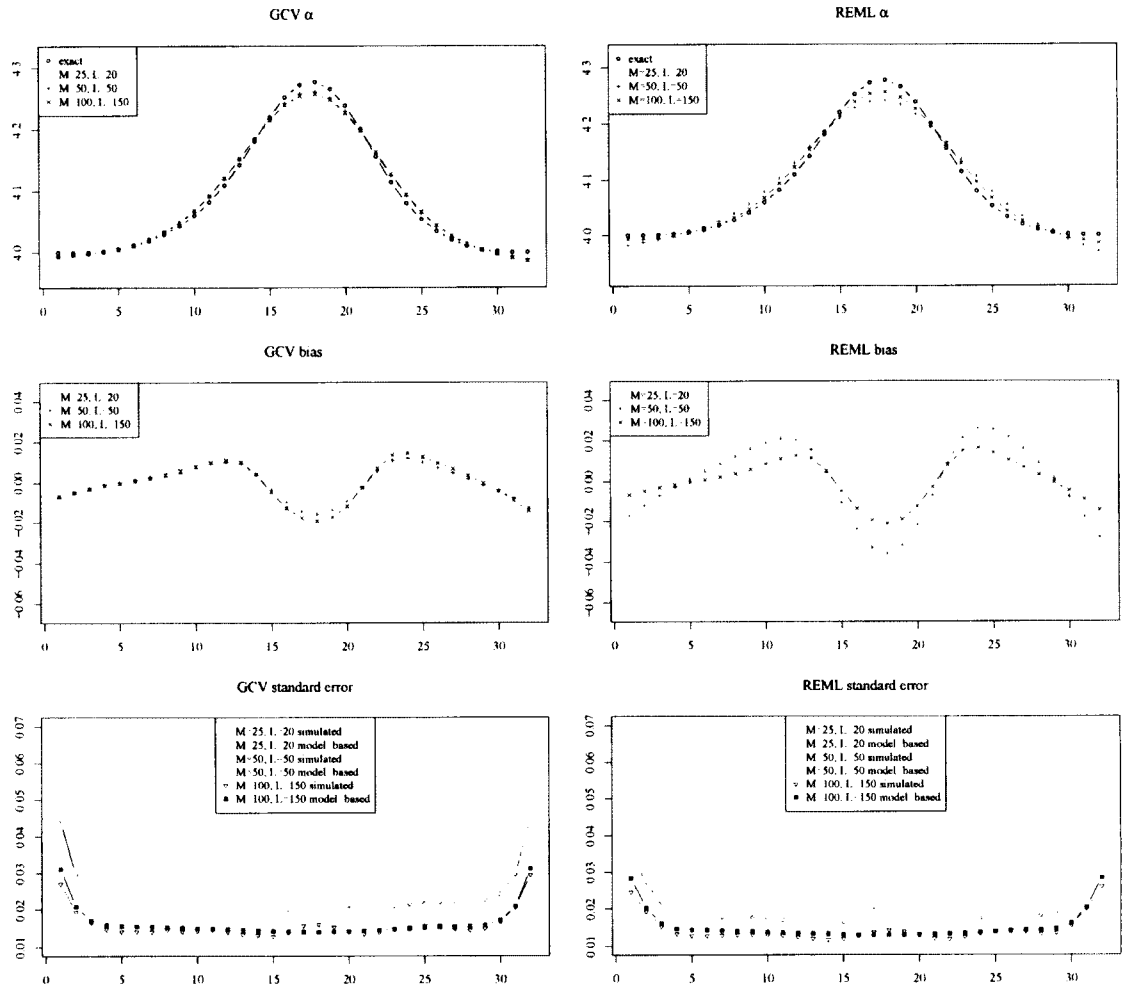

Figure 16: Simulation results for 1d independent Poisson model for different sample sizes. Finite element interpolation has $n = 64$ degrees of freedom. Each test case was estimated over $B = 1000$ random samples.

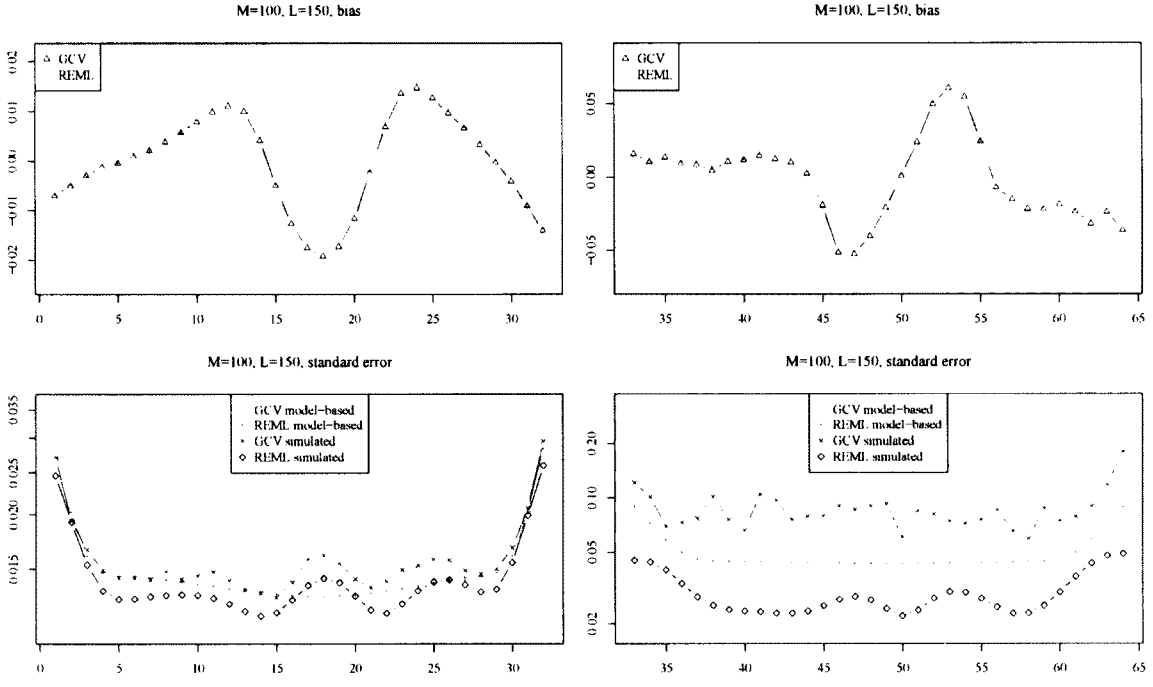

Figure 17: Biases and standard errors in the test case of $M = 100, L = 150$ for independent Poisson model. The coefficients corresponding to values of $\alpha(s)$ are shown on the left, the coefficients corresponding to values of $\alpha'(s)$ are on the right.

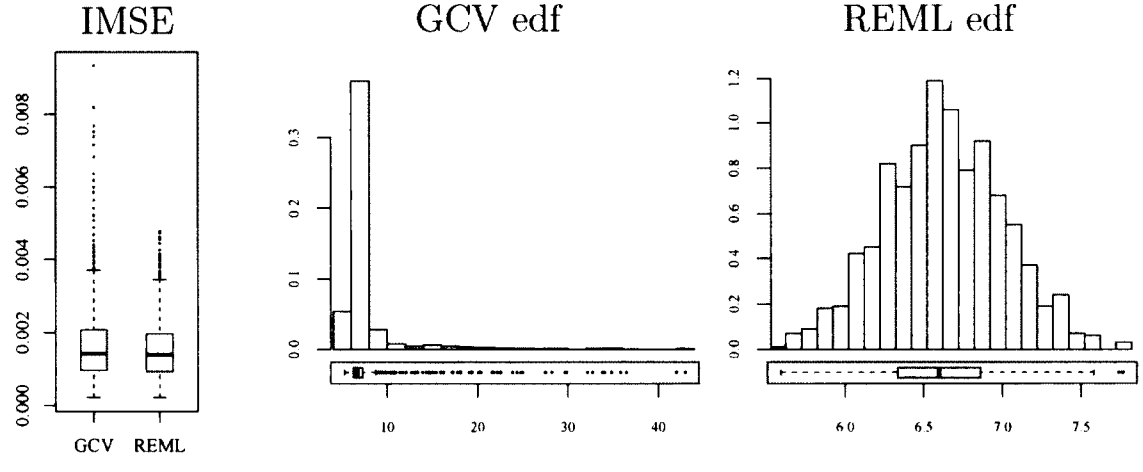

Figure 18: Histograms and boxplots of the distributions of edf for GCV and REML in the test case of $M = 100, L = 150$ for 1-d independent Poisson model.

$M = 100, L = 150$ are shown in figure 17. The estimates produced by GCV give on average slightly smaller bias and slightly larger standard errors when compared to those produced using REML. The average edf given by REML is 6.6 while GCV gives on average 7.6, however, as seen from the histograms and box plots in figure 18, the distribution of edf when GCV is used is skewed to the right. Figure 18 also shows box plots of the integrated mean squared error, which we compute by the formula

$$\mathrm{IMSE}_b = \int_\Omega \left(\boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}_b - \boldsymbol{\Phi}^T(s)\boldsymbol{\alpha}\right)^2 ds = \left(\boldsymbol{\alpha}_b - \boldsymbol{\alpha}\right)^T \mathbf{M} \left(\boldsymbol{\alpha}_b - \boldsymbol{\alpha}\right),$$

where $\mathbf{M}$ is the mass matrix defined in (2.43).

The two dimensional case

For the two dimensional simulation tests we use the domain $s = (x, y) \in \Omega = [0, 5]^2$ with the exact $\alpha(s)$ being the projection of $3 + 0.2q(x/5)q(y/5)$ onto $\mathbb{X}_h$, where q is defined in (3.10). For this choice of $\alpha(s)$ we have $20.08554 \leq \lambda_o(s) \leq 29.49918$ and the expected total count is $\Lambda_o(\Omega) \approx 526.0851$. Since this two dimensional α is constructed as the Cartesian product of a curve with seven degees of freedom with itself, we conclude that it has 49 degrees of freedom. The graph of $\alpha(s)$ is illustrated in figure 19.

Figure 19: α used in 2d simulations.

In two dimensions we require that edges between finite elements be aligned with the borders between the regions. Under this restriction, every finite element belongs to exactly one region and each region contains only whole finite elements. This makes integration over individual regions straight-forward and computationally efficient in the standard Finite Element implementation. This requirement may be relaxed at the expense of increased programming complexity.

For simplicity, we partition the domain into a regular grid of rectangular regions and we split each region into two triangular finite elements by drawing one of the two diagonals. Sample grids are shown in figure 20.

We solve two test cases with $M = 100$ on an $L = 8 \times 8$ grid and with $M = 300$ on an $L = 15 \times 15$ grid. The coarser grid gives a total of $n = 451$ degrees of freedom, while the finer grid has 1473 degrees of freedom. Figure 21 gives a comparison of the performance of the estimation in the two cases using GCV and REML. The graphs present a slice of the surfaces, which is perpendicular to the xy-plane and going along the line $y = x$, which covers all the features of the baseline intensity. The biases plot contains the bias curves as given by the interpolation, with the actual nodal values marked by symbols on each curve. In the standard error plots, the symbols represent the standard errors of the coefficients corresponding to function values. The horizontal axis in all plots in figure 21 is the distance from the origin along the $y = x$ line. We see from these plots that, similar to the 1-d case, the estimate has smaller bias and smaller variance on finer partitioning of Ω. In addition, REML gives estimates with a larger bias and lower variance than GCV. This is also confirmed by the histograms in figure 22.

Figure 23 presents perspective and contour plots of the REML-estimated $\alpha(x, y)$ and $\partial_x \alpha(x, y)$ surfaces for the case with $M = 300$ and $L = 15 \times 15$, as well as perspective

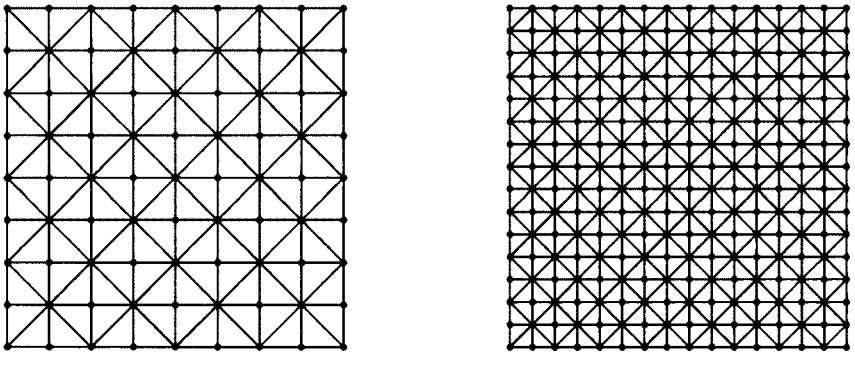

Figure 20: 8×8 and 15×15 finite element grids used in 2-d simulations.

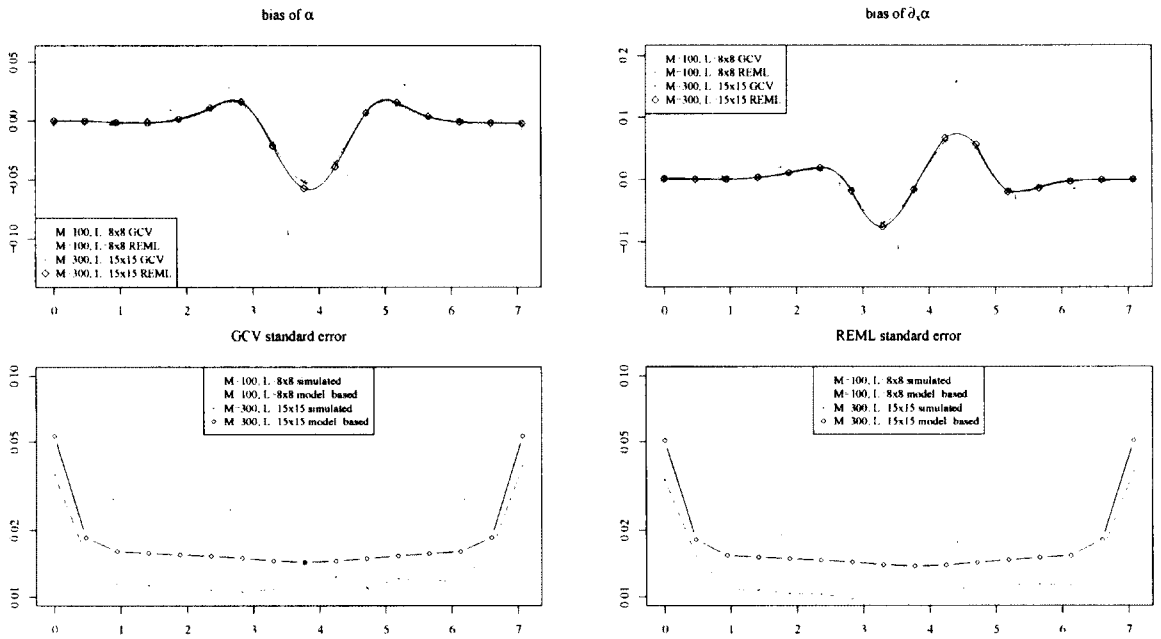

Figure 21: Comparison of the biases and standard errors along the line $y = x$ in the two dimensional independent Poisson model. Each test case simulation was replicated $B = 500$ times.

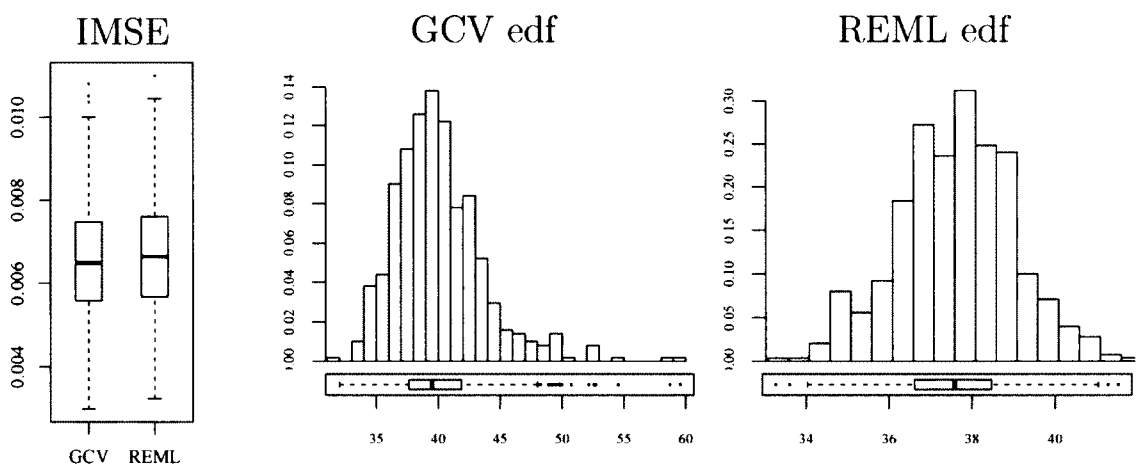

Figure 22: Histograms and boxplots of the distributions of edf for GCV and REML in the test case of $M = 300, L = 15 \times 15$ for 2-d independent Poisson model.

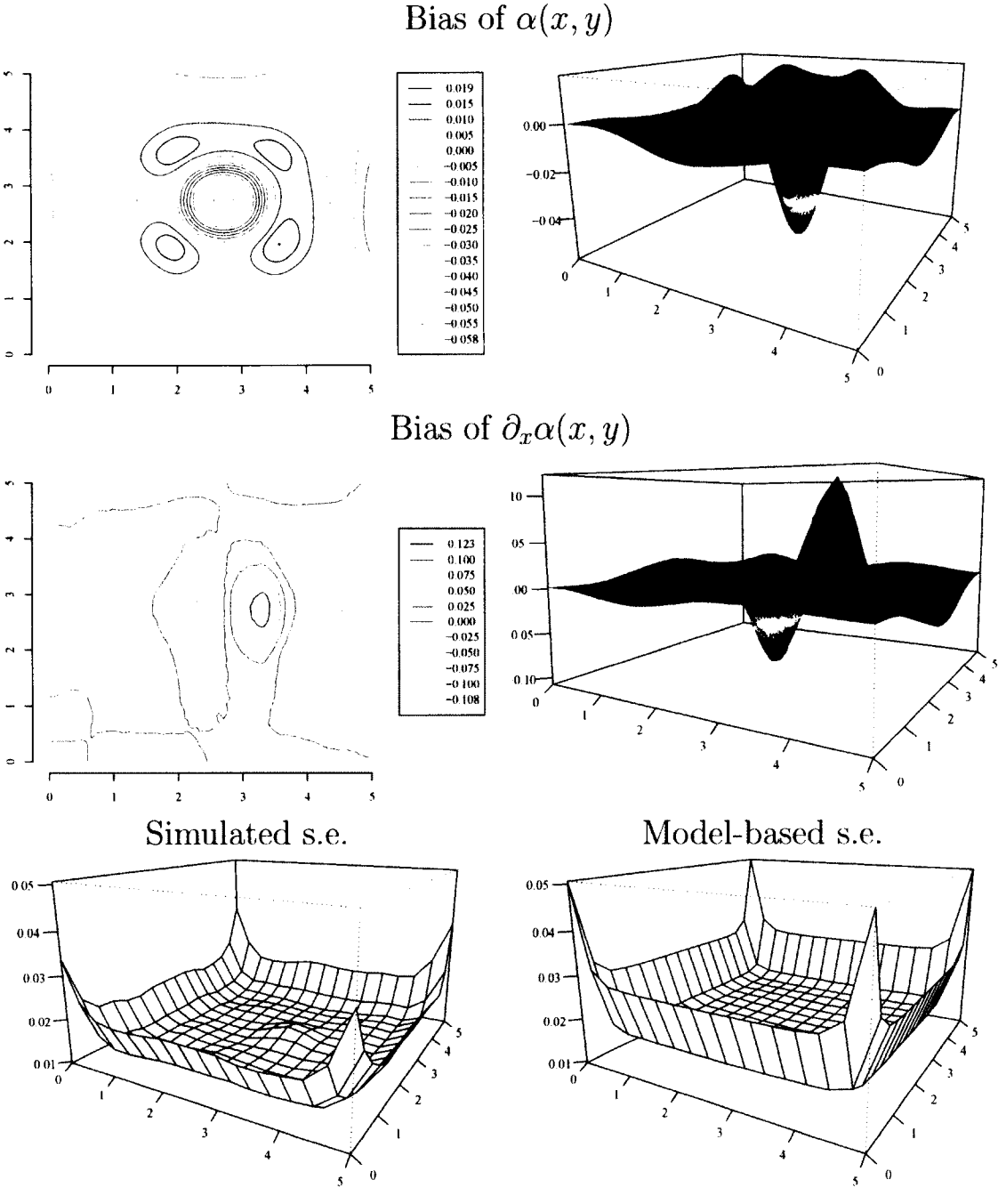

Figure 23: Surfaces and contour plots for the 2d independent Poisson model estimated using REML for $M = 300$, $L = 15 \times 15$.

plots of the simulated and model-based standard errors of the nodal values of $\alpha(x, y)$. It is interesting to notice in the second row of plots in figure 23 and in the top-right plot in figure 21 the discontinuity of the second derivatives of α. It is clearly demonstrated by sharp corners in the first partial x derivatives of α.

3.2.3 Negative binomial model

In this model we add individual-specific frailties, still without covariates. The equations in (3.9) apply with $\mathbf{V} = \mathbf{V}_i = \text{diag}(\boldsymbol{\mu}) + \tau \boldsymbol{\mu}\boldsymbol{\mu}^T$.

To generate a random sample from this model, we follow these two steps. First we generate a random sample $\{\nu_i\}_{i=1}^M$ of M independent random variables from a $\Gamma(\frac{1}{\tau}, \tau)$ distribution. Next, for each $i \in \{1, \ldots, M\}$ we draw L independent random variables from Poisson$(\nu_i \mu_j), j = 1, \ldots, L$ distributions.

The one dimensional case

Here we repeated the same simulations as we did in the independent Poisson case with two different values of the exact τ. The results for $\tau = 0.2$ are given in figure 24 and figure 25. The results in the case of $\tau = 0.01$ are nearly identical to those in the independent Poisson model. With the larger $\tau = 0.2$, we see that the bias and variances increase, however the performance remains adequate.

We observed numerically that the estimated values of τ with REML and GCV were identical in every simulation. This is not surprising, since the optimal value of τ is not influenced by the choice of the penalty parameter. The histograms of the estimated values of τ in the case with $(M, L) = (100, 150)$ for both values of the exact τ are shown in figure 26.

The two dimensional case

We ran simulations with the exact same setting as the simulations we presented in

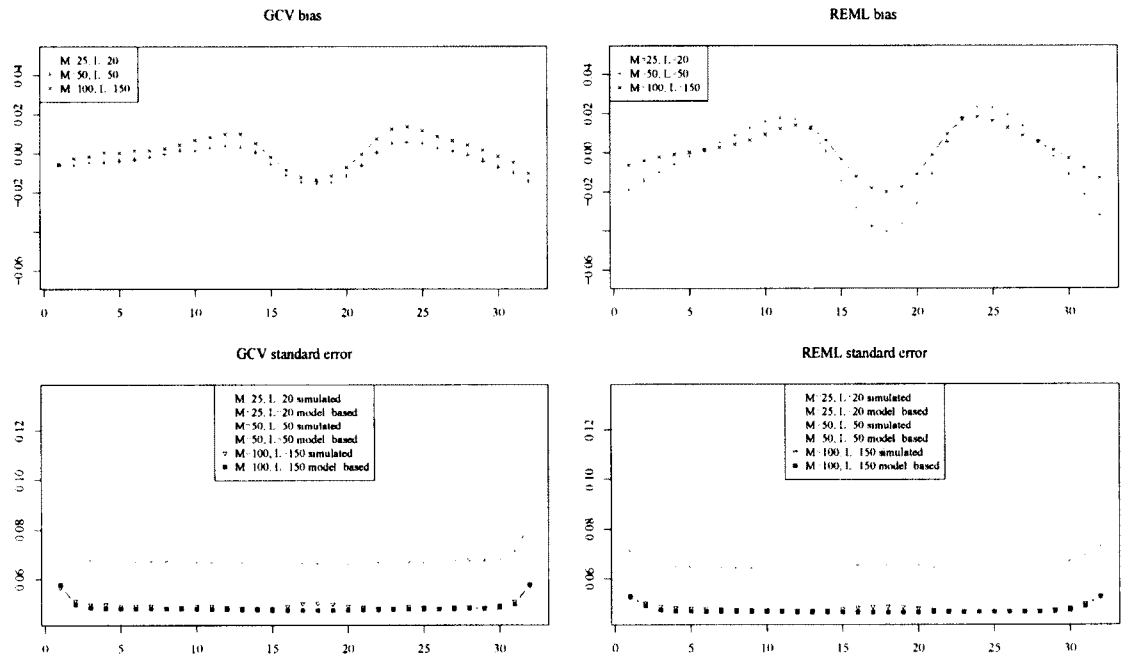

Figure 24: Comparison of bias and standard error for 1d negative binomial model with exact $\tau = 0.2$. Finite element interpolation has $n = 64$ degrees of freedom in each case. Each test case was replicated $B = 1000$ times.

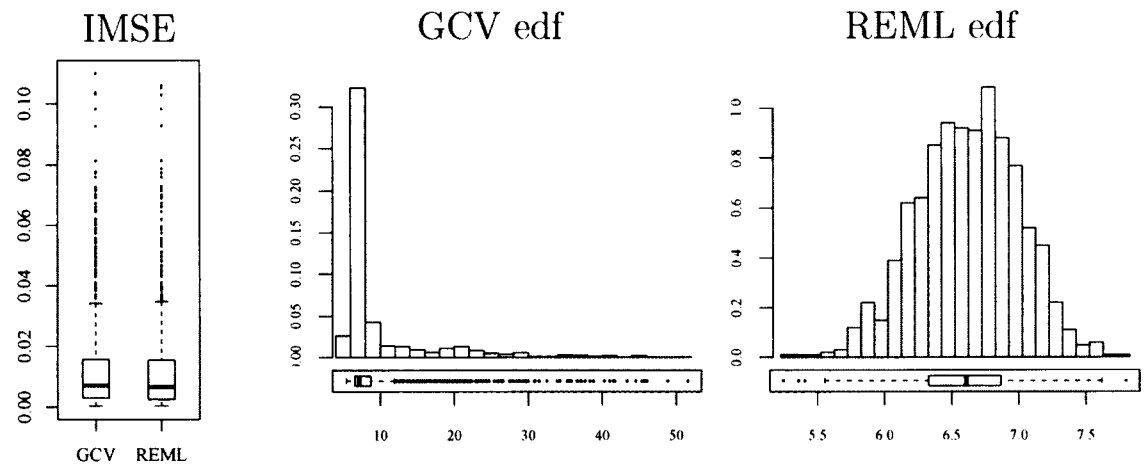

Figure 25: Histograms and boxplots of the distributions of edf for GCV and REML in the test case of $M = 100, L = 150$ for 1-d negative binomial model with exact $\tau = 0.2$.

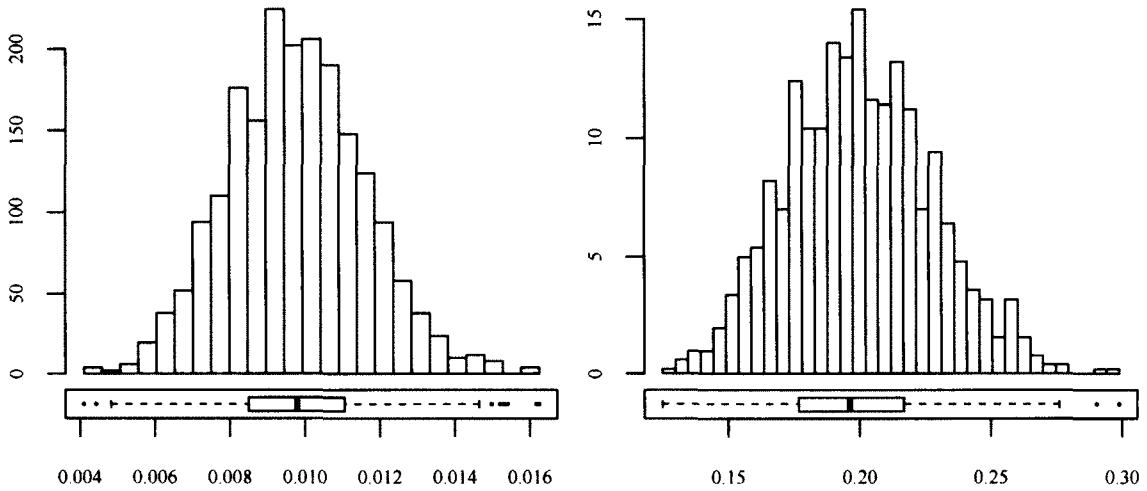

Figure 26: Histograms of estimated τ in 1-d case with exact $\tau = 0.01$ (left) and exact $\tau = 0.2$ (right)

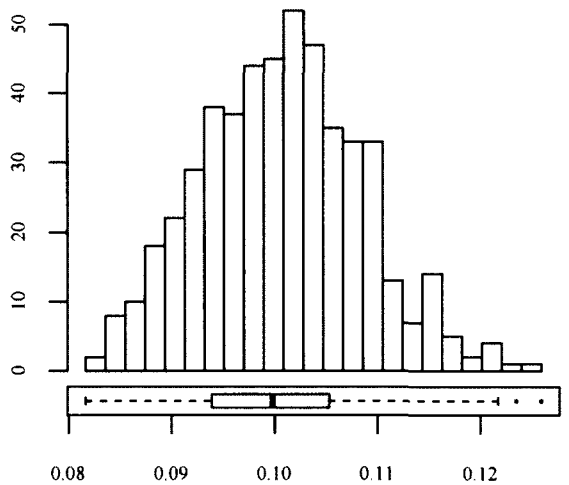

Figure 27: Histogram of τ for 2d mixed Poisson model for panel counts data. Exact $\tau = 0.1$

the 2-d independent Poisson case with the exact value of τ set to 0.1. The results for α are nearly identical to the independent Poisson model case. The histogram of τ is shown in figure 27.

3.3 Simulations in the case of event locations

In this section we discuss the computation of GCV in the event locations case, after which we present the results from our numerical tests with simulated data.

3.3.1 Computation of GCV

Here we work with expression (2.24) for J. Our goal is to rewrite the gradient of J in a form similar to (3.1) and (3.2), from which the rest of the derivation of GCV should follow the same way as in the panel data case. We start with the following expression for the gradient of J with respect to $\boldsymbol{\alpha}$

$$2\delta \mathbf{P}\boldsymbol{\alpha} + \nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M} \left[\sum_{j=1}^{N_i} \boldsymbol{\Phi}(s_{ij}) - \nabla_{\boldsymbol{\alpha}} \mu_i - \frac{\tau}{1+\tau\mu_i} (N_i - \mu_i) \nabla_{\boldsymbol{\alpha}} \mu_i \right]. \tag{3.12}$$

Denote $\mathbf{o}$ the $n \times 1$ vector of coefficients in our finite element interpolation of the function constant 1, i.e. $\boldsymbol{\Phi}^T(s)\mathbf{o} = 1$, for all $s \in \Omega$. Such vector $\mathbf{o}$ exists, since the function 1 belongs to $\mathbb{X}_h$. In fact, $\mathbf{o}$ is computed by solving the following linear system

$$\mathbf{M}\mathbf{o} = \int_{\Omega} \boldsymbol{\Phi}(s) ds. \tag{3.13}$$

The matrix $\mathbf{M}$ is the finite element mass matrix, which we introduced in equation (2.43). We also denote $\mathbf{L}_i$ the weighted mass matrix with weighting function $\lambda_i(s)$ given by

$$\mathbf{L}_i = \int_{\Omega} \lambda_i(s) \boldsymbol{\Phi}(s) \boldsymbol{\Phi}^T(s) ds \quad \text{with entries} \quad \left(\mathbf{L}_i\right)_{kl} = \int_{\Omega} \lambda_i(s) \varphi_k(s) \varphi_l(s) ds. \tag{3.14}$$

Notice that $\mathbf{L}_i$ is nonsingular, provided that the weighting function does not vanish on any subset of Ω with a nonempty interior. In addition, since in our case $\lambda_i(s) > 0$ for all $s \in \Omega$, the matrix $\mathbf{L}_i$ is positive definite. We can express μ_i and $\nabla_{\boldsymbol{\alpha}}\mu_i$ in terms of the quantities we just defined as follows

$$\nabla_{\boldsymbol{\alpha}}\mu_i = \int_{\Omega} \lambda_i(s)\boldsymbol{\Phi}(s)ds = \mathbf{L}_i\mathbf{o}$$
$$\mu_i = \int_{\Omega} \lambda_i(s)ds = \mathbf{o}^T\mathbf{L}_i\mathbf{o}$$

and also

$$N_i = \sum_{j=1}^{N_i} 1 = \sum_{j=1}^{N_i} \mathbf{o}^T\boldsymbol{\Phi}(s_{ij}).$$

Upon substituting these in (3.12), we derive the following

$$\begin{aligned} 2\delta\mathbf{P}\boldsymbol{\alpha} + \nabla_{\boldsymbol{\alpha}}J &= \sum_{i=1}^{M}\left[\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij}) - \mathbf{L}_i\mathbf{o} - \frac{\tau}{1+\tau\mu_i}\left(\sum_{j=1}^{N_i}\mathbf{o}^T\boldsymbol{\Phi}(s_{ij}) - \mathbf{o}^T\mathbf{L}_i\mathbf{o}\right)\mathbf{L}_i\mathbf{o}\right] \\ &= \sum_{i=1}^{M}\left[\left(\mathbf{I} - \frac{\tau}{1+\tau\mu_i}\mathbf{L}_i\mathbf{o}\mathbf{o}^T\right)\left(\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij}) - \mathbf{L}_i\mathbf{o}\right)\right] \\ &= \sum_{i=1}^{M}\left[\mathbf{L}_i\left(\mathbf{L}_i^{-1} - \frac{\tau}{1+\tau\mu_i}\mathbf{o}\mathbf{o}^T\right)\left(\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij}) - \mathbf{L}_i\mathbf{o}\right)\right]. \end{aligned} \tag{3.15}$$

The expectation of (3.15) is equal to the zero vector since

$$\begin{aligned} \mathrm{E}\left[\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij})\right] &= \mathrm{E}\left[\mathrm{E}\left[\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij})\,\middle|\, N_i\right]\right] = \mathrm{E}\big[N_i\mathrm{E}\big[\boldsymbol{\Phi}(s_{i1})\big|\, N_i\big]\big] \\ &= \mathrm{E}\left[N_i\int_{\Omega}\boldsymbol{\Phi}(s)\frac{\lambda_i(s)}{\Lambda_i(\Omega)}ds\right] = \int_{\Omega}\lambda_i(s)\boldsymbol{\Phi}(s)ds = \mathbf{L}_i\mathbf{o}. \end{aligned}$$

The covariance of s_{ij} and s_{kl} is zero for different i and k, since individuals/replicates are assumed independent. Within individual covariance can be computed also by conditioning on N_i so that,

$$\begin{aligned}
\mathbf{V}_i &= \mathrm{Cov}\left[\sum_{j=1}^{N_i}\mathbf{\Phi}(s_{ij}), \sum_{k=1}^{N_i}\mathbf{\Phi}(s_{ik})\right] \\
&= \mathrm{E}\left[\mathrm{Cov}\left[\sum_{j=1}^{N_i}\mathbf{\Phi}(s_{ij}), \sum_{k=1}^{N_i}\mathbf{\Phi}(s_{ik})\middle| N_i\right]\right] \\
&\qquad + \mathrm{Cov}\left[\mathrm{E}\left[\sum_{j=1}^{N_i}\mathbf{\Phi}(s_{ij})\middle| N_i\right], \mathrm{E}\left[\sum_{k=1}^{N_i}\mathbf{\Phi}(s_{ik})\middle| N_i\right]\right] \\
&= \mathrm{E}\left[N_i \mathrm{Cov}\left[\mathbf{\Phi}(s_{i1}), \mathbf{\Phi}(s_{i1})\middle| N_i\right]\right] \\
&\qquad + \mathrm{Cov}\left[N_i \int_\Omega \frac{\lambda_i(s)}{\Lambda_i(\Omega)}\mathbf{\Phi}(s)ds, N_i \int_\Omega \frac{\lambda_i(s)}{\Lambda_i(\Omega)}\mathbf{\Phi}(s)ds\right] \\
&= \mathrm{E}\left[N_i\left(\frac{1}{\Lambda_i(\Omega)}\int_\Omega \lambda_i(s)\mathbf{\Phi}(s)\mathbf{\Phi}^T(s)ds - \frac{1}{\Lambda_i^2(\Omega)}\int_\Omega \lambda_i(s)\mathbf{\Phi}(s)ds \int_\Omega \lambda_i(s)\mathbf{\Phi}^T(s)ds\right)\right] \\
&\qquad + \mathrm{Var}\left[N_i\right]\frac{1}{\mu_i^2}\int_\Omega \lambda_i(s)\mathbf{\Phi}(s)ds \int_\Omega \lambda_i(s)\mathbf{\Phi}^T(s)ds \\
&= \int_\Omega \lambda_i \mathbf{\Phi}(s)\mathbf{\Phi}^T(s)ds - \frac{1}{\mu_i}\int_\Omega \lambda_i(s)\mathbf{\Phi}(s)ds \int_\Omega \lambda_i(s)\mathbf{\Phi}^T(s)ds \\
&\qquad \frac{\mu_i + \tau\mu_i^2}{\mu_i^2}\int_\Omega \lambda_i(s)\mathbf{\Phi}(s)ds \int_\Omega \lambda_i(s)\mathbf{\Phi}^T(s)ds \\
&= \mathbf{L}_i - \frac{1}{\mu_i}\mathbf{L}_i\mathbf{o}(\mathbf{L}_i\mathbf{o})^T + \frac{\mu_i + \tau\mu_i^2}{\mu_i^2}\mathbf{L}_i\mathbf{o}(\mathbf{L}_i\mathbf{o})^T = \mathbf{L}_i + \tau\mathbf{L}_i\mathbf{o}\mathbf{o}^T\mathbf{L}_i.
\end{aligned}$$

In this derivation, we used the fact that the event locations given N_i are independent and that $\mathbf{L}_i$ is a symmetric matrix. By direct verification, we have

$$\left[\mathbf{L}_i^{-1} - \frac{\tau}{1+\tau\mu_i}\mathbf{o}\mathbf{o}^T\right]\left[\mathbf{L}_i + \tau\mathbf{L}_i\mathbf{o}\mathbf{o}^T\mathbf{L}_i\right] = \mathbf{I}. \tag{3.16}$$

Last but not least, we point out that the Jacobian matrix of $\mathbf{L}_i\mathbf{o}$ with respect to $\boldsymbol{\alpha}$ is exactly $\mathbf{L}_i$. With this, we have shown that (3.15) can be interpreted in the same

way as (3.2).

The gradient of J with respect to $\boldsymbol{\beta}$ can be written in a similar way. We have $\nabla_{\boldsymbol{\beta}}\mu_i = \mathbf{x}_i\mu_i = \mathbf{x}_i\mathbf{o}^T\mathbf{L}_i\mathbf{o}$, which allows us to rewrite equation (2.27) as follows

$$\begin{aligned}
\nabla_{\boldsymbol{\beta}} J &= \sum_{i=1}^{M} \frac{N_i - \mu_i}{\mu_i + \tau\mu_i^2}\nabla_{\boldsymbol{\beta}}\mu_i = \sum_{i=1}^{M} \nabla_{\boldsymbol{\beta}}\mu_i \frac{1}{\mu_i + \tau\mu_i^2}(N_i - \mu_i) \\
&= \sum_{i=1}^{M} \mathbf{x}_i\mathbf{o}^T\mathbf{L}_i\mathbf{o}\left[\frac{1}{\mu_i} - \frac{\tau}{1+\tau\mu_i}\right]\mathbf{o}^T\left(\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij}) - \mathbf{L}_i\mathbf{o}\right) \\
&= \sum_{i=1}^{M} \mathbf{x}_i\mathbf{o}^T\mathbf{L}_i\left[\mathbf{L}_i^{-1} - \frac{\tau}{1+\tau\mu_i}\mathbf{o}\mathbf{o}^T\right]\left(\sum_{j=1}^{N_i}\boldsymbol{\Phi}(s_{ij}) - \mathbf{L}_i\mathbf{o}\right)
\end{aligned}$$

The last line in the above derivation is justified by

$$\mathbf{o}^T\mathbf{L}_i\mathbf{o}\left[\frac{1}{\mu_i}\right]\mathbf{o}^T = \mu_i\left[\frac{1}{\mu_i}\right]\mathbf{o}^T = \mathbf{o}^T\mathbf{L}_i\mathbf{L}_i^{-1}.$$

If we denote the random variables $C_{il} = \sum_{j=1}^{N_i}\varphi_l(s_{ij})$ and $\mathbf{C}_i = \left[C_{i1}, \ldots, C_{in}\right]^T$, then we have $\mathrm{E}\left[\mathbf{C}_i\right] = \mathbf{L}_i\mathbf{o}$, $\mathbf{V}_i = \mathrm{Var}\left[\mathbf{C}_i\right] = \mathbf{L} + \tau\mathbf{L}_i\mathbf{o}\mathbf{o}^T\mathbf{L}_i$. We also denote $\mathbf{D}_{\alpha i} = \mathbf{L}_i$ the $n \times n$ Jacobian matrix of $\mathrm{E}\left[\mathbf{C}_i\right]$ with respect to $\boldsymbol{\alpha}$ and $\mathbf{D}_{\beta i} = \mathbf{L}_i\mathbf{o}\mathbf{x}_i^T$ the $n \times p$ Jacobian matrix of $\mathrm{E}\left[\mathbf{C}_i\right]$ with respect to $\boldsymbol{\beta}$. Then, the gradient of J is (compare to (3.1) and (3.2))

$$\nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M} \mathbf{D}_{\beta i}^T\mathbf{V}_i^{-1}(\mathbf{C}_i - \mathbf{L}_i\mathbf{o}) \tag{3.17}$$

$$\nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M} \mathbf{D}_{\alpha i}^T\mathbf{V}_i^{-1}(\mathbf{C}_i - \mathbf{L}_i\mathbf{o}) - 2\delta\mathbf{P}\boldsymbol{\alpha} \tag{3.18}$$

With this, the derivation of the GCV score follows the same steps as in the panel data case, which we discussed in section 3.2.1. The edf is then computed in an analogous

manner as in (3.6). The formula for GCV is

$$\mathrm{GCV}(\delta) = \frac{Mn}{\left(Mn - \mathrm{edf}(\delta)\right)^2} \sum_{i=1}^{M} \left(\sum_{i=1}^{N_i} \mathbf{\Phi}(s_{ij}) - \boldsymbol{\mu}_i \right)^T \mathbf{V}_i^{-1} \left(\sum_{i=1}^{N_i} \mathbf{\Phi}(s_{ij}) - \boldsymbol{\mu}_i \right). \quad (3.19)$$

We can show that the random vector $\mathbf{C}_i$ is in some sense analogous to the vector of random panel counts $\mathbf{N}_i$. Let $\mathbb{X}_L$ denote the space of all functions that are constant over each region, i.e. $\mathbb{X}_L = \left\{ v : v|_{R_j} = \mathrm{const} \right\}$. This is an L-dimensional subspace of $L_2(\Omega)$ with basis $\{1_{R_1}(s), \ldots, 1_{R_L}(s)\}$. It is also a reproducing kernel Hilbert space with respect to the standard inner product in L_2 with the reproducing kernel being

$$K_L(s,t) = \sum_{j=1}^{L} \frac{1}{|R_j|} 1_{R_j}(s) 1_{R_j}(t).$$

If we denote $\mathbf{1_R}(s) = [1_{R_1}(s), \ldots, 1_{R_L}(s)]^T$ the vector of basis functions of $\mathbb{X}_L$, we can write the random vector $\mathbf{N}_i$ as

$$\mathbf{N}_i = \sum_{j=1}^{N_i} \mathbf{1_R}(s_{ij}).$$

Similarly, in the case of event locations we have the finite element interpolation space $\mathbb{X}_h$, which is also finite dimensional subspace of $L_2(\Omega)$, its basis is $\{\varphi_1(s), \ldots, \varphi_n(s)\}$, and has the reproducing kernel

$$K_h(s,t) = \mathbf{\Phi}^T(s) \mathbf{M}^{-1} \mathbf{\Phi}(t).$$

So the random vector in (3.15), $\mathbf{C}_i = \sum_{j=1}^{N_i} \mathbf{\Phi}(s_{ij})$, is analogous to $\mathbf{N}_i$. Furthermore, both random vectors are partitionings of N_i in the sense that

$$N_i = \mathbf{1}^T \mathbf{N}_i = \sum_{j=1}^{N_i} \mathbf{1}^T \mathbf{1_R}(s_{ij}) \quad \text{and} \quad N_i = \mathbf{o}^T \mathbf{C}_i = \sum_{j=1}^{N_i} \mathbf{o}^T \mathbf{\Phi}(s_{ij}).$$

3.3.2 Independent Poisson Model

Here we have the same domain and exact baseline intensity as in the case of panel data presented above, except that this time the domain is not partitioned into regions, rather the exact event times are known. The random sample is generated in two steps. We draw M independent random variables, $\{N_i\}_{i=1}^{M}$, from the Poisson$\big(\Lambda_o(\Omega)\big)$ distribution. Then we draw a total of $\sum_{i=1}^{M} N_i$ independent random variables from a distribution with p.d.f. $\lambda_o(s)/\Lambda_o(\Omega)$ by the acceptance-rejection method. These random variables are assigned in turn as event times to the M individuals - the first N_1 of them to individual $i = 1$, the following N_2 to individual $i = 2$, etc.

The one dimensional case

We performed simulations of three cases with $M = 25, 50$, and 100. Each case was replicated $B = 1000$ times and the averages of the estimates of α were computed and analyzed the same way as we did before in the panel data case. The bias and standard errors are presented in figure 28 and the histograms are given in figure 29. Not surprisingly, the results are similar but slightly better than the panel data case as would be expected.

The two dimensional case

The simulations in the 2-d event locations case were performed on a grid constructed the same way as in the case of panel data, however this time the regions have no meaning in the model, but are only constructed as part of the grid generation. The grids we used here are 10×10 giving a total of $n = 683$ degrees of freedom in the interpolation basis. We solved for $M = 100$ and $M = 300$. Each simulation was replicated $B = 500$ times. The comparison between the two sample sizes is shown in figure 30 and as expected the larger sample produces more accurate estimates. Figures 32 and 31 show the usual comparison between REML and GCV in the case

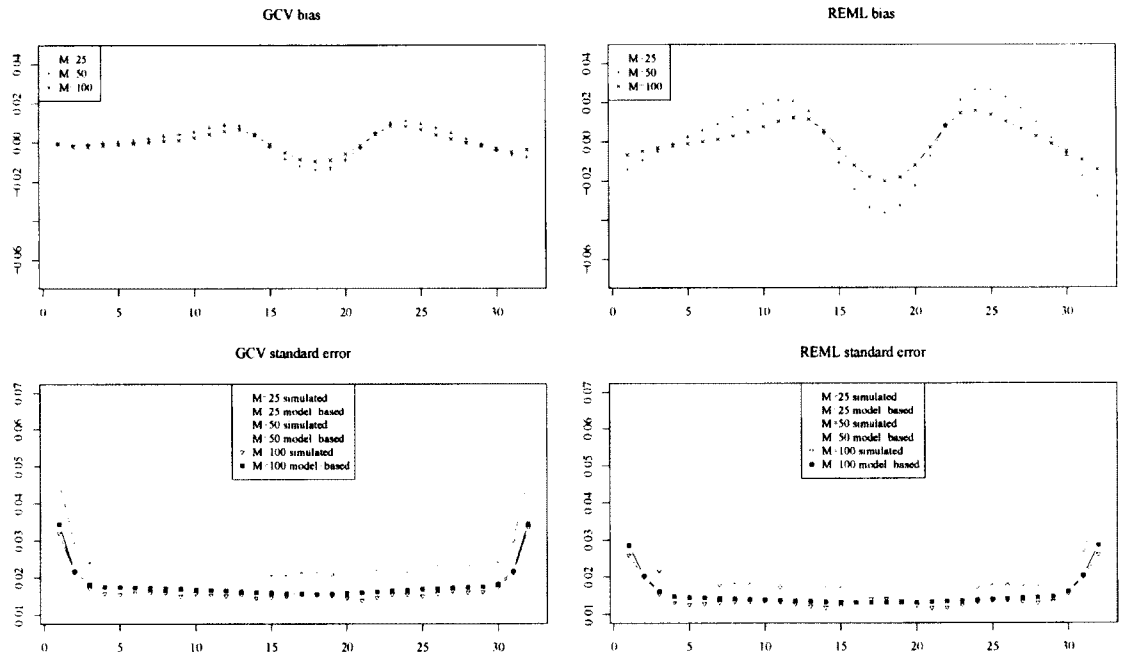

Figure 28: Comparison of bias and standard error for 1d event times model without frailty. Finite element interpolation has $n = 64$ degrees of freedom in each case. Each test case was replicated $B = 1000$ times.

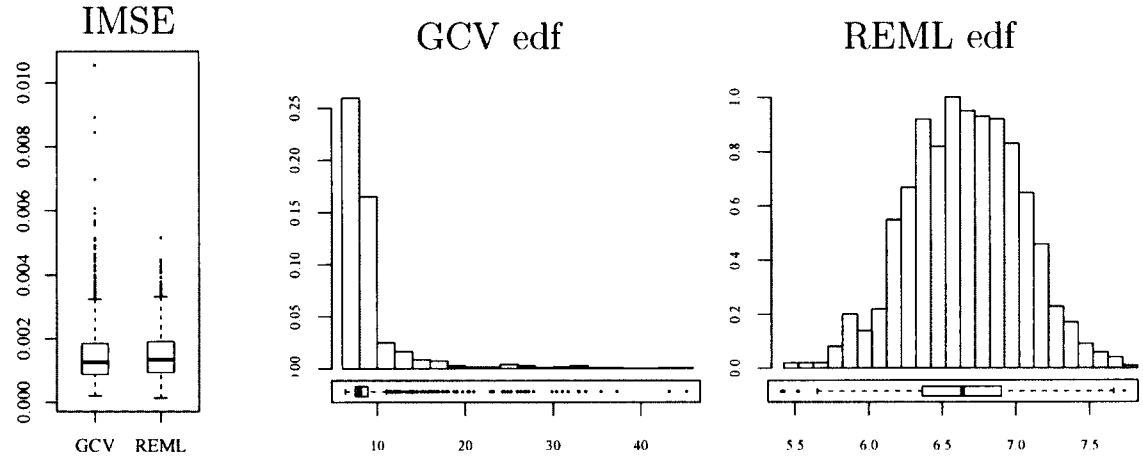

Figure 29: Histograms and boxplots of the distributions of edf for GCV and REML in the test case of $M = 100$ for 1-d event times Poisson model.

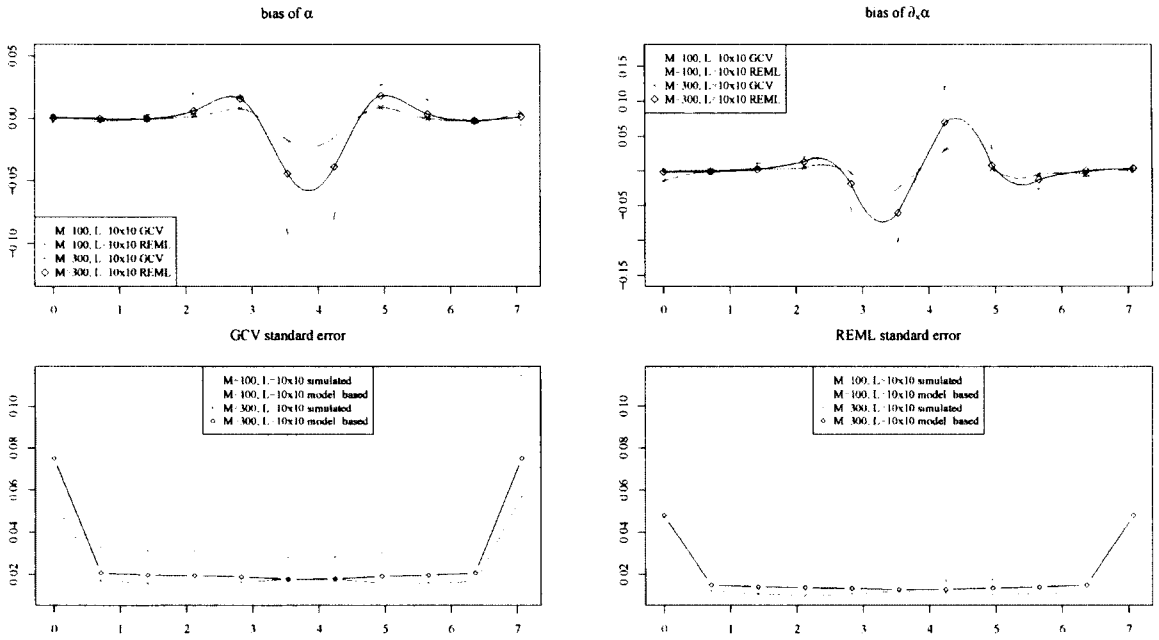

Figure 30: Comparison of the biases and standard errors along the line $y = x$ in the two dimensional event locations independent Poisson model. Each test case simulation was replicated $B = 500$ times.

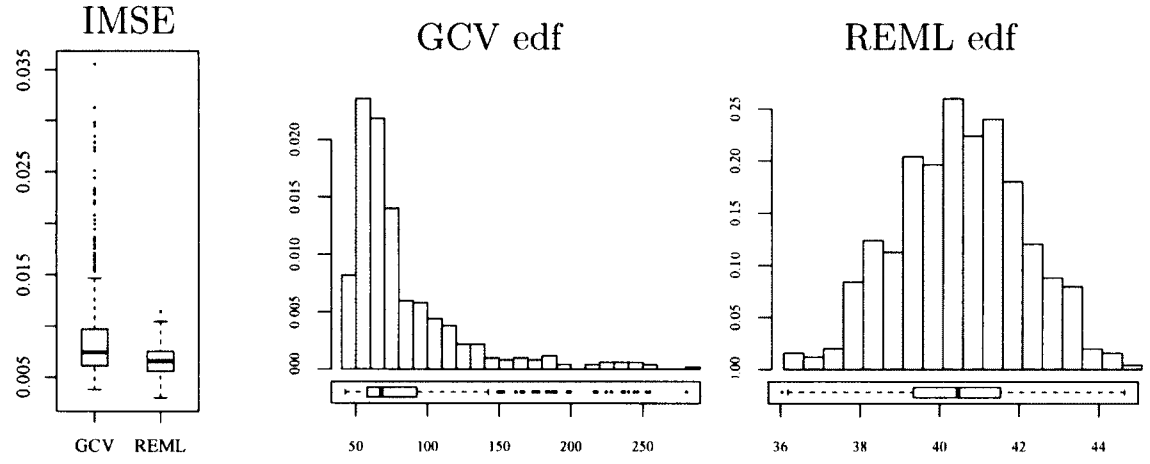

Figure 31: Histograms and boxplots of the distributions of edf for GCV and REML in the test case of $M = 300$ for 2-d event locations independent Poisson model.

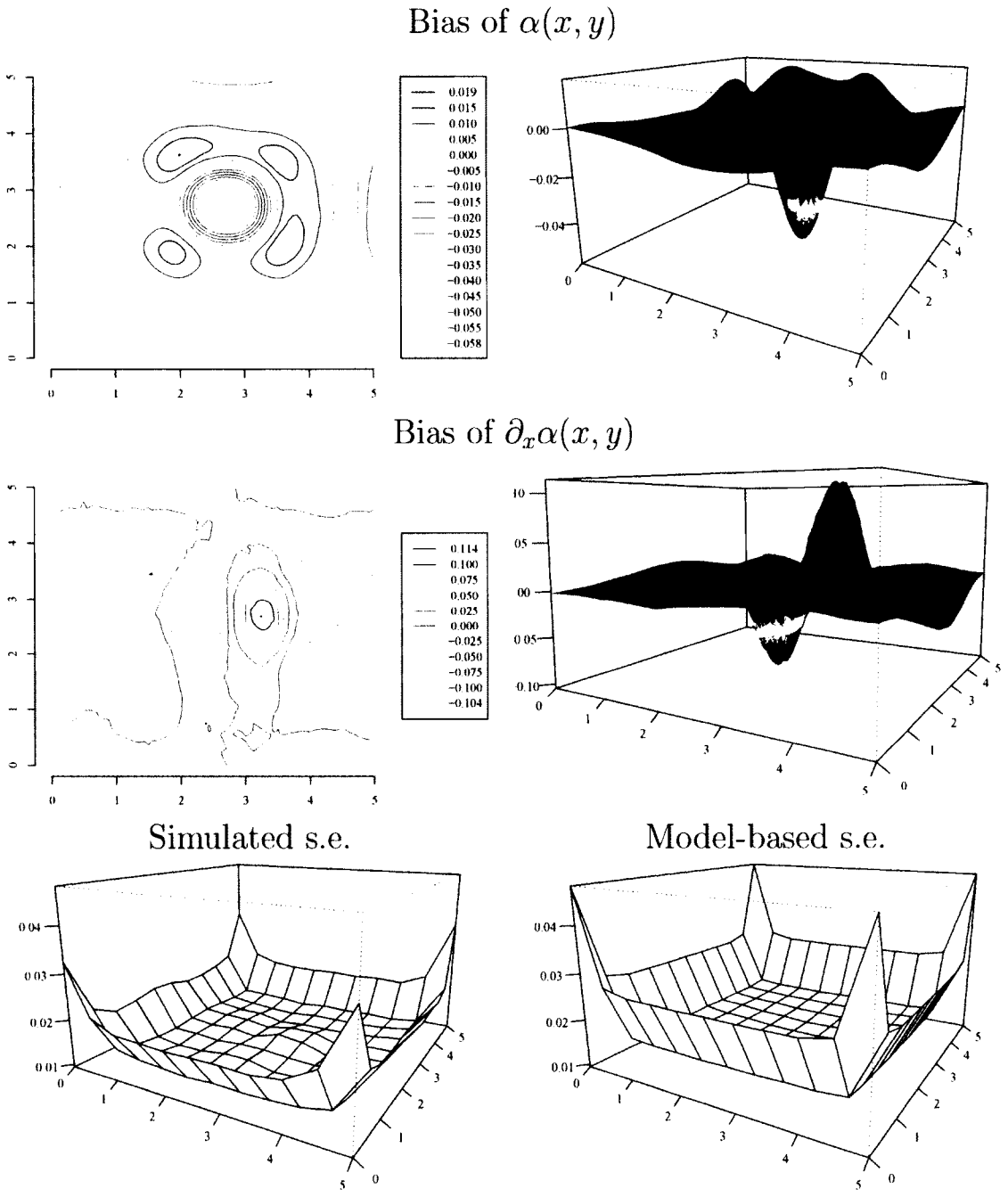

Figure 32: Surfaces and contour plots for the 2d event locations independent Poisson model estimated using REML for $M = 300$.

with $M = 300$. The results are good and comparable to the panel data case.

3.3.3 The case with frailty

To generate a random sample in this case, we draw M independent random variables ν_i, $i = 1, \ldots, M$, from the $\Gamma\left(\frac{1}{\tau}, \tau\right)$ distribution, after which we draw M independent random variables, N_i, each from Poisson$\left(\nu_i \Lambda_o(\Omega)\right)$ distributions. Lastly, we draw $\sum_{i=1}^{M} N_i$ independent random variables from a distribution with p.d.f. $\lambda_o(s)/\Lambda_o(\Omega)$ and assign them as event locations to each of the M individuals according to their corresponding total counts N_i.

The one dimensional case

We solved six test cases with $M = 25, 50, 100$ and $\tau = 0.01, 0.2$. Each test was replicated $B = 1000$ times. The case of $\tau = 0.01$ produced results that are nearly identical to the case without frailty presented above. The case of $\tau = 0.2$ is shown in figures 33 and 34, and similar to the panel data case, we see a slight increase in the bias and the standard errors as a result of the larger variance in the sample. The τ histograms for $M = 100$ and both values of τ are in figure 35. Here, as in the panel data case, the estimates for τ are the same for both REML and GCV.

The two dimensional case

In the last simulation test we solved the 2-d cases with $M = 100$ and 300 for $\tau = 0.1$. Similar to what we saw in the panel data case, the results here are nearly identical to the results in the 2d event times Poisson model. The histogram of τ is shown in figure 36.

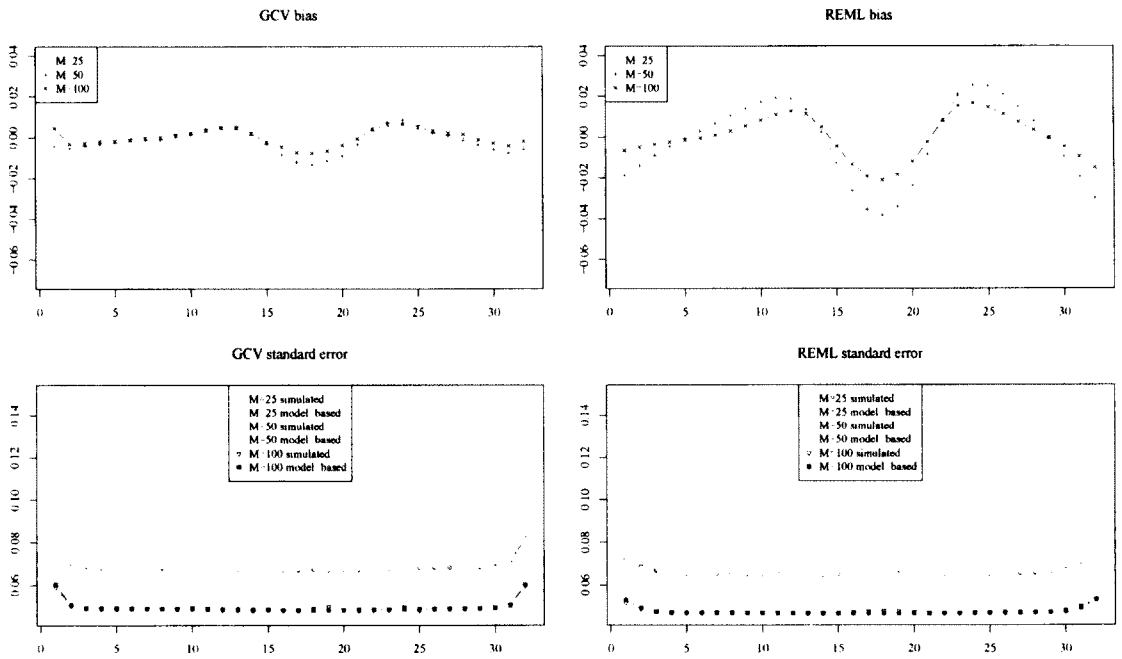

Figure 33: Comparison of bias and standard error for 1d event times negative binomial model with exact $\tau = 0.2$. Finite element interpolation has $n = 64$ degrees of freedom in each case. Each test case was replicated $B = 1000$ times.

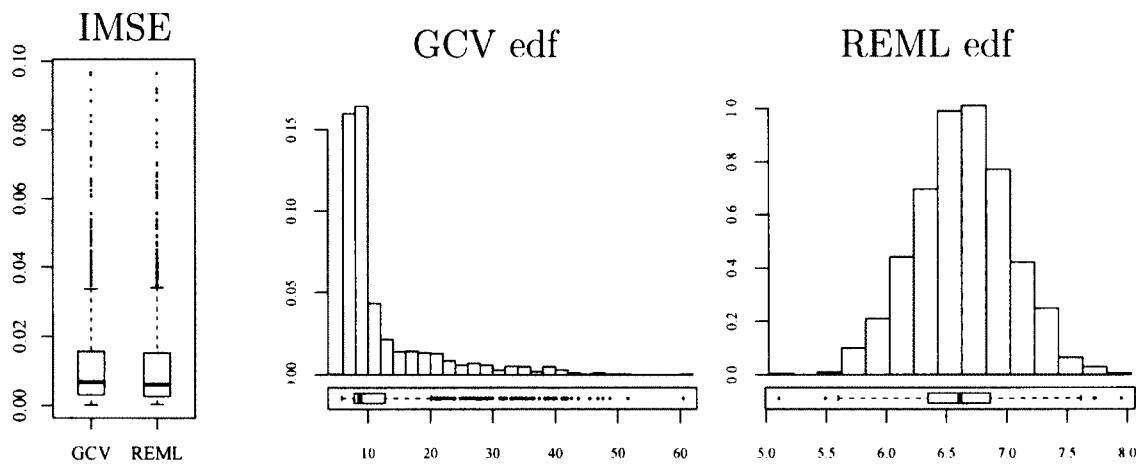

Figure 34: Histograms and boxplots of the distributions of edf for GCV and REML in the test case of $M = 100$ for 1-d event times negative binomial model with exact $\tau = 0.2$.

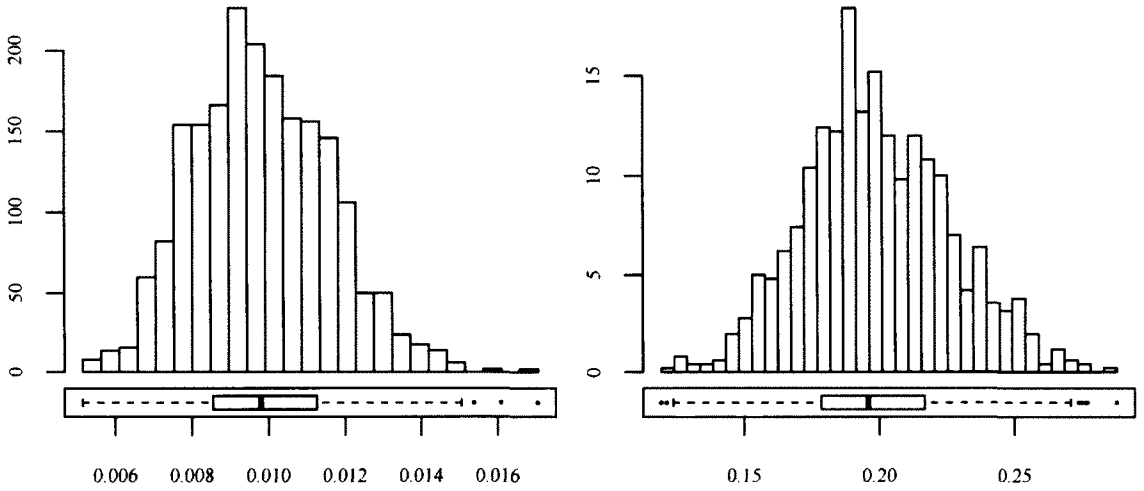

Figure 35: Histograms of estimated τ in 1d event times case with exact $\tau = 0.01$ (left) and exact $\tau = 0.2$ (right)

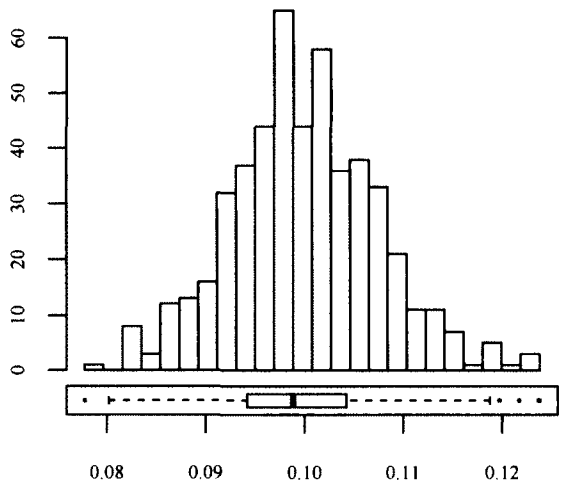

Figure 36: Histogram of τ for 2d mixed Poisson model for event locations data. Exact $\tau = 0.1$

3.4 Pennsylvania lung cancer data

As a real world application, we analyzed a dataset available in the SpatialEpi package on CRAN. The data comprises numbers of lung cancer cases in the 67 counties of Pennsylvania in 2002, stratified into 16 groups according to race, gender and age. The race variable has two values 'white' and 'non-white', while the age groups are four: 'under 40', '40 to 59', '60 to 69' and 'over 70'. The dataset also includes population sizes for each group in each county and geographic information describing the borders of the state and the counties. The map of Pennsylvania with the county borders is shown in figure 37

The matrix of observed counts, N_{ij}, has size $M = 16$ by $L = 67$. We use five indicator variables as covariates in our model, i.e. $p = 5$. The first indicates that race='white', the second indicates gender='male' and the third to fifth indicate each of the last three age groups. In this setting the base group is non-white female under 40. The population sizes for different counties and different groups are considerably different from one another, so the intensity function will model the rate per individual of the population. We denote the matrix of population sizes Z_{ij}, which has the same dimensions as the counts N_{ij}. We will use also the notation $\mathbf{Z}_i = \left[Z_{i1}, \ldots, Z_{iL}\right]^T$. In this sample we have just one observation (set of counts) for each of the 16 groups and the differences between the groups will be captured by the covariate effects. For this reason we use the independent Poisson model.

In the model we fit, N_{ij} are independent and have Poisson distributions. We will denote μ_{ij} the rate of lung cancer for a person in group i in county j. This way the expected counts are

$$\mathrm{E}\left[N_{ij}\right] = Z_{ij}\mu_{ij} = Z_{ij}\mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}}\Lambda_o(R_j) \tag{3.20}$$

With this notation, formulas (3.1) and (3.2) for the gradient of J are modified as

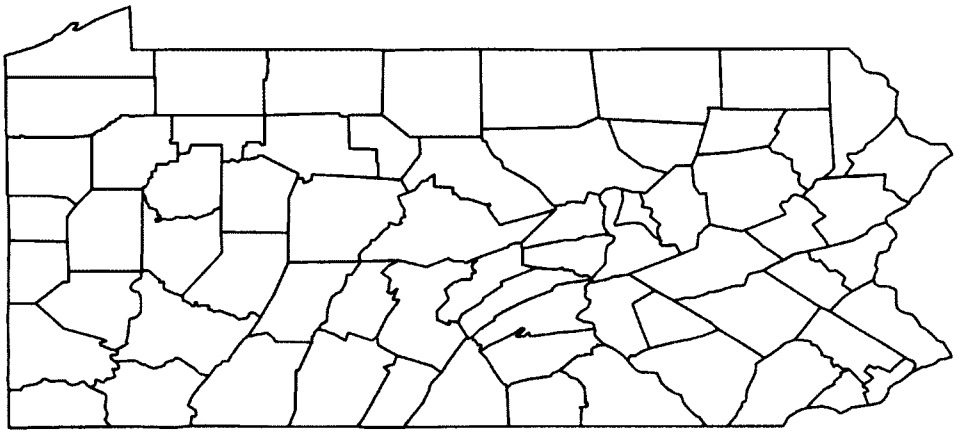

Figure 37: Map of the state of Pennsylvania showing county borders.

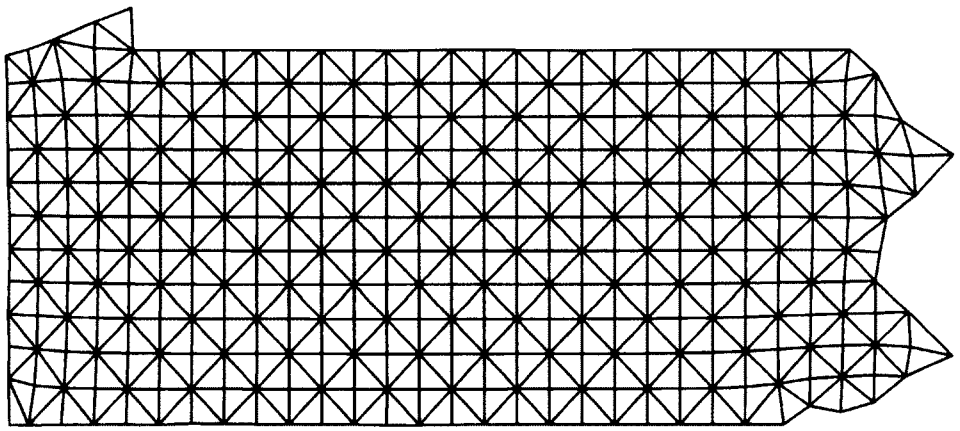

Figure 38: Finite element mesh over the state of Pennsylvania.

follows.

$$\nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M} \big(\operatorname{diag}(\mathbf{Z}_i)\mathbf{D}_{\beta i} \big)^T \mathbf{V}_i^{-1} \big[\mathbf{N}_i - \operatorname{diag}(\mathbf{Z}_i)\boldsymbol{\mu}_i \big] \tag{3.21}$$

$$\nabla_{\boldsymbol{\alpha}} J = \sum_{i=1}^{M} \big(\operatorname{diag}(\mathbf{Z}_i)\mathbf{D}_{\alpha i} \big)^T \mathbf{V}_i^{-1} \big[\mathbf{N}_i - \operatorname{diag}(\mathbf{Z}_i)\boldsymbol{\mu}_i \big] - 2\delta \mathbf{P}\boldsymbol{\alpha} \tag{3.22}$$

The negative expected Hessian in (3.3) is modified similarly, by multiplying each of $\mathbf{D}_{\beta i}$ and $\mathbf{D}_{\alpha i}$ by $\operatorname{diag}(\mathbf{Z}_i)$ on the left. In the three formulas, the variance of $\mathbf{N}_i$ is $\mathbf{V}_i = \operatorname{diag}(\mathbf{Z}_i) \operatorname{diag}(\boldsymbol{\mu}_i)$.

We generated a triangular mesh covering the teritory of the state; it is shown in figure 38. The mesh has 612 triangles and 345 nodes, resulting in a Finite Element interpolation space of dimension 1991. This number of degrees of freedom is larger than necessary to fit the baseline intensity, considering that the whole set of data consists of only 1072 counts. However, we needed this number of triangles in order to reasonably interpolate the state border.

We ran the estimation twice, choosing the penalty parameter by REML and GCV. The former method converged to optimal edf value of 44.9 while the latter gave edf 61.5. The estimated baseline intensities are shown in figures 39 and 40. Table 2 presents the estimated values of $\boldsymbol{\beta}$ and their model-based standard errors. We see that all of the coefficients are significant. According to these estimates, white persons

	REML		GCV	
indicator	β	s.e.	β	s.e
race (white=1)	−0.120530	0.0314080	−0.115797	0.0314024
gender (male=1)	0.536589	0.0198635	0.537303	0.0198467
age 40 to 59	4.126817	0.0371997	4.126610	0.0371921
age 60 to 69	5.664099	0.0355899	5.663278	0.0355792
age over 70	6.149401	0.0328230	6.147698	0.0328138

Table 2: Estimated covariate effects and their standard errors for the Pennsylvania lung cancer data.

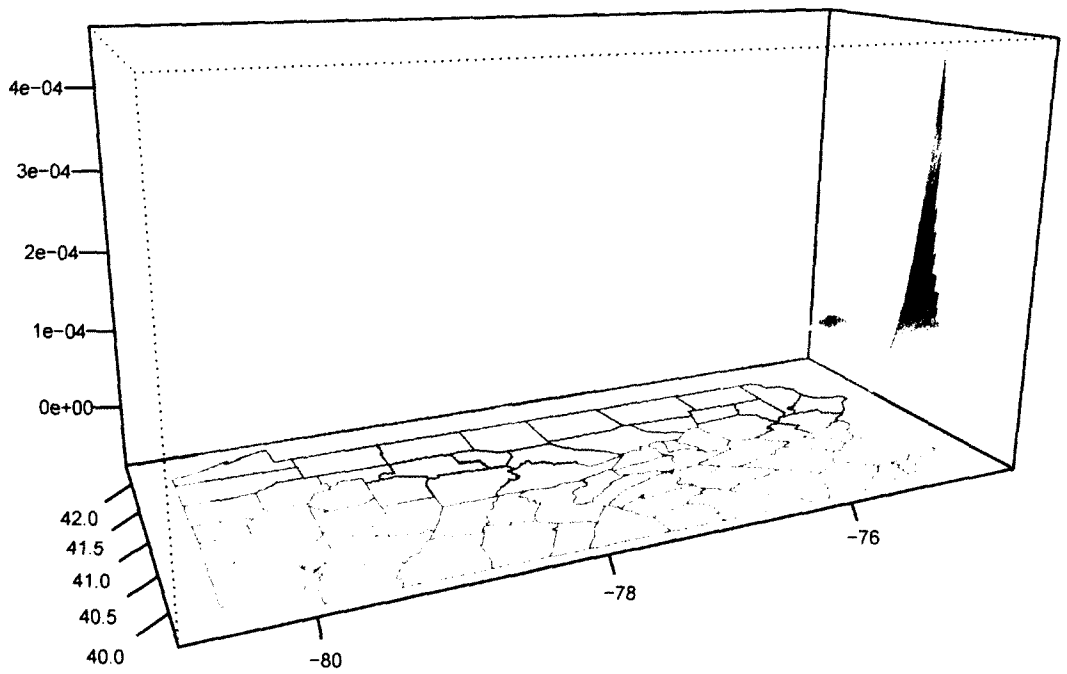

Figure 39: Baseline intensity surface for the Pennsylvania lung cancer data estimated using REML. Optimal edf 44.9.

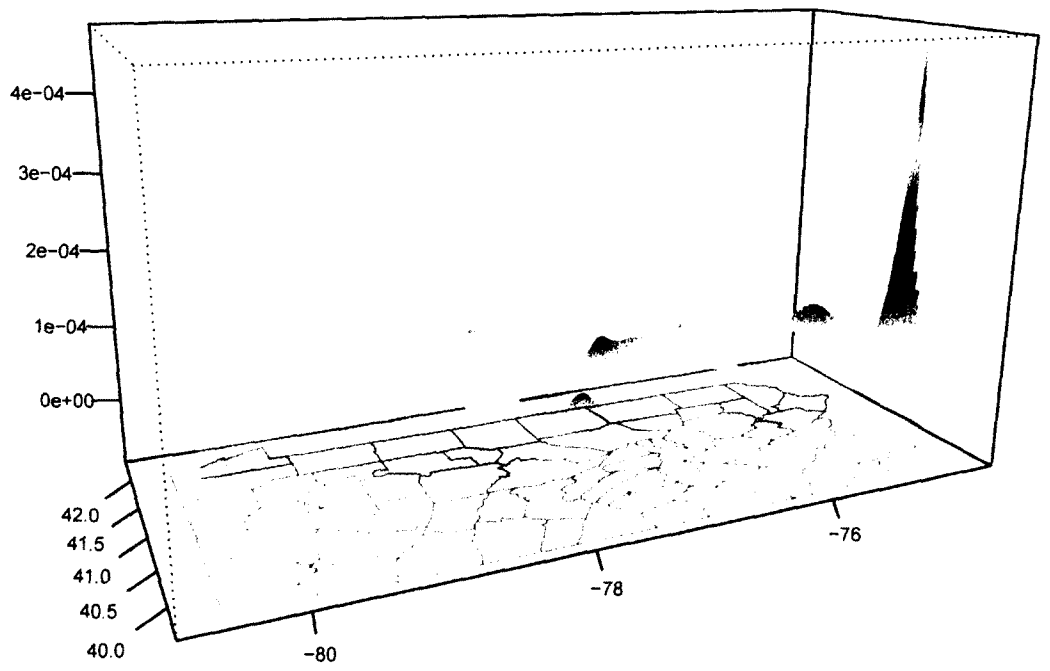

Figure 40: Baseline intensity surface for the Pennsylvania lung cancer data estimated using GCV. Optimal edf 61.5.

have slightly lower rate of lung cancer compared to other races, and males have a higher rate than females. The last three β's show that the rate of lung cancer jumps considerably between the first two age groups and continues to increase with age.

Conclusion

In this thesis we have demonstrated the applicability of the Finite Element Method to the non-parametric estimation of the intensity function of point processes in time and space. The advantages of FEM are particularly evident in two dimensional problems, where it can be applied in a domain of any shape. Furthermore, the case of event locations that we considered herein is in fact a method of density estimation. A natural extension of the presented methodology is its application to density estimation in a general setting, including two-, three-, and even higher dimensions. The FE interpolation can also be used in a similar manner to data smoothing.

It is well known that in data smoothing the bias is usually greater near the boundary. In kernel-based density estimations, the kernel also has to be modified near the boundary, which is a tedious and complicated procedure. If some information about the function is known at the boundary, the FEM can naturally incorporate this information either in the projection matrices or directly into the interpolation basis, potentially eliminating the problem other methods encounter near the boundary. However, this direction needs further investigation.

Another advantage of FEM that could be exploited has to do with the penalty term. In the examples we gave here we penalized the second derivative. However, in a more general setting the penalty could use an arbitrary differential operator, thus defining a prefered space of functions. A penalized regression or penalized maximum likelihood problem can be re-expressed as the solution of a differential equation, where the sum of squared residuals or log-likelihood appear as a non-linear external force

to the differential operator from the penalty term. This approach could also provide some insight into what boundary conditions are appropriate for different situations.

A Derivatives of J

The first and second partial derivatives of (2.24) and (2.25) with respect to τ and $\boldsymbol{\beta}$ are equal, since their difference is a function of $\boldsymbol{\alpha}$ alone. The first and second partial derivatives with respect to τ are found by direct differentiation and are given in equations (2.26) and (2.30). Before proceeding with the $\boldsymbol{\beta}$ derivatives, we point out that

$$\nabla_{\boldsymbol{\beta}}\mu_i = \nabla_{\boldsymbol{\beta}}\Big(\mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}}\Lambda_o(\Omega)\Big) = \mathrm{e}^{\mathbf{x}_i^T\boldsymbol{\beta}}\Lambda_o(\Omega)\mathbf{x}_i = \mu_i\mathbf{x}_i, \tag{A.1}$$

and that the following identity holds

$$\begin{aligned}\frac{1+\tau N_i}{1+\tau\mu_i} &= \frac{\mu_i\big(1+\tau N_i\big)+N_i-N_i}{\mu_i\big(1+\tau\mu_i\big)} = \frac{\mu_i-N_i+N_i\big(1+\tau\mu_i\big)}{\mu_i\big(1+\tau\mu_i\big)} \\ &= \frac{N_i}{\mu_i} - \frac{N_i-\mu_i}{\mu_i+\tau\mu_i^2}.\end{aligned}$$

Using these, the first and second derivatives of J with respect to $\boldsymbol{\beta}$ are

$$\begin{aligned}\nabla_{\boldsymbol{\beta}}J &= \sum_{i=1}^{M}\left[-\left(N_i+\frac{1}{\tau}\right)\frac{\tau\nabla_{\boldsymbol{\beta}}\mu_i}{1+\tau\mu_i}+N_i\mathbf{x}_i\right] = \sum_{i=1}^{M}\left[N_i\mathbf{x}_i-\frac{1+\tau N_i}{1+\tau\mu_i}\mu_i\mathbf{x}_i\right] \\ &= \sum_{i=1}^{M}\left[N_i\mathbf{x}_i-\left(\frac{N_i}{\mu_i}-\frac{N_i-\mu_i}{\mu_{h\,i}+\tau\mu_i^2}\right)\mu_i\mathbf{x}_i\right] = \sum_{i=1}^{M}\left[\frac{N_i-\mu_i}{1+\tau\mu_i}\mathbf{x}_i\right] \\ \nabla_{\boldsymbol{\beta}}\nabla_{\boldsymbol{\beta}}{}^T J &= \sum_{i=1}^{M}\left[\frac{-\nabla_{\boldsymbol{\beta}}\mu_i}{1+\tau\mu_i}\mathbf{x}_i^T-\frac{\big(N_i-\mu_i\big)\tau\nabla_{\boldsymbol{\beta}}\mu_i}{\big(1+\tau\mu_i\big)^2}\mathbf{x}_i^T\right] \\ &= \sum_{i=1}^{M}\left[\frac{-\mu_i}{1+\tau\mu_i}-\frac{\big(N_i-\mu_i\big)\tau\mu_i}{\big(1+\tau\mu_i\big)^2}\right]\mathbf{x}_i\mathbf{x}_i^T\end{aligned}$$

The mixed $\tau\boldsymbol{\beta}$ derivative is

$$\partial_\tau \nabla_{\boldsymbol{\beta}} J = \sum_{i=1}^{M} \left[-\frac{(N_i - \mu_i)\mu_i}{(1+\tau\mu_i)^2} \mathbf{x}_i \right]$$

In the event locations case, the first derivative of J with respect to $\boldsymbol{\alpha}$ is

$$\begin{aligned}
\nabla_{\boldsymbol{\alpha}} J &= \sum_{i=1}^{M} \left[-\left(N_i + \frac{1}{\tau}\right) \frac{\tau \nabla_{\boldsymbol{\alpha}} \mu_i}{1+\tau\mu_i} + \sum_{j=1}^{N_i} \boldsymbol{\Phi}(s_{ij}) \right] - 2\delta \mathbf{P}\boldsymbol{\alpha} \\
&= \sum_{i=1}^{M} \left[\sum_{j=1}^{N_i} \boldsymbol{\Phi}(s_{ij}) - \frac{1+\tau N_i}{1+\tau\mu_i} \nabla_{\boldsymbol{\alpha}} \mu_i \right] - 2\delta \mathbf{P}\boldsymbol{\alpha} \\
&= \sum_{i=1}^{M} \left[\sum_{j=1}^{N_i} \boldsymbol{\Phi}(s_{ij}) - \left(\frac{N_i}{\mu_i} - \frac{N_i - \mu_i}{\mu_i + \tau\mu_i^2} \right) \nabla_{\boldsymbol{\alpha}} \mu_i \right] - 2\delta \mathbf{P}\boldsymbol{\alpha} \\
&= \sum_{i=1}^{M} \left[\frac{N_i - \mu_i}{\mu_i + \tau\mu_i^2} \nabla_{\boldsymbol{\alpha}} \mu_i + \sum_{j=1}^{N_i} \left[\boldsymbol{\Phi}(s_{ij}) - \frac{\nabla_{\boldsymbol{\alpha}} \mu_i}{\mu_i} \right] \right] - 2\delta \mathbf{P}\boldsymbol{\alpha}
\end{aligned}$$

Differentiating the expression in the second line above, we find the second derivative.

$$\begin{aligned}
\nabla_{\boldsymbol{\alpha}} \nabla_{\boldsymbol{\alpha}}{}^T J &= \sum_{i=1}^{M} \left[-\left(1+\tau N_i\right) \frac{(1+\tau\mu_i) \nabla_{\boldsymbol{\alpha}} \nabla_{\boldsymbol{\alpha}}{}^T \mu_i - \nabla_{\boldsymbol{\alpha}} \mu_i (\tau \nabla_{\boldsymbol{\alpha}}{}^T \mu_i)}{(1+\tau\mu_i)^2} \right] - 2\delta \mathbf{P} \\
&= \sum_{i=1}^{M} \left[\frac{(1+\tau N_i)\tau}{(1+\tau\mu_i)^2} \nabla_{\boldsymbol{\alpha}} \mu_i \nabla_{\boldsymbol{\alpha}}{}^T \mu_i - \frac{1+\tau N_i}{1+\tau\mu_i} \nabla_{\boldsymbol{\alpha}} \nabla_{\boldsymbol{\alpha}}{}^T \mu_i \right] - 2\delta \mathbf{P}
\end{aligned}$$

Similarly we derive the first and second derivatives of J with respect to $\boldsymbol{\alpha}$ in the case of panel counts.

$$\begin{aligned}
\nabla_{\boldsymbol{\alpha}} J &= \sum_{i=1}^{M} \left[-\left(N_i + \frac{1}{\tau}\right) \frac{\tau \nabla_{\boldsymbol{\alpha}} \mu_i}{1+\tau\mu_i} + \sum_{j=1}^{L} \frac{N_{ij} \nabla_{\boldsymbol{\alpha}} \mu_{ij}}{\mu_{ij}} \right] - 2\delta \mathbf{P}\boldsymbol{\alpha} \\
&= \sum_{i=1}^{M} \left[\sum_{j=1}^{L} \frac{N_{ij}}{\mu_{ij}} \nabla_{\boldsymbol{\alpha}} \mu_{ij} - \left(\frac{N_i}{\mu_i} - \frac{N_i - \mu_i}{\mu_i + \tau\mu_i^2} \right) \nabla_{\boldsymbol{\alpha}} \mu_i \right] - 2\delta \mathbf{P}\boldsymbol{\alpha} \\
&= \sum_{i=1}^{M} \left[\frac{N_i - \mu_i}{\mu_i + \tau\mu_i^2} \nabla_{\boldsymbol{\alpha}} \mu_i + \sum_{j=1}^{L} \left(\frac{N_{ij}}{\mu_{ij}} - \frac{N_i}{\mu_i} \right) \nabla_{\boldsymbol{\alpha}} \mu_{ij} \right] - 2\delta \mathbf{P}\boldsymbol{\alpha}
\end{aligned}$$

$$\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}J = \sum_{i=1}^{M}\left[-\left(1+\tau N_i\right)\frac{\left(1+\tau\mu_i\right)\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_i - \nabla_{\boldsymbol{\alpha}}\mu_i\left(\tau\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_i\right)}{\left(1+\tau\mu_i\right)^2}\right]$$
$$+\sum_{i=1}^{M}\sum_{j=1}^{L}\left[N_{ij}\frac{\mu_{ij}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_{ij} - \nabla_{\boldsymbol{\alpha}}\mu_{ij}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_{ij}}{\mu_{ij}^2}\right] - 2\delta\mathbf{P}$$
$$= \sum_{i=1}^{M}\left[\frac{\left(1+\tau N_i\right)\tau}{\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\mu_i\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_i - \frac{1+\tau N_i}{1+\tau\mu_i}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_i\right]$$
$$+\sum_{i=1}^{M}\sum_{j=1}^{L}\left[\frac{N_{ij}}{\mu_{ij}}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_{ij} - \frac{N_{ij}}{\mu_{ij}^2}\nabla_{\boldsymbol{\alpha}}\mu_{ij}\nabla_{\boldsymbol{\alpha}}{}^{T}\mu_{ij}\right] - 2\delta\mathbf{P}$$

Finally, the mixed $\tau\,\boldsymbol{\alpha}$ and $\boldsymbol{\beta}\,\boldsymbol{\alpha}$ second derivatives are equal for both functions and are found by differentiating the first τ and $\boldsymbol{\beta}$ derivatives with respect to $\boldsymbol{\alpha}$.

$$\partial_{\tau}\nabla_{\boldsymbol{\alpha}}J = \sum_{i=1}^{M}\left[\left(\frac{1}{\tau^2}\right)\frac{\tau\nabla_{\boldsymbol{\alpha}}\mu_i}{1+\tau\mu_i} - \left(\frac{1+\tau N_i}{\tau}\right)\frac{\left(1+\tau\mu_i\right)\nabla_{\boldsymbol{\alpha}}\mu_i - \mu_i\tau\nabla_{\boldsymbol{\alpha}}\mu_i}{\left(1+\tau\mu_i\right)^2}\right]$$
$$= \sum_{i=1}^{M}\left[\frac{1+\tau\mu_i}{\tau\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\mu_i - \frac{1+\tau N_i}{\tau\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\mu_i\right]$$
$$= \sum_{i=1}^{M}\left[-\frac{N_i-\mu_i}{\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\mu_i\right]$$
$$\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\beta}}{}^{T}J = \sum_{i=1}^{M}\left[\frac{-\nabla_{\boldsymbol{\alpha}}\mu_i\left(1+\tau\mu_i\right) - \left(N_i-\mu_i\right)\tau\nabla_{\boldsymbol{\alpha}}\mu_i}{\left(1+\tau\mu_i\right)^2}\mathbf{x}_i^{T}\right]$$
$$= \sum_{i=1}^{M}\left[-\frac{1+\tau N_i}{\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\mu_i\mathbf{x}_i^{T}\right] = \sum_{i=1}^{M}\left[-\frac{1+\tau N_i}{\left(1+\tau\mu_i\right)^2}\nabla_{\boldsymbol{\alpha}}\nabla_{\boldsymbol{\beta}}{}^{T}\mu_i\right]$$

B Algorithm for minimization

In this appendix we present the computational details of the minimization algorithms used for finding the minimum of GCV.

B.1 Backtracking

The algorithm described here is presented in much detail and analysis in [16, section 3.1].

Suppose $f(x)$ is a function that we want to minimize and that x_c is the current approximation of the minimizer. Let $s = \text{sign}\left(f'(x_c)\right)$. We seek an improved approximation x_n in the form

$$x_n = x_c - s\alpha$$

where α is a positive number called *steplength*. It is clear that if $f(x)$ is continuously differentiable, then there will be an $\alpha^* > 0$ such that $f(x_n) < f(x_c)$ for all $\alpha \in (0, \alpha^*)$. In other words, if we try some steplength α and we find that the function doesn't decrease, this means that we have overshot the minimum and therefore we should try again with a smaller steplength. The backtracking algorithm is an implementation of this idea. For given $\alpha > 0$ and $\tau \in (0, 1)$ we compute

$$x^m = x_c - s\tau^m\alpha, \qquad \text{for } m = 0, 1, 2, \ldots,$$

until the smallest m is found, such that $f(x^m) < f(x_c)$ and $|f'(x^m)| < |f'(x_c)|$. The second inequality is required in order to guarantee that not only the function decreases, but we are actually getting closer to a minimum. Note also that the second inequality will never be satisfied if the objective function is concave. However, in this case $f(x)$ would have no local minima and it's global minimum would be $-\infty$. Clearly this is not the case when minimizing GCV, since GCV is always positive.

B.2 Minimization by fitting a cubic polynomial

Let x_1 and x_2 be two distinct points and let $f_1 = f(x_1)$, $f_2 = f(x_2)$, $p_1 = f'(x_2)$, and $p_2 = f'(x_2)$ be known. The unique cubic polynomial $a(x)$ that satisfies these four

interpolation conditions can be found by setting $a(x) = a_3x^3 + a_2x^2 + a_1x + a_0$ and solving the following linear system for the coefficients.

$$\begin{pmatrix} x_1^3 & x_1^2 & x_1 & 1 \\ x_2^3 & x_2^2 & x_2 & 1 \\ 3x_1^2 & 2x_1 & 1 & 0 \\ 3x_2^2 & 2x_2 & 1 & 0 \end{pmatrix} \begin{pmatrix} a_3 \\ a_2 \\ a_1 \\ a_0 \end{pmatrix} = \begin{pmatrix} f_1 \\ f_2 \\ p_1 \\ p_2 \end{pmatrix}$$

The derivative of $a(x)$ is the quadratic polynomial $a'(x) = 3a_3x^2 + 2a_2x + a_1$. If the discriminant $\Delta = a_2^2 - 3a_3a_1$ is zero or negative, then $a'(x)$ never changes its sign, and therefore $a(x)$ is either always increasing or always decreasing; either way it has no local extrema. In this case the algorithm fails to produce a better approximation of the minimizer. If the discriminant is positive, then $a'(x)$ has two distinct real roots $x_\pm = \frac{a_2 \pm \sqrt{\Delta}}{3a_3}$, one of which gives a local minimum and the other gives a local maximum of $a(x)$. The point that gives a local minimum is the one where the second derivative $a''(x) = 6a_3x + 2a_2$ is positive. We denote this point x_- and recognize that it gives an approximation of the minimizer of $f(x)$.

If $f(x_-) < \min\{f(x_1), f(x_2)\}$ and $|f'(x_-)|$ is sufficiently small then x_- is sufficiently close to the minimizer of $f(x)$ and we may stop the algorithm. Otherwise, we need to set up the next iteration by replacing one of the original points x_1 and x_2 with the new approximation x_-.

An obvious choice may be to keep the two points that give the two lowest values of f. Another obvious choice is to keep the two points that give the two values of f' that are closest to zero. However, we have an even better choice. Notice, that if the values of $f'(x_1)$ and $f'(x_2)$ have different signs, then the interpolant, $a(x)$, is guaranteed to have two local extrema, since $a'(x)$ is a quadratic that changes its sign somewhere between x_1 and x_2 and therefore must the kind of parabola that crosses

the x-axis twice. This justifies our choice to keep x_- and the x_i ($i = 1$ or 2) that gives $f'(x_i)f'(x_-) < 0$. This choice would guarantee that the next iteration of the algorithm will succeed.

Bibliography

[1] A. Andalaft-Chacur, M. M. Ali, and J. G. Salazar. Real options pricing by the finite element method. *Computers & Mathematics with Applications*, 61(9):2863 – 2873, 2011.

[2] P. K. Andersen, Ø. Borgan, R. D. Gill, and N. Keiding. *Statistical models based on counting processes.* Springer Series in Statistics. Springer-Verlag, New York, 1993.

[3] J. H. Argyris. *Energy theorems and structural analysis. A generalised discourse with applications on energy principles of structural analysis including the effects of temperature and non-linear stress-strain relations.* Co-author of Part II, S. Kelsey. Butterworths, London - Toronto - Sydney - Wellington - Durban, 1960.

[4] E. B. Becker, G. F. Carey, and J. T. Oden. *Finite elements. Vol. I.* The Texas Finite Element Series, I. Prentice Hall Inc., Englewood Cliffs, NJ, 1981. An introduction.

[5] S. A. Canann, S. Saigal, and S. J. Owen, editors. *Unstructured mesh generation.* John Wiley & Sons Ltd., Chichester, 2000. Selected papers from the symposium held at the University of Colorado, Boulder, CO, August 4–6, 1999, Internat. J. Numer. Methods Engrg. **49** (2000), no. 1-2.

[6] G. Casella and R. L. Berger. *Statistical inference.* The Wadsworth & Brooks/Cole

Statistics/Probability Series. Wadsworth & Brooks/Cole Advanced Books & Software, Pacific Grove, CA, 1990.

[7] P. G. Ciarlet. *The finite element method for elliptic problems.* North-Holland Publishing Co., Amsterdam, 1978. Studies in Mathematics and its Applications, Vol. 4.

[8] R. W. Clough. The finite element in plane stress analysis. In *Proceedings, 2nd ASCE Conference on Electronic Computations*, Pittsburgh, 1960.

[9] D. R. Cox. Regression models and life-tables. *J. Roy. Statist. Soc. Ser. B*, 34:187–220, 1972. With discussion by F. Downton, Richard Peto, D. J. Bartholomew, D. V. Lindley, P. W. Glassborow, D. E. Barton, Susannah Howard, B. Benjamin, John J. Gart, L. D. Meshalkin, A. R. Kagan, M. Zelen, R. E. Barlow, Jack Kalbfleisch, R. L. Prentice and Norman Breslow, and a reply by D. R. Cox.

[10] C. de Boor. *A practical guide to splines*, volume 27 of *Applied Mathematical Sciences.* Springer-Verlag, New York, revised edition, 2001.

[11] A. Ern and J.-L. Guermond. *Theory and practice of finite elements*, volume 159 of *Applied Mathematical Sciences.* Springer-Verlag, New York, 2004.

[12] P. L. George. *Automatic mesh generation.* John Wiley & Sons Ltd., Chichester, 1991. Application to finite element methods, Translated from the 1990 French original by the author.

[13] C. Gu. *Smoothing spline ANOVA models.* Springer Series in Statistics. Springer-Verlag, New York, 2002.

[14] N. E. Heckman and J. O. Ramsay. Penalized regression with model-based penalties. *Canad. J. Statist.*, 28(2):241–258, 2000.

[15] C. T. Kelley. *Iterative methods for linear and nonlinear equations*, volume 16 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1995. With separately available software.

[16] C. T. Kelley. *Iterative methods for optimization*, volume 18 of *Frontiers in Applied Mathematics*. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1999.

[17] C. T. Kelley. *Solving nonlinear equations with Newton's method*. Fundamentals of Algorithms. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003.

[18] Y. Kutoyants. *Statistical inference for spatial Poisson processes*. Lecture notes in statistics. Springer, 1998.

[19] M.-J. Lai and L. L. Schumaker. *Spline functions on triangulations*, volume 110 of *Encyclopedia of Mathematics and its Applications*. Cambridge University Press, Cambridge, 2007.

[20] J. F. Lawless. Negative binomial and mixed Poisson regression. *Canad. J. Statist.*, 15(3):209–225, 1987.

[21] T. Ramsay. Spline smoothing over difficult regions. *J. R. Stat. Soc. Ser. B Stat. Methodol.*, 64(2):307–319, 2002.

[22] S. M. Ross. *Stochastic processes*. Wiley Series in Probability and Statistics: Probability and Statistics. John Wiley & Sons Inc., New York, second edition, 1996.

[23] L. L. Schumaker. *Spline functions: basic theory*. Cambridge Mathematical Library. Cambridge University Press, Cambridge, third edition, 2007.

[24] D. L. Snyder and M. I. Miller. *Random Point Processes in Time and Space.* Springer-Verlag, 1991.

[25] M. J. Tomas and K. K. Yalamanchili. An application of finite elements to option pricing. *Journal of Futures Markets*, 21(1):19–42, 2001.

[26] G. Wahba. *Spline models for observational data*, volume 59 of *CBMS-NSF Regional Conference Series in Applied Mathematics.* Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 1990.

[27] Y. Yang. Consistency of cross validation for comparing regression procedures. *Ann. Statist.*, 35(6):2450–2473, 2007.

[28] O. C. Zienkiewicz and R. L. Taylor. *The finite element method. Vol. 1.* Butterworth-Heinemann, Oxford, fifth edition, 2000. The basis.