Computational Bayesian Inference Using Low Discrepancy Sequences.

A thesis

submitted in fulfilment

of the requirement for the degree

of

Doctor of Philosophy

at

The University of Waikato

by

Paul Tamataatoi Brown

THE UNIVERSITY OF
WAIKATO
Te Whare Wānanga o Waikato

January 2019

Abstract

The Integrated Nested Laplace Approximation (INLA) provides fast and accurate Bayesian inference for complex hierarchical models. For INLA, and other deterministic methods, the hyperparameter space is explored and points are laid out in a grid structure. These points are used in some numerical integration scheme for which marginal posterior distributions are computed. The main drawback is that the number of points increase exponentially with the number of hyperparameters. The grid is a type of quasi-Monte Carlo (QMC) point set. Low discrepancy sequences (LDS) are QMC point sets that are well known to have significant advantages over grids in terms of convergence and accuracy, and suffer less from the so-called curse of dimensionality.

This work makes several important contributions. We introduce a new method using LDS to compute marginal posterior distributions for hyperparameters, discuss the convergence properties of the approximations and show that they converge to the true posterior. We also show how these methods can be incorporated into the INLA inference framework, and we outline important extensions that improve the accuracy of our approximations with little extra computational effort needed. Lastly, we build a unique spatio-temporal model of residential crime in Hamilton, using INLA's stochastic partial differential equation approach to a Log-Gaussian Cox Process, and use an LDS to approximate the latent parameters of the model.

Our results show that for a fixed number of points or computational time, LDS methods can outperform general grid-based methods, leading to better marginal posterior approximations. Modifying the method for the purposes of incorporation to INLA, we show that we can outperform INLA's grid with respect to computational speed, and obtain accurate and flexible approximations to the model hyperparameters.

Acknowledgements

I would first like to acknowledge and thank my supervisors, Associate Professor Stephen Joe and Dr Chaitanya Joshi, for all their teaching, advice, and most of all, their patience! Stephen has bought a lot of wisdom, humour, and a real eye for detail which has always kept me on my toes. I would really like to thank Chaitanya for really pushing me to go for gold and for believing in me from the start (right from when I was a poor undergraduate), even when I did not think it was possible. Also, thank you to both for providing me the opportunity and funding for conferences in New Zealand and France, and the visit to King Abdullah University of Science and Technology (KAUST).

I would also like to thank all the lecturers and colleagues in the Department of Mathematics and Statistics that helped me and taught me so much throughout my time at Waikato University. I would especially like to thank Bronwyn Poki and the Te Paetata Mãori Student Support Unit for employing me, funding for conferences in Adelaide and Taupo, and giving me the opportunity to work with so many of our Mãori students. Kia ora!

A big thank you to Professor Håvard Rue for collaborating with us and for hosting me during the KAUST visit. I do not think I have ever learnt so much about a topic in such a short amount of time! Also, thank you to the INLA development team, in particular Haakon Baaka and Elias Krainski for their help and advice with the crime models. Thank you to the Waikato Police department and Nigel McCarter for the burglary data, and the Hamilton City Council for the graffiti data.

I would also like to acknowledge all those who supported me financially, the University of Waikato scholarships team, Te Kopua 2B3 Incorporated, Waikato-Tainui, Mãori Education Trust, Ngã Pae o te Mãramatanga, and Mum. Also, to Dr Ngapare

Hopa (Whãea Pare) and Vivian Morelle for their supporting letters, ngã mihi nui ki a kõrua.

To all my friends who started with me, and to those who I have met along the way, thank you for all the support. Special thanks to Gee, Han, Amber, Wise, Josh, and the Math Crew, for all the good times. To the Apiti and Wong whãnau, thank you for feeding me and warmly welcoming me into your family. To my own whãnau, Mum, Dad, Girlie, Maf, Niko, cousins, aunties, uncles, love you all and looking forward to coming home soon! Last, but not least, I would like to thank my fiancee Aotea Apiti for sharing this journey with me. I hope we can enjoy many more together very soon! Arohanui xx

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Bayesian methods are very flexible and widely used in many applications, yet from a computational perspective they can be challenging. The computational burden can be prohibitive when trying to perform inference on complex data and models with hierarchical structures. Increased access to computers in the mid to late 1980's led to an explosion of research in computational Bayesian inference, specifically in Markov chain Monte Carlo (MCMC) methods. Developments in this area continue to this day, though as data has become increasingly big and models increasingly more complex, MCMC methods in many cases can struggle to provide inference in a computationally efficient manner. However, MCMC methods remain very popular and are the most widely used class of methods for performing Bayesian inference.

Developments of alternative methods for Bayesian inference have gained prominence over the years. From Approximate Bayesian Computation (ABC) methods developed in the early 2000's, and the machine learning algorithms such as variational Bayes (VB) and expectation-propagation (EP), to name a few, these methods have gained some popularity amongst users. One such alternative method, the integrated nested Laplace approximation (INLA), was developed for the class of latent Gaussian models (LGM), a widely used class of generalised additive model with a hierarchical

structure. Through its clever use of the properties of Gaussian distributions, Gauss Markov random fields, sparse matrices, and a few other computational 'tricks', it is able to perform accurate inference for LGMs in much less time than MCMC, in some cases, several orders of magnitude faster!

1.1 Motivation

The development of INLA is still ongoing, and is being used extensively for many different statistical problems. However, INLA does have some computational challenges, particularly with LGMs with too many hyperparameters. One reason is that the addition of a hyperparameter in the model can add a large amount of additional latent parameters, thus increasing the amount of computation needed. Another reason is that finding the marginal posteriors for the hyperparameters is challenging and requires enough samples in the hyperparameter space to obtain a satisfactory approximation. Initially, INLA approximated the hyperparameter marginals via a brute force method, by laying out a grid point set and using numerical integration. Whilst grids are fine to use in small dimensions (two or three), in dimensions higher than two or three, they become extremely inefficient. Recent developments by the INLA team have gone a long way to overcoming this hurdle, but these new methods can sacrifice accuracy somewhat. Overall, the development of INLA has highlighted the challenges faced by grid-based Bayesian methods in general.

From a numerical integration perspective, the grid is a type of quasi-Monte Carlo (QMC) point set, and is known for performing poorly in high dimensions. Advancements in the QMC community led to the development of low discrepancy sequences (LDS), and over the years these have been successfully used to perform integration in hundreds, or even thousands of dimensions. We look to formally combine LDS point

sets and Bayesian inference via INLA in an attempt to further increase INLA's computational efficiency.

1.2 Thesis Objectives

We have two main objectives we wish to achieve in this thesis:

1. To explore the possibility of using LDS point sets to perform Bayesian inference. We start by looking at grid-based methods in a general setting and how they can be used to approximate marginal distributions, then look at ways of extending these ideas to build a general algorithm for using LDS point sets for approximations. For numerical integration, LDS point sets have significant computational advantages over grid point sets, thus we are interested in exploring what computational advantages any LDS-based method would have over grid-based methods. These objectives are explored primarily in Chapter 4 of this thesis.

2. To implement LDS-based methods into the general INLA inference framework, and to perform our new methods on a real-world challenging problem. We explore how this can be achieved and what computational advantages we might gain from the implementation. The incorporation of LDS-based methods into INLA is explored in Chapter 5, and implementation on a real-world challenging problem is done in Chapter 6. Parts of this work was done in collaboration with the INLA development team based in King Abdullah University of Science and Technology, Saudi Arabia (previously based out of Norwegian University of Science and Technology, Norway).

1.3 Thesis Outline

Here, we give a brief outline of each chapter in this thesis:

Chapter 2: Computational Bayesian Inference

We provide the reader with some preliminary material on Bayesian inference and hierarchical models. We also introduce the latent Gaussian model. Computational methods of Bayesian inference using sampling based techniques (such as MCMC) are discussed before providing the reader with the details of the INLA methodology.

Chapter 3: Quasi-Monte Carlo Methods

A short preliminary chapter that discusses some aspects of quasi-Monte Carlo integration, including discrepancy, error, and point sets. We introduce the reader to the significant point sets that we have used in the thesis, including the grid point set, the Korobov lattice, and the maximal-rank lattice. We also briefly discuss embedded and extensible lattices.

Chapter 4: Marginal Posterior Estimation using Low Discrepancy Sequences

We give some preliminary material on numerical integration for solving an $(s-1)$-dimensional integral with an s dimensional point set, and also introduce the orthogonal projection matrix that we make extensive use of for our approximations. We then introduce grid-based methods of Bayesian inference before introducing our own LDS-based methods. The LDS method makes use of a least-squares polynomial. We first prove that a weighted least-squares polynomial is equivalent to using an ordinary least-squares polynomial. We present convergence theorems and proofs that the LDS method with a least-squares polynomial converges to the true marginal, and converges much faster than grid-based methods.

Chapter 5: Incorporation of LDS Methods and the INLA Methodology

This chapter provides details about how we can incorporate LDS-based methods into the whole INLA methodology. We first introduce the reader to the so-called autore-

gressive (AR) model, before providing details and code about how to perform INLA inference on the autoregressive model of order 1 (known as an AR(1)) model using R. The LDS method described in the previous chapter may not be suitable for the purposes of INLA, so we propose certain modifications in a new algorithm to make our methods more compatible. Further accuracy gains can be made with an extension that can better account for skewness in the hyperparameters. We perform the full hyperparameter inference on two examples and compare our results to INLA's grid approximations and its default strategy known as the numerical integration free algorithm (NIFA). The results show that our method is superior to INLA's grid approximations in terms of accuracy and computational speed. The results also show that we can obtain more accurate results than the NIFA approximations though NIFA is faster computationally.

Chapter 6: Latent Field Approximations for Continuous Models of Crime Using LDS and INLA

The previous chapters have discussed hyperparameter inference, but recent developments have enabled us to perform inference on the latent parameters too. We give a brief introduction to modelling crime as a spatio-temporal process. We introduce the datasets and the spatial domain before diving into the methodology. The log-Gaussian Cox process (LGCP) is discussed as well as INLA's stochastic partial differential equation (SPDE) approach using a finite element mesh. The stationarity assumption is discussed and we introduce the so-called barrier model, which takes into account physical barriers inside the spatial dependence structure.

We constuct the spatio-temporal model of crime with barriers as an LGCP and use INLA's SPDE methodology to perform inference for all parameters. Using a new development which allows the user to design their own point set for INLA to use, we perform inference on the latent variables using LDS. We are interested in identifying potential future hotspots of crime, so we use the results to perform predictions of crime

hotspots and use standard hotspot predictive analysis to measure these predictions.

Chapter 7: Conclusions and Further Work

We outline our conclusions here and discuss potential further work that may answer any open questions that this thesis has not yet answered.

Chapter 2

Computational Bayesian Inference

We start by giving a brief overview of computational Bayesian inference. Though Bayesian inference is easy in theory, it can be very hard in practice due to computational issues surrounding the calculation of the so-called posterior distribution. This is especially true for models with complex structures. We start with Bayesian inference before introducing Bayesian hierarchical models and look at a specific type of model known as the latent Gaussian model (LGM). We also discuss some of the current computational methods used for performing Bayesian inference. For a full comprehensive look at Bayesian theory, please refer to [62, 68]. For more on Bayesian hierarchical models, see [20, 24].

From here, we move to the computational methods of Bayesian inference. Sampling-based methods are a common approach, the most common being Markov chain Monte Carlo (MCMC). We give details on MCMC before moving on to the integrated nested Laplace approximation (INLA). INLA is a relatively new method of computational Bayesian inference which provides accurate and fast inference for the class of LGMs. We provide details of how the INLA algorithm works, but for the most up-to-date account of INLA's features, see [75]. For a good overview of computational Bayesian inference from an applications perspecive, we refer the reader to [27].

2.1 Bayesian Inference

Probability models are used in statistical analyses to summarise an observed dataset $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ through a set of s unknown parameters $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_s)$. The Bayesian approach treats the unknown parameters as random variables with some probability distribution or "degree of belief". Let $\pi(\mathbf{y}|\boldsymbol{\theta})$ be the likelihood function, a function that gives the relative probabilities to all possible values of $\boldsymbol{\theta}$ that comes from the data. We also set a prior probability distribution $\pi(\boldsymbol{\theta})$ to represent our (subjective) beliefs of the values of $\boldsymbol{\theta}$ prior to any data being observed. The posterior distribution, denoted by $\pi(\boldsymbol{\theta}|\mathbf{y})$, represents the updated beliefs from our prior distribution after evidence from the data has been taken into account. The posterior distribution is found through Bayes' theorem

$$\pi(\boldsymbol{\theta}|\mathbf{y}) = \frac{\pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta})}{\pi(\mathbf{y})}. \tag{2.1}$$

The posterior is sometimes expressed in its unnormalised form,

$$\begin{aligned} \pi(\boldsymbol{\theta}|\mathbf{y}) &\propto \pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta}), \\ \pi(\boldsymbol{\theta}|\mathbf{y}) &\propto prior \times likelihood. \end{aligned} \tag{2.2}$$

The denominator in (2.1) is the marginal likelihood, and is obtained by integrating the numerator of (2.1) with respect to $\boldsymbol{\theta}$ over the support Θ

$$\pi(\mathbf{y}) = \int_{\Theta} \pi(\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\theta}) d\boldsymbol{\theta}, \tag{2.3}$$

and the value of $(\pi(\mathbf{y}))^{-1}$ is the proportionality constant.

The posterior distribution is a summary of everything we now believe about the parameter values. Bayesian point estimation, interval estimation and inferential procedures are all performed through the posterior. To perform Bayesian inference on an

individual parameter θ_i, we must marginalise the posterior by integrating out all other parameters. Let $\boldsymbol{\theta}_{-i}$ denote all parameters except θ_i. We have

$$\pi(\theta_i|\mathbf{y}) = \int_{\Theta \setminus \Theta_i} \pi(\boldsymbol{\theta}|\mathbf{y}) d\boldsymbol{\theta}_{-i}, \tag{2.4}$$

where $\Theta \setminus \Theta_i$ is the support after the exclusion of the i^{th} marginal space. For a comprehensive look at Bayesian inference, including prior and likelihood choices, see [6] and [10].

The Bayesian approach to statistics has its roots in the 18th century. Bayesian statistics enjoys many advantages over frequentist methods, such as the ability to add extra available knowledge through the prior distribution and the ability to make probability statements about the parameters. However, one of its big disadvantages is computational. In very few cases, this s-dimensional integral in (2.3) has a closed form solution. For most cases however, (2.3) does not have a closed form solution and must be approximated. Methods exist where one can approximate the posterior with a Gaussian distribution. Otherwise, some type of numerical integration strategy must be used. Since the 1980s, there have been many algorithms designed to approximate the posterior distribution, and to this day remains an open research topic. We discuss several methods of computing posteriors in the upcoming sections.

2.1.1 Bayesian Hierarchical Models

Bayesian hierarchical models are statistical models structured in several different stages that estimate parameters under the Bayesian paradigm. We have a stage of latent parameters between observed data at the bottom stage, and the parameters governing the process at the top stage (called the hyperparameters). This type of modelling can be used when we have data from different populations that we believe have the same distribution, but may have different parameter values. Observations $\mathbf{y} = \{y_1, \ldots, y_n\}$

may be arranged in K groups $\{g_1, \ldots, g_K\}$. Observations are no longer regarded as independent as individuals from one group would have similar traits to each other, but different traits to those from another group. This would indicate hidden factors that cause dependencies within each group.

We define the terminology similar to that of [86], where we have three stages. The observation stage is defined by data conditioned on all latent parameters $\boldsymbol{\phi}$ and hyperparameters $\boldsymbol{\theta}_1$. The latent process stage is defined by latent parameters $\boldsymbol{\phi}$ given hyperparameters $\boldsymbol{\theta}_2$ and the hyperparameter stage, defined by hyperparameters $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$. The model has the following form

$$\begin{aligned} \text{Hyperparameter stage} &: \boldsymbol{\theta} \sim \pi(\boldsymbol{\theta}), \\ \text{Latent process stage} &: \boldsymbol{\phi} \sim \pi(\boldsymbol{\phi}|\boldsymbol{\theta}_2), \\ \text{Observation stage} &: \mathbf{y} \sim \pi(\mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta}_1), \end{aligned} \tag{2.5}$$

where $\pi(\mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta}_1)$ is the conditional likelihood given latent parameters $\boldsymbol{\phi}$ and hyperparameters $\boldsymbol{\theta}_1$, $\pi(\boldsymbol{\phi}|\boldsymbol{\theta}_2)$ is the conditional distribution of latent parameters $\boldsymbol{\phi}$ given hyperparameters $\boldsymbol{\theta}_2$, and $\pi(\boldsymbol{\theta})$ is the joint distribution of the hyperparameters. This three-stage hierarchical Bayes model has a joint density that is the product of the three stages

$$\pi(\mathbf{y}, \boldsymbol{\phi}, \boldsymbol{\theta}) = \pi(\boldsymbol{\theta})\pi(\boldsymbol{\phi}|\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta}).$$

Applying Bayes' theorem, we have

$$\pi(\boldsymbol{\phi}, \boldsymbol{\theta}|\mathbf{y}) = \frac{\pi(\boldsymbol{\theta})\pi(\boldsymbol{\phi}|\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta})}{\int_{(\Theta,\Phi)} \pi(\boldsymbol{\theta})\pi(\boldsymbol{\phi}|\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta}) d(\boldsymbol{\theta}, \boldsymbol{\phi})}.$$

Marginalisation for both latent and hyperparameter posteriors can be computed via

integration:

$$\pi(\phi_i|\mathbf{y}) = \int \pi(\phi_i|\boldsymbol{\theta}, \boldsymbol{y})\pi(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta},$$
$$\pi(\theta_i|\mathbf{y}) = \int \pi(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}_{-i}.$$

The integrals cannot usually be solved analytically. We introduce a specific Bayesian hierarchical model known as latent Gaussian models, which generally describe all the models studied throughout this thesis. In upcoming sections we discuss how the integrals above can be approximated, and how inference can be performed on this model.

2.1.2 Latent Gaussian Models

Latent Gaussian models (LGM) are a subclass of so-called structured additive regression models, and are widely used in statistical applications [24]. Let the response variable $\mathbf{y} = \{y_1, \ldots, y_{n_d}\}$ belong to the exponential family, where the mean μ_i is linked to a structured additive predictor η_i through some link function $g(\cdot)$ such that $g(\mu_i) = \eta_i$. This predictor η_i accounts for the different covariates additively. A structured additive regression model is given by,

$$\eta_i = \alpha + \sum_{j=1}^{n_f} f_{(j)}(u_{ji}) + \sum_{k=1}^{n_\beta} \beta_k z_{ki} + \epsilon_i. \tag{2.6}$$

The α term represents the overall mean, $f_{(j)}(\cdot)$ are unknown functions of the covariates $\mathbf{u}_i$, the β_k's represent linear effects of the known covariates $\mathbf{z}_i$, and the ϵ_i's are the error terms. Let $\boldsymbol{\phi}$ be a vector of all unknown latent parameters $\{\eta_i, \alpha, f_{(j)}(\cdot), \beta_k\}$. Note that we include the η_i's rather than the ϵ_i's in $\boldsymbol{\phi}$. Next, we define the latent Gaussian model.

Definition 2.1.1. *A latent Gaussian model (LGM), is a structured additive regression model* (2.6)*, where all latent parameters $\boldsymbol{\phi}$ are assigned Gaussian priors.*

Let $\boldsymbol{\theta}$ be the vector of hyperparameters, with $\dim(\boldsymbol{\theta}) = s$. These hyperparameters

may or may not be Gaussian. We can also describe the LGM using a hierarchical structure,

$$\begin{aligned}
\text{Hyperparameter stage} &: \boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) \sim \pi(\boldsymbol{\theta}), \\
\text{Latent process stage} &: \boldsymbol{\phi}|\boldsymbol{\theta}_2 \sim \pi(\boldsymbol{\phi}|\boldsymbol{\theta}_2) = \mathcal{N}(\boldsymbol{\phi}; \mu(\boldsymbol{\theta}_2), \mathbf{Q}^{-1}(\boldsymbol{\theta_2})), \\
\text{Observation stage} &: \mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta}_1 \sim \pi(\mathbf{y}|\boldsymbol{\phi}, \boldsymbol{\theta}_1) = \prod_{i=1}^{n_d} \pi(y_i|\phi_i, \boldsymbol{\theta}_1). \qquad (2.7)
\end{aligned}$$

The bottom stage is the observation stage, which is formed by the likelihood function of the n_d independent observations $\mathbf{y}$, given the latent parameters $\boldsymbol{\phi}$ and possible hyperparameters $\boldsymbol{\theta}_1$. For the middle stage (the latent process stage), the conditional distribution of the latent parameters, given hyperparameters $\boldsymbol{\theta}_2$, is also known as the *latent Gaussian field*. As such, it has a multivariate Gaussian distribution with $\mu(\boldsymbol{\theta}_2)$ a mean vector, and covariance matrix (denoted here as the inverse precision matrix) $\mathbf{Q}^{-1}(\boldsymbol{\theta}_2)$. The top stage is the hyperparameter model (or hyperpriors), which are the prior distribution for the hyperparameters. The joint posterior for the unknowns $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ is given by:

$$\begin{aligned}
\pi(\boldsymbol{\phi}, \boldsymbol{\theta}|\mathbf{y}) &\propto \pi(\theta)\pi(\boldsymbol{\phi}|\boldsymbol{\theta}) \prod_{i=1}^{n_d} \pi(y_i|\phi_i, \boldsymbol{\theta}) \\
&\propto \pi(\boldsymbol{\theta})|\mathbf{Q}(\boldsymbol{\theta})|^{n_d/2} \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T \mathbf{Q}(\boldsymbol{\theta})\boldsymbol{\phi}\right) \exp\left(\sum_{i=1}^{n_d} \log(\pi(y_i|\phi_i, \boldsymbol{\theta}))\right) \\
&\propto \pi(\boldsymbol{\theta})|\mathbf{Q}(\boldsymbol{\theta})|^{n_d/2} \exp\left(-\frac{1}{2}\boldsymbol{\phi}^T \mathbf{Q}(\boldsymbol{\theta})\boldsymbol{\phi} + \sum_{i=1}^{n_d} \log(\pi(y_i|\phi_i, \boldsymbol{\theta}))\right).
\end{aligned}$$

Applications of LGMs are numerous due to the additive structure and the flexibility and different forms that the unknown functions $f_{(j)}(\cdot)$ can take.

We revisit parameter estimation and Bayesian inference for LGMs in upcoming sections. Various different methods can do this, though one method in particular is specifically designed to perform inference in an efficient way under this setting.

2.2 Bayesian Computing

Although Bayesian theory has been known and developed for over two centuries, *doing* Bayesian inference was (and still can be) hard due to computational issues surrounding the integration of various quantities. For many years, Bayesian inference was limited only to a very small class of problems. This was when the likelihood is assumed to be from the exponential family of distributions, and a corresponding prior was chosen from the conjugate family. See [9, 10] for more on conjugate priors.

Since the 1980s, there have been many developments in the field of Bayesian computing. This section highlights sampling-based methods for performing Bayesian inference, starting with Monte Carlo methods before moving into Markov Chain Monte Carlo (MCMC) methods. Sampling-based methods can be computationally intensive but have become more popular as computers have become more powerful - to the point where they are the most widely used methods in Bayesian inference.

2.2.1 Monte Carlo Integration

Suppose we wish to calculate the integral over some domain D,

$$I = \int_D g(x) f(x)\, dx, \tag{2.8}$$

where $g(x)$ is some function of the random variable X and $f(x)$ is the density function of X. We can approximate the integral numerically by generating a random sample of values $\{x_{(1)}, x_{(2)}, \ldots, x_{(n)}\}$ drawn independently and identically (iid) from $f(\cdot)$ and calculating the empirical mean

$$\hat{I} = \frac{1}{n} \sum_{i=1}^{n} g(x_{(i)}). \tag{2.9}$$

The strong law of large numbers [5, 35] dictates that, as $n \to \infty$, $\hat{I}$ converges almost surely to the true value I, that is

$$\frac{1}{n}\sum_{i=1}^{n} g(x_{(i)}) \xrightarrow{a.s} \int_D g(x)f(x)dx = I. \tag{2.10}$$

The estimator $\hat{I}$ is an unbiased estimator, and the accuracy of the approximation is inversely proportional to the square root of n. For more details of the properties of this estimator, refer to [70].

We know that Bayesian inference revolves around the posterior distribution of the parameter θ (for simplicity, assume here that θ is univariate). We know the posterior distribution up to a normalising constant (2.2). The posterior mean is given by

$$E(\theta|\mathbf{y}) = \int_\Theta \theta\, \pi(\theta|\mathbf{y})\, d\theta.$$

We can also calculate the posterior mean of a function $g(\cdot)$ with

$$E(g(\theta)|\mathbf{y}) = \int_\Theta g(\theta)\, \pi(\theta|\mathbf{y})\, d\theta.$$

The integral above has the same form of (2.8), with the density function $f(x)$ being the posterior distribution $\pi(\theta|\mathbf{y})$. Assuming we can sample from the posterior, we can use Monte Carlo integration to find the posterior mean, by setting $g(\theta) = \theta$ and generating an iid random sample $\{\theta_{(1)}, \theta_{(2)}, \ldots, \theta_{(n)}\}$ directly from $\pi(\theta|\mathbf{y})$ and finding the empirical mean similar to (2.9).

This is straightforward when $\pi(\theta|\mathbf{y})$ is a known distribution. When we do not have that information, indirect Monte Carlo sampling methods such as Sampling-Importance Resampling can be used. We choose not to provide details about these methods, but interested readers may refer to [30, 69] for an overview of all these types of methods.

These methods become very inefficient for high dimensional parameter space however, and are only recommended for $s \leq 2$.

2.2.2 Markov Chain Monte Carlo

The Markov Chain Monte Carlo (MCMC) method is currently the dominant method of performing Bayesian inference. MCMC is a sampling-based method that generates a correlated sample from the posterior distribution from which all inferences can be made. Given that MCMC is a popular method and the surrounding theory is very well known, we very briefly outline MCMC and discuss some variants that are currently used today.

A random or stochastic process is a process that evolves over an index set, governed by some probabilistic law. It is a collection of (dependent) random variables $\{X^{(t)}, t \in T\}$ for some index set T. The possible values that a stochastic process can take are collectively called the state space of the process, denoted by χ. A stochastic process with state space χ is a Markov chain if, for any $A \subset \chi$

$$q(X^{(t)} \in A | X^{(0)}, X^{(1)}, \ldots, X^{(t-1)}) = q(X^{(t)} \in A | X^{(t-1)}).$$

The equation above states that the probability distribution of $X^{(t)}$ conditioned on all past variables depends only on the variable at the last state $X^{(t-1)}$. The conditional probability distribution $q(X^{(t)} \in A | X^{(t-1)})$ is called the transition probability.

An MCMC algorithm generates (correlated) values from a chosen target distribution π, typically the posterior, by drawing values from a Markov chain that has a long-run stationary distribution equal to π. As π is the stationary distribution, if $X^{(t)} \sim \pi$, then $X^{(t+1)} \sim \pi$, so once the chain reaches π, the distribution of $X^{(t)}$ does not change with t. For a discrete state space, the Markov chain must satisfy certain properties in order

for us to obtain a unique stationary distribution, namely irreducibility, aperiodicity, and positive recurence. A Markov chain satisfying these three properties is called an *ergodic* Markov chain. It has a unique stationary distribution π that characterises the behaviour of the chain after it runs for a sufficient period of time (referred to as the burn-in time), regardless of the starting value. The ergodicity theorem states that the empirical average of the ergodic Markov chain converges almost surely to the expected value $E(g(X))$ for $l \to \infty$:

$$\frac{1}{l}\sum_{t=1}^{l} g(X^{(t)}) \xrightarrow{a.s} \int_{\chi} g(x)\pi(x)dx = E(g(X)).$$

The ergodicity theorem is similar to the law of large numbers used for iid MC samples, but differs by assuming non-independent correlated samples. For more on the Ergodic theorem and proofs, see [32]. For more on Markov chains with continuous state space, see [82].

In practice, the target distribution π is set to be the posterior distribution $\pi(\theta|\mathbf{y})$ which is known up to the proportionality constant. We generate a sequence of l values $\{\theta_{(1)}, \ldots, \theta_{(b)}, \theta_{(b+1)}, \ldots, \theta_{(l)}\}$ from a proposed Markov kernel $q(\cdot|\cdot)$, where value $\theta_{(b)}$ is the point at which the chain is considered to have converged to the target. This value is found via various tests and diagnosis plots, such as traceplots. The values $\{\theta_{(b)}, \theta_{(b+1)}, \ldots, \theta_{(l)}\}$ are considered to be a correlated, random sample from the posterior distribution. Although we have the ergodicity theorem, methods such as thinning can be used to reduce the dependency of the sample. All inferences are then made with the final sample.

The most basic MCMC algorithm is the Metropolis-Hastings algorithm [33, 59], which provides the starting point for the more advanced MCMC algorithms. Given a set of parameters $\boldsymbol{\theta} = \{\theta_1, \ldots, \theta_s\}$, we give a simple version of the Metropolis Hast-

ings method in Algorithm 1, where all parameters $\boldsymbol{\theta}$ are evaluated at once.

Algorithm 1 Metropolis-Hastings algorithm

Start at value $\boldsymbol{\theta}_{(0)}$
for $t = 1, \ldots, l$ **do**
  Generate $\boldsymbol{\theta}'$ from a proposal $q(\cdot|\boldsymbol{\theta}_{(t-1)})$
  Compute the acceptance probability $p_t = \min\left(1, \frac{\pi(\boldsymbol{\theta}')q(\boldsymbol{\theta}_{(t-1)}|\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta}_{(t-1)})q(\boldsymbol{\theta}'|\boldsymbol{\theta}_{(t-1)})}\right)$
  Generate $u_t \sim U(0, 1)$
  if $u_t \leq p_t$ **then**
    $\boldsymbol{\theta}_{(t)} = \boldsymbol{\theta}'$
  else
    $\boldsymbol{\theta}_{(t)} = \boldsymbol{\theta}_{(t-1)}$

The choice of proposal q gives flexibility to the algorithm, but this also determines its performance. Strategies such as the random walk, independent sampler and the Gibbs sampler are widely used and all have their advantages and disadvantages. For instance, the random walk is very simple and easy to implement, but usually takes a long time to move through the parameter space. The Gibbs sampler, first introduced by the authors of [28], uses a blocking structure $\boldsymbol{\theta} = (\boldsymbol{\theta}_k, \boldsymbol{\theta}_{-k})$ and the proposal density for each block of parameters is the true conditional density given all parameters outside that block and the data. The acceptance probability will always be 1 and every new value $\boldsymbol{\theta}'$ will be accepted. Despite this, the Gibbs sampler will move slowly around the parameter space if the parameters in the blocks are highly correlated. Since the introduction of these methods, many different variants of these MCMC algorithms have been developed with the aim of increasing their computational speed and accuracy. However, given all the developments, for large datasets and complex models such as LGMs they can be computationally burdensome. Reasons for this include the potentially high correlation between components of the latent field, and the dependency of hyperparameters and latent parameters when the number of latent parameters is large. For this reason and more, great emphasis has been placed on developing MCMC algorithms that are more computationally efficient through the choice of proposal q.

The Metropolis-adjusted Langevin algorithm (MALA) [71] is a variant of the random walk Metropolis Hastings algorithm and uses a stochastic differential equation (SDE) instead of a random walk to move through the parameter space. MALA relies on the gradient and the second derivative of the (unnormalised) log-posterior to make proposals that move into areas of the parameter space with higher density. There is a large body of empirical evidence that at the extra price of computing the gradients, MALA provides a substantial speed-up in convergence on certain types of problems [31]. MALA can also be used on some LGMs successfully, for instance in [19] and [81]. However, instability can occur if the posterior is light-tailed, see [12], and thus the construction of this algorithm needs care. Improvements and extensions of MALA exist, such as manifold-MALA, simplified manifold-MALA and position-dependent MALA, all made to improve the stability issues.

Many other MCMC methods exist, such as Hamiltonian (or hybrid) Monte-Carlo, adaptive MCMC, particle MCMC, and are widely used. We choose not to cover these here but interested readers can refer to [31] and its corresponding references for a complete look at these methods. Over the last decade, deterministic approaches such as variational Bayes [51] and expectation-propagation [57] approaches have become especially prominent in machine learning. In the next section, we provide an in-depth look at a particular deterministic method, designed specifically for performing fast Bayesian inference on LGMs.

2.3 Integrated Nested Laplace Approximations

Section 2.2 gave a brief overview of sampling-based methods for Bayesian inference. These methods relied on generating a random sample from the posterior, and inferences made using the sample. A downfall of these methods is they can be computationally ex-

pensive. A methodology, called the Integrated Nested Laplace Approximation (INLA), developed by the authors of [74] and designed specifically for LGMs, is a deterministic algorithm and uses Laplace approximations and a set of carefully selected points to approximate posterior distributions. It has been shown to be orders of magnitudes faster than many MCMC algorithms, whilst providing accurate results. It also has the advantage of having the ability to easily yield model comparison quantities and various predictive measures for full Bayesian analyses. Since its initial development, it has become widely used on a very wide range of applications (see [56] and [75] for a comprehensive — but not full — list).

The starting point for INLA is the Laplace approximation. We start this section with an overview of the Laplace approximation before briefly discussing its relationship with the LGM. A more in-depth look at inference and the algorithm are given later.

2.3.1 Laplace's Method

Laplace's method [83] is used as a technique to approximate the posterior distribution analytically. Suppose we wish to approximate the integral

$$\int f(x)dx$$

where $f(x)$ is some twice-differentiable function, and has a unique global maximum. We can represent $\log(f(x))$ through a Taylor series expansion of the second order around $x = x_0$:

$$\log(f(x)) \approx \log(f(x_0)) + (x - x_0)\left.\frac{d\log(f(x))}{dx}\right|_{x=x_0} + \frac{(x-x_0)^2}{2!}\left.\frac{d^2\log(f(x))}{dx^2}\right|_{x=x_0}$$

Let $x^* = \text{argmax}_x \log(f(x))$ be the mode and set $x_0 = x^*$. Then the first derivative of the above becomes zero, and the approximation becomes

$$\log(f(x)) \approx \log(f(x^*)) + \frac{(x-x^*)^2}{2!} \left. \frac{d^2 \log(f(x))}{dx^2} \right|_{x=x^*}.$$

We can rewrite $f(x)$ as $\exp(\log(f(x))$ and approximate the integral,

$$\begin{aligned}
\int f(x)dx &= \int \exp(\log(f(x))dx \\
&\approx \int \exp\left(\log(f(x^*)) + \frac{(x-x^*)^2}{2!} \left. \frac{d^2 \log(f(x))}{dx^2} \right|_{x=x^*}\right) dx \\
&= \exp(\log(f(x^*)) \int \exp\left(\frac{(x-x^*)^2}{2!} \left. \frac{d^2 \log(f(x))}{dx^2} \right|_{x=x^*}\right) dx.
\end{aligned}$$

We can see that the integrand looks similar to that of a Gaussian distribution. If we set $\sigma^{2*} = \left(\left. \frac{d^2 \log(f(x))}{dx^2} \right|_{x=x^*} \right)^{-1}$ we get

$$\int f(x)dx \approx f(x^*) \int \exp\left(-\frac{1}{2\sigma^{2*}}(x-x^*)^2\right) dx.$$

The integrand above is the kernel of a Gaussian distribution with mean x^* and variance σ^{2*}. The integral $\int f(x)dx$ on the interval (α, β) can be approximated by

$$\int_\alpha^\beta f(x)dx \approx f(x^*)\sqrt{2\pi\sigma^{2*}}(\Phi(\beta) - \Phi(\alpha)), \tag{2.11}$$

where $\Phi(\cdot)$ is the cumulative density function of the Gaussian distribution with mean x^* and variance σ^{2*}.

Note that Laplace's method can easily be extended out to the multivariate case, where $\mathbf{x} = (x_1, \ldots, x_s)$ is an s-dimensional vector, and

$$\left. \frac{d^2 \log(f(\mathbf{x}))}{d\mathbf{x}^2} \right|_{\mathbf{x}=\mathbf{x}^*}$$

gives a matrix of second derivatives, or Hessian matrix.

2.3.2 INLA and LGMs

Section 2.1.2 gave an introduction to the LGM. We attach a Gaussian prior to all latent parameters $\boldsymbol{\phi}$, and assume that the latent field is Gaussian (see (2.7)). All LGMs we consider for the INLA method have two main properties.

- **Assumption 1**: The latent Gaussian field admits conditional independence properties, that is:

  $$\phi_i \perp \phi_j | \phi_{-(i,j)} \iff Q_{ij} = 0.$$

  Here, $\perp$ denotes independence, and $\phi_i \perp \phi_j | \phi_{-(i,j)}$ states that ϕ_i and ϕ_j are conditionally independent, given $\phi_{-(i,j)}$.

  The statement above says that the conditional independence property implies that the latent Gaussian field is a Gauss-Markov random field (GMRF) with a sparse precision matrix. This is a big key to INLA's fast computational time, as sparse matrices are much faster to deal with than dense covariance matrices. For details of GMRFs and the innovative algorithms that can be used to deal with sparse matrices, see [72].

- **Assumption 2**: The dimension of $\boldsymbol{\theta}$ (i.e, the number of hyperparameters in the model) must be "small".

  Recommendations are for the dimension to be typically around two to five. Models exist where the hyperparameter dimension is six or more, and work has been done by the INLA team to come up with solutions to increase this number, see

[56] and [75]. New recommendations are that under certain conditions, the number of hyperparameters can be up to, but not higher than 20.

2.3.3 INLA Inference

Given an LGM, INLA performs Bayesian inference on all latent parameters and hyperparameters via computation of their posterior marginals

$$\pi(\phi_i|\mathbf{y}) = \int \pi(\phi_i|\boldsymbol{\theta},\mathbf{y})\pi(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta} \quad i = 1,\ldots,n_\phi$$
$$\pi(\theta_j|\mathbf{y}) = \int \pi(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}_{-j} \quad j = 1,\ldots,s,$$

where n_ϕ are the number of latent parameters. These integrals cannot be solved analytically, so the posterior marginals above are approximated by

$$\tilde{\pi}(\phi_i|\mathbf{y}) = \int \tilde{\pi}(\phi_i|\boldsymbol{\theta},\mathbf{y})\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta} \quad i = 1,\ldots,n_\phi \tag{2.12}$$
$$\tilde{\pi}(\theta_j|\mathbf{y}) = \int \tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})d\boldsymbol{\theta}_{-j} \quad j = 1,\ldots,s. \tag{2.13}$$

We denote $\tilde{\pi}$ as an approximation to π.

INLA makes use of both Laplace's Method (Section 2.3.1), and the Laplace approximation, as shown in [83]. The approximation is based on finding a marginal distribution by dividing out by the conditional distribution rather than through integration of a joint distribution. For example, let the joint distribution of x and z be $\pi(x,z)$. Then

$$\begin{aligned}\pi(x,z) &= \pi(x|z)\pi(z)\\ \pi(z) &= \frac{\pi(x,z)}{\pi(x|z)}.\end{aligned}$$

The equations above tell us we can find the marginal $\pi(z)$ by dividing the joint distribution by the conditional distribution $\pi(x|z)$, rather than integrating out x. This leads

to the identity

$$\tilde{\pi}(z) \propto \frac{\pi(x,z)}{\tilde{\pi}(x|z)}, \tag{2.14}$$

where the conditional distribution $\tilde{\pi}(x|z)$ is approximated with a Gaussian distribution, and $\tilde{\pi}(z)$ is the approximate marginal distribution of z.

The process for approximating the posterior marginals in (2.12) and (2.13) starts with approximating the joint posterior of the hyperparameters by the Laplace approximation, using (2.14), and performing numerical integration techniques to integrate out $\boldsymbol{\theta}_{-\boldsymbol{j}}$ for $\tilde{\pi}(\theta_j|\mathbf{y})$. The next task is to compute $\tilde{\pi}(\phi_i|\boldsymbol{\theta},\mathbf{y})$, which along with $\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})$ can be used to find latent posterior marginals $\tilde{\pi}(\phi_i|\mathbf{y})$ through numerical integration.

The approximation of the joint posterior of the hyperparameters is as follows:

$$\begin{aligned}
\pi(\boldsymbol{\theta}|\mathbf{y}) &= \frac{\pi(\boldsymbol{\phi},\boldsymbol{\theta}|\mathbf{y})}{\pi(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})} \\
&= \frac{\pi(\boldsymbol{\phi},\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\phi},\boldsymbol{\theta})}{\pi(\mathbf{y})}\frac{1}{\pi(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})} \\
&\propto \frac{\pi(\boldsymbol{\theta})\pi(\boldsymbol{\phi}|\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\phi},\boldsymbol{\theta})}{\pi(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})} \\
&\approx \left.\frac{\pi(\boldsymbol{\theta})\pi(\boldsymbol{\phi}|\boldsymbol{\theta})\pi(\mathbf{y}|\boldsymbol{\phi},\boldsymbol{\theta})}{\tilde{\pi}_G(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})}\right|_{\boldsymbol{\phi}=\boldsymbol{\phi}^*(\boldsymbol{\theta})} \\
&= \tilde{\pi}(\boldsymbol{\theta}|\mathbf{y}).
\end{aligned} \tag{2.15}$$

The quantity $\pi(\mathbf{y})$ is the marginal likelihood, giving the proportionality sign in the third line of (2.15). The fourth line of (2.15) approximates $\pi(\boldsymbol{\theta}|\boldsymbol{\phi},\mathbf{y})$ with a Gaussian $\tilde{\pi}_G(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})$, using the multivariate version of Laplace's method (described in Section 2.3.1), and $\boldsymbol{\phi}^*(\boldsymbol{\theta})$ is the mode for a given configuration of $\boldsymbol{\theta}$. Due to $\pi(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})$ being a priori distributed like a GMRF, the approximation $\tilde{\pi}_G(\boldsymbol{\phi}|\boldsymbol{\theta},\mathbf{y})$ is sufficiently accurate [74]. Posterior marginals $\tilde{\pi}(\theta_j|\mathbf{y})$ can then be found by numerical integration. We ex-

plain various methods to do this in upcoming sections.

The next task is more involved, due to the computation of $\tilde{\pi}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ and the dimension of $\boldsymbol{\phi}$ being generally much larger than $\boldsymbol{\theta}$. Three methods are available to compute $\tilde{\pi}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$.

1. **Gaussian Approximation**: The first, and computationally fastest, is to compute them directly from the Gaussian approximation $\tilde{\pi}_G(\boldsymbol{\theta}|\boldsymbol{\phi}, \mathbf{y})$, where all that is left is to compute the marginal variances using the identity found in [80]. Although fast, this is the least accurate method due to potential errors in location and/or skewness [73].

2. **Laplace Approximation**: A more accurate approximation is to treat $\boldsymbol{\phi}$ as the vector $\boldsymbol{\phi} = (\phi_i, \boldsymbol{\phi}_{-i})$ and use another Laplace approximation as in (2.14),

$$\begin{aligned}
\pi(\phi_i|\boldsymbol{\theta}, \mathbf{y}) &= \frac{\pi((\phi_i, \boldsymbol{\phi}_{-i})|\boldsymbol{\theta}, \mathbf{y})}{\pi(\boldsymbol{\phi}_{-i}|\phi_i, \boldsymbol{\theta}, \mathbf{y})} \\
&= \frac{\pi(\boldsymbol{\phi}, \boldsymbol{\theta}|\mathbf{y})}{\pi(\boldsymbol{\theta}|\mathbf{y})} \frac{1}{\pi(\boldsymbol{\phi}_{-i}|\phi_i, \boldsymbol{\theta}, \mathbf{y})} \\
&\propto \frac{\pi(\boldsymbol{\phi}, \boldsymbol{\theta}|\mathbf{y})}{\pi(\boldsymbol{\phi}_{-i}|\phi_i, \boldsymbol{\theta}, \mathbf{y})} \\
&\approx \left. \frac{\pi(\boldsymbol{\phi}, \boldsymbol{\theta}|\mathbf{y})}{\tilde{\pi}_G(\boldsymbol{\phi}_{-i}|\phi_i, \boldsymbol{\theta}, \mathbf{y})} \right|_{\boldsymbol{\phi}_{-i} = \boldsymbol{\phi}^*_{-i}(\phi_i, \boldsymbol{\theta})} \\
&= \tilde{\pi}_{LA}(\phi_i|\boldsymbol{\theta}, \mathbf{y}),
\end{aligned}$$

   where $\tilde{\pi}_{LA}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ is the Laplace approximation of $\pi(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ and $\boldsymbol{\phi}^*_{-i}(\phi_i, \boldsymbol{\theta})$ is the mode for a given configuration of ϕ_i and $\boldsymbol{\theta}$. The conditional distribution $\pi(\boldsymbol{\phi}_{-i}|\phi_i, \boldsymbol{\theta}, \mathbf{y})$ is approximated by $\tilde{\pi}_G(\boldsymbol{\phi}_{-i}|\phi_i, \boldsymbol{\theta}, \mathbf{y})$ using Laplace's method described in Section 2.3.1.

   This strategy is the most accurate. However, because this strategy needs to recompute $\tilde{\pi}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ for every value of $\boldsymbol{\phi}_i$ and $\boldsymbol{\theta}$, it is the most computationally

prohibitive. Modifications exist to ease the computational burden (see [74]) but this is regarded as the most expensive strategy.

3. **Simplified Laplace Approximation**: This new approximation method performs a third-order Taylor series expansion on $\tilde{\pi}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$. This allows for the correction to the location and skewness to the Gaussian approximation via the skew-normal distribution [3]. For details of the simplified Laplace approximation computations, see [74]. This approximation is sufficiently accurate and fast, and is the default strategy for the approximation of $\pi(\phi_i|\boldsymbol{\theta}, \mathbf{y})$.

With the computation of $\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})$ and $\tilde{\pi}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ complete, $\tilde{\pi}(\phi_i|\mathbf{y})$ in (2.12) is found numerically through a finite weighted sum

$$\tilde{\pi}(\phi_i|\mathbf{y}) \approx \sum_k \tilde{\pi}(\phi_i|\boldsymbol{\theta}^{(k)}, \mathbf{y})\tilde{\pi}(\boldsymbol{\theta}^{(k)}|\mathbf{y})\Delta_k \tag{2.16}$$

for some chosen integation points $\boldsymbol{\theta}^{(k)}$ with a corresponding set of weights Δ_k. The choice of integration points are the main focus of the next section.

2.3.4 INLA Algorithm

The mechanics within INLA to compute the necessary approximations are performed in several parts:

Exploration of $\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})$: Rather than represent the joint posterior parametrically, INLA searches the hyperparameter space and selects a set of suitable evaluation points $\{\boldsymbol{\theta}^{(k)}\}$ for the numerical integration in (2.16). Note that the hyperparameter space is usually reparameterised in order to deal with more regular densities. Two different strategies are used to choose the points, and both require the following steps:

1. Find the mode $\boldsymbol{\theta}^*$ of $\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})$ by optimising its log density with respect to $\boldsymbol{\theta}$ through some quasi-Newton method.

2. At the mode $\boldsymbol{\theta}^*$, compute the negative Hessian matrix $\mathbf{H}$.

3. Let $\boldsymbol{\Sigma} = \mathbf{H}^{-1}$. Compute the eigen-decomposition $\boldsymbol{\Sigma} = \mathbf{V}\boldsymbol{\Lambda}^{1/2}\mathbf{V}^T$, where $\mathbf{V}$ is the matrix of eigenvectors and $\boldsymbol{\Lambda}$ is the diagonal matrix of eigenvalues. Then, define a new variable $\mathbf{z}$ with standardised and orthogonal components, such that

$$\boldsymbol{\theta}(\mathbf{z}) = \boldsymbol{\theta}^* + \mathbf{V}\boldsymbol{\Lambda}^{1/2}\mathbf{z}.$$

4. Explore $\log(\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y}))$ through the $\mathbf{z}$-parameterisation to locate the bulk of the probability density. There are two exploration strategies, the grid strategy and the central composite design (CCD).

   (i) The grid strategy starts with a point at the mode ($\mathbf{z} = \mathbf{0}$), and goes in the positive direction of the first axis z_1 with some step length δ_z and continues as long as

$$\log(\tilde{\pi}(\boldsymbol{\theta}(\mathbf{0})|\mathbf{y})) - \log(\tilde{\pi}(\boldsymbol{\theta}(\mathbf{z})|\mathbf{y})) < \delta_\pi, \tag{2.17}$$

   where δ_π is the maximum log-density difference at which we stop taking points. Then we switch in the opposite direction and repeat the process. This is done for all axes $(z_1, \ldots, z_s)$. We then fill in all the intermediate values by taking all different combinations of the points already chosen so long as (2.17) holds. This builds up a set of point laid out in a type grid structure where all points are evenly spaced, and the weights Δ_k in (2.16) are taken to be equal.

   (ii) The CCD approaches the integration problem as a design problem. Starting with the mode $\boldsymbol{\theta}^*$ and Hessian $\mathbf{H}$, relevant points in the $\boldsymbol{\theta}$-

space are selected for performing a second-order approximation to a response variable (see Section 6.5 of [74] for full details of the design).

Approximating $\pi(\theta_j|\mathbf{y})$: After selecting suitable evaluation points $\{\boldsymbol{\theta}^{(k)}\}$ using either the grid or CCD, INLA re-uses these points along with an interpolation algorithm to approximate the hyperparameter marginals. For the grid, INLA takes the mean of the function evaluations over all rows and columns of the grid and fits an interpolant (spline) through the mean of the function evaluations (we discuss this more in Chapter 4). An interpolating scheme called the asymmetric Gaussian interpolation is used with the CCD points. Details of this scheme can be found in [56], but we will not discuss this scheme further.

Also available is a numerical integration free algorithm (NIFA), a method which bypasses numerical integration completely. For NIFA, the following structure of $\tilde{\pi}(\theta_i|\mathbf{y})$ is assumed

$$\tilde{\pi}(\theta_i|\mathbf{y}) = \begin{cases} N(0, \sigma^2_{i+}), \theta_i > 0 \\ N(0, \sigma^2_{i-}), \theta_i \leq 0. \end{cases} \tag{2.18}$$

Thus, the posterior is assumed to be Gaussian with different variances on each side of the mode. The computation of σ^2_{i+}, and σ^2_{i-}, for all $i = 1, \ldots, s$ is explained in [75]. This strategy is the most computationally efficient since it requires no numerical integration. However, the assumption that the marginal posteriors of $\boldsymbol{\theta}$ are unimodal is a drawback. Due to its computational speed, NIFA is currently the default method in INLA for the computation of hyperparameter marginals. We explore this more with some examples in Chapter 5.

Approximating $\pi(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ and $\pi(\phi_i|\mathbf{y})$: For a chosen strategy to compute $\tilde{\pi}(\phi_i|\boldsymbol{\theta}, \mathbf{y})$ (Gaussian, Laplace or simplified Laplace), for each point in $\{\boldsymbol{\theta}^{(k)}\}$, the posteriors

$\tilde{\pi}(\phi_i|\boldsymbol{\theta}^{(k)}, \mathbf{y})$ are evaluated on a grid of selected values for ϕ_i. The latent posterior marginals $\pi(\phi_i|\mathbf{y})$ are computed via numerical integration as in (2.16).

2.4 Discussion

Latent Gaussian models are an important and large class of models. The older and more vanilla MCMC methods struggle with this model, due to the dependency of the parameters of the latent field, and also the strong dependence of $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$. INLA has been designed specifically for inference on LGMs and use various techniques and properties of the Gaussian to overcome a lot of the computational burden. However, there has been some success with the more modern MCMC algorithms such as MALA. In fact, in a few cases, there have been claims that there is little difference in the performance between the two algorithms, and a case where MALA supposedly outperforms INLA [81]. However, the authors of INLA do not see their method as a rival for MCMC, rather an alternative method for certain models where a fast approximation is preferred, rather than a slow "exact" method.

INLA does have its limitations. First, the system will break down if the latent field $\boldsymbol{\phi}|\boldsymbol{\theta}$ is not unimodal. Whilst this is a drawback, multimodal posteriors are generally very hard to compute using any method, including MCMC. Secondly, the number of hyperparameters must be small. The grid strategy is the most accurate method to compute $\pi(\theta_j|\mathbf{y})$, but in higher dimension this requires many more points as the computational cost grows exponentially with the number of hyperparameters. Whilst the CCD and NIFA strategies have gone a long way to increasing the number of hyperparameters by decreasing the number of integration points needed, it is at the expense of some accuracy for the computation of $\tilde{\pi}(\theta_j|\mathbf{y})$. In later chapters we look at alternatives to grid-based sampling methods, and using alternative deterministic point sets to potentially increase computational benefits.

Chapter 3

Quasi-Monte Carlo Methods

This chapter will give a brief overview of the topics in Quasi-Monte Carlo (QMC) methods that we use throughout the thesis. We introduce QMC integration and discuss several properties of low discrepancy point sets, before looking into the construction of the specific point sets that we will use.

3.1 Quasi-Monte Carlo Integration

Recall that Section 2.2.1 described Monte Carlo (MC) integration as applied to Bayesian inference. We discuss this briefly in a more general and multivariate setting and a slight change in notation. We wish to estimate

$$I_f = \int_{[0,1)^s} f(\mathbf{u})\, d\mathbf{u}, \tag{3.1}$$

where $\mathbf{u} = (u_1 \ldots, u_s)$ is an s-dimensional vector in the unit hypercube $[0,1)^s$, with $f : [0,1)^s \to \mathbb{R}$ a real-valued function. We can draw a random sample of independent and identically (iid) distributed values and approximate (3.1) with the estimator

$$\hat{I}_N = \frac{1}{N} \sum_{i=1}^{N} f(\mathbf{u}_{(i)}), \tag{3.2}$$

where the points $\mathcal{P}_N = \{\mathbf{u}_{(1)}, \ldots, \mathbf{u}_{(N)}\}$ are sampled over $[0,1)^s$. The estimator is an unbiased estimator and converges almost surely to the true value as $N \to \infty$. Also, the central limit theorem tells us that

$$\frac{\hat{I}_N - I_f}{\sigma_f/\sqrt{N}} \xrightarrow{d} \mathcal{N}(0,1) \tag{3.3}$$

where $\xrightarrow{d}$ represents convergence in distribution and σ_f is the standard deviation of f. We can construct confidence intervals for the estimator $\hat{I}_N$ of the form

$$\left(\hat{I}_N \pm z_{\alpha/2}\frac{\hat{\sigma}}{\sqrt{N}}\right) \tag{3.4}$$

where $z_{\alpha/2}$ is the $100(1-\alpha/2)$th percentile of the standard normal distribution, and $\hat{\sigma}$ is the sample standard deviation found by

$$\hat{\sigma} = \left(\frac{1}{N-1}\sum_{i=1}^{N}(f(\mathbf{u}_{(i)}) - \hat{I}_N))^2\right)^{1/2}. \tag{3.5}$$

The probabilistic error is therefore in $\mathcal{O}(1/\sqrt{N})$. Although this is independent of the dimension s, the rate is considered to be slow. Improvements to the performance can be made in two ways, either by reducing the variance σ_f^2 of the function, or by generating a point set that is more uniformly distributed than the random sample (discussed more in Section 3.1.1). We choose not to discuss variance reduction techniques, but interested readers can consult [47] for more details.

Quasi-Monte Carlo (QMC) integration differs from MC integration by replacing random point sets with deterministic point sets. The aim is to improve efficiency by generating a QMC point set that is more uniform than the random point set. Uniformity of a point set can be measured through the idea of *discrepancy*, which is a measure of the deviation of a point set from the uniform distribution. The next section discusses

the generation of particular QMC point sets that have low discrepancy, known as low discrepancy point sets, or low discrepancy sequences (LDS). Note that, even though a point set is a subset of an infinite sequence, we use the abbreviation LDS for both low discrepancy point sets and low discrepancy sequences since in practice, we use only the finite subset of each sequence.

3.1.1 Discrepancy and Error

Discrepancy is a distance measure of the empirical distribution of a point set and the uniform distribution, calculated via the Kolmogorov-Smirnoff statistic. The most common measure of discrepancy is the *star discrepancy* which we define shortly. First, we consider the set of hyper-rectangles

$$J(\mathbf{v}) = \{\mathbf{u} \in [0,1)^s : 0 \leq u_j < v_j, 1 \leq j \leq s\},$$

where $\mathbf{v} = (v_1, \ldots, v_s) \in [0,1)^s$. Note that these hyper-rectangles have a corner at the origin, and are sometimes referred to anchored hyper-rectangles. Given a point set $\mathcal{P}_N$, we count how many of the points lie within the hyper-rectangle, which we denote as $C(\mathcal{P}_N, \mathbf{v})$. Dividing this quantity by N gives us a measure of the empirical distribution induced by $\mathcal{P}_N$. We compare this quantity with the volume of the hyper-rectangle $\prod_{j=1}^{s} v_j$ through the Kolmogorov-Smirnoff statistic which gives the star discrepancy.

Definition 3.1.1. *The star discrepancy of a point set $\mathcal{P}_N$, denoted by $D^*(\mathcal{P}_N)$, is given by*

$$D^*(\mathcal{P}_N) = \sup_{\mathbf{v} \in [0,1)^s} \left| \prod_{j=1}^{s} v_j - \frac{C(\mathcal{P}_N, \mathbf{v})}{n} \right|. \tag{3.6}$$

A low discrepancy sequence is a sequence of points that has a star discrepancy $D^*(\mathcal{P}_N) = \mathcal{O}(N^{-1}(\log(N))^s)$. An important result in the QMC theory is the Koksma-Hlawka inequality, which gives an upper bound to the integration error.

Definition 3.1.2. *The Koksma-Hlawka inequality states that, for the integration error*

$\epsilon_N = |I_f - \hat{I}_N|$,

$$\epsilon_N \leq V(f)\, D^*(\mathcal{P}_N), \tag{3.7}$$

where $D^*(\mathcal{P}_N)$ is the star discrepancy of the pointset P_N, and $V(f) < \infty$ is the total variation of the function f in the sense of Hardy and Krause.

Hence, for functions with bounded variance, the integration error is bounded above by $\mathcal{O}(N^{-1}(\log(N))^s)$. Comparing this with the MC error of $\mathcal{O}(1/\sqrt{N})$, for fixed dimension s, the QMC error converges faster to zero than the MC error for functions that are sufficiently smooth, and for a point set with sufficiently large N. The error bound however suggests that the accuracy deteriorates as the dimension s increases.

3.2 Low Discrepancy Sequences

Low discrepancy sequences are split into two main families, lattices rules and digital nets/sequences. We have chosen to use lattices in this work, due to their ease of coding and the availability of tables and software tools to generate optimal lattice point sets. For a more in-depth look at lattices, interested readers can refer to [79], and for information on digital nets and sequences see [22, 23]. This section will focus on the rank-1 lattice and point sets related to it, the Korobov lattice and the maximal-rank lattice. We briefly look at the idea of extensible lattices before giving a small introduction to the main software tool we use to generate the lattices.

One extra property of many low discrepancy sequences that we choose to define here, is the fully projection regular property. If a point set $\mathcal{P}_N$ has low discrepancy, it would be good if the projections of $\mathcal{P}_N$ also has low discrepancy, and also that the number of projection points is N. For a given subset $X = (j_1, \ldots, j_d) \subseteq \{1, \ldots, s\}$ of indices, $\mathcal{P}_N(X)$ denotes the d-dimensional point set

$$\mathcal{P}_N(X) = \{(u_{i,j_1}, \ldots u_{i,j_d}), i = 1, \ldots, N\}.$$

We have the following definition

Definition 3.2.1. *A pointset $\mathcal{P}_N$ is Fully Projection Regular if all its projections $\mathcal{P}_N(X)$ contains N unique points.*

This property is important in both the context of LDS in general, and has an affect on the construction of our algorithms in the following chapter.

3.2.1 Lattices

We give a brief introduction to lattice point sets before moving into the particular constructions.

Definition 3.2.2. *Given a dimension s, a lattice point set $\mathcal{P}_N$ is defined by the integration lattice L_s that takes the form*

$$L_s = \{v_1\mathbf{w}_1 + \cdots + v_s\mathbf{w}_s, \mathbf{v} \in \mathbb{Z}^s\},$$

where the vectors $\mathbf{w}_1, \ldots \mathbf{w}_s \in \mathbb{R}^s$ form a linearly independent basis. The lattice point set is obtained by taking all integer linear combinations that lie in the unit hypercube,

$$\mathcal{P}_N = L_s \cap [0,1)^s.$$

Note that different bases $\mathbf{w}_1, \ldots \mathbf{w}_s$ can lead to the same point set.

Let $\mathbf{W}$ be the $s \times s$ matrix whose i^{th} row is $\mathbf{w}_i$. The number of points in P_N can be shown to be $1/det(\mathbf{W})$, where $det(\mathbf{W})$ is the determinant of $\mathbf{W}$ [47]. The components of the basis vectors must be rational numbers, and can be expressed as fractions of the form l/N, where N is the number of points in P_N.

The notion of *rank* r and *invariants* $N_1, \ldots, N_r$ are used to reduce the number of possible bases. Here, r is the smallest integer satisfying three conditions.

1. $N_l | N_{l+1} \ \forall \ l < r$.

2. $N_1 \cdots N_r = N$.

3. $\mathcal{P}_N$ can be written as

$$\mathcal{P}_N = \left\{ \left(\frac{i_1}{N_1} \mathbf{z}_1 + \ldots + \frac{i_r}{N_r} \mathbf{z}_r \right) \right\} \tag{3.8}$$

for some vectors $\mathbf{z}_1, \ldots \mathbf{z}_r \in \mathbb{Z}^s$.

The choice of vectors $\mathbf{z}_1, \ldots \mathbf{z}_r$ is not unique, however the rank and invariants can be uniquely determined. Thus, in the context of parameter searches for good lattice point sets in paractice, we fix the parameters s, n, and r, and search for "good" vectors $\mathbf{z}_1, \ldots \mathbf{z}_r$. We refer to these vectors as *generating vectors*.

Many different types of lattice point sets exist, see [79] for a good overview of the different types of constructions. However, we choose to focus on those that are relevant to this thesis.

Rank-1 Lattice

The rank-1 lattice is a popular lattice point set. It is often referred to as the *method of good lattice points*. We denote the rank-1 lattice as $\mathcal{R}_{N,s}$ which signifies the rank-1 lattice has N points and s dimensions. We define the rank-1 lattice below and give its representation based on (3.8).

Definition 3.2.3. *Given one generating vector $\mathbf{z} = (z_1, \ldots, z_s)$ of s integers, the rank-1 lattice point set, denoted by $\mathcal{R}_{N,s}$, is given by*

$$\mathcal{R}_{N,s} = \left\{ \frac{i-1}{N} (z_1, \ldots, z_s) \ mod \ 1, i = 1, \ldots, N \right\},$$

where the modulo operation is performed component-wise.

The components of the generating vector should be chosen to be relatively prime with N, so $\gcd(z_j, N) = 1$, for $j = 1, \ldots, s$. Choosing a generating vector as such results in the rank-1 lattice being fully projection regular (see Definition 3.2.1).

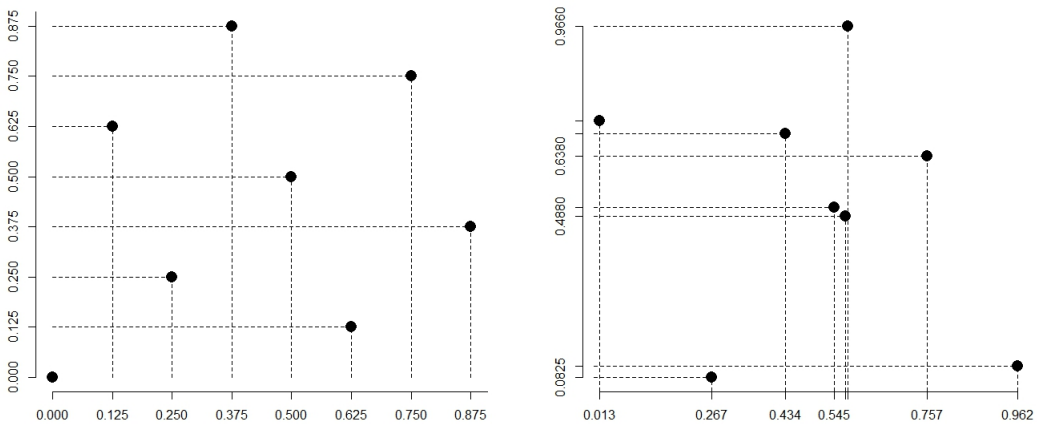

Figure 3.1: Left: A rank-1 lattice $\mathcal{R}_{8,2}$ is an example of a fully projection regular point set. Note that the projections onto each axis are equally spaced, thus the projections are also a low discrepancy sequence. The figure on the right in a random generated point set. It too is fully projection regular, though the projections lead to uneven spacing in each dimension.

Korobov Lattice

The Korobov lattice, proposed by Korobov in [42], is a special case of the rank-1 lattice. For a fixed dimension s and number of points N, we define a generating constant α_z chosen to be an integer, for which $1 \leq \alpha_z \leq N - 1$. We denote the Korobov lattice by $\mathcal{K}_{N,s}$ which signifies the Korobov lattice as having N points in s dimensions. The generating vector for a Korobov lattice is given by

$$\mathbf{z} = (1, \alpha_z, \alpha_z^2, \ldots, \alpha_z^{s-1}), \tag{3.9}$$

We define the Korobov lattice point set.

Definition 3.2.4. *Given the generating vector in* (3.9)*, the Korobov lattice point set*

(hereafter referred to as Korobov Lattice), denoted by $\mathcal{K}_{N,s}$ is given by

$$\mathcal{K}_{N,s} = \left\{ \frac{i-1}{N}(1, \alpha_z, \alpha_z^2, \ldots, \alpha_z^{s-1}) \; \textit{mod}\, 1, i = 1, \ldots, N \right\}. \tag{3.10}$$

Since s and N are predetermined, the only parameter to be chosen is the generating constant α_z. Similar to rank-1 lattices, α_z should be chosen to be relatively prime with N. However, some choices based on this criteria may not always yield desirable point sets. For example, choosing $\alpha_z = 1$ is relatively prime with N, but substituting into (3.10) gives

$$\mathcal{K}_{N,s} = \{(0, \ldots, 0)(1/N, \ldots, 1/N), \ldots, ((N-1)/N, \ldots, (N-1)/N)\}.$$

These points lie on a diagonal line. This is shown in the right hand side plot in Figure 3.2. Though care should be taken when choosing α_z, there are several tables available to determine a good choice of α_z for a given s and N available in the literature. See, for instance [11, 45].

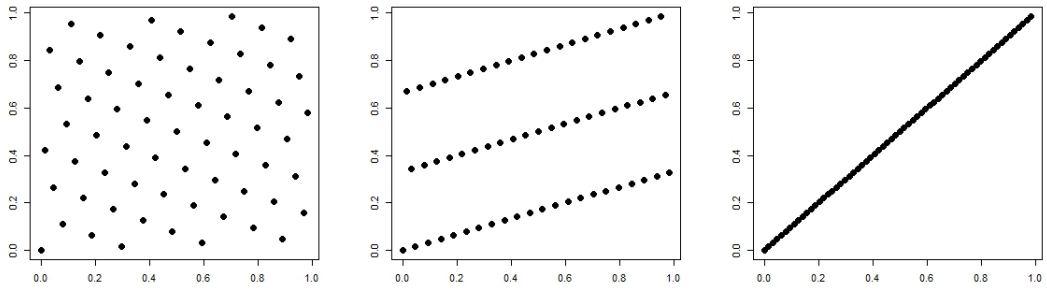

Figure 3.2: Three examples of a two-dimensional Korobov lattice with $N = 64$ points, with $\alpha_z = 27, 43, 1$ respectively.

Maximal-Rank Lattices

The maximal-rank lattice, or rank-s lattice, is a lattice point set with number of generating vectors equal to the dimension s. Maximal-rank lattices are constructed by scaling

a rank-1 lattice and copying it into subcubes over $[0,1)^s$. Dividing each dimension into m partitions leads to m^s subcubes. We define the maximal-rank lattice point set below

Definition 3.2.5. *The maximal-rank lattice point set, denoted by $\mathcal{M}_{n,m,s}$, is given by*

$$\mathcal{M}_{n,m,s} = \left\{ \left(\frac{i-1}{n} \mathbf{z} \bmod 1 \right) + \frac{(k_1, \ldots, k_s)}{m}, \right.$$
$$\left. i = 1, \ldots, n, k_j = 0, \ldots, (m-1),\ j = 1, \ldots, s \right\}, \tag{3.11}$$

where n is the number of points generated in the rank-1 lattice component of (3.11).

The total number of points N generated by the maximal-rank lattice is $n \times m^s$, or number of points generated by the rank-1 lattice times the number of subcubes. An obvious disadvantage of the maximal-rank lattice is that by construction, the point set will not be fully projection regular. Since we are taking m^s copies of a point set, we will have only $m \times n$ distinct points, rather than N. Figure 3.3 shows both the construction of the maximal-rank lattice $\mathcal{M}_{8,2,2}$ (left), and the projections onto each one dimensional axis (right).

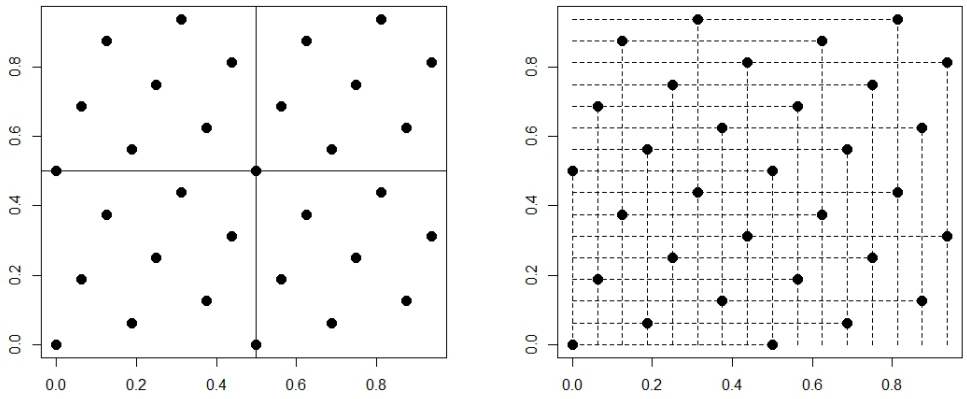

Figure 3.3: A maximal-rank lattice point set $\mathcal{M}_{8,2,2}$. The plot on the left shows the copies of a two-dimensional lattice with 8 points is copied twice in each dimension. The right shows that the maximal-rank lattice is not fully projection regular. There is a total of 64 points, however we only have 32 unique points after projecting.

Extensible Lattices

Usually, for generation of a rank-1 or Korobov lattice for an s-dimensional problem, we fix the number of points N and conduct a search for a best generating vector (or constant in the case of a Korobov lattice). However, a problem arises where, if more points are required, a new point set must be generated. Extensible lattices, first proposed by the authors of [34], were constructed such that it is possible to increase the number of points N without discarding points already generated. These constructions make use of the radical inverse function. For integers $i \geq 0$ and base $b \geq 2$, the radical inverse function $\psi_b(i)$ is as follows; if $i = \sum_{a=1}^{\infty} i_a b^{a-1}$, where $i_a \in \{0, 1, \ldots, b-1\}$, then

$$\psi_b(i) := \sum_{a=1}^{\infty} \frac{i_a}{b^a}.$$

For example, if $i = (\cdots, i_3, i_2, i_1)_b$ is the base b representation of i, then $\psi_b(i) = 0.(i_1, i_2, i_3, \cdots)_b$. The extensible rank-1 lattice sequence, having a generating vector $\mathbf{z} = (z_1, \ldots, z_s) \in \mathbb{Z}^s$ has its i^{th} point given by

$$\mathbf{u}_{(i)} = \psi_b(i-1)\ \mathbf{z} \bmod 1 \tag{3.12}$$

for $i \geq 1$ and where $\psi_b(i)$ is the base b radical inverse function applied to i. The extensible Korobov lattice is similar to (3.12), but with generating vector given by (3.9).

3.2.2 The Rectangular Grid

The rectangular grid (hereafter referred to as the grid, or n-point grid) is a type of QMC point set that can be used for numerical integration, but is only effective for small s. It is not an LDS as its discrepancy is not $\mathcal{O}(N^{-1}(\log(N))^s)$, and lacks some other properties of LDS as well. However, in the context of this thesis, it makes up an important part of our work and we choose to mention it here.

The n-point grid is simply made up of rows and columns of n points in s dimensions, and we denote the n-point grid by $\mathcal{G}_{n,s}$. We define an equally spaced n-point grid as follows.

Definition 3.2.6. *The points of an equally spaced, n-point grid is given by*

$$\mathcal{G}_{n,s} = \left\{ \left(\frac{g_1 - 1}{n-1}, \cdots \frac{g_s - 1}{n-1} \right), g_j = 1, \ldots, n, j = 1, \ldots, s \right\}, \tag{3.13}$$

where the total number of points $N = n^s$.

It has been shown by Niederreiter in [60] that the star discrepancy is

$$D^*(\mathcal{G}_{n,s}) = 1 - (1 - 1/N)^s,$$

thus $D^*(\mathcal{G}_{n,s}) \in \mathcal{O}(N^{-1/s})$. The convergence, whilst going to zero with large N, is slower than the MC convergence for $s > 2$.

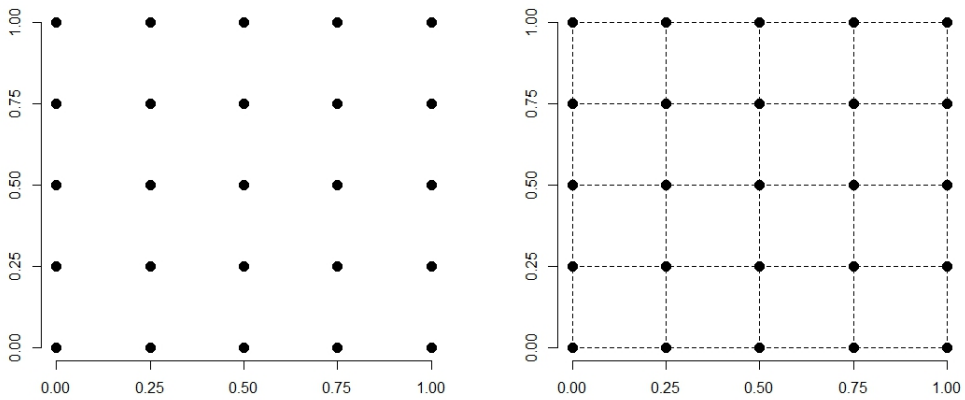

Figure 3.4: A grid $\mathcal{G}_{5,5}$ with a total of 25 points. The right hand side plot shows that the grid is not fully projection regular, with only $n = 5$ unique points after projecting.

The grid is not fully projection regular. Similar to the maximal-rank lattice, the

points line up in such a way that the projection onto a single dimension yields just n unique points rather than N. Figure 3.4 shows a simple example of the structure of the grid (left) and its projections (right).

3.2.3 Lattice Builder

Lattice Builder is a software tool developed by the authors of [46] that uses a number of algorithms to find generation vectors/constant for rank-1 and Korobov lattices. It can perform random and/or exhaustive searches for a given number of points N, dimension s and for various measures of discrepancy. It can be used to generate embedded lattice point sets $\mathcal{P}_{N_1} \subset \mathcal{P}_{N_2} \subset \ldots \subset \mathcal{P}_{N_k}$ with increasing number of points $N_1 < N_2 < \ldots < N_k$ for some positive number k. When $k = \infty$, this is an extensible lattice, but for practical purposes, Lattice Builder assumes that k is finite. Figure 3.5 shows an example of an embedded Korobov lattice where a best generating constant was found using the Lattice Builder software. Note that we denote an embedded Korobov lattice by $\mathcal{K}^*_{N,s}$.

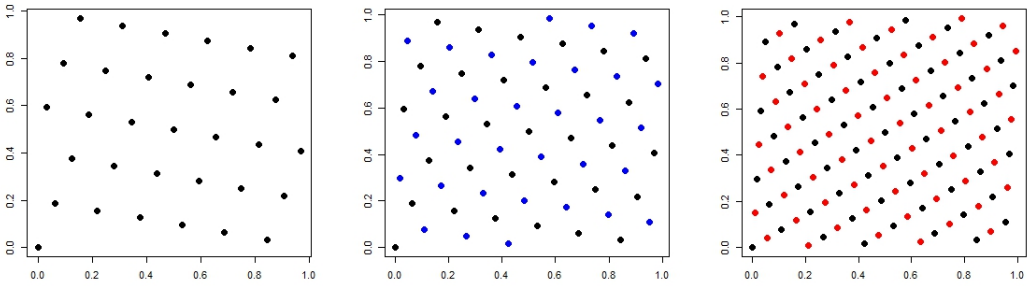

Figure 3.5: An example of an embedded Korobov lattice, where the generating constant is kept fixed while we increase the number of points from 32 to 64, and 64 to 128 points. For these Korobov lattices, the generating constant $\alpha_z = 19$. The blue dots in the middle plot show the new points added to the left plot as we move from a $\mathcal{K}^*_{32,2}$ to $\mathcal{K}^*_{64,2}$. Similarly, the red points on the right plot show the new points as we increase N from 64 to 128.

3.3 Discussion

We have given a very brief overview of QMC integration and the relevant point sets in the class of LDS, namely lattices. We briefly looked at the most relevant lattices to us, namely rank-1 lattices, Korobov lattices and maximal-rank lattices. Extensible and embedded lattices are also mentioned, as well as the software tool of Lattice Builder, which we use to generate the optimum point set for a given set of parameters. We also mention the grid, which was briefly described in Section 2.3.4 (though the algorithm that generates that grid is different to the formula described in (3.13), they are essentially the same).

The plan for the next phase of this thesis is to combine computational Bayesian inference and LDS. Chapter 2 mentioned that the grid can be used to perform Bayesian inference on a vector of hyperparameters $(\theta_1, \ldots, \theta_s)$ through INLA. The next chapter touches upon how grids can be used to perform Bayesian inference, before proposing a new method of using the various lattice points described in Section 3.2.1.

Chapter 4

Marginal Posterior Estimation using Low Discrepancy Sequences

The previous chapters discussed Bayesian inference and elements of QMC integration. We combine the two ideas to introduce new methods of Bayesian inference, with the main emphasis on using LDS points to estimate hyperparameters. As mentioned previously, the INLA methodology uses a grid to perform Bayesian inference (see Section 2.3.4). We take a numerical integration perspective here, and introduce the general algorithm for using grids to solve an $(s-1)$ dimensional integral. We then propose a new method which uses LDS points. The computational efficiency gained from using LDS points over the grid is significant, and as such we present the surrounding theory, convergence theorems, and examples.

INLA and its use of grids have inspired other works, such as [2, 37, 61, 87]. In the case of estimating hyperparameters with a grid-based approach, it is relatively simple in concept and fairly straightforward to implement. A set of grid points is placed in an appropriate support region. After evaluating the function at those points, we average out over rows and columns and fit an interpolant between the function evaluation means. The accuracy of the approximation is dependent on several things. The construction of

the support is important as we need to generate points and evaluate the function where the bulk of the probability density lies. Also, if the initial approximation to the function (such as the Laplace approximation $\tilde{\pi}(\boldsymbol{\theta}|\mathbf{y})$ (2.15)) is poor, the approximations will be of no use. The quality of the pointset used is important, and ties in with the idea of discrepancy (see Definition 3.1.1). We focus on the quality of points used for now and assume that the functional approximation is exact, and that an appropriate support is available.

Two major drawbacks of using the grid are (1) the potential inability to properly estimate the shape of the distribution, and (2) it is computationally prohibitive in higher dimensions. The first problem can occur when the distribution being estimated is highly skewed or multimodal and the grid points are laid out in such a way that fails to capture the behaviour of the function. The second (and larger) problem can be attributed to the number of grid points increasing exponentially with the number of dimensions. We propose an alternative point set to the grid, a maximal-rank lattice point set, to tackle the first problem. Due to the structure of the maximal-rank lattice (described in (3.11)), we regard this as a grid-based approach. We also propose a fully-projection regular LDS point set (such as the Korobov lattice) with a least-squares polynomial smoother to tackle the first problem, and also argue that this method is far more computationally efficient than any grid-based approach.

4.1 Preliminaries and Notation

We introduce some useful preliminary ideas as well as the notation that we will use throughout this chapter. Let $\pi(\boldsymbol{\theta})$ denote the s-dimensional joint distribution, with $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_s)$ the parameters of interest. Note that we drop the dependence on the data $\mathbf{y}$ for notational convenience. The usual situation in Bayesian analysis applies here, in that $\int_\Theta \pi(\boldsymbol{\theta}) d\boldsymbol{\theta}$ cannot be computed analytically. We also assume that the joint

posterior is absolutely integrable, that is $\pi \in \mathbf{L}^1[\boldsymbol{\Theta}]$ as $\pi \geq 0$. We wish to approximate the marginal densities of $\boldsymbol{\theta}$, that is $\pi(\theta_1), \ldots, \pi(\theta_s)$.

4.1.1 Approximation to $(s-1)$-Dimensional Integrals

Consider the Monte-Carlo approximation to the following s-dimensional integral

$$\begin{aligned} I_\pi = \int_\Theta \pi(\boldsymbol{\theta}) d\boldsymbol{\theta} &\approx \prod_{k=1}^{s} (b_k - a_k) \times \frac{1}{N} \sum_{j=1}^{N} \pi(\boldsymbol{\theta}_{(j)}), \\ &= \text{Vol}(\Theta) \times \frac{1}{N} \sum_{j=1}^{N} \pi(\boldsymbol{\theta}_{(j)}), \end{aligned} \tag{4.1}$$

where $\Theta = [\mathbf{a}, \mathbf{b}]$ is the support, with $\mathbf{a} = (a_1, \ldots a_s) \in \mathbb{R}^s$ and $\mathbf{b} = (b_1, \ldots b_s) \in \mathbb{R}^s$. Also, $\{\boldsymbol{\theta}_{(j)}\} \in \Theta$ for $j = 1, \ldots, N$ denotes any point set generated within the support, at which the function π is to be evaluated. The volume term $\text{Vol}(\Theta)$ is required since $\Theta \neq [0, 1)$. Equation (4.1) shows us that the approximation is simply the product of the volume of the region and the mean function evaluations. We can approximate the marginal densities in a similar fashion. Let θ_i be the i^{th} component of $\boldsymbol{\theta}$, with $\pi(\theta_i)$ being the i^{th} marginal density of $\pi(\boldsymbol{\theta})$. To marginalise for our variable of interest θ_i, we integrate out all other variables $\boldsymbol{\theta}_{-i}$ over the new support $\Theta \setminus [a_i, b_i)$

$$\pi(\theta_i) = \int_{\Theta \setminus [a_i, b_i)} \pi(\boldsymbol{\theta}) d\boldsymbol{\theta}_{-i}. \tag{4.2}$$

We can approximate the $(s-1)$-dimensional integral in (4.2) as

$$\begin{aligned} \int_{\Theta \setminus [a_i, b_i)} \pi(\boldsymbol{\theta}) d\boldsymbol{\theta}_{-i} &\approx \frac{\prod_{k=1}^{s} (b_k - a_k)}{b_i - a_i} \times \frac{1}{N} \sum_{j=1}^{N} \pi(\theta_i; \boldsymbol{\theta}_{(j)}), \\ &= \text{Vol}(\Theta \setminus [a_i, b_i]) \times \frac{1}{N} \sum_{j=1}^{N} \pi(\theta_i; \boldsymbol{\theta}_{(j)}). \end{aligned}$$

Here $\pi(\theta_i; \boldsymbol{\theta}_{(j)})$ are the so-called orthogonal projections of $\pi(\boldsymbol{\theta}_{(j)})$ projected onto the i^{th} marginal space $[a_i, b_i)$ (we define these in upcoming sections). Note that the volume

of this space is excluded in the volume since we are not integrating out θ_i.

Consider that the marginal distribution of θ_i is to be evaluated at n distinct points $\theta_i = \theta_{i_l}$ for $l = 1, \ldots, n$. Suppose also that we have ν function evaluations at each θ_{i_l}, thus giving a total of $N = n \times \nu$ points. As an example, for the n-point grid, $\nu = n^{s-1}$. The marginal distribution of θ_i at each distinct point can be approximated by

$$\begin{aligned} \pi(\theta_i = \theta_{i_l}) &= \int_{\Theta \setminus [a_i, b_i]} \pi(\theta_1, \ldots, \theta_i = \theta_{i_l}, \ldots \theta_s) d\boldsymbol{\theta}_{-i} \\ &\approx \text{Vol}(\Theta \setminus [a_i, b_i]) \times \frac{1}{\nu} \sum_{j=1}^{\nu} \pi(\theta_i = \theta_{i_l}; \boldsymbol{\theta}_{(j)}) = \hat{\pi}(\theta_{i_l}). \end{aligned} \tag{4.3}$$

We refer to $\hat{\pi}(\theta_{i_l})$ as the *pointwise mean*, which is obtained by averaging out all ν function evaluations at the point θ_{i_l}.

4.1.2 Orthogonal Projections

We give some details on orthogonal projection. First, we express the point set $\{\boldsymbol{\theta}_j\} \in \Theta, j = 1, \ldots, N$ in matrix form

$$\begin{pmatrix} \boldsymbol{\theta}_{(1)} \\ \boldsymbol{\theta}_{(2)} \\ \vdots \\ \boldsymbol{\theta}_{(N)} \end{pmatrix}^T = \begin{pmatrix} \theta_{1,1} & \theta_{2,1} & \ldots & \theta_{s,1} \\ \theta_{1,2} & \theta_{2,2} & \ldots & \theta_{s,2} \\ \vdots & \vdots & \ddots & \vdots \\ \theta_{1,N} & \theta_{2,N} & \ldots & \theta_{s,N} \end{pmatrix}^T \tag{4.4}$$

and use each row to evaluate the joint posterior, thus giving us a vector of function evaluation points. Let Ψ be the augmented matrix of points and function evaluations,

having size $(s+1) \times N$,

$$\mathbf{\Psi}_{(s+1)\times N} = \begin{pmatrix} \theta_{1,1} & \theta_{2,1} & \dots & \theta_{s,1} & \pi(\boldsymbol{\theta}_{(1)}) \\ \theta_{1,2} & \theta_{2,2} & \dots & \theta_{s,2} & \pi(\boldsymbol{\theta}_{(2)}) \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ \theta_{1,N} & \theta_{2,N} & \dots & \theta_{s,N} & \pi(\boldsymbol{\theta}_{(N)}) \end{pmatrix}^T \in \mathbb{R}^{s+1}.$$

An orthogonal projection matrix P_i is used to project $\mathbf{\Psi}$ onto the i^{th} marginal space. Let $P_i = A(A^TA)^{-1}A^T$, where $A_{(s+1)\times 2}$ is a unit basis vector for $\mathbb{R}^2$ with the i^{th} entry being 1 and the rest zeroes, and the $(s+1)^{th}$ entry also being 1 with the rest being zeroes. As an example, the unit basis vector A and corresponding orthogonal projection matrix P_i for $s = 3$ and $i = 2$ is

$$A_{(4\times 2)} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix}, P_{(4\times 4)} = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

To approximate the i^{th} marginal $\pi(\theta_i)$, we orthogonally project $\mathbf{\Psi}$ onto the i^{th} marginal space by multiplying $P_i \times \mathbf{\Psi}$

$$P_i\mathbf{\Psi} = \psi_i \quad = \quad \begin{pmatrix} 0 & \dots & \theta_{i,1} & 0 & \dots & \pi(\boldsymbol{\theta}_{(1)}) \\ 0 & \dots & \theta_{i,2} & 0 & \dots & \pi(\boldsymbol{\theta}_{(2)}) \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & \theta_{i,N} & 0 & \dots & \pi(\boldsymbol{\theta}_{(N)}) \end{pmatrix}^T \tag{4.5}$$

Ignoring all columns with zeroes, we have

$$\begin{pmatrix} \theta_{i,1} & \pi(\boldsymbol{\theta}_{(1)}) \\ \theta_{i,2} & \pi(\boldsymbol{\theta}_{(2)}) \\ \vdots & \vdots \\ \theta_{i,N} & \pi(\boldsymbol{\theta}_{(N)}) \end{pmatrix}^T \in \mathbb{R}^2. \tag{4.6}$$

Since all but the i^{th} and $(s+1)^{th}$ columns are all zeroes, this reduces the dimension of ψ_i from $\mathbb{R}^{(s+1)}$ to $\mathbb{R}^2$.

4.2 Grid-Based Methods for Bayesian Inference

This section will present a general algorithm in which to perform Bayesian inference using the grid point set, as described in Definition 3.2.6 and (3.13). Whilst other works that have used grid-based inference differ in the details, they all follow the same general structure. We also present the details of the steps that we will use throughout the chapter with regards to grid-based inference.

4.2.1 The Grid Algorithm

We present the general algorithm for grid-based inference:

Algorithm 2 Grid-based method for Bayesian inference

1) Optimise $\pi(\boldsymbol{\theta})$ for mode π^* and Hessian H_π
2) Construct an appropriate support $[\mathbf{a}, \mathbf{b})$
3) Generate a grid $\mathcal{G}_{n,s}$ over the support and evaluate $\pi(\boldsymbol{\theta}_{(j)}), \forall \boldsymbol{\theta}_{(j)} \in \mathcal{G}_{n,s}$
4) Numerical integration for pointwise means $\hat{\pi}$, fit a smoother between pointwise means, and normalise
5) Re-use $\mathcal{G}_{n,s}$ and $\pi(\boldsymbol{\theta})$ for the estimation of the latent parameters (see Section 2.3.3 and (2.16))

The first four steps in the algorithm are used for the estimation of the marginal pos-

teriors of the hyperparameters, while Step 5 is for the marginal posteriors of the latent parameters. We focus here on the first four steps for the approximation of the hyperparameter posterior marginals. The optimisation step can be performed using any type of quasi-Newton method, such as the BFGS algorithm [16]. The mode and Hessian are important for the construction of the support, in which the points $\pi(\boldsymbol{\theta}_{(j)}) \in \mathcal{G}_{n,s}$ are generated. A simple method is to use the diagonal quantities of the Hessian to approximate the marginal standard deviations, then expand out by $\pm\delta, \delta > 0$ standard deviations from the mode on every axis. The INLA method uses the mode and Hessian to create a set of Eigen-axes to explore the space, fills the space with grid points and stops when the density is sufficiently close to zero (see Eq. (2.18)), thus partially fulfilling the requirements of Step 3.

Step 4 requires us to numerically integrate our function evaluations to obtain the pointwise means, for which we then fit a smoother and normalise. The numerical integration here requires computing pointwise means, as explained in (4.3). The function evaluations $\pi(\boldsymbol{\theta}_{(j)})$ all project to an abscissa point θ_{i_l} and are all averaged out to obtain pointwise means $\hat{\pi}(\theta_{i_l})$. For each marginal, we have n pointwise means. An interpolant, such as a cubic spline, is fitted through the pointwise means. A cubic spline estimation to the marginal would be

$$\tilde{\pi}_{sp}(\theta_i) = \begin{cases} P_1^3(\theta_i), & a_i = \theta_{i,1} \leq \theta_i < \theta_{i,2} \\ P_2^3(\theta_i), & \theta_{i,2} \leq \theta_i < \theta_{i,3} \\ & \vdots \\ P_{n-1}^3(\theta_i), & \theta_{i,n-1} \leq \theta_i < \theta_{i,n} = b_i \end{cases},$$

where $\tilde{\pi}_{sp}(\theta_i)$ is the cubic spline approximation to $\pi(\theta_i)$ and $P_k^3(\theta_i)$ is the k^{th} cubic polynomial fit between the co-ordinates $(\theta_{i,k}, \hat{\pi}(\theta_{i,k}))$ and $(\theta_{i,k+1}, \hat{\pi}(\theta_{i,k+1}))$. Normalisation requires us to integrate $\tilde{\pi}_{sp}(\theta_i)$ over the support $[a_i, b_i)$ which can be performed

analytically or otherwise. The final approximation for the marginal posterior is

$$\pi(\theta_i) \approx \frac{\tilde{\pi}_{sp}(\theta_i)}{\int_{a_i}^{b_i} \tilde{\pi}_{sp}(\theta_i) d\theta_i}. \tag{4.7}$$

Example 4.2.1. *To demonstrate the approximation of the marginals using the grid, we start with a very easy example. Let* $(\theta) = (\theta_1, \theta_2)$ *be our parameters of interest, and assume that the joint posterior distribution is a bivariate Gaussian, with mode centred at the origin and covariance matrix* $\Sigma = I_2$*, that is the* 2×2 *identity matrix. The joint posterior is given by*

$$\pi(\boldsymbol{\theta}) = (2\pi)^{-1} \exp\left(-\frac{1}{2}\boldsymbol{\theta}^T\boldsymbol{\theta}\right). \tag{4.8}$$

This of course can be done analytically via separation of variables, using the property of exponentials ($\exp(u + v) = \exp(u)\exp(v)$*) and integrating out each variable of interest. The result is that each marginal is the standard Gaussian*

$$\begin{aligned} \pi(\theta_1) &= (2\pi)^{-1/2} \exp\left(-\frac{1}{2}\theta_1^2\right) \\ \pi(\theta_2) &= (2\pi)^{-1/2} \exp\left(-\frac{1}{2}\theta_2^2\right). \end{aligned} \tag{4.9}$$

We construct the support in Step 2 by setting $\delta = 3$*, thus going out three standard deviations from the mode in every direction, giving the support* $[(-3, -3), (3, 3)]$*. We generate* $\mathcal{G}_{5,2}$ *inside the support, giving a total of* $N = 25$ *points. The support box and the position of the points with respect to the joint distribution can be seen in Figure 4.1. Looking at the marginal space for* θ_1*, we can see the grid points project to five different abscissa points on the first axis,* $(-3, -1.5, 0, 1.5, 3)$*. To complete Step 3, we evaluate the function at all points. For an approximation of the density at each abscissa point, we average across all function evaluations that project to it giving the pointwise means. The results are shown in Table 4.1.*

Figure 4.2 shows the process for Step 4. The pointwise means are found and plotted.

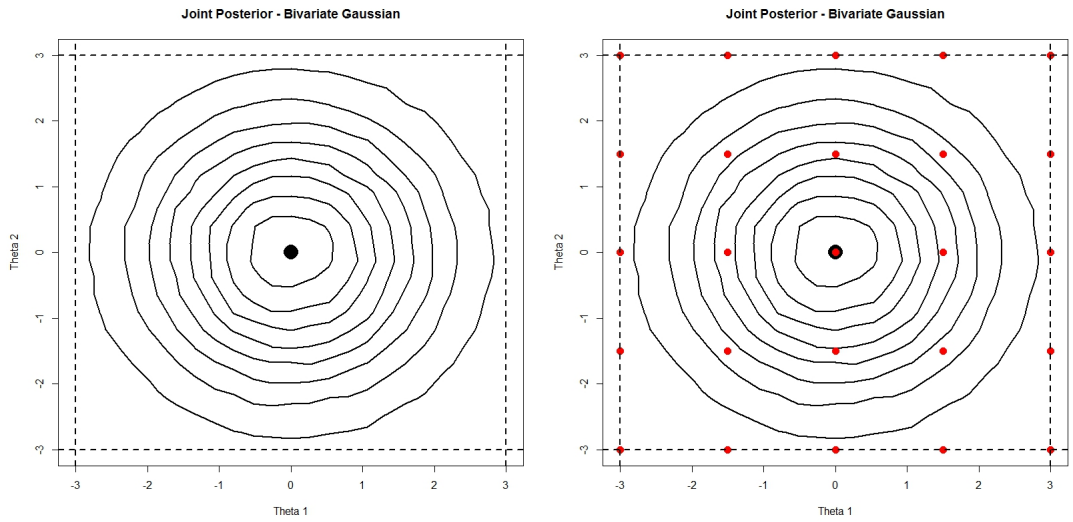

Figure 4.1: Contour plot of the joint posterior, with mode centred at the origin. The left figure shows the support box (dashed line) generated by going out three standard deviations from the mode (step two). The 5-point grid $\mathcal{G}_{5,2}$ is placed within the support box (red points), as shown by the figure on the right.

Table 4.1: Pointwise means for each abscissa point for θ_1.

Abscissa	Pointwise Mean
-3	0.000591
-1.5	0.017274
0	0.053206
1.5	0.017274
3	0.000591

An interpolant (cubic spline) is fitted through the pointwise means giving us the shape of the marginal. Lastly, we normalise the curve by integrating the cubic spline with respect to θ_1 (as described in (4.7)). The bottom graphs in Figure 4.2 shows the comparison between the grid approximation to the marginals versus the true marginal given by (4.9). *We see that the approximation is near exact.*

4.2.2 Using Maximal-Rank Lattice point sets

Example 4.2.1 illustrates the simplicity of using the grid to approximate marginal posteriors. In low dimensions it requires very few points, and for univariate, symmetric distributions it provides good approximations. However, for marginals that are highly

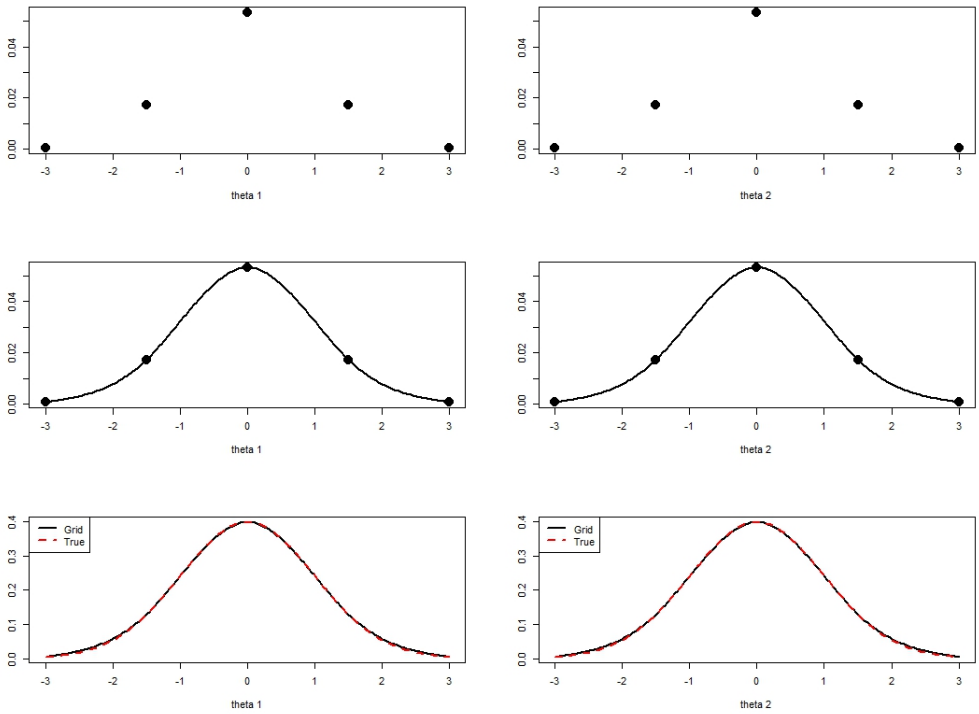

Figure 4.2: The process of Step 4 - averaging out for pointwise means (top), fitting a spline through pointwise means (middle) and normalising (bottom). The bottom figures also gives a comparison between the grid approximation (black, solid line) and the true density (red, dashed line).

skewed or multimodal it may require many more points than are computationally desirable. Figure 4.3 shows a four-dimensional mixture distribution that was approximated by $\mathcal{G}_{5,4}$ ($N = 625$ points) and $\mathcal{G}_{10,4}$ ($N = 10000$ points). This highlights the two main criticisms and how they can be related. The 5-point grid failed to capture the true shape of the marginal. Whilst it did capture the multimodal nature of the mixture distribution, it failed to locate the modes properly as we did not average out at enough abscissa points. Therefore in order to correctly capture the shape, we needed to double the amount of abscissa points to evaluate at, thus requiring a 10-point grid. This lead to a very good approximation. Unfortunately this came at the cost of increasing the number of points by 16 times what we originally used.

We propose an alternative point set to the grid, the maximal-rank lattice. Due to

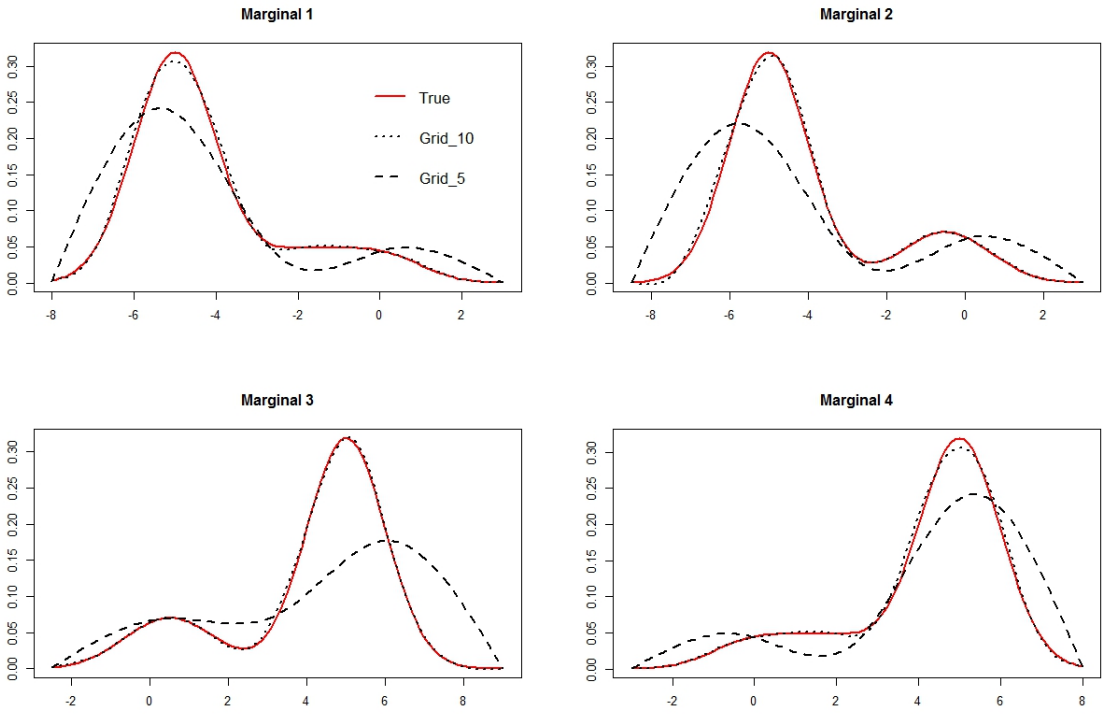

Figure 4.3: Approximating the marginals of a four dimensional mixture distribution using $\mathcal{G}_{5,4}$ and $\mathcal{G}_{10,4}$. The $\mathcal{G}_{10,4}$ approximated this shape very well, though required 10000 points to do so.

the grid-like structure of the maximal-rank lattice (as described in (3.11)), we can also use these as the point set described in Algorithm 2. We first demonstrate how it works, with the same bivariate Gaussian example as demonstrated with the grid. We then give another example which illustrates the advantage of using the maximal-rank lattice over the grid. The advantage is that we can evaluate the joint posterior at more locations, which can lead to a better approximation of the shape of the distribution, especially those which are highly skewed or multimodal. In lower dimension (say $s = 2$ or 3) the number of points used is comparable to the grid. In higher dimensions, we show that we can obtain accurate estimates of marginal posteriors with less points than the grid.

Example 4.2.2. *We estimate the marginals of the standard bivariate Gaussian, this time with the maximal-rank lattice. The density is given by (4.8) and the marginals by (4.9). We use the same support box and place a maximal-rank lattice inside. We generate $\mathcal{M}_{2,4,2}$, a two-point LDS copied four times in each dimension. With $n = 2$,*

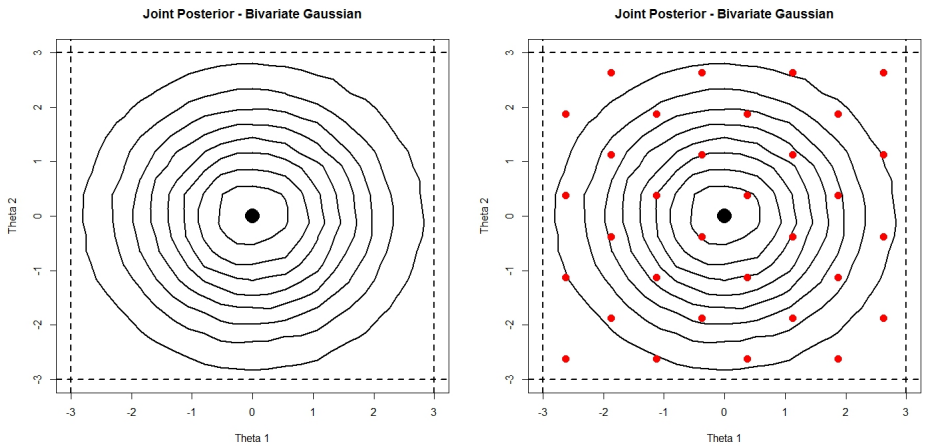

Figure 4.4: Contour plot of the joint posterior, with mode centred at the origin. The left figure shows the support box (dashed line) generated by going out three standard deviations from the mode (step two). The maximal-rank lattice $\mathcal{M}_{2,4,2}$ is placed within the support box and centred, as shown by the red points on the right-hand side graph.

$m = 4$, *and* $s = 2$, *this gives the total points* $N = 2 \times 4^2 = 32$, *with* $n \times m = 8$ *abscissa points. Due to how the maximal-rank lattice is generated, we centre it within the support bounds by shifting the intial two-point LDS. This is shown in Figure 4.4.*

Figure 4.5 shows the process of Step 4. We can see that we generate eight pointwise means, fit the spline and normalise. Similar to the grid, the approximated marginals are very good compared to the true marginals.

Example 4.2.3. *We compare the results of the grid and maximal-rank lattice on a multivariate beta distribution. The joint posterior is the product of several beta distributions, given by*

$$\pi(\boldsymbol{\theta}) = \prod_{i=1}^{s} \frac{\theta_i^{(\alpha_i - 1)}(1-\theta_i)^{(\beta_i - 1)}}{B(\alpha_i, \beta_i)}, \boldsymbol{\theta} \in [0,1]^s,$$

where $B(\alpha, \beta)$ *is the beta function, and where* $\alpha_i > 0$ *and* $\beta_i > 0$ *for all* $i = 1, \ldots, s$ *are the shape parameters. Whilst this is very easy to do analytically, we use it to demonstrate the difference between the grid approximations and the maximal-rank approximations. Let* $s = 5$. *We generate a* 5*-point grid* $\mathcal{G}_{5,5}$, *giving* $N = 5^5 = 3125$

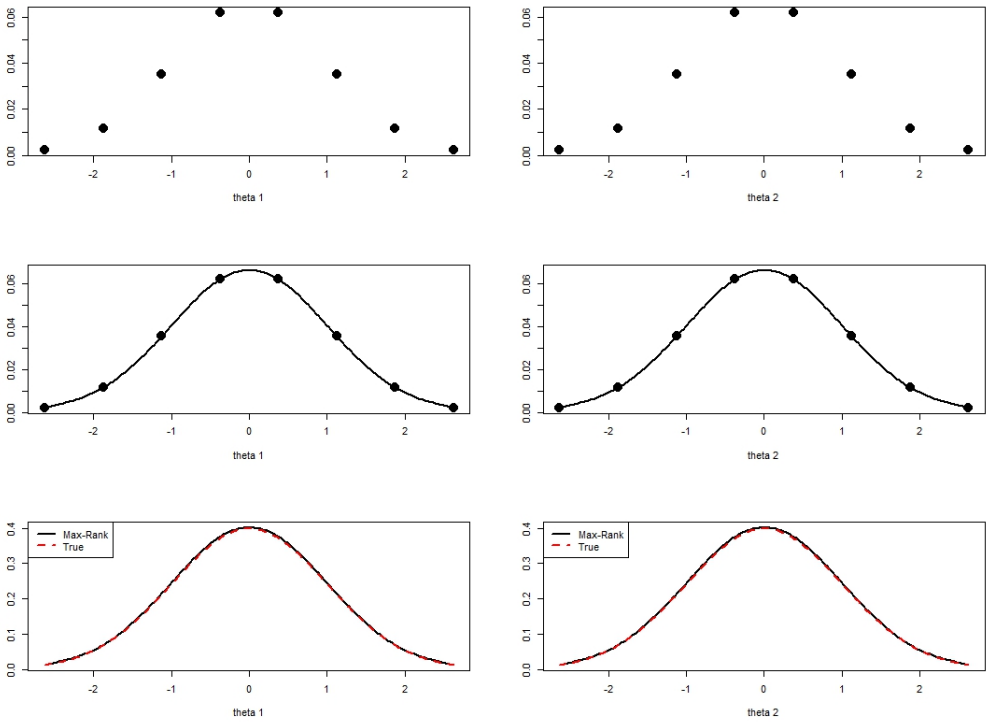

Figure 4.5: The process of Step 4 - averaging out for pointwise means (top), fitting a spline through pointwise means (middle) and normalising (bottom). The bottom figures also gives a comparison between the maximal-rank approximation (black, solid line) and the true density (red, dashed line).

points. We also generate a maximal-rank lattice $\mathcal{M}_{2,4,5}$ *with two points and four copies in each dimension, giving* $N = 2 \times 4^5 = 2048$ *points.*

Table 4.2: Kullback-Leibler divergence (K-L Div) and Hellinger distance (H.Dist) results for the multivariate beta approximations. The maximal-rank lattice outperformed the grid for all marginals according to both measures.

	K-L.Div		H.Dist	
Marginal	$\mathcal{G}_{5,5}$	$\mathcal{M}_{2,4,5}$	$\mathcal{G}_{5,5}$	$\mathcal{M}_{2,4,5}$
1	0.004276	0.000075	0.03417	0.00434
2	0.012995	0.000682	0.06112	0.01339
3	0.009344	0.004773	0.05113	0.03650
4	0.001157	0.000749	0.05421	0.01390
5	0.026038	0.005657	0.08541	0.04011

The approximations, as presented in Figure 4.6, show that the maximal-rank lattice approximations were far more accurate then the grid approximations. This is despite

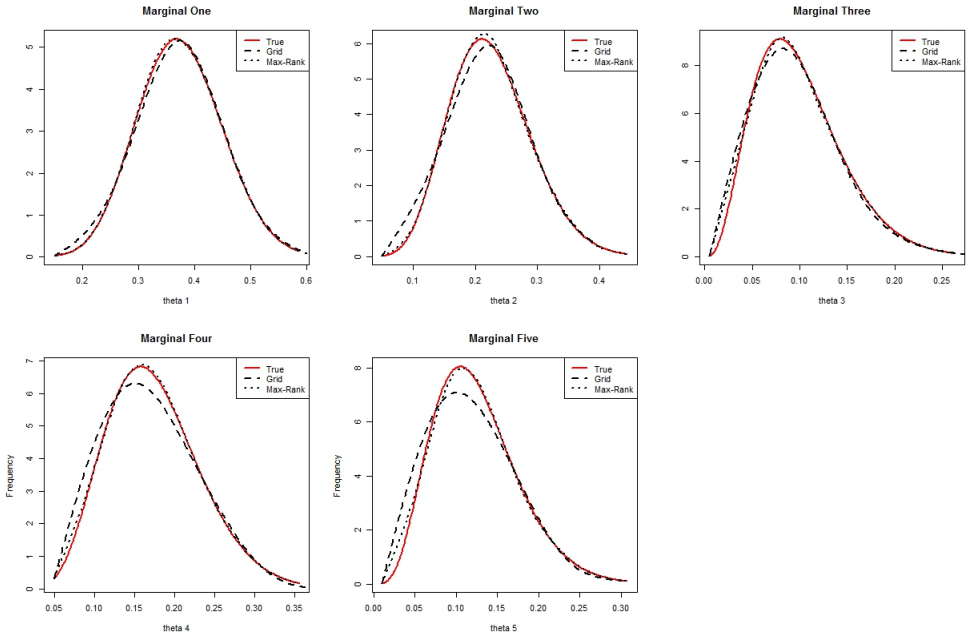

Figure 4.6: Approximation of the marginals of a multivariate beta distribution, using both the grid and a maximal-rank lattice.

the fact it uses fewer points. For the more skewed marginals (four and five), the grid was unable to properly capture the shape. In order for the grid to capture the shape, we may have to extend this out to a 6*-point grid, which more than doubles the amount of points of the* 5*-point grid (*7776 *points compared with* 3125 *points). We use two measures for accuracy, the Kullback-Leibler divergence [44] and Hellinger distance [48]. Both measures are commonly used to compute the distance between two probability distributions. The values in Table 4.2 indicate that the maximal-rank lattice approximations had the smaller distance values between the approximation and the true, indicating that these were the more accurate approximations. The overall accuracy for the maximal-rank lattice was satisfactory, with all KLD and Hellinger distance measures being very close to zero.*

4.2.3 Discussion of Grid-Based Inference Methods

We have introduced the algorithm to perform grid-based inference, demonstrated how it approximates marginals, and extended this out to maximal-rank lattices. Grid-based methods are nice to use in low dimension due to their simplicity. The grid approximation does well for univariate and symmetric distributions. For more skewed distributions and in higher dimensions we can do better with a maximal-rank lattice, both in terms of accuracy and computational efficiency, as shown in Example 4.2.3. However, the computational savings are not that great. Both the grid and the maximal-rank lattice increase exponentially as the dimension increases which is problematic for high dimensional problems. Next, we introduce a new method that not only captures shape, but also significantly reduces the number of points for high dimensional problems, thus easing the computational burden of approximating marginals.

4.3 LDS-Based Methods of Bayesian Inference

We discuss Bayesian inference using LDS. This is applicable to any LDS, but we restrict ourselves to Korobov lattices (Definition 3.2.4), due to their ease of use and the ability to find optimum generating vectors using the software Lattice Builder (Section 3.23). Recall from Section 3.2.1, that lattice point sets of rank-1 are *fully projection regular* (Definition 3.2.1). Therefore we cannot approximate in the same way as we did with the grid-based methods, as we have N unique abscissa points and nothing to average out over. However, we can get around this by fitting a polynomial through the function evaluations. We explain the details in the upcoming section.

4.3.1 The LDS algorithm

We present an algorithm to perform LDS-based inference:

Algorithm 3 LDS-based method for Bayesian inference

1) Optimise $\pi(\boldsymbol{\theta})$ for mode π^* and Hessian H_π
2) Construct an appropriate support $[\mathbf{a}, \mathbf{b}]$
3) Generate an LDS $\mathcal{L}_{N,s}$ over the support and evaluate $\pi(\boldsymbol{\theta}_{(j)}), \forall \boldsymbol{\theta}_{(j)} \in \mathcal{L}_{N,s}$
4) Orthogonally project $\pi(\boldsymbol{\theta}_{(j)})$ onto the i^{th} marginal for $\pi(\theta_i; \boldsymbol{\theta}_{(j)})$, fit a least-squares polynomial through $\pi(\theta_i; \boldsymbol{\theta}_{(j)})$ and normalise
5) Re-use $\mathcal{L}_{N,s}$ and $\pi(\boldsymbol{\theta})$ for the estimation of the latent parameters (see Section 2.3.3 and (2.16))

Performing the orthogonal projections (as described in Section 4.1.1 and Section 4.1.2) to the function evaluations results in a scatter of points on each marginal space. Since each point projects to its own one-dimensional component, we could think about it as its own pointwise mean. However, fitting a cubic spline interpolant through this would not result in an appropriate approximation, due to the scatter of points resulting from the orthogonal projections of LDS points (see Figure 4.8 (top)). A solution to this problem is to fit a least-squares polynomial through the scatter to approximate the shape. From there, the polynomial can easy be normalised (similar to (4.7)), and the LDS points can be re-used to estimate the latent parameters.

The details of fitting the least-squares polynomial are as follows. After generating $\boldsymbol{\theta}_{(j)} \in \mathcal{L}_{N,s}, j = 1, \ldots, N$ and projecting onto the i^{th} marginal, we have ψ_i as described in (4.5). The first column of ψ_i contain the abscissa points, and the second column contains the corresponding function evaluations. Let the vector of function evaluations be denoted by $\boldsymbol{\pi}$. Let V be the design matrix when fitting a least-squares polynomial of degree p. The design matrix is a Vandermonde matrix

$$V = \begin{pmatrix} 1 & \theta_{i,1} & \theta_{i,1}^2 & \cdots & \theta_{i,1}^p \\ 1 & \theta_{i,2} & \theta_{i,2}^2 & \cdots & \theta_{i,2}^p \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \theta_{i,N} & \theta_{i,N}^2 & \cdots & \theta_{i,N}^p \end{pmatrix}.$$

The least-squares approximation to the marginal posterior is

$$\tilde{\pi}_{LS}(\theta_i) = V(V^T V)^{-1} V^T \boldsymbol{\pi}$$

where $\tilde{\pi}_{LS}(\theta_i)$ is the least-squares approximation to $\pi(\theta_i)$. Normalisation gives us our final approximation

$$\pi(\theta_i) \approx \frac{\tilde{\pi}_{LS}(\theta_i)}{\int_{a_i}^{b_i} \tilde{\pi}_{LS}(\theta_i) d\theta_i}.$$

An open question here is what degree of polynomial should be chosen? A polynomial with low degree may not capture the shape properly, whereas a polynomial with too high a degree can lead to Runge's phenomenon [76]. This phenomenon occurs when oscillations occur at the edges of the interval when using a polynomial of high degree for interpolation. For now, we use the polynomial degree as a "tuning" parameter, but we will formulate strategies to answer this question in upcoming sections and the following chapter.

Example 4.3.1. *We demonstrate Algorithm 3 using the bivariate Gaussian example. After constructing the support box we place the Korobov lattice $\mathcal{L}_{32,2}$ within the support. The Korobov lattice we have chosen has $N = 32$ points, and the generating constant was found using Lattice Builder. After evaluating the function for all lattice points, we orthogonally project the function evaluations on each marginal space, resulting in a scatter of points (top graphs in Figure 4.8). Through those projections we fit a least-squares polynomial of degree eight (middle graphs of Figure 4.8). The resulting approximation after normalising results in a very close approximation.*

4.4 Convergence Theorems and Results

We discuss the approach of fitting a least-square polynomial through a set of orthogonally projected function evaluation points to approximate a marginal distribution. We

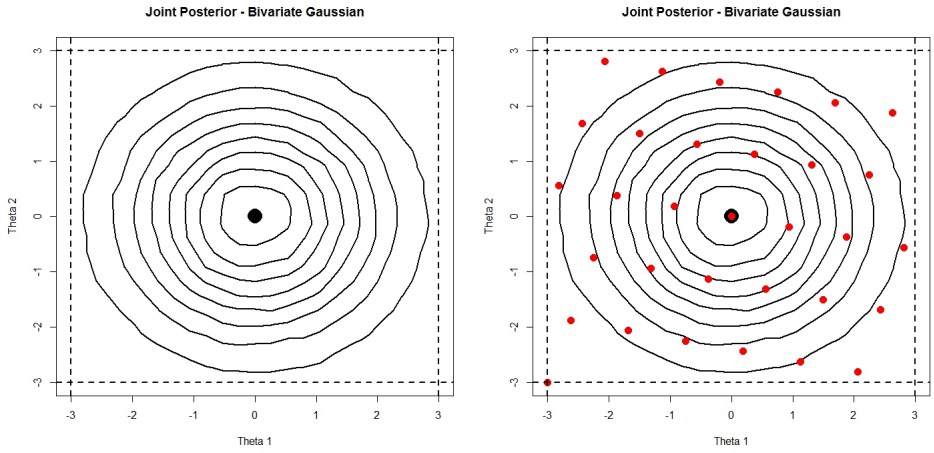

Figure 4.7: Contour plot of the joint posterior, with mode centred at the origin. The left figure shows the support box (dashed line) generated by going out three standard deviations from the mode (Step 2). The Korobov lattice $\mathcal{L}_{32,2}$ is placed within the support box, as shown by the red points on the right-hand side graph.

assume here that we have $N = n \times \nu$ points, such that $\pi(\theta_i)$ is evaluated at n distinct points $\theta_{i,l}$ for $l = 1, \ldots, n$ and that for each $\theta_{i,l}$, there are ν points whose i^{th} abscissa coordinate is $\theta_{i,l}$. This not only covers an n-point grid where $\nu = n^{s-1}$ or a maximal-rank lattice where $\nu = n \times m^{s-1}$, but any general point set when the n points $\theta_{i,l}$ are fixed, and ν points are selected (either randomly or determinstically) for each value of $\theta_{i,l}$. The choice of points are important, and determines the convergence properties and computational efficiency.

The orthogonal projections result in a scatter of points that have non-constant variance in each marginal space (see Figure 4.9). This suggests fitting a weighted least-squares polynomial through the projections $\pi(\theta_i = \theta_{i,l}; \boldsymbol{\theta}_{(j)})$ on θ_i, where the weights are proportional to the variances. We show however that a least-squares fit or its weighted version are equivalent when the degree of the polynomial is $n - 1$. We then study the convergence properties of the polynomial approach, and present some theorems and proofs. First, we give some matrix definitions for the upcoming least-squares analysis.

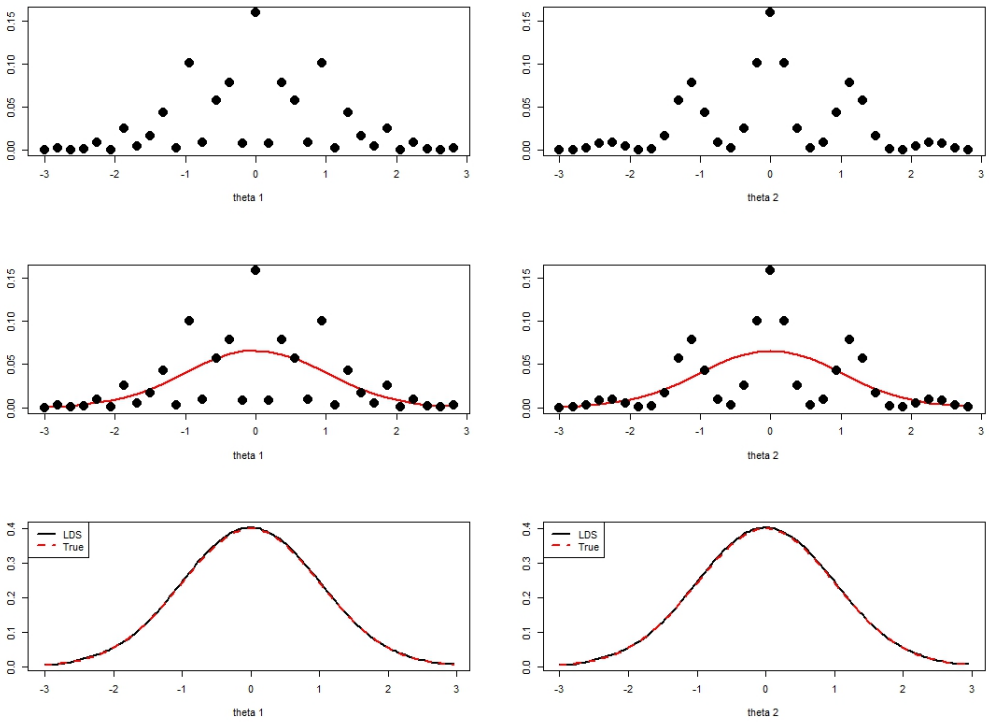

Figure 4.8: The process of step four for the LDS algorithm - projection of the function evaluations (top), fitting a least-squares polynomial through the projections (middle) and normalising (bottom). The bottom figures also gives a comparison between the LDS approximation (black, solid line) and the true density (red, dashed line).

4.4.1 Matrix Definitions

Let $\mathcal{P}_{N,s}$ be any point set as described at the beginning of this section. Let V be the design matrix when fitting a least-squares polynomial through the orthogonal projections $\pi(\theta_i; \boldsymbol{\theta}_{(j)})$ on θ_i. The coordinates of the projections correspond to n unique abscissa points $\theta_{i,l}, l = 1 \ldots, n$. The design matrix V is of size $N \times n$ and has a block structure

$$\underline{V} = \begin{pmatrix} \mathbf{1} & \mathbf{t}_1 & \mathbf{t}_1^2 & \cdots & \mathbf{t}_1^{n-1} \\ \mathbf{1} & \mathbf{t}_2 & \mathbf{t}_2^2 & \cdots & \mathbf{t}_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathbf{1} & \mathbf{t}_n & \mathbf{t}_n^2 & \cdots & \mathbf{t}_n^{n-1} \end{pmatrix},$$

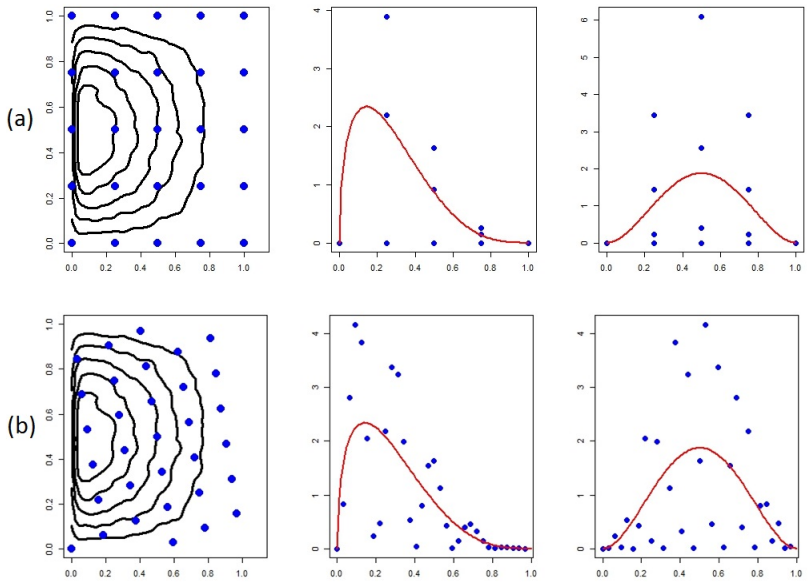

Figure 4.9: An example of the orthogonal projections for $\mathcal{G}_{5,2}$ and $\mathcal{K}_{32,2}$. Note that the projection onto each marginal space results in scatter of points that have non-constant variance.

where each element $\mathbf{t}_1^p \in \underline{V}, (p = 0, 1, \ldots, n-1)$ is a column vector of size $\nu \times 1$ and contains only the element $\theta_{i_l}^p$. We can more conveniently express $\underline{V}$ as the Kronecker product (see Appendix) of a Vandermonde matrix V and an $\nu \times 1$ column vector of ones,

$$\underline{V} = V \otimes \mathbf{1}_{\nu \times 1},$$

where

$$V = \begin{pmatrix} 1 & \theta_{i,1} & \theta_{i,1}^2 & \cdots & \theta_{i,1}^{n-1} \\ 1 & \theta_{i,2} & \theta_{i,2}^2 & \cdots & \theta_{i,2}^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \theta_{i,n} & \theta_{i,n}^2 & \cdots & \theta_{i,n}^{n-1} \end{pmatrix}.$$

Note that V is a Vandermonde matrix of size $n \times n$, is of full rank and is invertible since all its elements are unique.

For weighted least-squares, we give a weight w_l to the projections corresponding to

an abscissa point $\theta_{i,l}$. We define the weight matrix $\underline{W}$ that has size $N \times n$ by

$$\underline{W} = \begin{pmatrix} w_1 I_\nu & 0I_\nu & \cdots & 0I_\nu \\ 0I_\nu & w_2 I_\nu & \cdots & 0I_\nu \\ \vdots & \vdots & \ddots & \vdots \\ 0I_\nu & 0I_\nu & \cdots & w_n I_\nu \end{pmatrix},$$

where I_ν is an identity matrix of size $\nu \times \nu$. Similarly to $\underline{M}$, we can express $\underline{W}$ as a Kronecker product

$$\underline{W} = W \otimes I_\nu,$$

where

$$W = \begin{pmatrix} w_1 & 0 & \cdots & 0 \\ 0 & w_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & w_n \end{pmatrix}$$

is a square and diagonal matrix of weights, with size $n \times n$.

4.4.2 Theorems

Let $\tilde{\pi}_{LS}(\theta_i)$ denote the least-square polynomial approximation to $\pi(\theta_i)$ and $\tilde{\pi}_{WLS}(\theta_i)$ denote the weighted least-squares polynomial approximation to $\pi(\theta_i)$. Also, for notational convenience, we will express the othogonally projected function evaluations $\pi(\theta_i; \boldsymbol{\theta}_{(j)})$ simply as $\boldsymbol{\pi} = (\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \ldots, \boldsymbol{\pi}_n)^T$. Note that each element in $\boldsymbol{\pi}$ is a $\nu \times 1$ vector of function evaluations $\pi(\boldsymbol{\theta})$ corresponding to $\theta_{i,l}$. If $\mathcal{P}_{N,s}$ is any point set described at the beginning Section 4.5, then the following theorem holds. A basic knowledge of Kronecker products and their properties are needed for these proofs, many of which follow the basic rules of matrices. These can be found in most linear algebra textbooks, such as [52].

Theorem 4.4.1. *For any i, $\tilde{\pi}_{LS}(\theta_i) = \tilde{\pi}_{WLS}(\theta_i)$.*

Proof. The least-squares approximation $\tilde{\pi}_{LS}(\theta_i)$ can be expressed as

$$\begin{aligned}
\tilde{\pi}_{LS}(\theta_i) &= \underline{V}(\underline{V}^T\underline{V})^{-1}\underline{V}^T\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})\left((V \otimes \mathbf{1})^T(V \otimes \mathbf{1})\right)^{-1}(V \otimes \mathbf{1})^T\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})\left((V^T \otimes \mathbf{1}^T)((V \otimes \mathbf{1})\right)^{-1}(V^T \otimes \mathbf{1}^T)\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})(V^TV \otimes \mathbf{1}^T\mathbf{1})^{-1}(V^T \otimes \mathbf{1}^T)\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})\left((V^TV)^{-1} \otimes \nu^{-1}\right)(V^T \otimes \mathbf{1}^T)\boldsymbol{\pi} \\
&= \nu^{-1}(V \otimes \mathbf{1})(V^TV)^{-1}(V^T \otimes \mathbf{1}^T)\boldsymbol{\pi} \\
&= \nu^{-1}\left(V(V^TV)^{-1}V^T \otimes \mathbf{1}\mathbf{1}^T\right)\boldsymbol{\pi} \\
&= \nu^{-1}(I_n \otimes \mathbf{1}\mathbf{1}^T)\boldsymbol{\pi}, \qquad (4.10)
\end{aligned}$$

and the weighted least-squares approximation $\tilde{\pi}_{WLS}(\theta_i)$ can be expressed as

$$\begin{aligned}
\tilde{\pi}_{WLS}(\theta_i) &= \underline{V}(\underline{V}^T\underline{WV})^{-1}\underline{V}^T\underline{W}\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})\left((V \otimes \mathbf{1})^T(W \otimes I_\nu)(V \otimes \mathbf{1})\right)^{-1}(V \otimes \mathbf{1})^T(W \otimes I_\nu)\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})\left((V^T \otimes \mathbf{1}^T)(W \otimes I_\nu)((V \otimes \mathbf{1})\right)^{-1}(V^T \otimes \mathbf{1}^T)(W \otimes I_\nu)\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})(V^TWV \otimes \mathbf{1}^TI_\nu\mathbf{1})^{-1}(V^T \otimes \mathbf{1}^T)(V \otimes I_\nu)\boldsymbol{\pi} \\
&= (V \otimes \mathbf{1})\left((V^TWV)^{-1} \otimes \nu^{-1}\right)(V^T \otimes \mathbf{1}^T)(W \otimes I_\nu)\boldsymbol{\pi} \\
&= \nu^{-1}(V \otimes \mathbf{1})(V^TWV)^{-1}(V^T \otimes \mathbf{1}^T)\boldsymbol{\pi} \\
&= \nu^{-1}\left(V(V^TWV)^{-1}V^TW \otimes \mathbf{1}\mathbf{1}^TI_\nu\right)\boldsymbol{\pi} \\
&= \nu^{-1}(I_n \otimes \mathbf{1}\mathbf{1}^T)\boldsymbol{\pi} = \tilde{\pi}_{LS}(\theta_i) \qquad (4.11)
\end{aligned}$$

□

Theorem 4.4.2. *For any i, $\tilde{\pi}_{LS}(\theta_i)$ passes through $\hat{\pi}(\theta_{i,l})$, for $l = 1, \ldots, n$.*

Proof. From (4.10), we have that

$$\begin{aligned}\tilde{\pi}_{LS}(\theta_i) &= \nu^{-1}(I_n \otimes \mathbf{1}\mathbf{1}^T)\boldsymbol{\pi} \\ &= \nu^{-1}\begin{pmatrix} J_\nu & 0_\nu & \cdots & 0_\nu \\ 0_\nu & J_\nu & \cdots & 0_\nu \\ \vdots & \vdots & \ddots & \vdots \\ 0_\nu & 0_\nu & \cdots & J_\nu \end{pmatrix}\begin{pmatrix} \boldsymbol{\pi}_1 \\ \boldsymbol{\pi}_2 \\ \vdots \\ \boldsymbol{\pi}_n \end{pmatrix} = \begin{pmatrix} \hat{\pi}(\theta_{i,1}) \\ \hat{\pi}(\theta_{i,2}) \\ \vdots \\ \hat{\pi}(\theta_{i,n}) \end{pmatrix}, \end{aligned} \tag{4.12}$$

where each element J_ν or 0_ν is a square matrix of size $\nu \times \nu$ that contains either all 1's or all 0's respectively. □

Theorem 4.4.2 implies that this approach is equivalent to the interpolating polynomial approach, where a polynomial with degree $n-1$ is fitted to n function evaluations, for which the convergence properties can be studied using the existing literature in numerical analysis. For an arbitrary set of fixed abscissa points $\theta_{i,l}$, the convergence properties for the interpolating polynomial is poor in general. If the points $\theta_{i,l}$ are chosen either as Chebyshev nodes (linearly transformed to $[a_i, b_i]$, see [43]), or as equidistant points, the resulting interpolating polynomial will converge to the true function under strong smoothness conditions on the function. For the next two theorems, we will assume that $\theta_{i,l}$ are chosen as either Chebyshev nodes or equidistant points respectively. The points $\theta_{(j)} \in \Theta \setminus [a_i, b_i)$ can be sampled randomly, as a grid, or as an LDS. Note that the theorems and proofs we present from here until the end of this section were originally proved by the lead author in [38].

Theorem 4.4.3. *If $\pi(\theta_i)$ is infinitely differentiable such that*

$$\max_{\xi \in [a_i, b_i)} |\pi^{(n)}(\xi)| \leq L, \, \forall n,$$

for some $L < \infty$ such that $\frac{L}{2^{(n-1)}}\left(\frac{b_i - a_i}{2}\right)^n \ll (n-1)!$, $\forall n$, and $\theta_{i,l}$ for $l = 1 \ldots, n$ cor-

respond to Chebyshev nodes on the interval $[a_i, b_i]$*, then* $\tilde{\pi}_{LS}(\theta_i) \to \pi(\theta_i)$ *as* $n \to \infty$*.*

Proof. As $\nu \to \infty$,

$$\hat{\pi}(\theta_{i,l}) = \frac{\prod_{k=1}^{s}(b_k - a_k)}{b_i - a_i} \times \frac{1}{\nu}\sum_{j=1}^{\nu} \pi(\theta_i = \theta_{i,l}; \theta(j)) \to \pi(\theta_i = \theta_{i,l}) \tag{4.13}$$

Equation (4.13) holds due to Koksma-Hlawka inequality if the points $\theta(j)$ are sampled using a LDS, and due to the Law of Large numbers if $\theta(j)$ are sampled randomly. Then, from Theorem 4.4.2 and the standard result in approximation theory [18], it can be seen that

$$\max_{\theta_i \in [a_i, b_i)} |\pi(\theta_i) - \tilde{\pi}_{LS}(\theta_i)| \leq \max_{\theta_i \in [a_i, b_i)} \frac{|\pi^{(n)}(\theta_i)|}{n!} \max_{\theta_i \in [a_i, b_i)} \prod_{i=1}^{n} |\theta_i - \theta_{i,l}|.$$

This implies that

$$\max_{\theta_i \in [a_i, b_i)} |\pi(\theta_i) - \tilde{\pi}_{LS}(\theta_i)| \quad \leq \quad \frac{L}{n!} \max_{\theta_i \in [a_i, b_i)} \prod_{i=1}^{n} |\theta_i - \theta_{i,l}|$$

It can be shown (see for example [18]) that if the points $\theta_{i,l}$ correspond to the Chebyshev nodes on $[a_i, b_i)$, then

$$\max_{\theta_i \in [a_i, b_i)} \prod_{i=1}^{n} |\theta_i - \theta_{i,l}| \leq \frac{1}{2^{(n-1)}} \left(\frac{b_i - a_i}{2}\right)^n,$$

and therefore,

$$\max_{\theta_i \in [a_i, b_i)} |\pi(\theta_i) - \tilde{\pi}_{LS}(\theta_i)| \quad \leq \quad \frac{L}{2^{(n-1)} n!} \left(\frac{b_i - a_i}{2}\right)^n.$$

□

Theorem 4.4.4. *If $\pi(\theta_i)$ is infinitely differentiable such that*

$$\max_{\xi \in [a_i, b_i)} |\pi^{(n)}(\xi)| \leq L, \, \forall n$$

for some $L < \infty$ such that $L\left(\frac{b_i - a_i}{n-1}\right)^n \ll 1$, $\forall n$, and $\theta_{i,l}$ are equidistant points then $\tilde{\pi}_{LS}(\theta_i) \to \pi(\theta_i)$ as $\nu \to \infty$ and $n \to \infty$.

Proof. Using (4.13) and again from [18], it can be shown that if $\theta_{i,l}$ are equidistant then

$$\max_{\theta_i \in [a_i, b_i)} \prod_{l=1}^{n} |\theta_i - \theta_{i,l}| \leq \frac{(n-1)!}{4} \left(\frac{b_i - a_i}{n-1}\right)^n$$

and thus,

$$\begin{aligned} \max_{\theta_i \in [a_i, b_i)} |\pi(\theta_i) - \tilde{\pi}_{LS}(\theta_i)| &\leq \frac{1}{4(n+1)} L \left(\frac{b_i - a_i}{n}\right)^{n+1} \\ &\ll \frac{1}{4(n+1)}. \end{aligned}$$

□

If the function is only n times differentiable then the results of the previous two theorems indicate that interpolation obtained using a polynomial of degree $n-1$ will still be good as long as the derivatives are sufficiently bounded. Although these results give us the conditions in which $\tilde{\pi}_{LS}(\theta_i)$ converges to $\pi(\theta_i)$, the results may not be of practical importance since they require $O(\nu n s)$ function evaluations to evaluate all one dimensional marginals. This may not be computationally efficient or even feasible depending on the size of s.

A more computationally feasible approach is to use an LDS. We generate an LDS where $N = n \times \nu$ is (equally) partitioned into n parts, thus each partition having ν points. Let $[\theta_{i_u}, \theta_{i_{u+1}})$ be the u^{th} partition on the axis θ_i for $u = 0, \ldots, n-1$. We have

the following approximation,

$$\frac{1}{(\theta_{i_{u+1}} - \theta_{i_u})} \int_{\Theta \setminus [a_i, b_i)} \int_{\theta_{i_u}}^{\theta_{i_{u+1}}} \pi(\boldsymbol{\theta})\, d\theta_i\, d\boldsymbol{\theta}_{-i} \approx \frac{\prod_{k=1}^{s}(b_k - a_k)}{b_i - a_i} \times \frac{1}{\nu} \sum_{j=1}^{\nu} \pi(\boldsymbol{\theta}_{(j)}), \tag{4.14}$$

where $\boldsymbol{\theta}_{(j)} \in \boldsymbol{\Theta} \setminus [a_i, b_i) \times [\theta_{i_u}, \theta_{i_{u+1}})$ and $u = 1, \ldots, n$. We denote the RHS of (4.14) by $\tilde{\pi}(\theta_{i_{u+1}})$. We fit a least-squares polynomial of degree $n-1$ to the orthogonal projections of $\pi(\boldsymbol{\theta}_{(j)})$ on the i^{th} marginal space.

Theorem 4.4.5. *$\hat{\pi}_{LS}(\theta_i)$ passes through $\hat{\pi}(\theta_{i_u})$ for $u = 0, \ldots, n-1$.*

Proof. Follows from Theorems 4.4.1 and 4.4.2, though the block sizes in the Vandermonde matrix is now $n \times \nu$. □

Note that Theorem 4.4.5 also implies that the approach is equivalent to an interpolating polynomial approach. For the following theorem, let $\Delta\theta_i$ denote the length of a partition on the axis of θ_i. We have $\Delta\theta_i = \theta_{i_{u+1}} - \theta_{i_u}$, so then $\theta_{i_{u+1}} = \theta_{i_u} + \Delta\theta_i$.

Theorem 4.4.6. *$\hat{\pi}(\theta_{i_{u+1}}) \to \pi(\theta_{i_u})$ as $\nu \to \infty$ and $\Delta\theta_i \to 0$.*

Proof. We have that, as $\nu \to \infty$,

$$\tilde{\pi}(\theta_{i_{u+1}}) \to \frac{1}{(\theta_{i_{u+1}} - \theta_{i_u})} \int_{\Theta \setminus [a_i, b_i)} \int_{\theta_{i_u}}^{\theta_{i_{u+1}}} \pi(\boldsymbol{\theta}|\mathbf{y})\, d\theta_i\, d\boldsymbol{\theta}_{-\boldsymbol{i}}. \tag{4.15}$$

Equation (4.15) holds due to the Koksma-Hlawaka inequality. Note that

$$\frac{1}{\Delta\theta_i} \to \frac{d}{d\theta_{i_u}} \text{ as } \Delta\theta_i \to 0. \tag{4.16}$$

Therefore as $\nu \to \infty$ and $\Delta\theta_i \to 0$,

$$\tilde{\pi}(\theta_{i_u}) \to \frac{d}{d\theta_{i_u}} \int_{\theta_{i_u}}^{\theta_{i_{u+1}}} \int_{\Theta \setminus [a_i, b_i)} \pi(\boldsymbol{\theta})\, d\boldsymbol{\theta}_{-i}\, d\theta_i,$$

using (4.15) and (4.16) and changing the order of the integration in (4.15). Theorem

4.4.6 is then proven since we have that,

$$\begin{aligned}\frac{d}{d\theta_{i_u}}\int_{\theta_{i_u}}^{\theta_{i_{u+1}}}\int_{\Theta\setminus[a_i,b_i)}\pi(\boldsymbol{\theta})\,d\boldsymbol{\theta_{-i}}\,d\theta_i &= \frac{d}{d\theta_{i_u}}\int_{\theta_{i_u}}^{\theta_{i_{u+1}}}\pi(\theta_i)\,d\theta_i \\ &= \pi(\theta_{i_u}),\end{aligned}$$

with the last equality following from the fundamental theorem of calculus.

□

The partitions $\theta_{i_u}, u = 1, \ldots, n$ are all equally spaced. The last theorem shows the the least-squares approximation converges to the true marginal.

Theorem 4.4.7. *If $\pi(\theta_i)$ is infinitely differentiable such that*

$$\max_{\xi\in[a_i,b_i)}|\pi^{(n)}(\xi)| \leq L(n),\ \forall n,$$

$\exists L < \infty$ such that $L(n) \leq L$ and $L\left(\frac{b_i-a_i}{n}\right)^{n+1} \ll 1\,\forall n$, and $\theta_{i,l}$ are equidistant points then $\tilde{\pi}_{LS}(\theta_i) \to \pi(\theta_i)$ as $\nu \to \infty$ and $n \to \infty$.

Proof. The result follows from Theorems 4.4.3, 4.4.4 and 4.4.5 □

Note that if the function is only n times differentiable then the results in Theorem 3.6 indicate that interpolation obtained using a polynomial of degree $(n-1)$ will still be good as long as the derivatives are sufficiently bounded. This approach requires $O(\nu n)$ function evaluations, where typically $\nu < n^{(s-1)}$ and therefore this approach is computationally efficient compared to using an n-point grid.

4.4.3 Examples

We use some examples to illustrate both the convergence properties and the computational advantages of the LDS method over grid-based methods. Firstly, we an ex-

ponential example to illustrate Theorem 4.4.7. We revisit the multimodal example, introduced in Figure 4.3. Last of all, we demonstrate how the LDS approach works in higher dimension at which point the grid is not computationally feasible.

Example 4.4.1. *The exponential distribution, like many distributions, is smooth and has smooth derivatives. However the derivative does not exist at zero. We show that it still satisfies smoothness conditions imposed by Theorem 4.4.7. Suppose that one of the marginal posterior distributions is exponential with rate parameter $\lambda > 0$. We have that*

$$\pi(\theta_i) = \lambda \exp(-\lambda\theta_i).$$

The n^{th} derivative is

$$\pi^{(n)}(\theta_i) = (-1)^n \lambda^{n+1} \exp(-\lambda\theta_i),$$

and also

$$\sup_{\theta_i} |\pi^{(n)}(\theta_i)| = \lim_{\theta_i \to 0+} |\pi^{(n)}(\theta_i)| = \lambda^{n+1}.$$

The support $\Theta = [a, b) = [0, b)$ for some finite $b < \infty$. Then, $\exists n' > 0$ and $c < 1$ such that $\forall n' > n' + 1, \frac{b}{n-1} \leq \frac{1}{nc} \leq 1$. Furthermore, for any $\lambda < \infty, \exists n'' > n'$ such that $\forall n > n'', \lambda^{n+1}(\frac{1}{nc})^n \ll 1$.

It can be seen here that the conditions for Theorem 4.4.7 are met and $\tilde{\pi}_{LS}(\theta_i) \to \pi(\theta_i)$ as $\nu \to \infty$ and $n \to \infty$. Figure 4.10 demonstrates this, with the approximations converging as n and ν increase.

Example 4.4.2. *Recall Figure 4.3, which illustrated the need for the grid to have more abscissa points to estimate the shape for a multimodal distribution. Using a Korobov lattice, we can more than halve the amount of points needed to accurately approximate the posterior marginals for this example. We generate a $\mathcal{K}_{4096,4}$ and approximate the marginals using a least-squares polynomial. Figure 4.11 shows both the grid approx-*

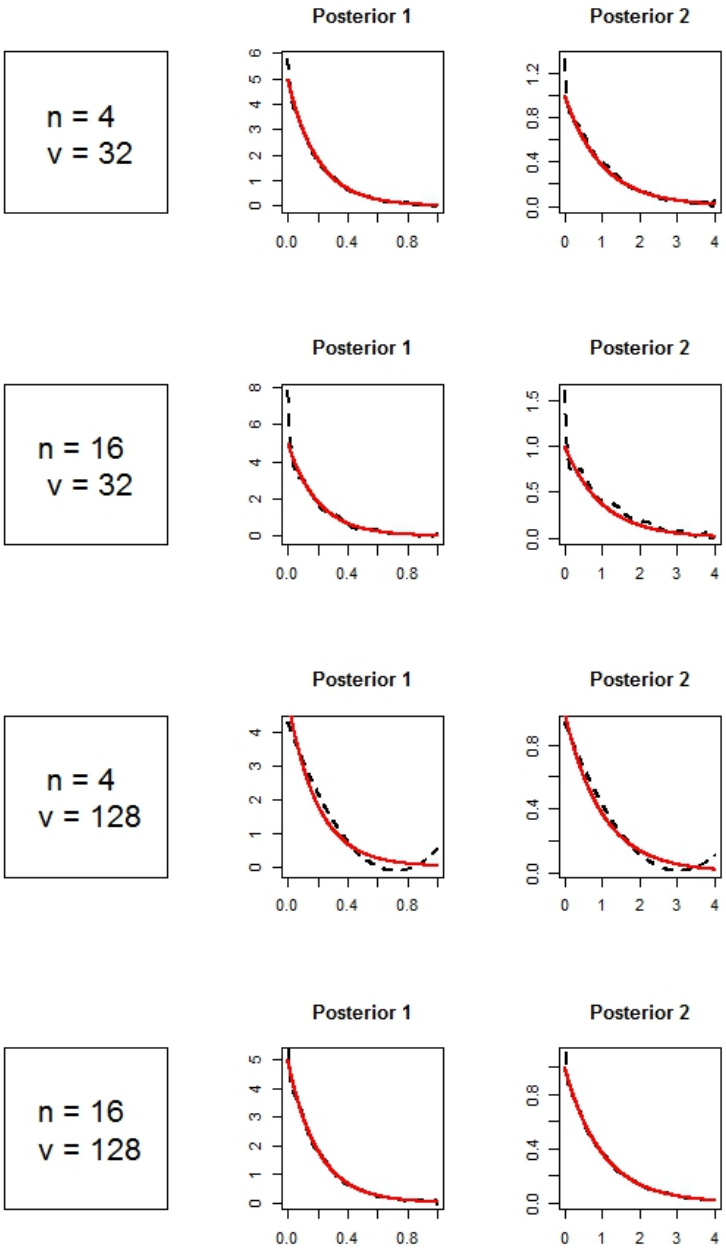

Figure 4.10: The least-squares approximation to the posterior marginals where the marginals are exponential distributions. The dashed line is our approximation, and the red solid line is the true marginal. The points used were from a Korobov lattice. Note, as both n and ν increase, the approximation converges to the true marginal.

imation and the LDS approximation. Both methods approximated the marginals well, as shown by the low Kullback-Leibler divergence and Hellinger distances (Table 4.3). The Korobov was slightly more accurate in all but the first marginal.

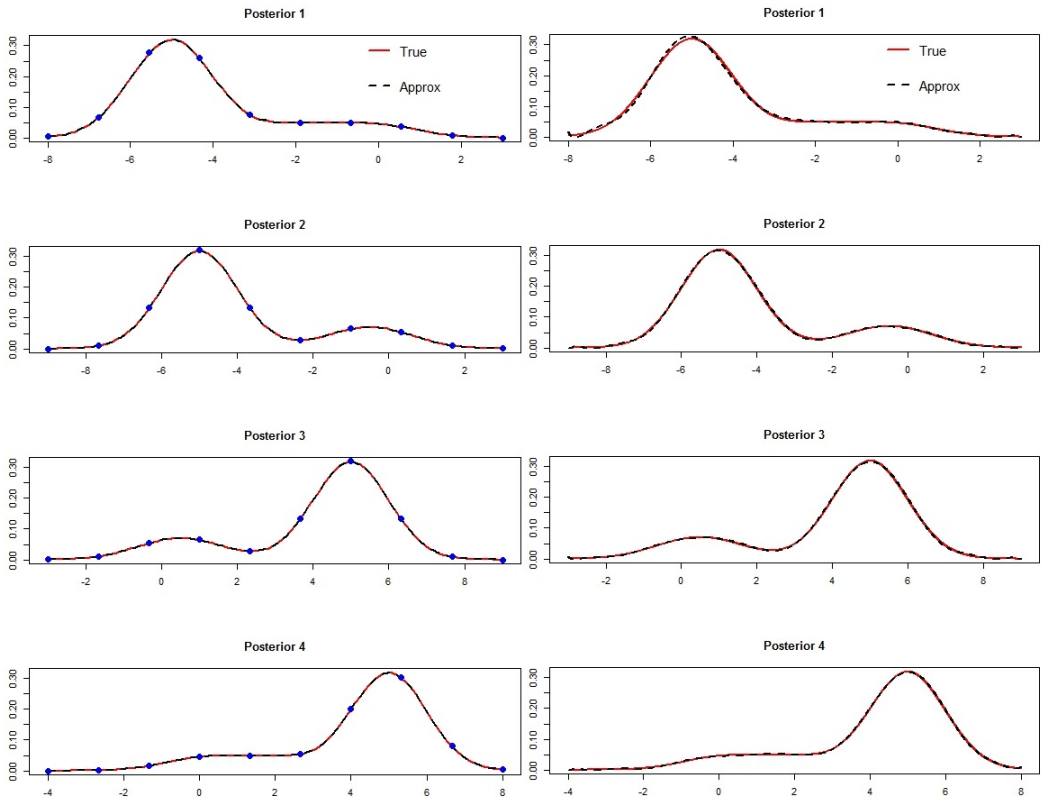

Figure 4.11: The multimodal mixture distribution example, estimated by $\mathcal{G}_{10,4}$ with a cubic spline, and $\mathcal{K}_{4096,4}$ with a least-squares polynomial. Both strategies accurately captured the shape and approximated well, though the Korobov lattice did this with less than half the points (4096 compared to 10000 grid points).

Example 4.4.3. *We provide an example that is not computationally feasible for the grid, but can be performed using a Korobov lattice. Let our joint posterior be a product of s gamma distributions, so we have*

$$\pi(\boldsymbol{\theta}) = \prod_{i=1}^{s} \frac{\beta_i^{\alpha_i}}{\Gamma(\alpha_i)} \theta_i^{\alpha_i - 1} \exp(-\beta_i \theta_i),$$

where $\alpha_i > 0$ and $\beta_i > 0$ are rate and shape parameters respectively, and $\Gamma(\cdot)$ is the Gamma function. Let $s = 12$. Our standard 5-point grid would need 5^{12} points, which is around a quarter of a billion points. A 3-point grid would require around half a million points which is feasible, but would not capture the shape effectively, especially

Table 4.3: Kullback-Leibler divergence (K-L Div) and Hellinger distance (H.Dist) results for multivariate, multimodal example given in Example 4.4.2. The 10-point grid was more accurate for the first marginal, but the Korobov was more accurate for the other three marginals. Note the Korobov lattice generated had less than half the points of the grid.

	K-L.Div		H.Dist	
Marginal	$\mathcal{G}_{10,4}$	$\mathcal{K}_{4096,4}$	$\mathcal{G}_{10,4}$	$\mathcal{K}_{4096,4}$
1	0.00123	0.00148	0.01751	0.01788
2	0.04944	0.00468	0.02445	0.01406
3	0.04944	0.00476	0.02445	0.01406
4	0.01913	0.01033	0.02426	0.01471

since the marginals are skewed. Figure 4.12 shows the results for the Korobov approximation with a least-squares fit. We were able to get a good approximation with $\mathcal{K}_{2^{17},12}$*, a Korobov lattice with* 131072 *points.*

4.5 Discussion

This chapter gave an overview of grid-based methods for Bayesian inference, which are appealing due to their simplicity and ease of implementation. However, they suffer the drawbacks of a limited ability to capture the target distributions shape, and the fact that the number of points increases exponentially with dimension. We introduced the maximal-rank point set, which goes some way to improving the ability to capture the shape. Although the maximal-rank can outperform the grid in terms of accuracy, the number of points also increase as dimension increases.

We introduced LDS-based methods for Bayesian inference as a way of overcoming the computational burden of approximating marginals in high dimension. We fit a least-squares polynomial through function evaluations that are projected onto each marginal space. We provided the theory that shows that the approximation converges, and also provided some examples of the computational benefits of using an LDS-based approach. We argue that this approach will outperform grid-based methods with respect to accuracy and computational efficiency in almost all cases. Our goal now is to

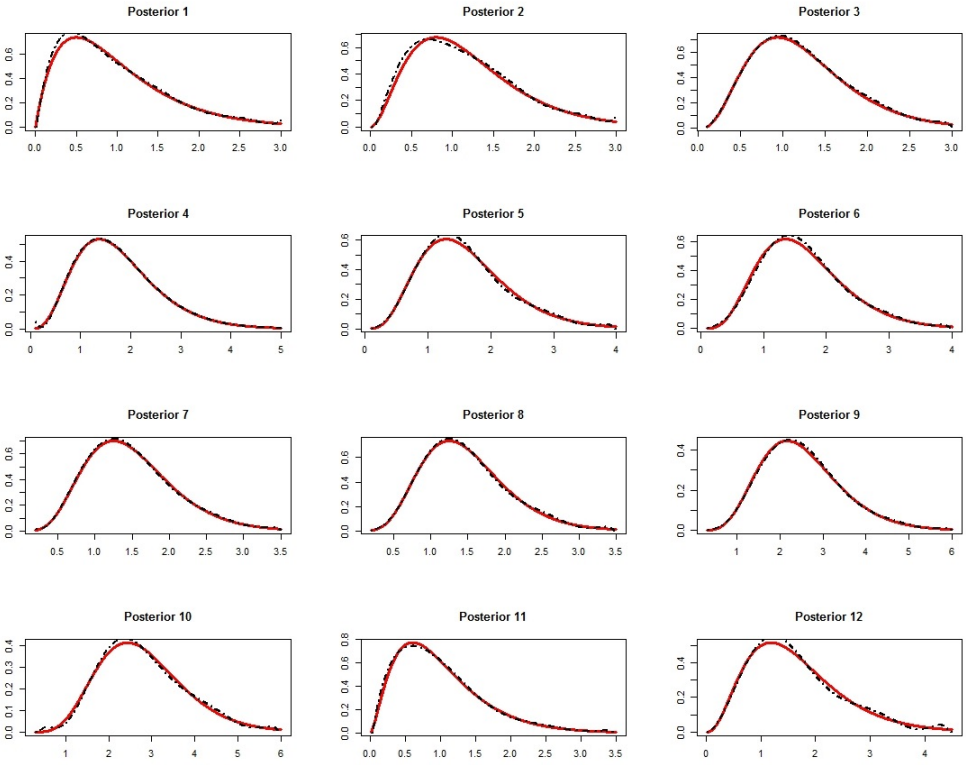

Figure 4.12: Results for the 12D gamma example. We used a $\mathcal{K}_{2^{17},12}$ with a least-squares polynomial fit. The approximations (dashed line) are close to the true marginal (red solid line) for all marginals.

use these methods in a full Bayesian inference framework. The next chapter will look at incorporating this method into INLA, including modifictions to the LDS-algorithm, an extension to improve accuracy even further for little to no cost, as well as three examples to assess performance.

Chapter 5

Incorporation of LDS Methods and the INLA Methodology

We proposed a method in Chapter 4 for marginal posterior estimation using LDS point sets. We showed through examples and convergence theorems that these methods can outperform grid-based methods with respect to computational speed and accuracy. Although the LDS-based method works quite well, there can be issues with regards to the polynomial chosen to fit through the function evaluations. For instance, there is some chance of unstable approximations if the degree of the polynomial is too high. From a practical perspective, this is inconvenient as it creates more tuning to find a polynomial degree that is suitable.

This chapter discusses several modifications to the LDS-based methods proposed in Chapter 4, in the hopes of making it more useful in practice. One of the key modifications comes about through INLA's use of variance-stabilising transformations, to reparameterise the hyperparameters into something that closely resembles a Gaussian. Starting from here, we continue to develop the method towards something that is fast, accurate and useful. We start this chapter by giving the reader a brief introduction of INLA using the software package R [63], as well as introducing the autoregres-

sive model. We then propose several modifications and use the autoregressive model to illustrate our new methods. We finish by analysing two real-world latent Gaussian models, which we will use to compare our proposed methods with INLA's integration strategies.

Note that all computations in this chapter and the following chapter, were performed using an Intel(R) Core i7-4770 CPU @ 3.40GHz, 64-bit Windows OS, with 8GB RAM.

5.1 Using R-INLA

The purpose of this section is to demonstrate how INLA works in a practical setting. The INLA methodology is almost exclusively performed using R-INLA (see www.r-inla.org), which contains all the necessary code and functions to perform the required inference. Since its inception, R-INLA has been widely used for a number of applications [56, 75]. We give all necessary code to perform basic inference on a simple autoregressive model, and continue on using this model to show the incorporation of LDS methods into INLA. The R package $\{\text{INLA}\}$ cannot be downloaded from the usual R website, but is available from the R-INLA website. For more details about R-INLA, see [78].

5.1.1 The Autoregressive Model

The autoregressive (AR) model is a commonly used model in time series analysis. The AR model represents a type of stochastic process, where the variable of interest ϕ_t depends linearly on its previous values and a random error term. We use the index t to represent the value at time t. The AR process of order P, denoted as AR(P), is defined as

$$\phi_t = \sum_{i=1}^{P} \rho_i \phi_{t-i} + \epsilon_t$$

where $\rho_1, \ldots, \rho_P$ are the model parameters and ϵ_t are the error terms. We consider the case where $P = 1$, which gives rise to the autoregressive model of order 1, denoted as AR(1). The AR(1) process for a multivariate Gaussian vector $(\phi_1, \ldots, \phi_{n_d})$ is given by

$$\phi_t = \rho\phi_{t-1} + \epsilon_t, \qquad t = 2, \ldots, n_d$$

where $\epsilon_t \sim \mathcal{N}(0, \tau_x^{-1})$, and the initial value is given by

$$\phi_1 \sim \mathcal{N}(0, (\tau_x(1-\rho^2))^{-1}), \qquad |\rho| < 1.$$

We have two hyperparameters in this model, the precision parameter τ_x and correlation parameter ρ. We perform a reparameterisation to represent the hyperparameters as a function that is "more Gaussian". Let θ denote a hyperparameter, and let f be the function corresponding to the reparameterisation. We have

$$f(\theta) = \theta_z, \tag{5.1}$$

where θ_z denotes the reparameterised hyperparameter. A suitable reparameterisation of these hyperparameters is given by the following. Let κ be the marginal precision given by

$$\kappa = \tau_x(1-\rho^2).$$

Then, the hyperparameter θ_κ is given by

$$\theta_\kappa = \log(\kappa).$$

The hyperparameter θ_ρ is represented as

$$\theta_\rho = \log\left(\frac{1+\rho}{1-\rho}\right).$$

We can simulate this process using the `arima.sim(...)` function from the R package {stats}. In addition to this process, we will generate observations $\mathbf{y} = \{y_1, \ldots, y_{n_d}\}$ assuming the following model

$$y_t = \phi_t + e_t, \tag{5.2}$$

where $e_t \sim \mathcal{N}(0, \tau_y^{-1})$ and ϕ_t is the AR(1) process. This adds a third hyperparameter, the precision for the Gaussian observations τ_y. We reparameterise τ_y with the following representation

$$\theta_\tau = \log(\tau_y).$$

Thus, our hyperparameters are given by $\boldsymbol{\theta}_z = \{\theta_\kappa, \theta_\rho, \theta_{\tau_y}\}$ and latent parameters $\{\phi_t\}, t = 1, \ldots, n_d$. Note that $\boldsymbol{\theta}_z$ denotes the reparameterised hyperparameters. We will assume this model through the majority of this chapter. We will show how to simulate data and perform inference on this model using R-INLA, before moving on to performing inference using modifications of the LDS-based methods introduced in the previous chapter.

5.1.2 Data Simulation and Inference

We start by generating some data based on (5.2). We let $n_d = 100$, $\rho = 0.65$, $\kappa = 1$, and $\tau_y = 100$. We can use the following R-code to perform this:

```
### Generate data ###
set.seed(781984)
nd = 100; rho = 0.65; tau = 100
phi = arima.sim(nd, model = list(ar = rho))
y = phi + rnorm(nd, sd = 1/sqrt(tau))
```

The data generated is shown in Figure 5.1. All inference using INLA is done via the `inla(...)` function. There are many options as to the inputs we can use inside this function, see [55]. However, we will go through the necessary inputs in this section.

We first organise all the data in a data frame. We have observations $\mathbf{y}$, and we set up the index from 1 to n_d. Any other covariates would be added to this data frame. Next, we must define the model. We can use the function `f(...)` to define the AR process, and we can input

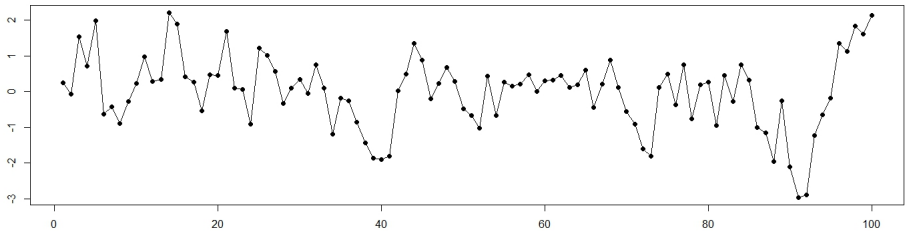

Figure 5.1: Simulated data for the AR(1) process.

other necessary details in this function, such as priors for the hyperparameters corresponding to the AR process (in our case, θ_κ and θ_ρ). Covariate information need not be input using `f(...)`, but can be input using its column name in the data frame. We will call this model "`formula`" in the code. Lastly, we use the `inla(...)` function to perform the inference for all parameters in the model. The following code will perform inference on our AR(1) process:

```
### Libraries ###
library(INLA)

### Data frame ###
data = data.frame(y, idx = 1:n)

### Formula ###
formula = y ~ -1 + f(idx, model="ar1",
                  hyper = list(rho = list(fixed = FALSE,
                                          prior = "betacorrelation",
                                          param = c(5, 1)),
                               prec = list(fixed = FALSE,
                                          prior = "loggamma",
                                          param = c(1, 1))))

### inla(...) call ###
res = inla(formula, data = data, family = "gaussian",
          control.family = list(hyper = list(prec = list(
                                       fixed = FALSE,
                                       prior = "loggamma",
                                       param = c(100, 1)))),
          control.inla = list(int.strategy = "ccd",
```

```
strategy = "simplified.laplace"))
```

The formula in the code includes a '`-1`' to specify that there is no intercept in this model. We have assigned a scaled-Beta prior for θ_ρ and log-Gamma priors for θ_κ and θ_{τ_y}. The scaled-Beta prior is a prior primarily used for a correlation parameter since $\rho \in (-1, 1)$, and is a Beta distribution scaled to have domain in $(-1, 1)$. The prior for θ_{τ_y} is defined in the `control.family` argument in the `inla(...)` function itself, rather than in the `formula`. We also define the likelihood model for our observations through the `family` argument. The `control.inla` argument allows the user to use different integration point sets and integration strategies as outlined in Section 2.3.3. We have used the central composite design (CCD) as our point set and the simplified Laplace strategy for the approximations of the latent field, though it should be noted that these are currently the default options in INLA and is not necessary to give these arguments.

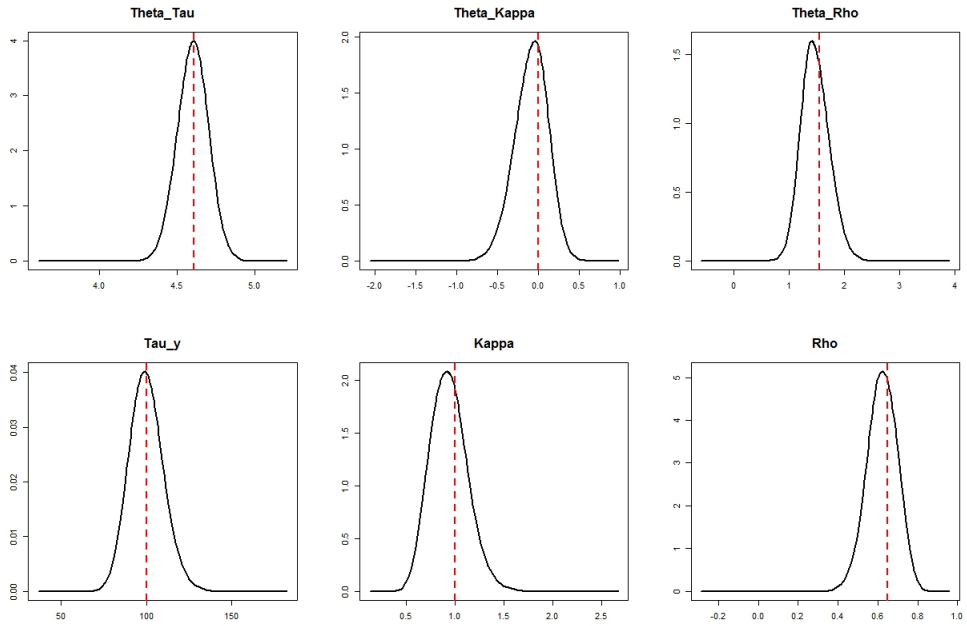

Figure 5.2: Posterior marginal estimates for the hyperparameters of the AR(1) example using R-INLA default settings. The top row are the estimates of $\{\theta_{\tau_y}, \theta_\kappa, \theta_\rho\}$, with the bottom row the estimates for $\{\tau_y, \kappa, \rho\}$. The red-dashed line is the true value of the hyperparameter.

The object `res` stores all of the computations. Using the function `names(res)` outputs a list of elements that can be extracted from the object, including summaries and marginals

of all model parameters, marginal log-likelihood and other useful diagnostics. We can use the `summary` function to output the estimates for the parameters and also show the time used to perform the calculations. The posterior mean of the hyperparameters are very close to the true values. The inferential process using INLA takes less than 1.3 seconds.

```
> summary(res)

Call:
c("inla(formula = y ~ -1 + f(...

Time used:
 Pre-processing    Running inla Post-processing           Total
         0.4500          0.6170          0.2180          1.2851

The model has no fixed effects

Random effects:
Name      Model
 idx   AR1 model

Model hyperparameters:
                                          mean     sd 0.025quant 0.5quant
Precision for the Gaussian observations 99.9298 9.9978    81.4311  99.5465
Precision for idx                        0.9366 0.1889     0.5983   0.9276
Rho for idx                              0.6217 0.0753     0.4686   0.6233
                                        0.975quant    mode
Precision for the Gaussian observations   120.7410 98.9161
Precision for idx                           1.3359  0.9146
Rho for idx                                 0.7619  0.6230

Expected number of effective parameters(std dev): 97.98(0.00)
Number of equivalent replicates : 1.021

Marginal log-Likelihood:  -125.63
```

We can obtain the marginals for the latent field and hyperparameters in two ways. The first is to simply use the plot function on the inla object, `plot(res)`. Another way is to extract them directly from the object itself. The following code performs this task:

```
### Obtain posterior marginals to thetas
theta.tau = res$internal.marginals.hyperpar[[1]]
```

```
theta.kappa = res$internal.marginals.hyperpar[[2]]
theta.rho = res$internal.marginals.hyperpar[[3]]

### Obtain posterior marginals to hyperparameters
tau = res$marginals.hyperpar[[1]]
kappa = res$marginals.hyperpar[[2]]
rho = res$marginals.hyperpar[[3]]

### Approximations to the latent field
latent = res$summary.random
```

The above code extracts the estimated marginals for the reparameterised hyperparameters $\{\theta_{\tau_y}, \theta_\kappa, \theta_\rho\}$, and the true hyperparameters $\{\tau_y, \kappa, \rho\}$. Plotting these estimates gives Figure 5.2, which shows that the marginal posteriors are consistent with the true values. We can also plot the means of the latent parameters, which is shown in Figure 5.3.

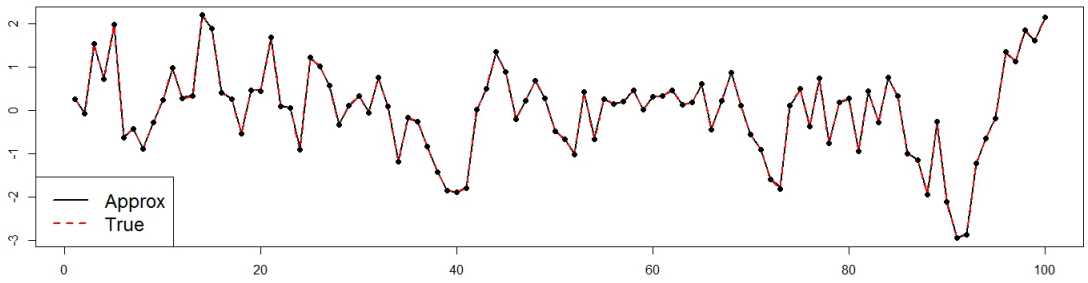

Figure 5.3: Latent field approximation for the AR(1) example. The approximation closely match the data.

The last INLA function of interest to us is the `inla.hyperpar(...)` function, which gives improved estimates of the hyperparameters by re-running INLA but with a grid point set. We demonstrate this with the following code:

```
res.hp = inla.hyperpar(res,
                       diff.logdens = 20,
                       dz = 0.5)
```

The `diff.logdens` and `dz` arguments specify the grid structure. The `diff.logdens` argument determines how far away from the mode you go (see criterion (2.17)), and the `dz`

argument specifying how dense the grid will be. Interested readers can try the code and obtain the summary output and the plots for the object `res.hp`. We obtain only very slightly better estimates, but increase computation time drastically relative to our first estimates (around 9.3 seconds), due to the fact that the grid specified uses upwards of 8000 points compared with the 15 points used previously.

We now move back to LDS-methods and present some modifications to improve performance. We also present an extension to help improve accuracy further with no significant computational burden.

5.2 LDS Method Modifications

The LDS methods presented in the previous chapter do come with some drawbacks. We know that our polynomial approximations converge to the true posterior distribution as the number of points n increases. In practice however, it can be hard to find a suitable degree of polynomial to choose. We can make n partitions along the parameter axis and fit a polynomial of degree $n-1$. If we do not have enough points in each partition, we will not get a good approximation to the pointwise means and thus, the estimation will not be of use. Also, as we increase the number of partitions, the polynomial approximation can become unstable and lead to Runge's phenomenon.

We give details on how the LDS methods can be modified to improve performance and help with the problem of the polynomial degree. Since INLA uses a reparameterisation on each hyperparameter to make the shape more Gaussian, this implies we can fit a quadratic polynomial in the $\log(\boldsymbol{\theta}_z)$-scale. This also suggests that we can make a minimum of three partitions along the parameter axes. Care must be made when finding the pointwise means however. The function evaluations must be found and averaged out in the actual-scale rather than the log-scale as $\log(\text{mean}) \neq \text{mean}(\log)$. The pointwise means are transformed in the $\log(\boldsymbol{\theta}_z)$-scale, then the polynomial can be fitted in this scale before transforming the polynomial from the $\log(\boldsymbol{\theta}_z)$-scale to the $\boldsymbol{\theta}_z$-scale. After normalisation, we can transform back the reparameterised hyperparame-

ters $\boldsymbol{\theta}_z$ back to the original hyperparameters $\boldsymbol{\theta}$ and perform inference on these.

Although the hyperparameter transformation makes the process of marginalisation easier, it is not guaranteed that the hyperparameters $\boldsymbol{\theta}_z$ are perfectly Gaussian. Thus, the strategy of fitting a quadratic very much constricts us to a Gaussian estimate after transformation. To account for potential discrepancies in scale, location and skew, we propose another iteration of the algorithm that fits a cubic polynomial found by analysing the residuals of the quadratic, i.e the difference between the pointwise means and the fitted quadratic. This can be done with little extra computational effort. We discuss details of the so-called polynomial correction in upcoming sections.

5.2.1 Modified LDS for Hyperparameter Estimation

Recall Algorithm 3 in Section 4.3.1. We modify Step 4 to include partitioning, transformations and polynomial fitting with the following algorithm. Since we begin with a reparameterisation of the hyperparameters, Steps 1, 2, and 3 are all performed for the purposes of approximating the marginals $\pi(\theta_{z,i})$. Approximating $\pi(\theta_i)$ simply requires an inverse transformation of the hyperparameter $f^{-1}(\theta_z) = \theta$.

Algorithm 4 Step Four Modifications for LDS-based Bayesian Inference

4a) Orthogonally project function evaluations $\pi(\boldsymbol{\theta}_{z,(j)})$ onto the i^{th} marginal for $\pi(\theta_{z,i}; \boldsymbol{\theta}_{z,(j)})$ for $j = 1, \ldots, N$
4b) Create n partitions of the $\theta_{z,i}$ axis. Here, $[\theta_{z,i,u}, \theta_{z,i,u+1})$ is the u^{th} partition, denoted as $\theta_{z,i,u'}$, and where $u = 1, \ldots, n$
4c) Find pointwise mean for each partition similarly to (4.3), where the pointwise mean of the u^{th} partition is given by $\hat{\pi}(\theta_{z,i,u'})$
4d) Transform pointwise means to $\log(\boldsymbol{\theta}_z)$-scale, and fit a least-squares quadratic polynomial through $\log(\hat{\pi}(\theta_{z,i,u})), u = 1, \ldots, n$
4e) Transform polynomial to θ_z-scale and normalise for each marginal for $\tilde{\pi}(\theta_{z,i})$, then back-transform to θ-scale for $\tilde{\pi}(\theta_i)$

We discuss Algorithm 4 in more detail. After projection of the function evaluation coordinates, we partition the i^{th} marginal space with n partitions with equal interval lengths $\Delta\theta_{z,i}$. We do not assume that each partition will have the same number of points (for example, having

$N = 2^x$ points and $n = 3$ partitions will never yield the same number of points in each partition). However, since we are generating a fully projection LDS that yields equally spaced points in its one-dimensional projections, equally spaced partitions will have a very similar number of points. Let ν_u denote the number of points in the u^{th} partition. Obtaining the pointwise means is very similar to the grid, though the function evaluations do not project onto a single point, but project onto an interval. The natural abscissa point to take would be to take the midpoint of the interval $0.5 \times (\theta_{z,i,u+1} + \theta_{z,i,u})$ which is the one we use in practice. The ordinate for the pointwise mean is given by

$$\hat{\pi}(\theta_{z,i,u'}) = \frac{1}{\nu_u} \sum_{j=1}^{\nu_u} \pi(\theta_{z,i,u'}; \boldsymbol{\theta}_{z,u',(j)}), \tag{5.3}$$

where $\boldsymbol{\theta}_{z,u',(j)}$ are the points sampled that lie in the u^{th} partition. Doing this for all $u = 1, \ldots, n$ partitions giving us n pointwise means.

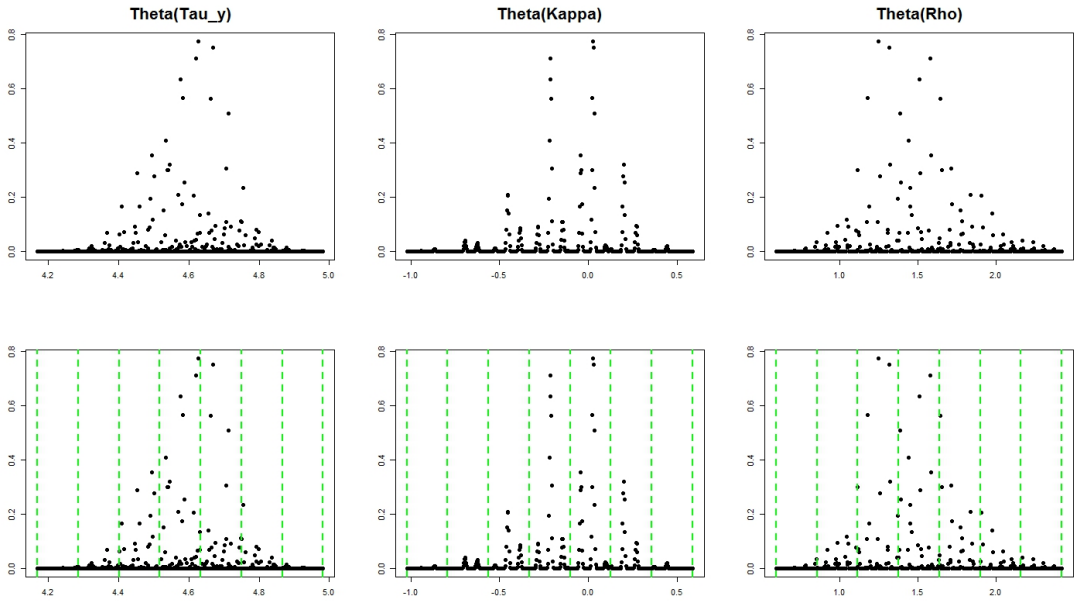

Figure 5.4: An illustration of Step 4a (orthogonal projection) and Step 4b (partitioning) of Algorithm 4. We orthogonally project function evaluations onto each marginal space, and partition each space. In this example, we have $n = 7$ partition of equal length and $N = 512$ points.

We transform the n pointwise means to the $\log(\boldsymbol{\theta}_z)$-scale, and fit the quadratic polynomial

via least-squares. The quadratic polynomial is given by

$$\tilde{\pi}_{\log}(\theta_{z,i}) = \tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}\theta_{z,i} + \tilde{\beta}_{i,2}\theta_{z,i}^2, \tag{5.4}$$

where $\tilde{\pi}_{\log}(\theta_{z,i})$ is the unnormalised log-approximation to the marginal posterior $\pi(\theta_{z,i})$, in which the coefficients $\tilde{\beta}$ are found through a least-squares approximation. The quadratic is then transformed back to the $\boldsymbol{\theta}_z$-scale and normalised, which gives the approximation to the reparameterised hyperparameter,

$$\tilde{\pi}_{MLS}(\theta_{z,i}) = \frac{\exp(\tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}\theta_{z,i} + \tilde{\beta}_{i,2}\theta_{z,i}^2)}{\int_{[a_i,b_i]} \exp(\tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}\theta_{z,i} + \tilde{\beta}_{i,2}\theta_{z,i}^2) d\theta_{z,i}}, \tag{5.5}$$

where $\tilde{\pi}_{MLS}(\theta_{z,i})$ is the approximation of the marginal posterior for $\theta_{z,i}$ via the modified LDS method. For the approximation of the true hyperparameter, let $f^{-1}(\theta_z) = \theta$. So, we have

$$\begin{aligned}
\tilde{\pi}_{MLS}(f^{-1}(\theta_{z,i})) &= \frac{\exp(\tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}f^{-1}(\theta_{z,i}) + \tilde{\beta}_{i,2}(f^{-1}(\theta_{z,i}))^2)}{\int_{[a_i,b_i]} \left(\exp(\tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}f^{-1}(\theta_{z,i}) + \tilde{\beta}_{i,2}(f^{-1}(\theta_{z,i}))^2)\right) d\theta_{z,i}} \\
&= \frac{\exp(\tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}\theta_i + \tilde{\beta}_{i,2}\theta_i^2)}{\int_{[a_i,b_i]} \exp(\tilde{\beta}_{i,0} + \tilde{\beta}_{i,1}\theta_i + \tilde{\beta}_{i,2}\theta_i^2) d\theta_i} \\
&= \tilde{\pi}_{MLS}(\theta_i)
\end{aligned} \tag{5.6}$$

Example 5.2.1. *We demonstrate the process for Algorithm 4 by continuing with the AR(*1*) example described in Section 5.1.1. For now, we use INLA output from the* `res` *object to construct a suitable support boundary in which to generate a Korobov Lattice, as well as calculate the function evaluations for each point in the lattice. Figure 5.4 shows the projected function evaluations for each hyperparameter (note that we have evaluated for hyperparameters* $\boldsymbol{\theta}_z$*) and the partitioning of the marginal space. We have* $n = 7$ *partitions and* $N = 512$ *points in this example. Note that all partitions have the same number of points, except for the first partition that has one extra point, as* $512 = 74 + (6 \times 73)$.

Figure 5.5 shows us the pointwise mean coordinates (red points). We can see that the abscissa point is the midpoint of each partition interval, with the pointwise mean ordinate given by (5.3)*. The logs are taken of the pointwise mean coordinates, and the quadratic polynomial is*

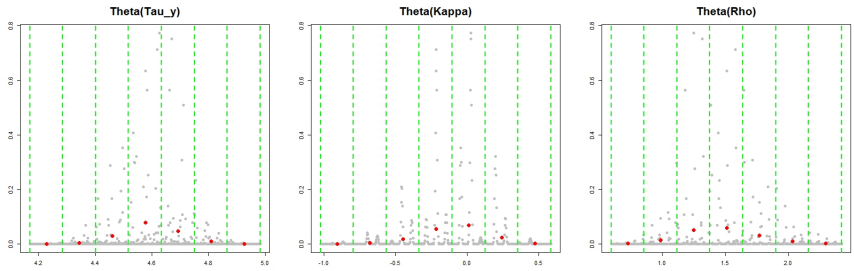

Figure 5.5: Step 4c in Algorithm 4. For each marginal, and for each partition, find the pointwise means. This is done in the $\boldsymbol{\theta}_z$-scale.

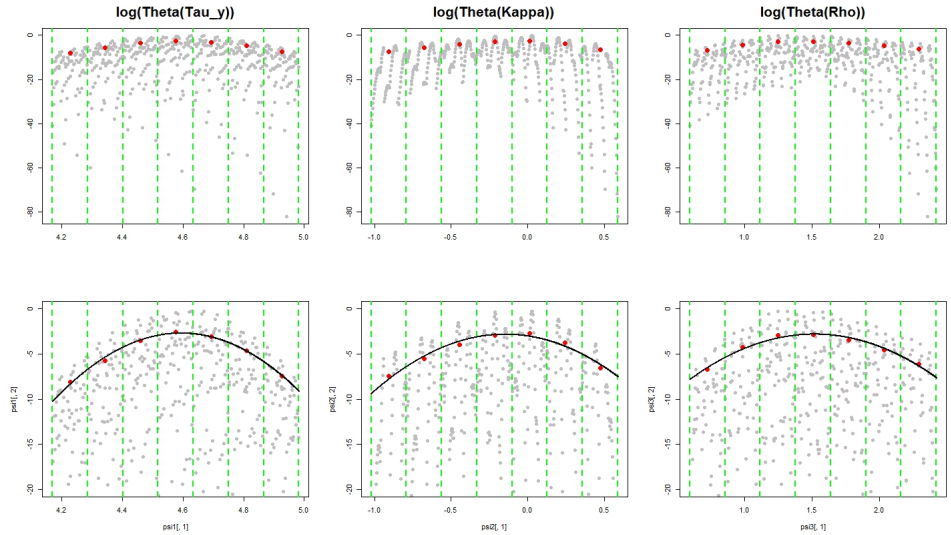

Figure 5.6: Transformation of the pointwise means in the log-scale, and fitting a quadratic polynomial through the pointwise means. The top set of plots show the pointwise means (red points) with the logged function evaluations (grey points). The bottom set of plots show each set of pointwise means fit with a quadratic polynomial. This is Step 4d in Algorithm 4.

fitted through them, as shown in Figure 5.6. The top plots in Figure 5.7 give our approximation given by (5.5)*, which is the quadratic transformed back to* $\boldsymbol{\theta}_{\boldsymbol{z}}$*-scale and normalised (red solid line). The bottom plots show the approximations transformed back to* θ *from* θ_z*. We compare this to INLA's grid approximation. Note that we use the* `inla.hyperpar` *function and set a very dense grid (over 8000 points), thus these are very close to what we would consider the true marginals. Our modified approach approximations are close to INLA's approximations, with both approximating the marginals and parameter estimates well (parameter estimates are shown in Table 5.1). The main drawback however is that we cannot capture any skewness of the reparameterised hyperparameters, since we are only fitting a quadratic polynomial.*

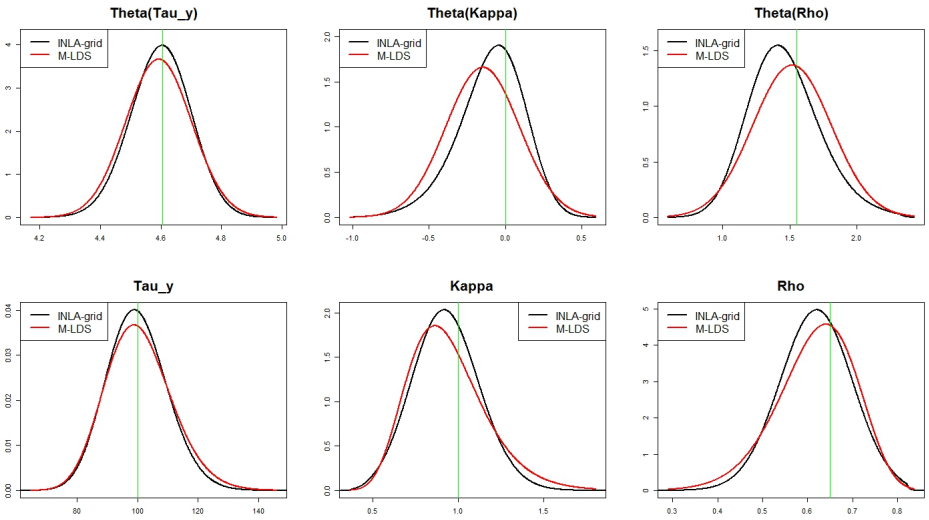

Figure 5.7: The final posterior marginal approximations, in both $\boldsymbol{\theta}_z$ and $\boldsymbol{\theta}$ scale. We compare our approximations (M-LDS, red lines) with INLA's grid approximation (INLA-grid, black lines). The true value is given by the green line.

Table 5.1: Posterior means of the $\boldsymbol{\theta}_z$ marginals for both INLA and modified LDS, with comparisons to the true value of each hyperparameter.

Parameter	True	INLA grid	Modified LDS
θ_{τ_y}	4.6052	4.6042	4.5939
θ_κ	0	-0.0735	-0.1296
θ_ρ	1.5506	1.4409	1.5179

5.2.2 Cubic Correction

The hyperparameter marginal approximations using the method set out in Algorithm 4 gives a Gaussian approximation to the reparameterised hyperparameter marginals. However, it is not guaranteed that all hyperparameters $\boldsymbol{\theta}_z$ will have a Gaussian shape. We propose a correction to the quadratic approximation by analysing the residuals, given by the difference between the polynomial fit given by (5.4) and the pointwise means. We can view (5.4) as the initial approximation that can be updated by a cubic polynomial to correct for location, scale and skew.

The process changes Algorithm 4 slightly, by adding the polynomial correction process after Step 4d. The polynomial correction is summarised in Algorithm 5. Since we have n

Algorithm 5 Cubic Correction to Quadratic Approximation

4d) Transform pointwise means to $\log(\boldsymbol{\theta}_z)$-scale, and fit a least-squares quadratic polynomial through $\log(\hat{\pi}(\theta_{z,i,u}))$, $u = 1, \ldots, n$
4d) (i) Obtain residuals found by the difference between $\tilde{\pi}_{\log}(\theta_{z,i})$ evaluated at u^{th} abscissa point and the corresponding pointwise mean
4d) (ii) Fit a cubic polynomial through residuals via least-squares
4d) (iii) Correct initial quadratic approximation with the cubic coefficients

pointwise means, we can find the u^{th} residual for marginal i, denoted as $R_u(\theta_{z,i})$, by

$$R_u(\theta_{z,i}) = \tilde{\pi}_{\log}(\theta_{z,i})|_u - \log(\hat{\pi}(\theta_{z,i,u})) \tag{5.7}$$

where $\tilde{\pi}_{\log}(\theta_{z,i})|_u$ is the quadratic polynomial evaluated at the u^{th} abscissa point, and $\log(\hat{\pi}(\theta_{z,i,u}))$ is the u^{th} pointwise mean. The cubic polynomial fitted through the residuals is done via least-squares, which we denote as

$$P_R(\theta_{z,i}) = \beta'_{i,0} + \beta'_{i,1}\theta_{z,i} + \beta'_{i,2}\theta^2_{z,i} + \beta'_{i,3}\theta^3_{z,i}. \tag{5.8}$$

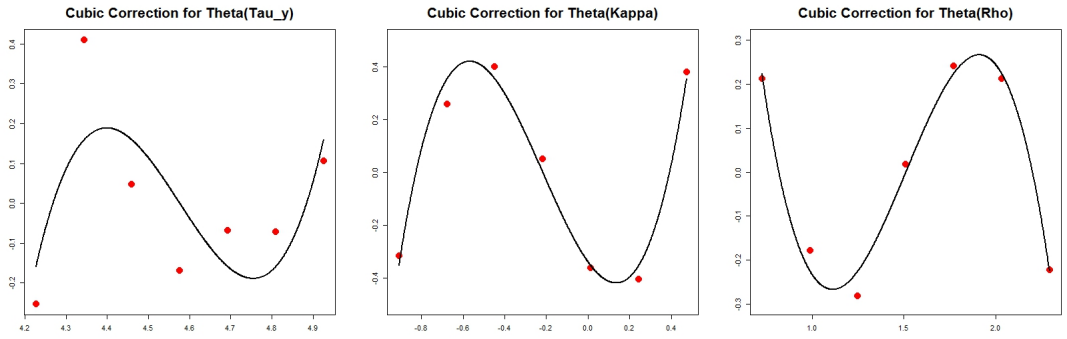

Figure 5.8: Fitting a cubic polynomial through the residuals of the quadratic approximation and the log pointwise means. The coefficients of the cubic are used to update the quadratic.

The polynomial correction update for the unnormalised log-approximation to the marginal posterior $\pi(\theta_{z,i})$ is

$$\tilde{\pi}_{\log,P}(\theta_{z,i}) = (\tilde{\beta}_{i,0} - \beta'_{i,0}) + (\tilde{\beta}_{i,1} - \beta'_{i,1})\theta_{z,i} + (\tilde{\beta}_{i,2} - \beta'_{i,2})\theta^2_{z,i} + \beta'_{i,3}\theta^3_{z,i}. \tag{5.9}$$

From here, Step 4e in Algorithm 4 can be performed to find the final approximations by using

the expression in (5.9) and substituting into (5.5) and (5.6).

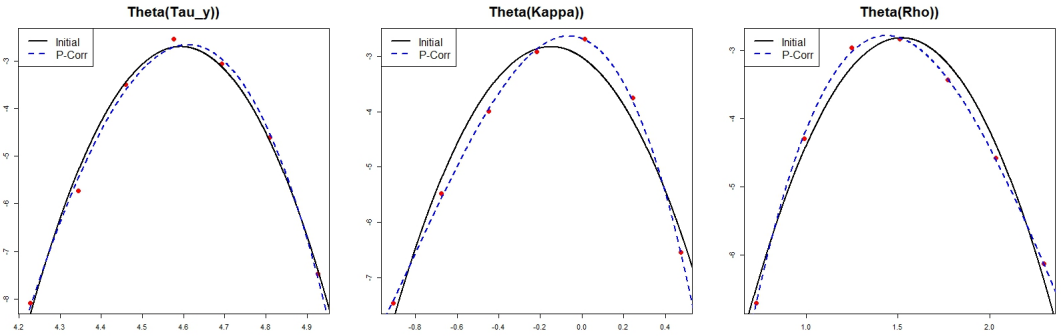

Figure 5.9: Cubic correction for all three hyperparameter approximations. The residuals described in (5.7) are given by the red points.

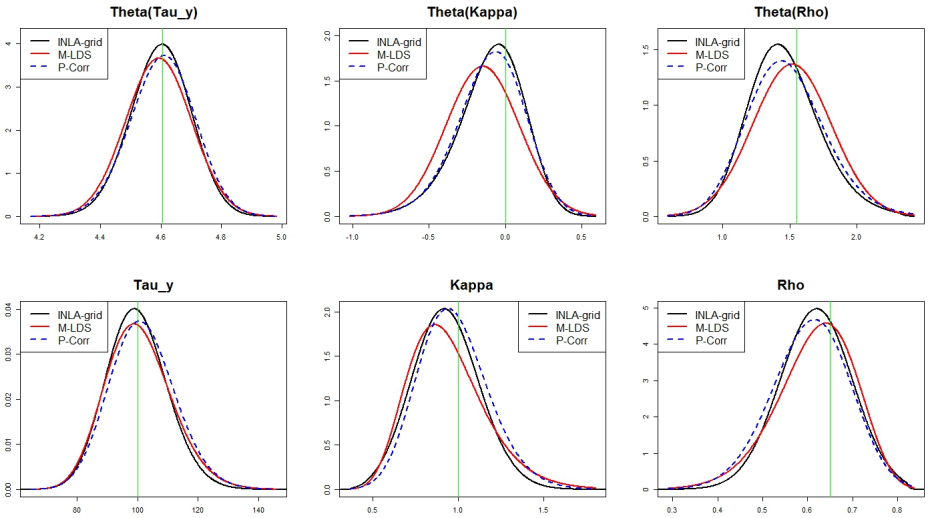

Figure 5.10: Marginal posterior approximations with INLA grid (black solid line), modified LDS fit (red solid line) and the cubic correction (blue dashed line). The cubic correction moves away from the quadratic fit and towards INLAs grid.

Example 5.2.2. *Continuing from Example 5.2.1, where Figure 5.7 gave us the initial approximation given by* (5.5)*. We update this approximation by going back to the quadratic fit in Figure 5.6 and updating this with a cubic polynomial fit through the residuals. The cubic polynomials used to correct the quadratic are shown in Figure 5.8. Figure 5.9 shows both the initial fit and the cubic correction. It is clear to see that the correction fits the pointwise means much closer, especially for* θ_κ *and* θ_ρ*. Performing the necessary transformation yields the final results in*

Figure 5.10. The cubic corrections (blue dashed lines) have moved away from the initial fit and moved towards INLA's dense grid approximation.

5.2.3 Latent Field Approximation

A recent update by the INLA developers have allowed the user to define their own points in which to evaluate the hyperparameters, rather than setting up a grid or CCD pointset. Since these points are used in the numerical integration process by (2.16) this allows us to use any point set to evaluate the latent field posteriors. Note that INLA will not use the user-defined points to estimate the hyperparameter marginals, but will estimate using NIFA (see Section 2.3.4). Details for the user-defined points can be found by running the R-script `vignette("int-design", package = "INLA")`, which gives details on how to construct the data frame of points with corresponding weights. Let `design` be the data frame of our user defined points and weights. The following code runs the same AR(1) example but with the `design` point set:

```
### Libraries ###
library(INLA)

### Formula ###
formula = y ~ -1 + f(idx, model="ar1",
                  hyper = list(rho = list(fixed = FALSE,
                                          prior = "betacorrelation",
                                          param = c(5, 1)),
                               prec = list(fixed = FALSE,
                                          prior = "loggamma",
                                          param = c(1, 1))))

### inla(...) call ###
res = inla(formula, data = data, family = "gaussian",
          control.family = list(hyper = list(prec = list(
                                       fixed = FALSE,
                                       prior = "loggamma",
                                       param = c(100, 1)))),
          control.inla = list(int.strategy = "user.std", int.design = design,
                              strategy = "simplified.laplace", force.diagonal = TRUE))
```

Here, we assume the design point set corresponds to the points in the $\boldsymbol{\theta}_z$-scale, thus using the command `int.strategy = "user.std"` in the `control.inla` argument. If we

generated the points in the typical $\boldsymbol{\theta}$-scale, we would run the command `int.strategy = "user"`. Figure 5.11 is the output from running the model again, using all the available options for `int.strategy` (emperical Bayes, CCD and grid), as well as using the above `res` call, with a LDS design ($\mathcal{L}_{64,3}$), all having equal weights. We see little to no difference between all strategies. However, we must note that it is important that our support for the design must be in the same region as the grid or CCD. Taking a support too large or too small can result in some deviation from the other approximations.

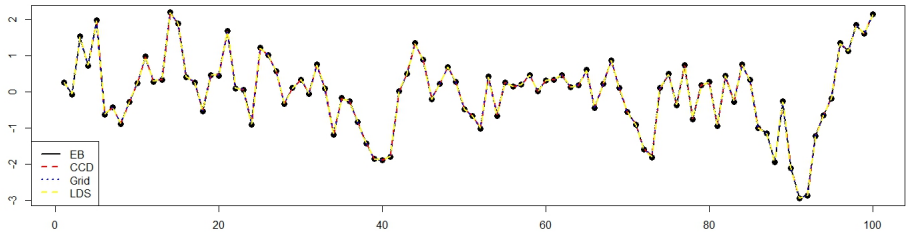

Figure 5.11: Latent field approximation for the AR(1) example using all available integration strategies. There is little to no difference in the approximations between the three INLA strategies and using the LDS.

5.3 Applications and Results

We have proposed modifications to the LDS-based methods described in Chapter 4, first by partitioning and fitting a quadratic as our approximation to the reparameterised hyperparameters $\boldsymbol{\theta}_z$, then by updating the approximation via analysing residuals and correcting the quadratic with a cubic. The AR(1) example used exclusively in the previous sections showed that the modifications yielded good approximations. We apply the modifications and the cubic correction to two more examples, with the aim of demonstrating further the new methodology for estimating hyperparameter marginal posteriors. The first example is a case study of a spatial analysis of childhood undernutrition in Zambia. The second is a spatio-temporal study of low birth weights in Georgia. Information, details and data for the Zambia example can be found on the R-INLA website and [54]. Details and links to the data and materials for the Georgia example can be found in [8].

5.3.1 Child Undernutririon in Zambia

The Zambia dataset was first introduced by [40] who used spatial factors to analyse undernutrition among children in the 57 regions that comprise Zambia. Child undernutrition is measured by the height of a child relative to their age. A Z-score is used to determine the stunting of a child, which is defined by

$$Z_i = \frac{AI_i - MAI}{\sigma}, i = 1 \ldots, n_d$$

where AI_i is the i^{th} child's anthropometric indicator (height relative to age), MAI and σ are the median and standard deviation of the referenced population. We assume the scores are conditionally independent Gaussian random variables with unknown mean η_i and unknown precision τ_z.

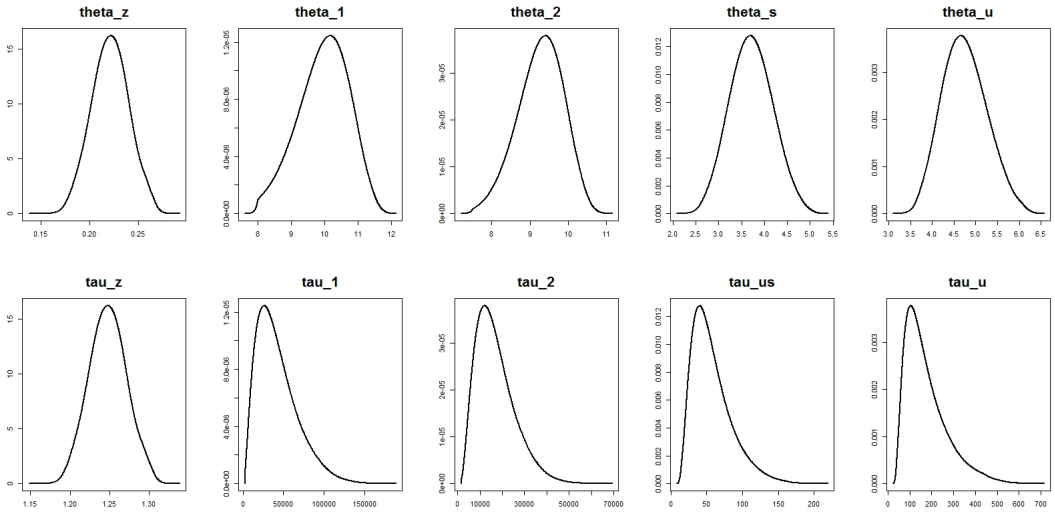

Figure 5.12: INLA approximations of marginal posteriors for $\boldsymbol{\theta_z}$ and $\boldsymbol{\theta}$ in the Zambia model. For this, we use an extremely dense grid and consider these the most accurate approximation.

Several factors are considers such as age, body mass index of the child's mother, and several categorical variables including gender, education, mothers employment status and locality. This dataset has been used by [41] as an introduction to *BayesX* for the analysis of Bayesian semiparametric regression using MCMC techniques. It was also used to introduce the same

idea using INLA. In both studies, they considered the model presented by [40]

$$\boldsymbol{\eta} = \mu + \mathbf{z}_i^T\boldsymbol{\gamma} + bmi_i \times f_1(district_i) + f_2(age_i) + f_s(district_i) + f_u(district_i). \quad (5.10)$$

Here, μ is the overall mean, $\mathbf{z}_i^T\boldsymbol{\gamma}$ represent all covariates $\mathbf{z}$ as having a linear effect. Also, $f_u(district_i)$ is the spatially unstructured component that is i.i.d Gaussian distributed with mean 0 and unknown precision τ_u, and $f_s(district_i)$ is the spatially structured component which varies smoothly from region to region. This is modelled as an intrinsic Gauss Markov random field (IGMRF) – a conditional autoregressive (CAR) prior [7] – with unknown precision τ_s. Previous studies believed that age covariate has a non-linear effect, and that the bmi covariate can be used as a weight for the IGMRF $f_1(\cdot)$. Both of these components have unknown precision τ_1 for bmi and τ_2 for age.

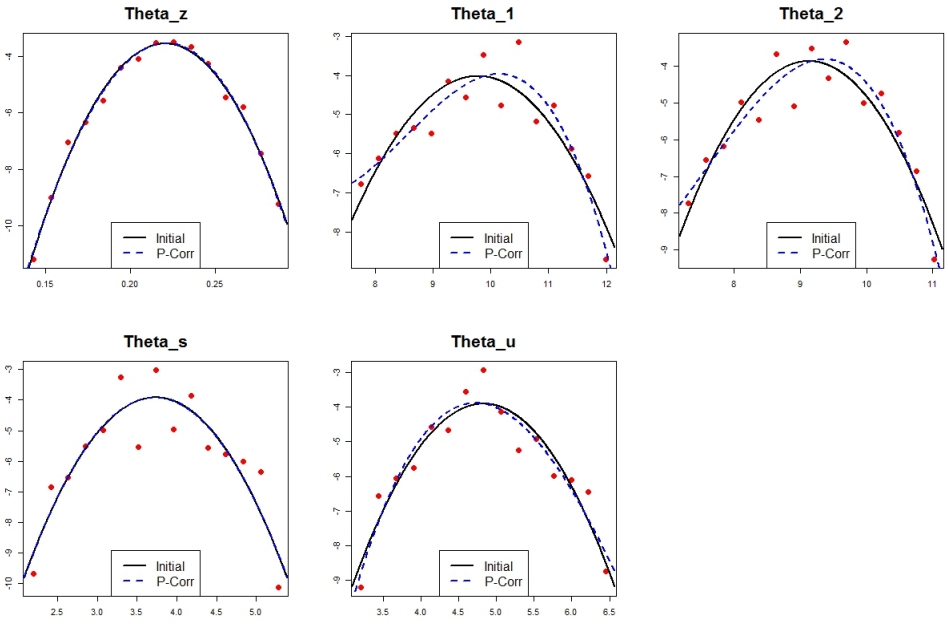

Figure 5.13: Zambia: Plots of initial quadratic approximation (black solid line) and cubic correction (blue dashed line) with the red points as pointwise means.

The latent Gaussian field for (5.10) is $\boldsymbol{\phi} = \{\mu.\{\boldsymbol{\gamma}\}, \{f_u(\cdot)\}, \{f_s(\cdot)\}, \{f_1(\cdot)\}, \{f_2(\cdot)\}, \boldsymbol{\eta}\}$, with hyperparameters $\boldsymbol{\theta} = \{\tau_Z, \tau_u, \tau_s, \tau_1, \tau_2\}$. We assign vague Gamma priors for each element. We are interested in estimating the posterior marginal for each hyperparameter. We

will first use the grid strategy in INLA before using the methods proposed in Algorithm 4 and Algorithm 5. The reparameterisation INLA uses for the precision hyperparameters is the $\log$ function, thus $\boldsymbol{\theta}_z = \{\log(\tau_Z), \log(\tau_u), \log(\tau_s), \log(\tau_1), \log(\tau_2)\}$. Note that the grid we use is extremely dense (a total of 59442 points), and as such we consider this an extremely accurate approximation (INLA grid approximations shown in Figure 5.12).

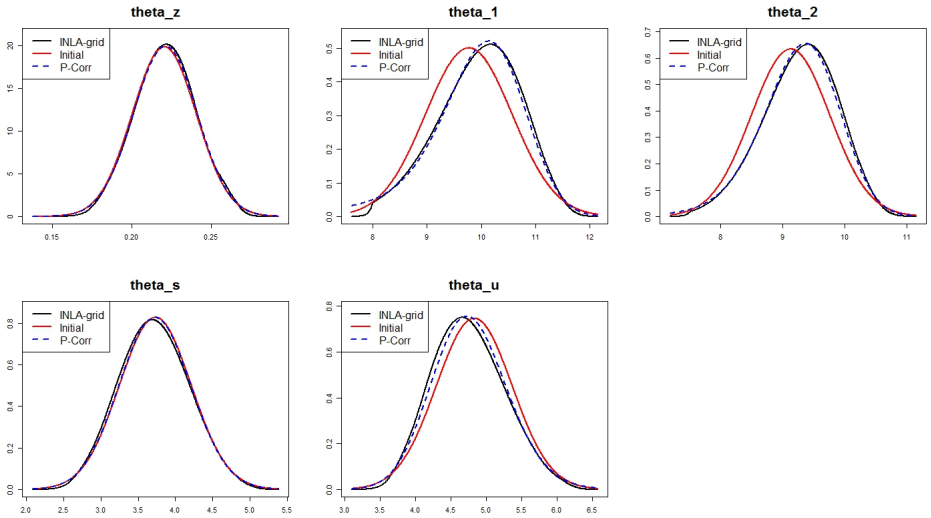

Figure 5.14: Zambia: Marginal posterior approximations for the elements of $\boldsymbol{\theta}_z$, showing the INLA dense grid approximation (black), initial fit (red) and the cubic correction (blue-dashed). In all cases, the cubic correction performed well with respect to INLA's dense grid.

Initial Results

We apply the steps in Algorithm 5 by updating the quadratic approximation with the coefficients found by fitting a cubic through the residuals. Prior to this, we generate an embedded Korobov Lattice $\mathcal{K}_{512,5}$, for which the generating constant $\alpha_z = 19$. Note that 512 points is less than 116 times the number of points used by INLA's dense grid. For the puposes of this exercise, we use the approximations of the mode and Hessian found by INLA to generate the support (± 3 standard deviations from the mode on each $\boldsymbol{\theta}_z$-axis). We also use INLA to manually compute the function evaluations for each point in $\mathcal{K}_{512,5}$. After projecting function evaluations onto each axis, we make 15 equally spaced partitions and find the pointwise means. The logged pointwise means and the initial quadratic polynomial fit through those points are shown in Figure 5.13

with the red points and solid black line respectively.

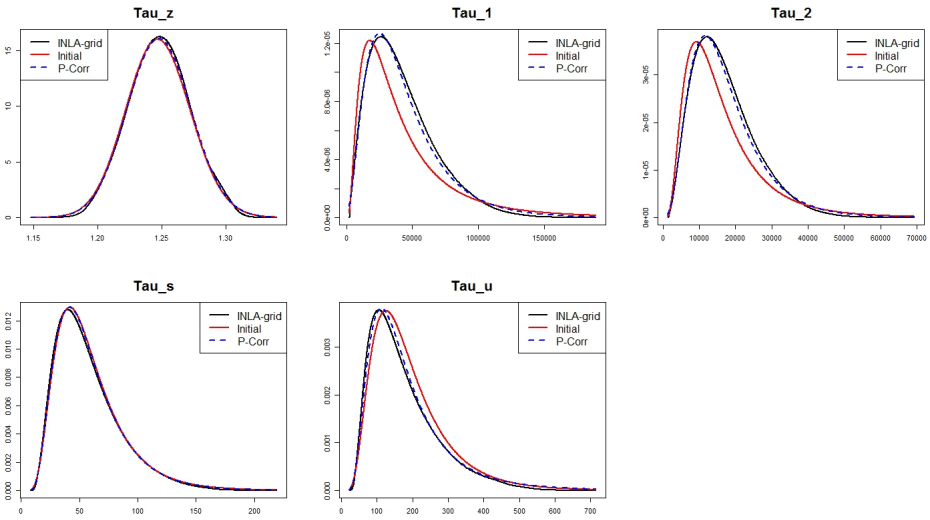

Figure 5.15: Zambia: Marginal posterior approximations for the elements of $\boldsymbol{\theta}$ after transformation from $\boldsymbol{\theta}_z$.

A cubic polynomial is fitted to the residuals of the initial fit and the coefficients are used to update the initial fit. The updated fit is shown in Figure 5.13 as the dashed blue line. The initial fit for θ_Z and θ_s are very good, and as such the cubic correction does not update them. There is a slight skew to θ_1 and θ_2 which cannot be captured by the initial fit. However, the cubic correction captures this skew well. The cubic correction also updates θ_u slightly, with a small shift in location and skew. The transformations from the approximations in the $\log(\boldsymbol{\theta}_z)$-scale to $\boldsymbol{\theta}_z$ scale shows how the initial approximation (red solid line) has shifted towards INLAs dense grid approximation (Figure 5.14). It especially highlights how the cubic correction has approximated the two more heavily skewed posterior marginals θ_1 and θ_2. For completeness, we give the results for the second transformation from $\boldsymbol{\theta}_z$ to $\boldsymbol{\theta}$ in Figure 5.15, which are the final approximations and follow from the results shown in Figure 5.14.

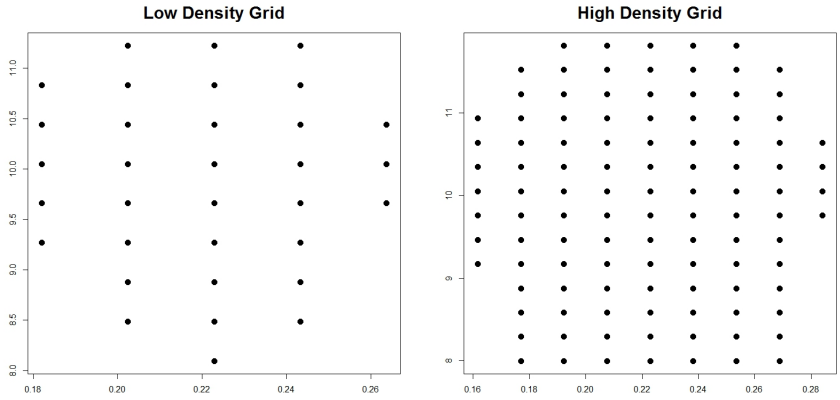

Figure 5.16: Zambia: The grid structures in the dimensions θ_Z and θ_1 for INLA's low density grid and INLA's high density grid.

Comparisons with INLA Low Density Grid

We give comparisons of the cubic correction with a grid that is less dense than that given earlier in this section, and has a more comparative number of points to our Korobov Lattice. We choose a grid that has 2655 points which is around five times more than that of our Korobov lattice and is far less dense than that of our earlier grid. Figure 5.16 gives the density of the grid in just two dimensions (θ_Z and θ_1). Whilst this might not look like much difference, remember that the number of points increases exponentially with dimension. The resulting approximations shown in Figure 5.17 show that the less dense grid failed to capture the shape and spread for all posterior marginals and is outperformed by the cubic correction, despite having less points. This is consistent with the results of our method even in its most vanilla form.

Comparisons with Numerical Integration Free Algorithm (NIFA)

NIFA (2.18) is the current default setting in INLA for the estimation of hyperparameters (see Section 2.3.4). Though it is extremely fast and quite accurate, there is some room for improvement with respect to accuracy. We compare the cubic correction with NIFA in Figure 5.18, and give Kullback-Leibler divergence and Hellinger distance results in Table 5.2. Note that we only show the $\boldsymbol{\theta}_z$ approximations. It follows that the distance measure between two distributions is invariant under transformation.

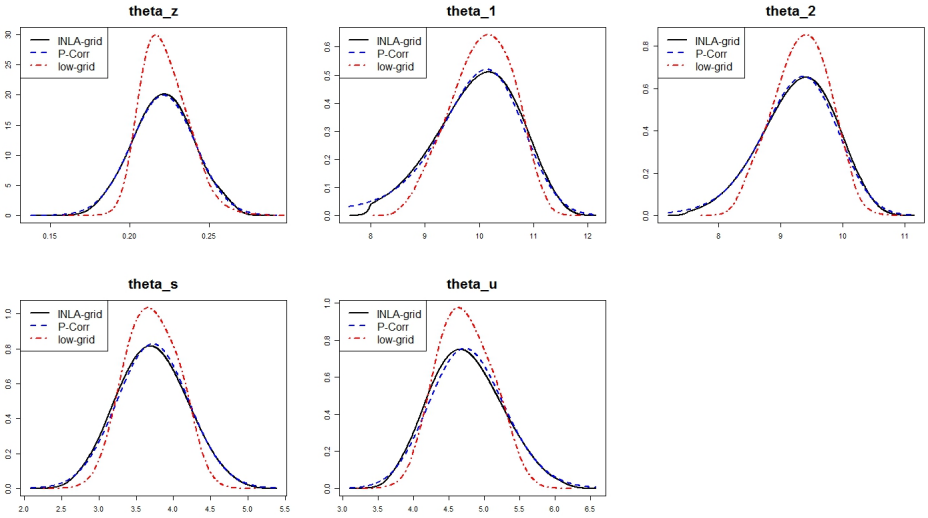

Figure 5.17: Zambia: Comparisons between INLA's low density grid and the cubic correction. The low density grid approximations are poor, despit having roughly five times more points than the cubic correction.

The results show that there is very little difference between the NIFA and the cubic correction. From the naked eye, there is very little between the approximations. However, the Kullback-Leibler divergence and Hellinger distances in Table 5.2 shows that for all marginals, the cubic correction did approximate better than the NIFA. The NIFA approximation is fixed, thus this is the best approximation this method can give. It also assumes that the marginal is Gaussian, with different standard deviations on each side of the mode. Therefore, this approximation can only give univariate estimates to $\pi(\theta_{z,i}|\mathbf{y})$. We give an example of this problem in the second application.

Table 5.2: Zambia: Distance measures comparing INLA's dense grid with both the NIFA and cubic correction methods. The cubic correction gave the more accurate approximations for each hyperparameter marginal according to both the Kullback-Leibler divergence and Hellinger distances, though the differences are very small.

	K-L.Div		H.Dist	
Parameter	NIFA	Cubic	NIFA	Cubic
θ_Z	0.00501	0.00329	0.03961	0.03233
θ_1	0.01194	0.00495	0.05664	0.04088
θ_2	0.00329	0.00290	0.03058	0.02964
θ_s	0.01199	0.00248	0.05797	0.02655
θ_u	0.00787	0.00533	0.04953	0.03967

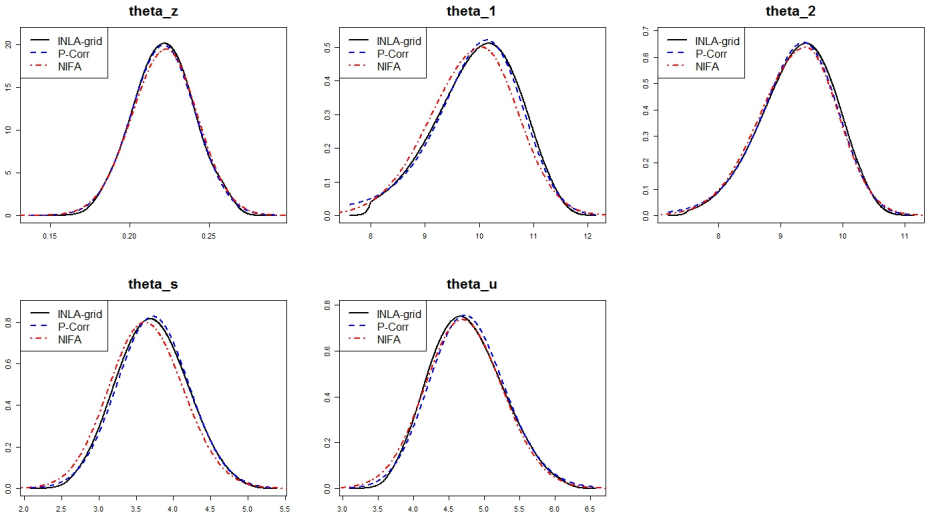

Figure 5.18: Zambia: Comparisons between INLA's NIFA and the cubic correction. They are both accurate, though the cubic correction approximates slightly better for some marginals.

Embedded Lattices for Extra Computational Efficiency

A small illustration here on how we can effectively decrease the number of points without the hassle of re-computation. Recall from Section 3.2.1, that an extensible, or embedded lattice, can be generated to give the user the ability to add more points without discarding all the old points. By the same token, we can also take points away. For example, let $\mathcal{K}^*_{2^x,s}$ be an embedded Korobov lattice with 2^x points in s dimensions. Expressing the point set in matrix form as in (4.4), keeping the first row and taking out every second row after will give $\mathcal{K}^*_{2^{x-1},s}$, and keeping the first row and keeping every fourth row will give $\mathcal{K}^*_{2^{x-2},s}$, and so on.

As an example, since we have $\mathcal{K}^*_{2^9,5}$, we may decide that $2^9 = 512$ points is too many. We keep the first row of the Korobov lattice, and proceed to keep every 16^{th} point afterwards, leaving us with $\mathcal{K}^*_{2^6,5}$, a Korobov lattice with 64 points. We present the posterior marginal approximations in Figure 5.19. This was our best approximation with such few points, and the posterior marginals were very close to the dense grid.

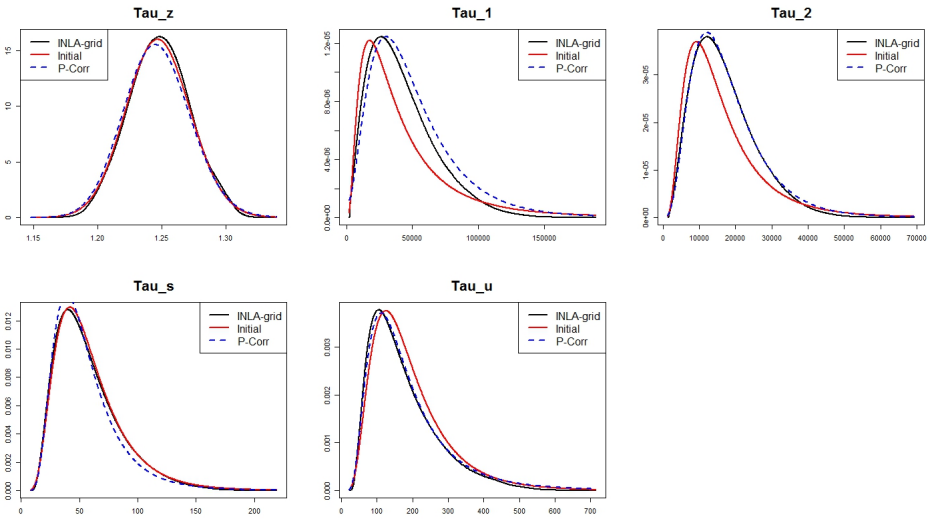

Figure 5.19: Zambia: Posterior approximations using $\mathcal{K}^*_{64,5}$. Whilst the approximations are not as precise as the approximations using $\mathcal{K}^*_{512,5}$, we are using much less points.

5.3.2 Low Birth Weight Counts in Georgia

This example considers the count of new-borns with very low birth weight (less than 2500gm) in the counties of Georgia, USA. The data was collected over ten years from 2000-2010, with the aim to perform spatio-temporal disease mapping. This particular example has been used by [8] to illustrate a spatio-temporal Poisson nonparametric approach. This includes interactions between different spatial and temporal components.

Our model is a space-time interaction model, which has the form

$$\eta_{i,t} = \mu + f_s(county_i) + f_u(county_i) + f_{t_1}(year_t) + f_{t_2}(year_t) + f_{u,t_2}(area_i \times year_t),$$

where $i = 1, \ldots, 159$ and $t = 1, \ldots, 11$. We have μ as the overall mean, $f_s(county_i)$ as the spatially structured component, modelled as a conditional autoregressive prior with unknown precision τ_s, and $f_u(county_i)$ as the spatially unstructured component modelled as i.i.d Gaussian with mean 0 and unknown precision τ_u. Out time components are $f_{t_1}(year_t)$, which is modelled as a random walk of order two [49] with unknown precision τ_{y_1}, and $f_{t_2}(year_t)$ is the unstructured time effect modelled as i.i.d Gaussian with mean 0 and precision τ_{y_2}. We also have

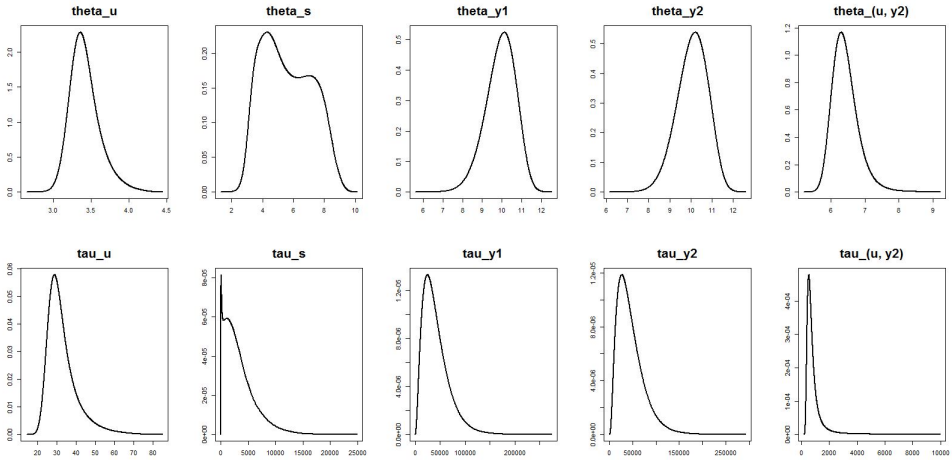

Figure 5.20: INLA's approximation to the marginal posteriors for each hyperparameter in the Georgia model. With a very dense grid, the second posterior is shown to be multimodal.

an interaction term $f_{u,t_2}(area_i \times year_t)$. We choose an interaction between the unstructured space and unstructured time variables, thus placing no spatial or temporal structure on the interaction, and so modelling this as i.i.d Gaussian with mean 0, and unknown precision τ_δ. Thus we have five hyperparameters $\boldsymbol{\theta} = \{\tau_u, \tau_s, \tau_{y_1}, \tau_{y_2}, \tau_\delta\}$. Taking the log of each component will give $\boldsymbol{\theta}_z = \{\theta_u, \theta_s, \theta_{y_1}, \theta_{y_2}, \theta_\delta\}$.

Again, we use a dense grid within INLA to approximate the marginals. The approximations are shown in Figure 5.20. We see from the outset that we have a multimodal marginal posterior for τ_s which the dense grid has captured. To capture this, we needed quite a dense grid, and as such used over 100,000 grid points. We first give our initial approximations before giving a potential solution to the problem of multimodality.

Initial Results

We take the same approach as we did in the Zambia example by generating an embedded Korobov Lattice $\mathcal{K}^*_{512,5}$, computing the function evaluations and partitioning making 15 equally spaced partitions for each axis. We follow the processes outlined in Algorithm 4 and Algorithm 5, by fitting an initial quadratic polynomial through the pointwise means, finding the residuals, fitting a cubic polynomial and updating the initial approximation. We display the approxima-

tions for the components of $\boldsymbol{\theta}_z$ in Figure 5.21. The initial approximations were not good as some of the components of $\boldsymbol{\theta}_z$ were quite skewed. However, we see that the cubic correction was able to rectify this well and gave much more accurate approximations. Of course, the approximation of θ_s was not appropriate due to it being multimodal in shape.

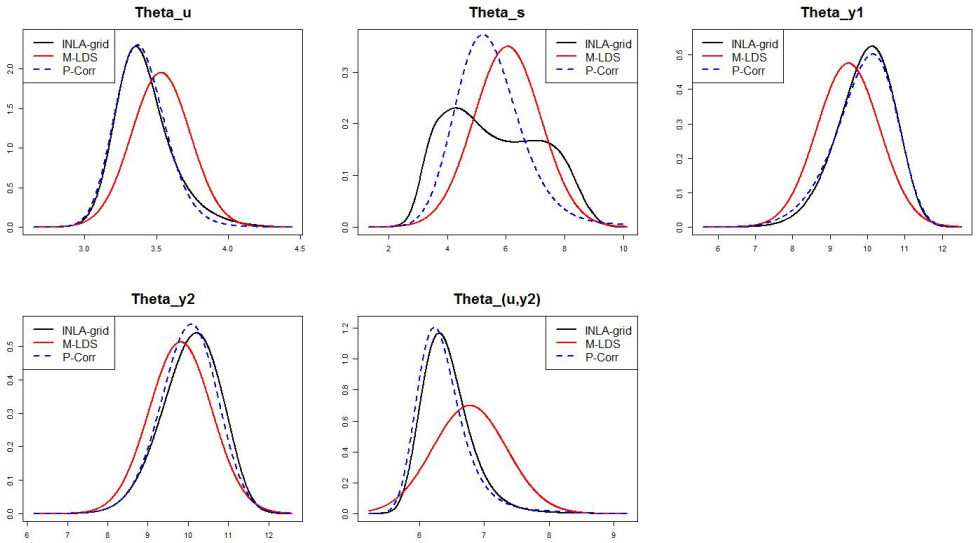

Figure 5.21: Georgia: Initial Approximation and cubic corrections for $\boldsymbol{\theta_z}$ (red solid and blue dashed lines respectively), and comparisons with INLA's grid. The cubic correction was needed as the initial fits were not very good. However, we cannot approximate θ_s well, as expected.

Multimodal Hyperparameters

Up until now we have chosen to fit a cubic polynomial to the residuals to correct our initial quadratic approximation for skewness. If a density is unimodal, this is all that is necessary. However, for special cases such as multimodal densities, we can fit a higher order polynomial correction. Since we fit each hyperparameter independently of the other, this is easily done. We focus solely on the marginal θ_s and fit both a quartic and quintic correction.

Figure 5.22 shows the process of fitting the marginal τ_s using three different corrections. As shown here and in the initial results, the cubic correction was unable to capture the multimodal shape. The quartic polynomial has up to three turning points, so can detect up to two modes. It

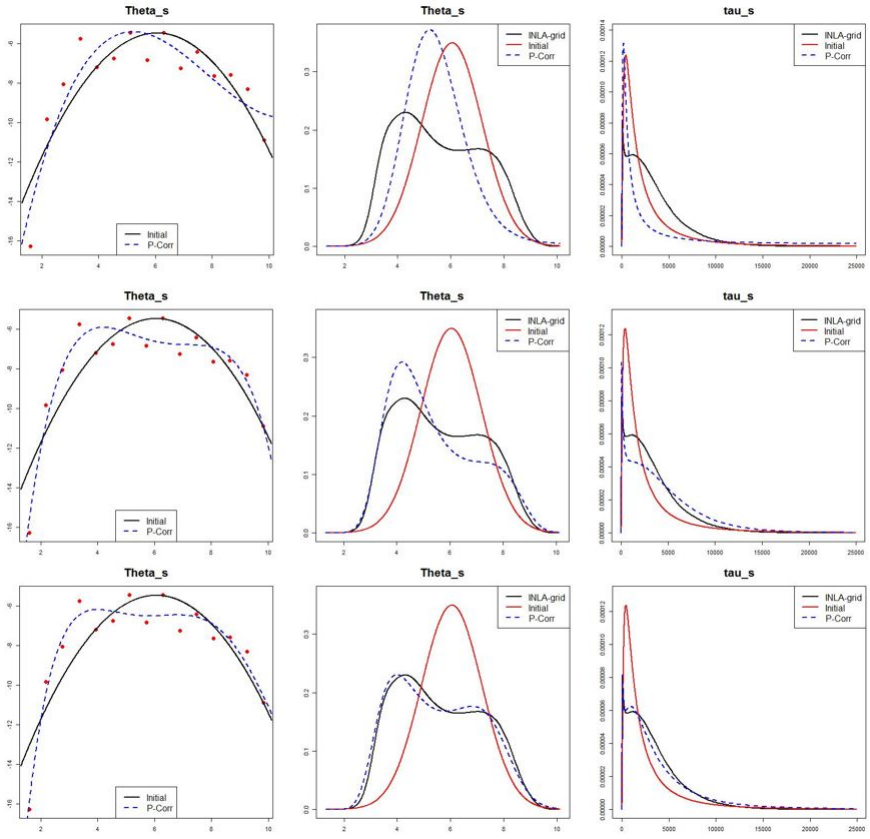

Figure 5.22: Georgia: The process of approximating τ_s by fitting a cubic correction (top row), a quartic correction (middle row) and quintic correction (bottom row). Note that the first column is fitting the initial quadratic, and the polynomial update, whilst the second and third columns are approximating the θ_s and τ_s respectively. Going to a higher order polynomial worked well for approximating the multimodal density.

did not quite capture both modes. However it was a much better approximation than the cubic. Finally the quintic correction detected both modes and gave a very nice approximation.

Comparisons with NIFA

We end this section by giving some comparisons with INLA's NIFA method. As discussed previously, the speed and accuracy of NIFA is very good. However, NIFA does assume the marginal is a Gaussian with different standard deviations on each side, hence only being able to give univariate approximations. We give visual comparisons (Figure 5.23) and Kullback-Leibler divergence and Hellinger distances between INLA's dense grid (regarded as the true density) and the approximations in Table 5.3.

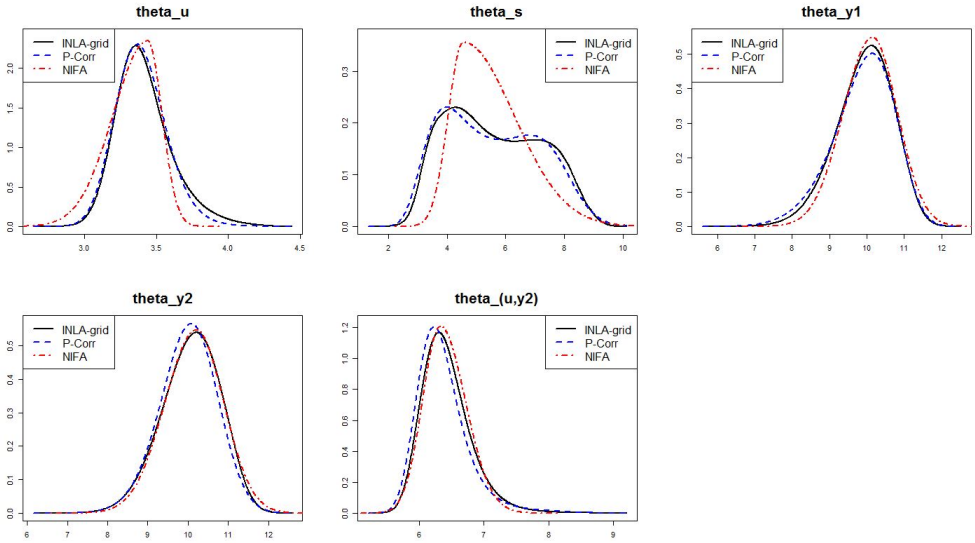

Figure 5.23: Georgia: Comparisons between NIFA and polynomial correction (cubic correction for all but θ_s which has a quintic correction).

The polynomial correction we use is cubic for all marginals, except for θ_s, for which we use a quintic correction. As expected, for θ_s, the NIFA approximated this with a unimodal density. The approximation for θ_u was slightly off for the NIFA too, with the Kullback-Leibler and Hellinger distance being much higher than the cubic correction. The other posterior marginals were well approximated by both methods, and NIFA outperformed the cubic correction for the marginal θ_{y_2}.

Table 5.3: Georgia: Distance measures comparing INLA's dense grid with both the NIFA and polynomial correction methods. Cubic polynomials were used for all but the second marginal, which used a quintic. The corrections gave the more accurate approximations for each hyperparameter except θ_{y_2} according to both the Kullback-Leibler divergence and Hellinger distances.

	K-L.Div		H.Dist	
Parameter	NIFA	Poly	NIFA	Poly
θ_u	0.30047	0.00923	0.17790	0.04445
θ_s	0.25381	0.00601	0.22173	0.04080
θ_{y_1}	0.01446	0.00250	0.05836	0.02597
θ_{y_2}	0.00459	0.00671	0.03598	0.04059
θ_δ	0.02541	0.01550	0.06262	0.05712

5.4 Discussion

We discuss several different modifications to make LDS-based methods more useful in practice. Starting from INLA's reparameterisation of the hyperparameters to something that is "more Gaussian", we use the projected function evaluations and log-pointwise means to fit a quadratic polynomial. This can be considered to be a Gaussian approximation to the reparameterised hyperparameter and it works fairly well. However, we can do better by examining the residuals between the quadratic fit and pointwise means, and come up with a cubic correction polynomial to update the quadratic polynomial and account for any disparities in the location, spread and/or skew.

The results show that the method works well, and outperforms grid-based methods in terms of both computational speed and accuracy. Using an embedded lattice also allows us to make further computational gains by simply removing points systematically from the LDS. INLA's default method of estimating hyperparameter marginals is the NIFA, which is both fast and accurate. Our methods are not as computationally efficient as NIFA, but they can be more accurate and much more flexible, as shown with the multimodal issue. The methods proposed work well in practice and they can easily be incorporated into the INLA inference framework as an alternative to grid-based approximations or the NIFA, and the LDS points can easily be re-used in the estimation of latent parameters.

Chapter 6

Latent Field Approximations for Spatio-Temporal Models of Crime using INLA and LDS

We have predominantly focussed our attention on hyperparameter estimations using LDS-based methods. This chapter will focus on the problem of latent parameter estimation. Due to recent upgrades in the INLA methodology, we can input a point set design of our choice and use this in INLA to perform inference (see Section 5.2.3 for details). We use this new upgrade on the challenging example of modelling a spatio-temporal point process model of crime in Hamilton, New Zealand. The results for this work are very much preliminary at this stage with respect to both the application of an LDS point set to approximate latent fields, and the crime model itself. We start with brief introductions to crime modelling and the data we are working with. Also, we present a class of model used for point patterns, called the log-Gaussian Cox process, and how INLA performs inference on such models. Finally, we present the results of our study of crime and conclude about how LDS performs when used for approximating latent parameters.

6.1 Introduction to Crime Modelling

Efficient policing requires optimal use of policing resources, and is especially necessary for taking more preventative actions against crime as opposed to reactive measures. Intelligence-led policing, where data-driven methods used in conjunction with criminal theory drives police resource allocation and decision making, are becoming ever more popular and necessary [53, 65]. Given that criminal activity and some corresponding factors occur within a geographical context including space and time [25], spatio-temporal modelling may lead to more accurate prediction of crime than models that do not take space and time variables into consideration. Prediction analysis can be made in the form of hotspot analysis [64, 66], where certain areas are identified as having a high intensity of crime.

6.1.1 Spatial Models of Crime

Observations of crime that are geographically referenced in space (such as in latitude/longitude coordinates or northings/eastings coordinates) is a type of spatial data, and can be defined as realisations of a stochastic process with spatial indices

$$Y(\mathbf{s}) \equiv \{y(\mathbf{s}), \mathbf{s} \in \Omega\}, \tag{6.1}$$

where the spatial domain Ω is a fixed subset of $\mathbb{R}^s$ (usually, and here, $s = 2$), and $\mathbf{s}$ is the spatial coordinate or unit. According to [21], there are three general ways to treat spatial data, areal/lattice, point-referenced, and spatial point patterns/process. Note that the lattices used in this context are not to be confused with the lattice point sets we have been using. Previous studies (for example, see [15, 25]) treat the data as area or lattice data, where $y(\mathbf{s})$ is a random aggregate value over an area unit $\mathbf{s}$ with well defined boundaries in Ω. Typically, a lattice treatment is preferred, where the spatial domain is partitioned into regular polygons and the aggregate number of crimes is the variable of interest, though some studies analysed crime rates at the suburban level (for example, see [14]). We take an alternative approach by treating the spatial domain as continuous (as close as possible), rather than partitioning discretely. We treat the data as a spatial point process, where $y(\mathbf{s})$ represents the occurence (or not) of an event, and the

locations $\mathbf{s}$ are random. The values of $y(\mathbf{s})$ take values of either 0 or 1. We also wish to add covariate information, making this a *marked point process*.

The spatial process defined in (6.1) can be generally extended to include a temporal process. We include a time dimension such that the data is defined by a process indexed by both space and time

$$Y(\mathbf{s}, t) \equiv \{y(\mathbf{s}, t), (\mathbf{s}, t) \in \Omega \subset \mathbb{R}^s \times \mathbb{R}\}, \tag{6.2}$$

and are observed at n spatial locations and at T time points. Note the spatial domain is expanded to include a temporal dimension, thus Ω is our spatio-temporal domain.

Research suggests that socio-economic and environmental factors are important to consider for predicting potential crimes. Past offenders of petty crimes tend to live in areas of high unemployment and low socio-economic status. Urban environments with high amounts of physical deterioration leave an impression of lawlessness that can lead to anti-social behaviour, and may empower some offenders to commit crimes due to a perceived lack of risk [14, 65].

Criminology literature also suggests that a strong predictor of crime are repeat and near-repeat victimisation [17, 67, 84]. Repeat victimisation is a type of crime pattern where a target is subject to victimisation multiple times, and are empirically likely to be targeted again. Near-repeat victimisation is the empirically observed pattern where potential targets close to an initial incident are at a heightened risk of being actual targets. As the distance between the initial victim and potential targets increase, and as time progresses, the likelihood of a potential target becoming a victim of crime decreases. This suggests that crimes tend to occur in clusters over space and time.

6.1.2 Data

The main dataset consists of petty (under $5000 worth of stolen or damaged goods), residential burglaries in the Hamilton City region, Waikato, New Zealand. Hamilton is New Zealand's

fourth largest city and is the major centre of the Waikato province. As of June 2017, Hamilton has an urban population of 198,600, and has enjoyed a steady annual population growth since the 1960's.

We are looking at a three month period from January - March 2014. All burglary locations are geo-coded using the New Zealand traverse mercator (NZTM) northings and eastings. We have excluded certain outer suburbs of Hamiton that we have considered too rural (mainly consisting of farmland and lifestyle blocks) such as Tamahere, Gordonton and Whatawhata, and have also excluded Temple View (roughly four kilometres away from the nearest suburb). A shapefile was created and acts as the spatial domain, which excludes the two main physical barriers, namely the Waikato river (split into five segments) and Lake Rotoroa (commonly referred to as Hamilton Lake). Figure 6.1 shows the Hamilton region with physical barriers (in blue), and the incidence of crime over the four months considered.

We consider three other ancillary datasets. The New Zealand Index of Deprivation (NZDEP) is a measure of socio-economic status of a particular area. This area is usually defined on the suburb level, but recent reports define the deprivation index on a smaller geographic area. The NZDEP measures a range of factors, such as income, employment and population density. For full details, see [1]. We represent social and physical environments through off-licence liquor stores and recent incidence of graffiti. Off-licence liquor stores are a primary source for cheap alcohol and may encourage anti-social behaviour. For each incidence of burglary, we measure the distance from the burglary to the nearest liquor store and use this distance as a covariate. Graffiti is also an act of anti-social behaviour and can be used as a measure of perceived lawlessness and physical degradation of an area. At each incidence of burglary, we count the number of incidence of graffiti in the near vicinity (a circle with radius of 500 metres) over the previous two-week period, with the number of recent graffiti incidence used as another covariate.

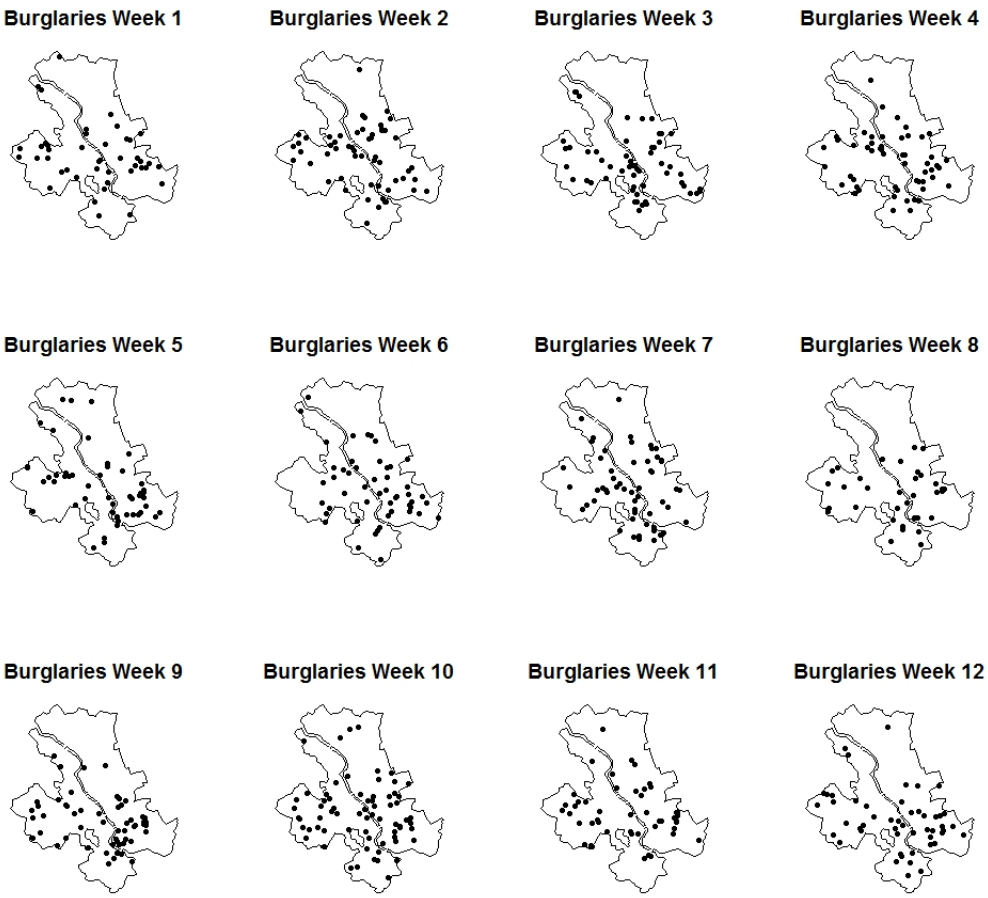

Figure 6.1: All residential burglaries in Hamilton between January and March 2014.

6.2 Methodology

We give a brief overview of the modelling and methodology using INLA's stochastic partial differential equation (SPDE) approach. We discuss a new feature of the INLA SPDE approach which takes into account physical barriers, before discussing approximations using the INLA SPDE approach with an LDS point set.

6.2.1 Log-Gaussian Cox Processes

The log-Gaussian Cox process (LGCP) is a class of models commonly used for the analysis of spatial point patterns (see for example [13, 58]). Given a bounded region Ω defined similarly in

(6.2), a point process model is an inhomogeneous Poisson process where the number of points in a region $\omega \in \Omega$ is Poisson distributed with mean $\Lambda(\omega) = \int_\omega \lambda(\mathbf{s})ds$, where $\lambda(\mathbf{s})$ is the intensity of the point process. The likelihood of an inhomogeneous point process, given an intensity λ for some point pattern Y is given by

$$\pi(Y|\lambda) = \exp\left\{|\Omega| - \int_\Omega \lambda(\mathbf{s})ds\right\} \prod_{\mathbf{s}_i \in Y} \lambda(\mathbf{s}_i).$$

The likelihood is usually intractible given the integrand $\int_\Omega \lambda(\mathbf{s})ds$ cannot be calculated explicitly and must be approximated via numerical methods or otherwise.

Cox Processes arise from considering the intensity $\lambda(\mathbf{s})$ itself as a stochastic process (these are also referred to sometimes as a doubly stochastic process), by treating the intensity as a realisation of a random field. Modelling the random field by $\log(\lambda(\mathbf{s})) = Z(\mathbf{s})$, where $Z(\mathbf{s})$ is a Gaussian random field, gives the log-Gaussian Cox process (LGCP). The LGCP can be framed as a Bayesian hierarchical model, and is a type of latent Gaussian model (2.6) and thus can be approximated in various ways using INLA.

The most common way to perform an LGCP analysis is to discretise the spatial domain Ω into a fine grid (for example, see [36]). The authors of [77] argue that there is a large amount of computational waste. Since $Z(\mathbf{s})$ is a Gaussian random field, the corresponding multivariate Gaussian vector has a dense covariance matrix, which limits how fine you can make the lattice. If $Z(\mathbf{s})$ is assumed stationary, the covariance matrix has a block-Toeplitz structure [29] that can be used to speed computations, though these are unsuitable for second-order approximations. We can model $Z(\mathbf{s})$ as a conditional autoregressive (CAR) model similar to the two examples in Section 5.3. However, it has been shown that the binning the observations is the main source of error in the fine grid, thus requiring an extremely fine grid and wasting computational resources (see [77] - Corollary A1).

6.2.2 The Stochastic Partial Differential Equation Approach

The authors of [50] proposed the stochastic partial differential equation (SPDE) approach, where a continuous spatial process (such as a Gaussian random field) is represented by a discretely indexed spatial stochastic process (such as a GMRF). We begin with the linear fractional SPDE proposed by [85]

$$(\kappa^2 - \Delta)^{\alpha/2}(\tau Z(\mathbf{s})) = \mathcal{W}(\mathbf{s}), \tag{6.3}$$

where Δ is the Laplacian, $\kappa > 0$ is a scale parameter, α is a smoothness parameter, τ is the variance parameter, and $\mathcal{W}(\mathbf{s})$ is a Gaussian spatial white noise process. The stationary solution to this equation is a stationary Gaussian field $Z(\mathbf{s})$ with the Matèrn covariance function

$$\text{Cov}(Z(\mathbf{s}_i), Z(\mathbf{s}_j) = \frac{\sigma^2}{\Gamma(\zeta)2^{\zeta-1}}(k||\mathbf{s}_i - \mathbf{s}_j||)^{\zeta} B_{\zeta}(k||\mathbf{s}_i - \mathbf{s}_j||). \tag{6.4}$$

Here, $||\mathbf{s}_i - \mathbf{s}_j||$ represents the Euclidean distance between two locations $\mathbf{s}_i$ and $\mathbf{s}_j$. Also, the term B_{ζ} denotes the modified Bessel function of the second kind, and order $\zeta > 0$. The Bessel function is a measure of smoothness and is kept fixed. The parameter σ^2 is the marginal variance, and $\kappa > 0$ is the scaling parameter. The scaling parameter is related to the range parameter r which is defined as the distance from a point at which the spatial correlation becomes negligible. The authors of [50] measured the empirically derived definition for the range to be $r = \sqrt{8\zeta}/\kappa$, and [8] assesses the appropriateness of this quantity for different values of ζ. For a full discussion of these parameters, including the empirical results for the range parameter, see [50].

The solution to (6.3) (i.e the Gaussian field) can be approximated through a finite element mesh, using a basis function representation defined by the triangulation of the spatial domain Ω

$$Z(\mathbf{s}) = \sum_{g=1}^{G} \Upsilon_g(\mathbf{s})\tilde{Z}_g, \tag{6.5}$$

where G is the number of triangularisation vertices, $\{\Upsilon_g\}$ is the set of deterministic basis functions, and $\{\tilde{Z}_g\}$ are the set of Gaussian weights. The basis functions are chosen to have a local

support, and are piecewise linear in each triangle, so Υ_g has the value 1 at vertex g and 0 for all other values. From [50], Neumann boundary conditions are used and the precision matrix $\mathbf{Q}$ for the Gaussian weight vector $\tilde{\mathbf{Z}}$ is given by

$$\mathbf{Q} = \tau^2(\kappa^4\mathbf{C} + 2\kappa^2\mathbf{G} + \mathbf{G}\mathbf{C}^{-1}\mathbf{G}), \tag{6.6}$$

where $\mathbf{C}$ is a diagonal matrix, with elements $C_{ii} = \int \Upsilon(\mathbf{s})d\mathbf{s}$, and the elements of the sparse matrix $\mathbf{G}$ is $G_{ij} = \int \nabla\Upsilon_i(\mathbf{s})\nabla\Upsilon_j(\mathbf{s})d\mathbf{s}$, where ∇ is the gradient. Since $\mathbf{G}$ is sparse, the precision matrix $\mathbf{Q}$ is sparse and is dependent on τ and κ. Also $\mathbf{Z}$ is a GMRF that has the distribution $N(\mathbf{0}, \mathbf{Q}^{-1})$ and this represents the solution to the SPDE (in the stochastically weak sense).

6.2.3 Non-Stationary Gaussian Fields

Stationarity assumes that the spatial correlation is constant throughout the entire spatial domain. For modelling real-world situations, this may not be appropriate. For example, the spatial correlation of some phenomena may be influenced by physical barriers, or features such as mountains, lakes and rivers. In these cases, a non-stationary spatial process may be more appropriate.

The SPDE approach is quite flexible and can extend the stationary case by specifying parameters $\kappa(\mathbf{s})$ and $\tau(\mathbf{s})$, which vary over the spatial domain. This changes the SPDE given in (6.3) to

$$(\kappa(\mathbf{s})^2 - \Delta)^{\alpha/2}(\tau(\mathbf{s})Z(\mathbf{s})) = \mathcal{W}(\mathbf{s}),$$

and the precision matrix given by (6.6) is modified to be

$$\mathbf{Q} = \mathbf{T}(\mathbf{K}^2\mathbf{C}\mathbf{K}^2 + \mathbf{K}^2\mathbf{G} + \mathbf{G}\mathbf{K}^2 + \mathbf{G}\mathbf{C}^{-1}\mathbf{G})\mathbf{T},$$

where $\mathbf{T} = \text{diag}(\tau(\mathbf{s}_i))$ and $\mathbf{K} = \text{diag}(\kappa(\mathbf{s}_i))$. Note locations $\mathbf{s}_i$ are the mesh vertices. For more on the SPDE extension to the non-stationary case, see (cite Bolin and Lindgren, Ingebretson 2013).

The so-called barrier model, developed recently by [4], models spatial dependence in the presence of physical barriers, where the spatial correlation becomes null where a physical barrier lies. Rather than treating the Matèrn covariance (given by (6.4)) as a correlation function of the shortest distance between two points, the barrier model views it as a collection of paths through a Simultaneous Autoregressive (SAR) model. The local dependencies are manipulated in such as way as to cut off paths the are crossing physical barriers. We do not describe the full details here, but details can be found in [4].

6.3 Spatio-Temporal Point Patterns of Burglaries in Hamilton

We present the results of the study of spatio-temporal point patterns of petty burglaries in Hamilton, New Zealand. For this study, we are interested in building a spatio-temporal model of burglaries over eight weeks, and using this to predict crimes for the week proceeding the eight weeks. We use the LGCP models discussed in 6.2.1 and use INLA's SPDE approach to perform inference on the unknown Gaussian random field. We assume the Gaussian field is non-stationary, and consider the physical features of Lake Rotoroa and the Waikato river as barriers. Thus, the spatial correlation along the boundaries of the barriers is 0.

We have two main objectives. The first is to analyse the spatio-temporal point patterns of crime and perform hotspot analysis on the crime map predictions. The second is to use an LDS point set inside INLA for the purposes of estimating the latent parameters. We do this in the same way as shown in Section 5.2.3.

6.3.1 Model Setup

We consider three different models. The first model is a pure spatio-temporal model

$$\text{Mod1}: \quad \eta_{i,t} = \mu + f_{barrier}(\mathbf{s}_i, t; r, r_b = 0, \sigma, \rho). \tag{6.7}$$

where $i = 1, \ldots, n_s$ indexes the spatial units and $t = 1, \ldots, T$ indexes the time units. Here, $\eta_{i,t} = Z(\mathbf{s}_i, t)$. The parameter μ is the overall mean and the function $f_{barrier}$ represents the spatio-temporal correlation function for barrier models. For this example, this is a Matèrn covariance function with parameters r, r_b, σ, and ρ. The parameter r is the practical range discussed in Section 6.2.2. The parameter r_b is the spatial correlation where a boundary occurs and is treated as a constant with value 0. The parameter σ is the marginal variance and ρ is the temporal correlation parameter and is modelled as an AR(1) process.

As stated in Section 6.1.2, we have three covariates. Deprivation indices, distance to nearest liquor store, and instances of recent graffiti observations are given the acronyms Dep, Liq and $Graf$ respectively. For the second model, we add the deprivation index as a covariate

$$\text{Mod2}: \quad \eta_{i,t} = \mu + \beta_{dep} Dep_{i,t} + f_{barrier}(\mathbf{s}_i, t; r, r_b = 0, \sigma, \rho), \tag{6.8}$$

where β_{dep} is the linear coefficient for the covariate Dep. The third model includes all covariates

$$\begin{aligned} \text{Mod3}: \quad \eta_{i,t} = \mu + \beta_{dep} Dep_{i,t} + \beta_{liq} Liq_{i,t} + \beta_{graf} Graf_{i,t} \\ + f_{barrier}(\mathbf{s}_i, t; r, r_b = 0, \sigma, \rho), \end{aligned} \tag{6.9}$$

where β_{liq} and β_{graf} are the linear coefficients for the covariates Liq and $Graf$ respectively.

The model fitting process requires a triangulated mesh described in (6.5), which discretises the spatial domain. Details on the mesh, and the principles of fitting good meshes can be found in [8], and for mesh setups for barrier models, see [4]. For the LGCP model, rather than binning all observations into cells, we evaluate the model at all vertices of the mesh, and at every observation point. This has the advantage of the data being modelled considering its actual location, rather than the position of cells. Covariates need to be defined at all observation locations and all vertices, which for our choice of covariates is easily done with little pre-processing time.

For the hyperparameters r and σ, we consider the so-called penalised complexity (PC)-

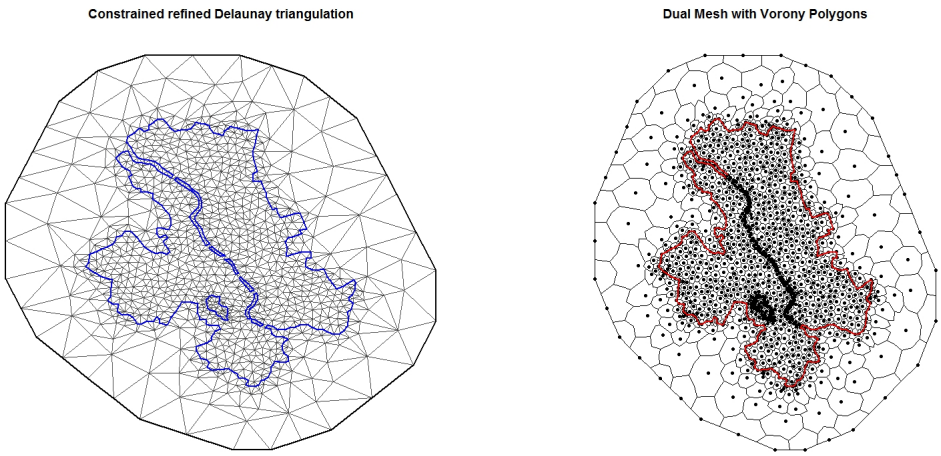

Figure 6.2: The left hand side triangulation mesh discretises the spatial domain. The right hand side dual mesh shows the Voronoi polygons, which are centred at the vertices of the triangulated mesh. These volume of the polygons define the integration weights. All vertices outside of the spatial domain (red solid line) are assigned a zero weighting.

prior, an informative prior derived in [26]. The PC-prior is defined by assigning the parameter a value, with a corresponding probablility. For the range parameter r, we were quite specific, assigning it a value of 0.5 (or 500 metres) with a 0.5 probability of being less than 500 metres. This is based on previous studies cited earlier, where the spatial correlation of burglaries tends to decline after 500 to 1000 metres. A PC-prior was also considered for the correlation parameter ρ, though we were less specific, assigning a value of 0, with probability 0.95 of being greater than 0. We keep these priors the same over all three models. We define the SPDE model on the mesh vertices and observation points, and to fit the LGCP model, these points are considered to be the integration points. The methods in [50, 77] define the expected number of events at each point to be proportional to the volume of the polygon from a dual mesh, thus mesh vertices with larger edges are expected to have a larger expected value. It is clear though in Figure 6.2 that mesh has vertices outside of the spatial domain, thus those values are given a weight of zero. The points in the dual mesh correspond to the triangulated mesh vertices, as shown in Figure 6.2. The corresponding Voronoi polygons give the integration weights of each point. See references mentioned for full details.

Recall that for our dataset, we have twelve weeks of burglary data. We use eight weeks of data to fit an LGCP model, then use this to predict the following week. Performing this over a 12 week period means that we first use the first eight weeks to predict the ninth week, then use weeks 2 – 9 (the actual week 9 data, not the predictions) to predict week 10 and so on. Note that this time period is from January to March 2014, thus we are essentially predicting for the four weeks in March. We will present the model fit and predictions for all three models described in (6.7), (6.8), and (6.9) respectively, using both INLA and LDS. Note that for computational reasons we use `int.strategy = "ccd"` and `strategy = "gaussian"`, as using `int.strategy = "grid"` crashed the INLA program. For LDS, we use a $\mathcal{K}^*_{64,3}$ with `strategy = "gaussian"`.

6.3.2 Spatio-Temporal Burglary Maps

We present here the spatio-temporal burglary maps for the first eight weeks, and the corresponding predicted map for the following week. We present the latent field posterior means for Mod1, Mod2, and Mod3 for the first run of eight weeks, using both INLA points and LDS points. We also present the corresponding posterior mean and standard deviation predictions for each model (Figures 6.3 to 6.14).

Comparing the two methods, there seems to be some difference in the latent field estimates. Using LDS gives a larger range of estimates, thus the intensities on the LDS maps look more profound then in the INLA estimates. However, on closer inspection, both LDS and INLA maps have high intensities in roughly the same areas. These are the Hamilton East area close to the University of Waikato, the south of Hamilton East close to the Waikato river, and Bader in the south of Hamilton, south-west of the Waikato river. The prediction maps tell a similar story, where the predicted intensities are most profound in Hamilton East and Bader. Areas of medium intensities include the central city (adjacent and to the west of the Waikato river), the south-east of Hamilton lake and the Fairfield-Enderly area, slightly north of Hamilton East. Areas of low intensity are mainly in the northern suburbs of Rototuna, Flagstaff and Huntington, which are areas of high socio-economic status and have experienced a high level of development over the

last decade or so.

The maps and predictions for the following three weeks were similar to the first week, with only very slight movements in intensities overall. The overall conclusion of the comparisons between INLA and LDS are the same, that there really is not much difference between the two methods with respect to estimation. The overall intensities did change from week-to-week, albeit only very slightly. The general areas of high intensities did not move much at all. We will show this in more detail with the upcoming hotspot analysis.

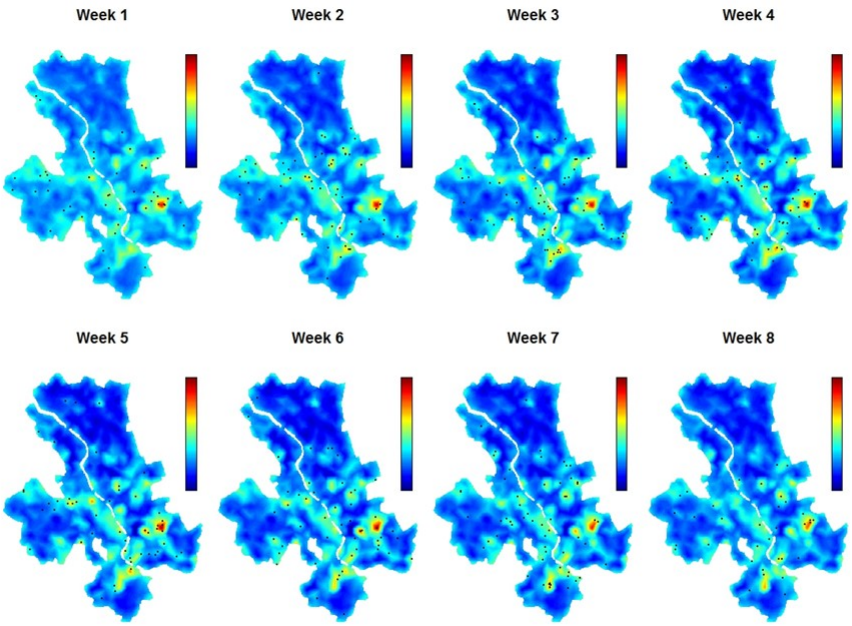

Figure 6.3: The eight week model fit for Mod1 using INLA's CCD points.

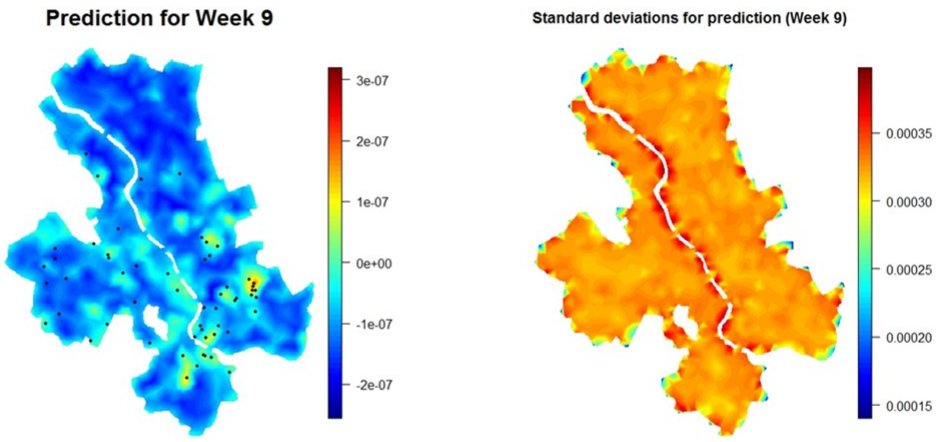

Figure 6.4: Posterior mean and standard deviation for week 9 predictions, for Mod1 using INLA's CCD points.

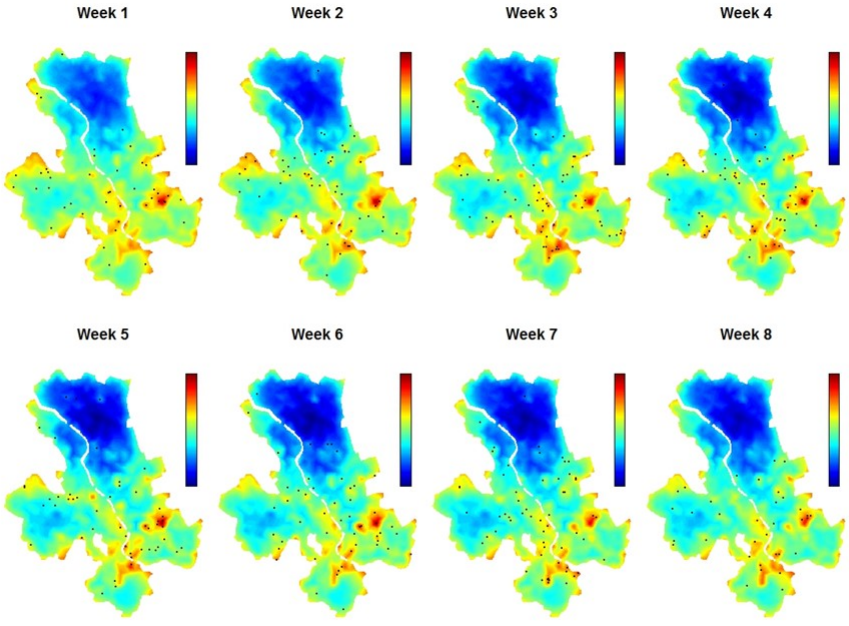

Figure 6.5: The eight week model fit for Mod1 using LDS points.

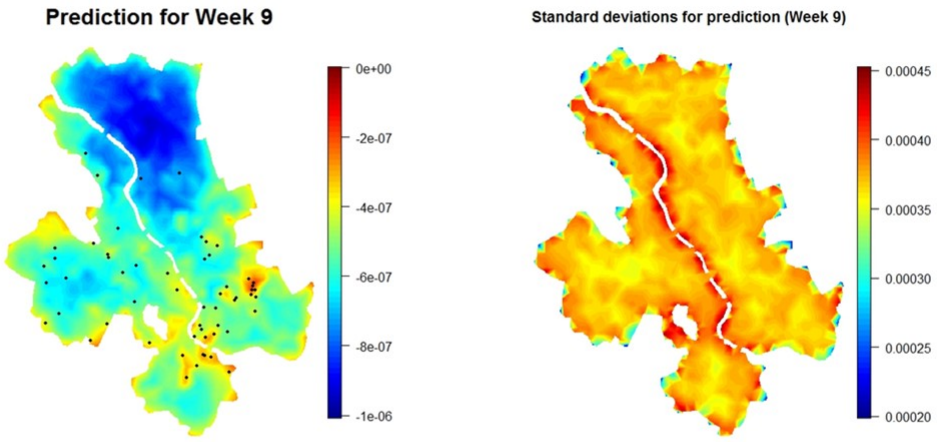

Figure 6.6: Posterior mean and standard deviation for week 9 predictions, for Mod1 using LDS points.

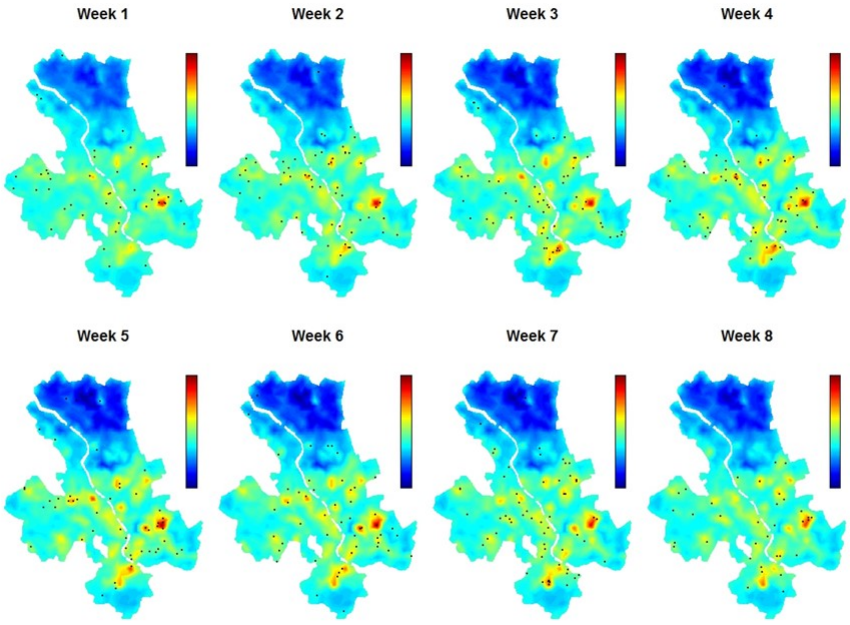

Figure 6.7: The eight week model fit for Mod2 using INLA's CCD points.

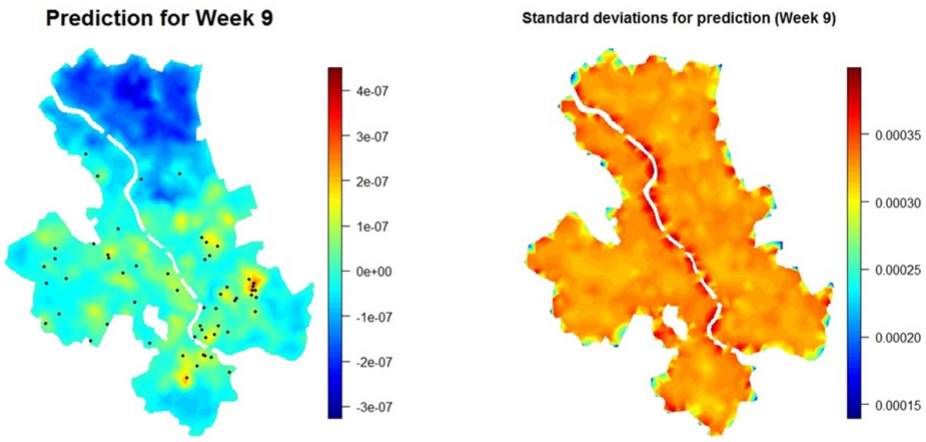

Figure 6.8: Posterior mean and standard deviation for week 9 predictions, for Mod2 using INLA's CCD points.

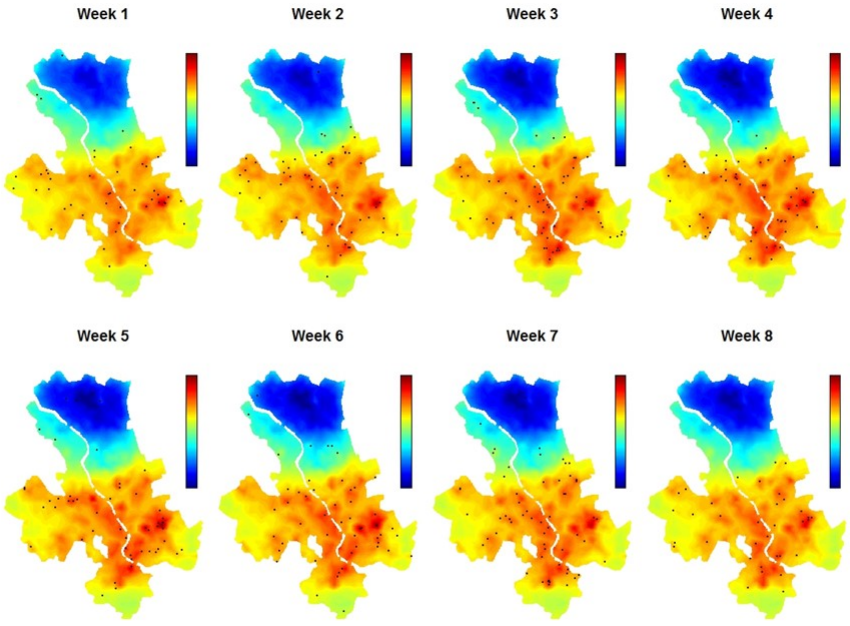

Figure 6.9: The eight week model fit for Mod2 using LDS points.

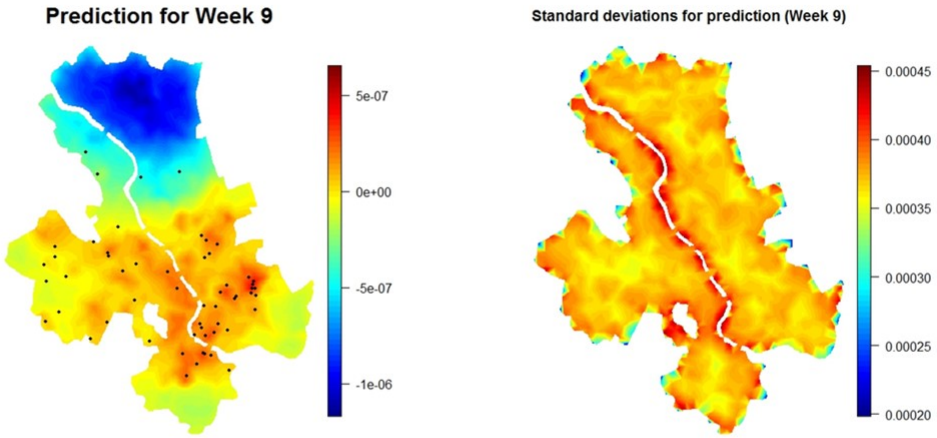

Figure 6.10: Posterior mean and standard deviation for week 9 predictions, for Mod2 using LDS points.

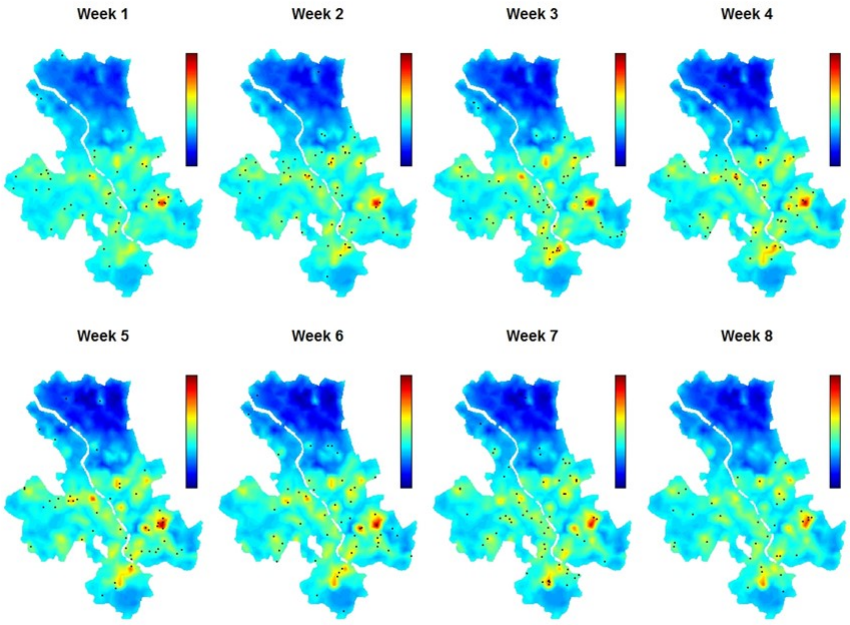

Figure 6.11: The eight week model fit for Mod3 using INLA's CCD points.

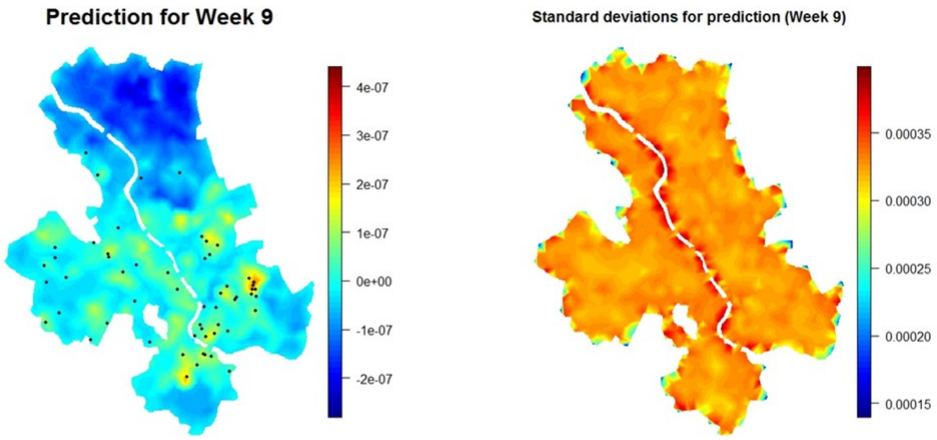

Figure 6.12: Posterior mean and standard deviation for week 9 predictions, for Mod3 using INLA's CCD points.

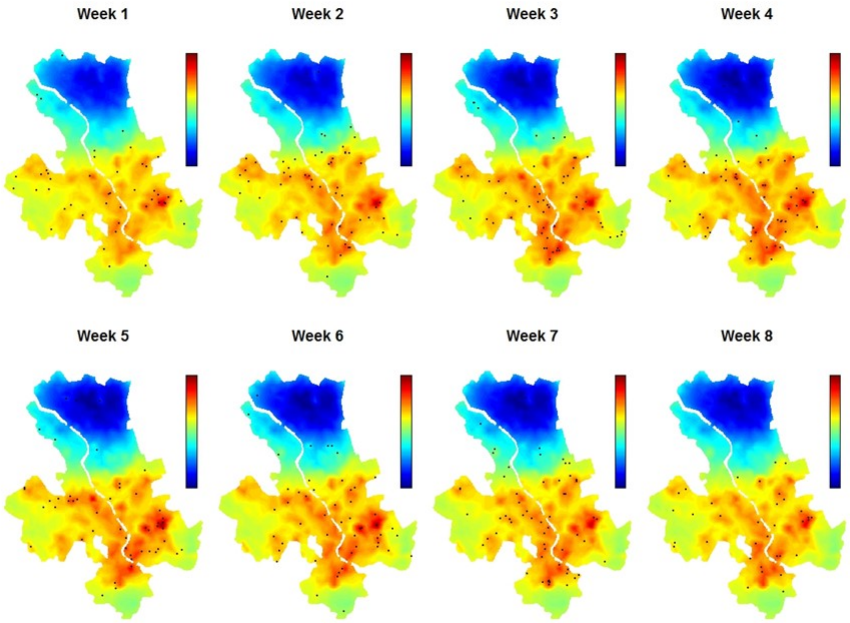

Figure 6.13: The eight week model fit for Mod2 using LDS points.

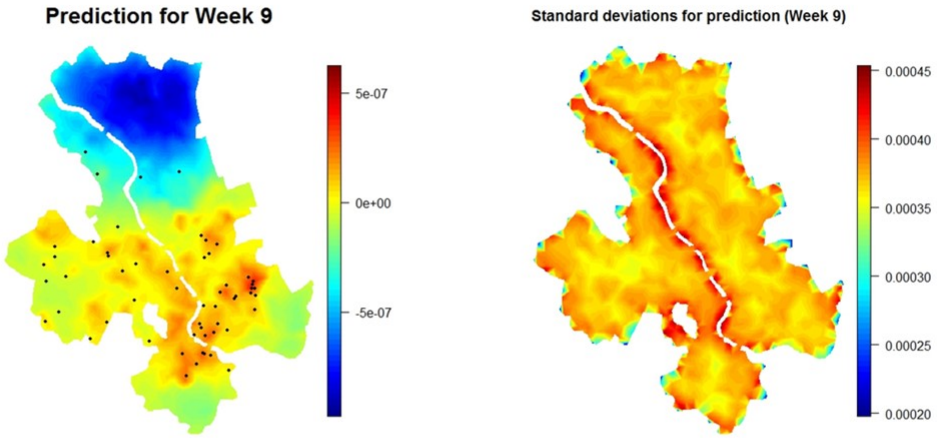

Figure 6.14: Posterior mean and standard deviation for week 9 predictions, for Mod3 using LDS points.

6.3.3 Parameter Estimates and Statistics

Marginal likelihoods can be used as a criteria for Bayesian model selection, as Bayes factors are defined as the ratio of marginal likelihoods of two competing models. INLA approximates the marginal log-likelihood, which we will use for model comparisons (for details of the approximation, see Section 6.2 in [74]). Table 6.1 shows the marginal log-likelihood results for all three models over the four weeks, using both INLA's CCD points and the LDS points. Comparing INLA and LDS, there is very little difference between the two methods for all three models over four weeks. Comparing the models, we see a significant increase in the marginal log-likelihoods of Mod2 compared to Mod1, thus adding the covariate Dep was advantageous. We see a further increase in marginal log-likelihood for Mod3, though the relative increase was not as profound.

The parameter estimates in Table 6.2 (covariate estimates) and Table 6.3 (hyperparameter estimates) also show very little difference between INLA and LDS. Since LDS-based meth-

Table 6.1: Marginal log-likelihoods for all models, using both INLA and LDS points.

	Marginal log-likelihood: INLA			Marginal log-likelihood: LDS		
Weeks	Mod1	Mod2	Mod3	Mod1	Mod2	Mod3
1 - 9	-2358.62	-1775.03	-1734.25	-2361.37	-1774.79	-1735.25
2 - 10	-2373.92	-1803.56	-1763.53	-2376.34	-1802.5	-1765.42
3 - 11	-2389.96	-1833.91	-1787.58	-2386.61	-18362.64	-1787.04
4 - 12	-2381.95	-1817.77	-1777.75	-2390.72	-1815.97	-1778.25

ods for hyperparameter estimates are not implemented in INLA, it will use NIFA algorithm to estimate the hyperparameters, so we expect little difference between INLA and LDS for hyperparameter estimation.

The main difference between INLA and LDS for this example was computational time. For this particular example, we could get INLA to run each model in 953.7 seconds on average. Using LDS, the computations took, on average, 4923.4 seconds. Although this may not be the best representation, since further optimisation may be available if LDS is fully implemented as an option in INLA (as opposed to using it as a substitute point set), we would expect that the computational time would be higher for LDS given that INLA's CCD only used 16 points, whereas the LDS method used a Korobov lattice with 64 points.

Table 6.2: Posterior means and standard deviations for the covariate coefficient parameters for week 1 only. These were obtained using both INLA and LDS points. The tables show very little difference between the covariate estimates.

Covariate coefficient estimates using INLA for Week 1						
	Mod1		Mod2		Mod3	
Parameter	Mean	SD	Mean	SD	Mean	SD
μ	-1.011	0.0204	-0.3418	0.0287	-0.2985	0.0293
Dep	NA	NA	-0.2349	0.0079	-0.2241	0.0083
Liq	NA	NA	NA	NA	-0.2702	0.0285
$Graf$	NA	NA	NA	NA	0.0505	0.0302
Covariate coefficient estimates using LDS for Week 1						
	Mod1		Mod2		Mod3	
μ	-1.0105	0.0203	-0.3415	0.0288	-0.2985	0.0293
Dep	NA	NA	-0.2349	0.0079	-0.2241	0.0083
Liq	NA	NA	NA	NA	-0.2702	0.0285
$Graf$	NA	NA	NA	NA	0.0505	0.0306

Table 6.3: Posterior means and standard deviations for model hyperparameters for week 1 only. These were obtained using both INLA and LDS points. The estimates are all very similar, since they both use the NIFA to approximate the hyperparameter posterior distributions.

Hyperparameter estimates using INLA for Week 1						
	Mod1		Mod2		Mod3	
Parameter	Mean	SD	Mean	SD	Mean	SD
$\theta_1(\sigma)$	-8.6603	0.7345	-8.6549	0.7347	-8.6548	0.7346
$\theta_2(r)$	-0.6254	0.9962	-0.6338	0.9951	-0.6348	0.9949
ρ	0.6693	0.4336	0.6692	0.4337	0.6692	0.4337
Hyperparameter estimates using LDS for Week 1						
	Mod1		Mod2		Mod3	
$\theta_1(\sigma)$	-8.6609	0.7356	-8.6567	0.7347	-8.6548	0.7346
$\theta_2(r)$	-0.6258	0.9960	-0.6255	0.9961	-0.6348	0.4337
ρ	0.6693	0.4335	0.6694	0.4338	0.6692	0.4337

6.3.4 Hotspot Analysis - Maps

We are interested in the areas of high intensity. We look at these areas more closely in the prediction maps. Four hotspot levels are chosen which represent the highest percentage level of intensity. These four levels are 1%, 2.3%, 5% and 10%. The 2.3% hotspot was chosen as this represents the realistic amount of space the police can monitor at any one time. The hotspot maps are shown in Figures 6.15 to 6.20, where the hotspots are shown in red. We also plot the actual observations of crime that occurred during that week. We present the week 1 results, though the hotspot maps were quite consistent over all predicted weeks.

There are some interesting observations that we can see. For Mod1 at the 5% and 10% level (Figure 6.15 and Figure 6.16), both INLA and LDS show hotspots on parts of the boundary. This is more profound in the LDS case, where we have hotspots in the west of Hamilton (around Nawton and Grandview Heights) and Te Rapa in the north-west. This may be due to boundary issues with the mesh, though this issue is not present for Mod2 and Mod3. The INLA hotspot maps show the hotspots are more scattered for 5% and 10% levels, whereas the LDS hotspot maps are less so, suggesting that it may perform better when trying to capture clusters of observations. Both models identify the main cluster of crimes around the Hamilton East/University area at 1% and 2.3%, and start to capture the Bader area when increasing the level to 5%. At 10%, the hotspots start to capture the central city and the south end of Lake Rotoroa.

There are no identifiable hotspots north of Enderly/Fairfield area.

Since we are using more points with the lattice than we are using CCD, we expect better performance in terms of accuracy, and the hotspot maps seem to suggest this. For Mod2 and Mod3, using INLA with an LDS seems to generate predictive hotspots which capture the clusters slightly better than using the CCD, especially at the higher hotspot levels. We perform more analysis on this in the next section.

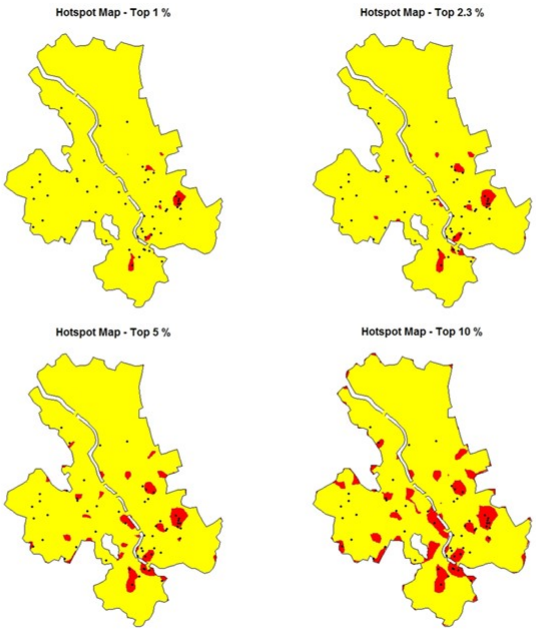

Figure 6.15: Hotspot analysis for Mod1 using INLA's CCD points.

6.3.5 Hotspot Analysis - Statistics

We present several statistics that are typically used to analyse hotspots. These are presented in Table 6.4 (using INLA's CCD) and Table 6.5 (using LDS points). We use the maps to count the number of occurrences that lie in a hotspot. Let n the number of burglaries that lie in a hotspot, and N be the total weekly amount of burglaries. The hit-rate is simply the ratio of hotspot

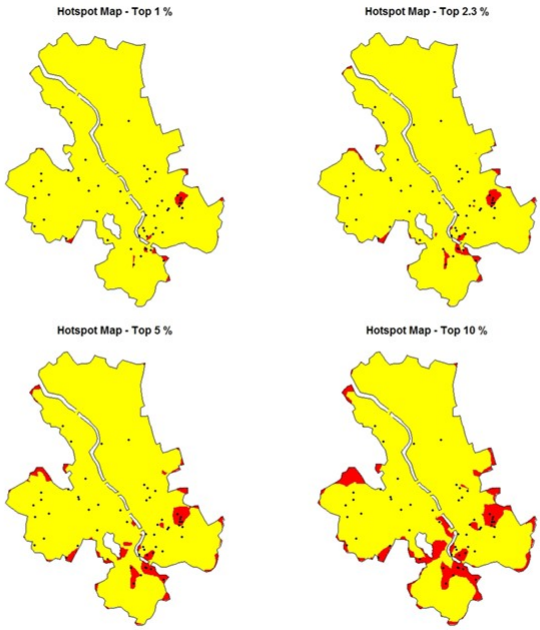

Figure 6.16: Hotspot analysis for Mod1 using LDS points.

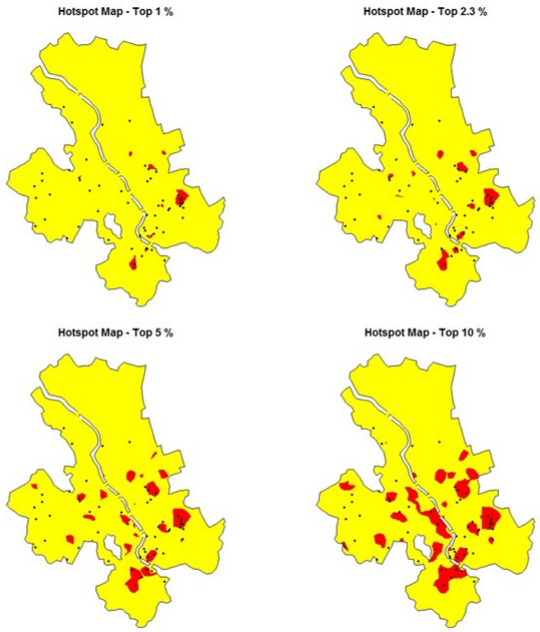

Figure 6.17: Hotspot analysis for Mod2 using INLA's CCD points.

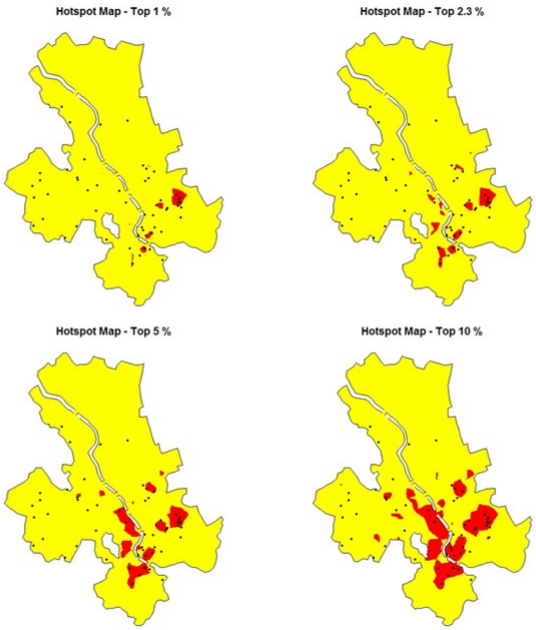

Figure 6.18: Hotspot analysis for Mod2 using LDS points.

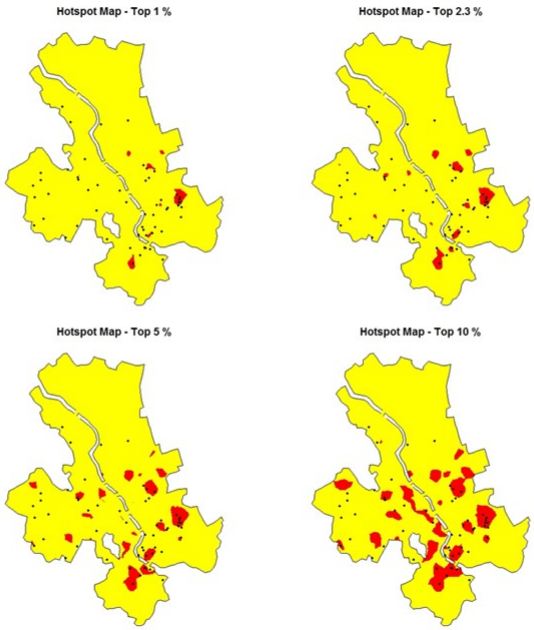

Figure 6.19: Hotspot analysis for Mod3 using INLA's CCD points.

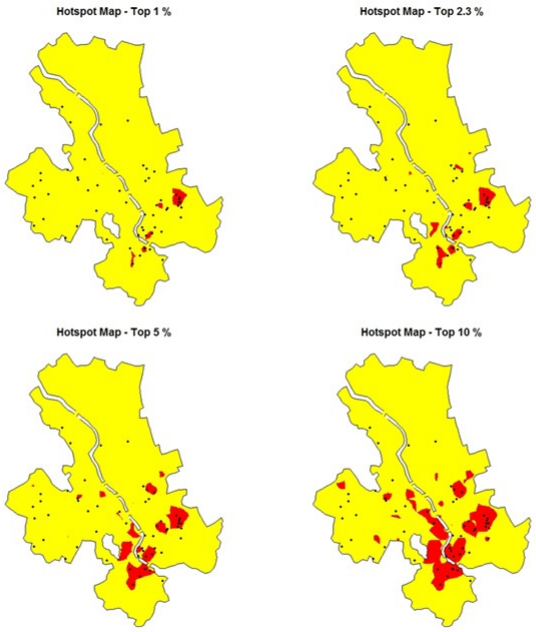

Figure 6.20: Hotspot analysis for Mod3 using LDS points.

burglaries and total number of burglaries

$$HR = \frac{n}{N}.$$

The predictive accuracy index (PAI) introduced by the authors of [17] is a measure of the accuracy of a hotspot, taking into account the size of the hotspot. Let a be the area of the hotspot, and A be the total area of the spatial domain. We have

$$PAI = \frac{n/N}{a/A}.$$

A feature of this commonly used statistic is that it will give larger PAI values as the size of the hotspot decreases, making it hard to compare the performances of each hotspot level. The authors of [39] introduced the penalised-PAI in which a parameter α is used to penalise the PAI as a decreases. The penalised-PAI is given by

$$PPAI = \frac{n/N}{(a/A)^{\alpha}}.$$

For our example, the penalising parameter α is found empirically for each level of hotspot, and the values are shown in Table 6.4 under the column "alpha". Note that when $\alpha = 1$ we have PAI, and when $\alpha = 0$, we have the HR, thus $0 \leq \alpha \leq 1$.

Table 6.4: Hotspot predictive statistics for INLA, Week 9 predictions

Predictive Statistics for INLA Mod1 - Week 9 Prediction					
Level	n	HR	PAI	alpha	PPAI
1%	7	0.125	12.5	0.76	4.139
2.3%	11	0.196	8.540	0.7	2.754
5%	20	0.357	7.143	0.6	2.155
10%	26	0.464	4.643	0.55	1.647
Predictive Statistics for INLA Mod2 - Week 9 Prediction					
Level	n	HR	PAI	alpha	PPAI
1%	7	0.125	12.5	0.76	4.139
2.3%	11	0.196	8.540	0.7	2.754
5%	18	0.3521	6.429	0.6	1.940
10%	25	0.446	4.446	0.55	1.584
Predictive Statistics for INLA Mod3 - Week 9 Prediction					
Level	n	HR	PAI	alpha	PPAI
1%	6	0.107	10.714	0.76	3.548
2.3%	11	0.196	8.540	0.7	2.754
5%	18	0.3521	6.429	0.6	1.940
10%	26	0.464	4.643	0.55	1.647

The conclusions we draw from the statistics given in Table 6.4 and Table 6.5 are similar to those that we drew from the hotspot maps. At the lower hotspot levels (1% and 2.3%), both INLA and LDS models performed similarly, though Mod3 using LDS points performed slightly better at the 2.3% level. Mod1 at the 10% level using LDS performed the worst, with the lowest PAI, PPAI and only predicted 19 burglaries, compared to 26 - 29 burglaries for every other model. The maps showed for Mod1 that there was a problem with predicting hotspots on the boundaries, and this was particularly a problem with the LDS estimates. However, at the 10% level, Mod3 using LDS performed the best, predicting 51.7% of all burglaries.

Over all weeks predicted, the results varied, depending on the distribution of all the crimes. Figure 6.21 show that for week 9, there were a higher concentration of burglaries in the suburbs of Hamilton East and Bader. The following weeks, the distribution of crimes were more spread,

Table 6.5: Hotspot predictive statistics for LDS, Week 9 predictions

Predictive Statistics for LDS Mod1 - Week 9 Prediction					
Level	n	HR	PAI	alpha	PPAI
1%	8	0.142	14.286	0.76	4.730
2.3%	11	0.196	8.540	0.7	2.754
5%	14	0.25	5	0.6	1.509
10%	19	0.339	3.393	0.55	1.204
Predictive Statistics for LDS Mod2 - Week 9 Prediction					
Level	n	HR	PAI	alpha	PPAI
1%	9	0.161	16.071	0.76	5.322
2.3%	11	0.196	8.540	0.7	2.754
5%	18	0.3521	6.429	0.6	1.940
10%	26	0.464	4.643	0.55	1.647
Predictive Statistics for LDS Mod3 - Week 9 Prediction					
Level	n	HR	PAI	alpha	PPAI
1%	10	0.179	17.857	0.76	5.913
2.3%	12	0.214	9.317	0.7	3.005
5%	18	0.3521	6.429	0.6	1.940
10%	29	0.517	5.187	0.55	1.837

Table 6.6: Weekly hit rates for hotspot level 2.3%. Though there was little difference between using INLA and LDS points, there was a significant drop in bumber of crimes predicted, due to the more evenly spread distribution of burglaries over weeks 10, 11, and 12 (see Figure 6.21).

Hit-rates at the hotspot level 2.3%						
	INLA			LDS		
Week	Mod1	Mod2	Mod3	Mod1	Mod2	Mod3
10	0.139	0.139	0.139	0.139	0.125	0.139
11	0.104	0.104	0.125	0.167	0.145	0.145
12	0.08	0.08	0.08	0.08	0.08	0.08

thus the hotspot analyses did not perform as well. Table 6.6 shows the hit-rates at the 2.3% hotspot level. These values were smaller than week 9, where the hit-rate at 2.3% hotspot level were between 19.6% and 21.4%. This was consistent over all hotspot levels, for both INLA's CCD and LDS point sets.

6.4 Discussion

The objective of this chapter was to use an LDS point set in INLA (as outlined in Section 5.2.3) to perform latent field approximations. We chose to do this on the challenging problem of

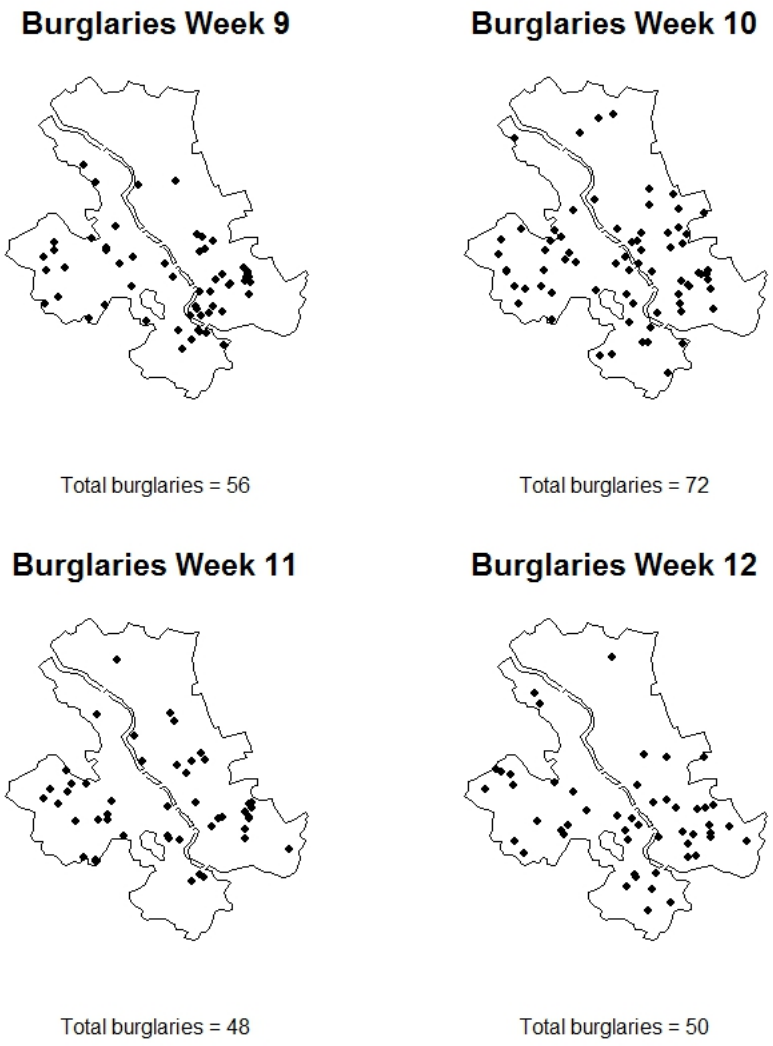

Figure 6.21: Occurrences of burglaries for weeks 9 to 12 with total burglary numbers. The distribution of burglaries in week 9 were much more concentrated around the Hamilton East and Bader areas.

spatio-temporal point process model, using INLA's SPDE approach. The primary result was that we could implement an LDS into INLA, and the approximations were very similar to those made using INLA's default CCD points. Computationally however, the LDS method took much longer to approximate the marginals for the latent field. Our preliminary results showed that using an LDS took around five times longer than using INLA's CCD strategy. However, we draw two positive conclusions from this. First, although using an LDS was slow, it was computationally feasible. The same could not be said using a grid, as INLA would crash with memory issues. Secondly, for this example, we did obtain better outcomes using LDS rather than CCD

since for Mod2 and Mod3, we were able to identify hotspots that had more occurrences of burglaries.

With regards to the crime model itself, this is a work in progress. The main aim is to develop a tool that could be used to help police allocate their time and resources more efficiently. Our model has incorporated some complexity in it by assuming the spatial correlations are not stationary, and takes into account physical barriers that may affect spatial correlations. Our models were able to predict burglary hotspots and potential clusters, and we showed that improvements can be made when taking into account socio-economic factors such as deprivation indices. Since this model is relatively new, there are several ways of improving the models with respect to accuracy and usability, such as using a more detailed spatial domain, or including more informative covariates.

Chapter 7

Conclusions and Further Work

This work on incorporating LDS-based methods has made several important contributions. A significant contribution is the development of the LDS-based method itself. We cannot numerically integrate in the same fashion as a grid point set, so using orthogonal projections and a least-squares polynomial is a particularly novel way of marginalising. Along with the initial development, we have outlined and proved several theorems regarding the convergence of these approximations. Another contribution was expanding on these ideas and modifying the approximations for compatibility within the framework of INLA. Using INLA's hyperparameter transformations, we can perform the approximations in two steps, by approximating with a quadratic polynomial, and updating that approximation with a cubic (or a polynomial of higher order if necessary). Comparisons using several examples show that these approximations outperform INLA's grid approximations for both accuracy and speed, while allowing more flexibility then INLA's NIFA approximation, since NIFA cannot approximate any distribution with more than one mode.

A recent update of the INLA software has allowed us to explore using LDS point sets to approximate latent parameters. We show that, although computation of the latent parameters using LDS are much slower than using INLA's CCD strategy, they are computationally feasible (unlike INLA's grid). Also, our example in Chapter 6 seems to suggest that we can obtain slightly better approximations for predictions. Lastly, we have developed a spatio-temporal

model of crime which can be a predictive tool to analyse future hotspots of crime. Although this model is in the early stages of development, with further exploration and work, this could be developed into something very useful in practice for police to help them allocate their scarce resources.

Whilst we have made contributions to the development and theory of LDS-based methods for Bayesian inference and their potential applications to the INLA methodology, we do not claim that this work is complete. There are a number of open questions that are yet to be answered. We give an overview of our conclusions before posing some unanswered questions for future work.

7.1 Conclusions

The motivation of this thesis is outlined in Section 1.1, with thesis objectives set out in Section 1.2. We were intially motivated by INLA's loss of computational efficiency for latent Gaussian models with a large amount of hyperparameters, especially when performed using a grid point set. The two main objectives were to explore the use of LDS is the marginalisation process, and to create a general algorithm that could be implemented into the INLA inferential framework which could be used on a real-world example. Our conclusions are as follows:

- **LDS-Based Methods**

  We show that grid-based methods can be improved upon in two ways. The first way is by substituting a grid with a maximal-rank lattice in the marginalisation process. This can improve accuracy, especially for marginals that are skewed in shape. However, it does little to improve speed, since the number of points required increases exponentially as dimension increases, similar to the grid. The second way is to bypass grids altogether and use LDS point sets. This requires orthogonally projecting all function evaluations onto each marginal space and using a least-squares polynomial fit through the scattered points. We conclude that for approximating marginal densities from multivariate joint sensities, that this method requires far fewer points than a grid.

The convergence theorems presented in Section 4.4.2 provide several conclusions about our methods:

1. Approximations using least-squares is equivalent to weighted least-squares.
2. Under sufficient smoothness conditions and abscissa points corresponding to either Chebyshev nodes or equidistant points, the least-squares approximation converges to the target as the number of points evaluated at each abscissa point increases without bound.
3. Given that the above statement may not be practical in some cases as this requires so many points, an alternative is to use LDS and partition the points into equally spaced intervals. Let n be the number of partitions made, and that the pointwise mean of each partition is found, a polynomial of degree $n-1$ will pass through the pointwise means.
4. As the partition lengths become small, and as the number of points increase in each partition, the least-squares approximation will converge to the true posterior.

However, there is the significant drawback of the choice of polynomial degree. Too low and we cannot capture the shape, too high inevitably leads to Runge's phenomenon. Thus from a practical perspective, LDS-based methods in its initial form may not be suitable for INLA.

- **Modifications and Compatibility with INLA**

  The modifications presented in Section 5.2, which includes a partitioning process, transformations to the log-scale, approximation with an initial quadratic fit and performing a skew correction, overcomes the problem of polynomial choice. It also sychronises well with INLA, since the hyperparameters are initially transformed to something resembling a Gaussian. We conclude using the examples presented in Section 5.3 that our new approximations perform far better than INLA's grid-based method with respect to both computational speed and accuracy. We also conclude that by performing the corrections by analysing residuals, that our method is more flexible than INLA's numerical integration free algorithm (NIFA), since NIFA can only approximate unimodal densi-

ties. However, we cannot claim that our methods are more computationally efficient than NIFA.

- **Latent Field Approximations using LDS**

  Recent upgrades to INLA has resulted in allowing user-defined points in the INLA algorithm, which allows us to estimate latent parameters using an LDS design. Performing this on a simple AR(1) model shows no real difference between using an LDS or any of INLA's strategies (empirical Bayes (EB), central composite designs (CCD) or grids) with respect to estimation, though computationally, the grid was far slower. Testing this on our challenging example in Chapter 6 highlighted a few key points. First, although using an LDS was computationally slower than using CCD, it was faster than the grid. In fact, the grid was not computationally feasible and crashed several times with memory issues. The LDS used 64 points, with the CCD only using 16, thus we would expect better approximations of the latent parameters. We observed that this was the case and that our predictive hotspot maps identified more instances of burglaries at higher hotspot levels than the CCD points.

7.2 Further Work

As stated in the introduction of this chapter, we cannot claim that this work is complete. There are a number of questions left outstanding, and we make the following suggestions for potential future work.

- The modified-LDS algorithm with polynomial corrections require a choice of the number of points and partitions. If we only fit a quadratic as our approximation, the number of partitions must be three or greater. Going further and fitting a cubic correction requires at least four partitions. We must also have a sufficient number of points in each partition to get an appropriate estimate for the pointwise mean. More work is needed to provide guidance on the best number of points or partitions that should be used, given the dimensionality of a problem. In our examples in Section 5.3 (both of them being five dimensional problems), we used 15 partitions with 512 points, which worked well as

there were a good number of points in each partition. For the Zambia problem, we could go as low as 64 points and 15 partitions, though we could not do this for the Georgia example. Further analysis into this problem of points and partitions would be beneficial and may help towards getting this implemented into INLA itself, which would be useful in solving the next two points.

- A full implementation of our methods into INLA would allow for a deeper analysis of performance. Currently, we are assessing computational efficiency based on the number of points used only, not necessarily actual computational time. Whilst we feel that this is enough to determine what methods are best with respect to speed, it would be beneficial to make comparisons between, say LDS and grids, with respect to computational time.

- Following from the previous point, full implementation of LDS methods into INLA would allow to make a full inference analysis of a problem. Estimations of the latent field in Chapter 6 are performed, but estimations of the model hyperparameters are performed (by default) by the NIFA strategy, since LDS methods are not implemented yet in INLA. We can of course estimate the hyperparameters separately, however analysis on the true computational time cannot be performed.

- We use Korobov lattices in the LDS methods we propose, due to their ease of use and ability to find good generating constants with current software. However, there are many other LDS point sets that could be explored in this problem. Our general feeling is that it may not greatly improve approximations, but it could still be a worthwhile task to explore different point sets.

- The crime models presented in Chapter 6 are currently a work in progress. There are several ways in which these models could be improved. We have chosen three covariates that we feel represent social and environmental factors of crime, though we do not claim that these are the best choice. Boundary problems can arise in the hotspot analysis for our spatio-temporal model with no covariates, suggesting that a finer mesh defined along the boundaries may be necessary. Finally, our spatial domain is quite naive, with only taking into account the physical boundaries of the river and lake. We have not factored in

areas such as parks, industrial areas and schools. A greater representation of the spatial domain would make these models more informative for the end-user.

Bibliography

[1] J. Atkinson, C. Salmond, and C. Crampton. *NZDEP2013 Index of Deprivation*. New Zealand Ministry of Health, NZ, (2014).

[2] H. Austad and N. Friel. Deterministic Bayesian inference for the p* model. *AISTATS, Journal of Machine Learning*, 9:41–48, (2010).

[3] A. Azzalini and A. Capitano. Statistical applications of the multivariate skew-normal distribution. *Journal of the Royal Statistical Society: Series B*, 61(2):579–602, (1999).

[4] H. Bakka, J. Vanhatalo, J. Illian, D. Simpson, and H. Rue. Non-stationary Gaussian models with physical barriers. *ArXiv e-prints*, (2016).

[5] R. Bartoszynski and M. Neiwiadomska Bugaj. *Probability and Statistical Inference, 2nd Edition.* Wiley, New Jersey, (1997).

[6] J.M. Bernado and A.F.M. Smith. *Bayesian Theory.* Wiley, Chichester, (2000).

[7] J. Besag. Spatial interaction and the statistical analysis of lattice systems (with discussion). *Journal of the Royal Statistical Society: Series B*, 36(2):179 – 195, (1974).

[8] M. Blangiardo and M. Cameletti. *Spatial and Spatio-temporal Bayesian Models with R-INLA*. Wiley, Chichester, UK, (2015).

[9] W.M. Bolstad. *Introduction to Bayesian Statistics, 2nd Edition*. Wiley, New Jersey, (2007).

[10] W.M. Bolstad. *Understanding Computational Bayesian Statistics*. Wiley, New Jersey, (2010).

[11] I. Borosh and H. Niederreiter. Optimal multipliers for pseudo-random number generation by the linear congruential method. *Bit*, 23:115 – 129, (1983).

[12] N. Bou-Rubee and M. Hairer. Non-asymptotic mixing of the MALA algorithm. *IMA Journal of Numerical Analysis*, 33(1):80–110, (2013).

[13] A. Brix and P.J. Diggle. Spatio-temporal prediction for log-Gaussian Cox processes. *Journal of the Royal Statistical Society: Series B*, 63(2):823 – 841, (2001).

[14] M. Brown. Modelling the spatial distribution of suburban crime. *Economic Geography*, 58(3):247 – 261, (1982).

[15] P.T. Brown, C. Joshi, S. Joe, and N. McCarter. Spatio-temporal modelling of crime using low discrepancy sequences. *Proceedings of the 31st International Workshop on Statistical Modelling*, 2:7 – 12, (2016).

[16] C.G Broyden. The convergence of a class of double-rank minimization algorithms. *Journal of the Institute of Mathematics and Its Applications*, 21(1):368–381, (1970).

[17] S.P. Chainey and B.F.A. da Silva. Examining the extent of repeat and near repeat victimisation of domestic burglaries in Belo Horizonte, Brazil. *Crime Science*, 5(1):1 – 10, (2016).

[18] W. Cheney and D. Kincaid. *Numerical Mathematics and Computing, 2nd Ed*. ITP, Belmont, CA., (1994).

[19] O. Christensen, G. Roberts, and M. Skold. Bayesian analysis of spatial GLMM using partially non-centred MCMC methods. *Journal of Computational and Graphical Statistics*, 15(1):1–17, (2006).

[20] P.D. Congdon. *Applied Bayesian Hierarchical methods*. Clarendon Press, Boca Raton, (2010).

[21] N. Cressie. *Statistics for Spatial Data*. Wiley, UK, (1993).

[22] J. Dick, F.Y. Kuo, and I.H. Sloan. High dimensional integration - the quasi-Monte Carlo way. *Acta Numerica*, 22:133–288, (2013).

[23] J. Dick and F. Pillichshammer. *Digital Nets and Sequences*. Cambridge University Press, Cambridge, (2010).

[24] L. Fahrmeir and G. Tutz. *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer, Berlin, (1994).

[25] J. Fitterer, T.A. Nelson, and F. Nathoo. Predictive crime mapping. *Police Practice and Research: An International Journal*, 16(2):121 – 135, (2014).

[26] G.A. Fuglstad, D. Simpson, F. Lindgren, and H. Rue. Constructing priors that penalize the complexity of Gaussian random fields. *ArXiv e-prints*, (2015).

[27] A. Gelman, J.B. Carlin, H.S. Stern, D.B. Dunson, A. Vehtari, and D.B. Rubin. *Bayesian Data Analysis, 3rd Edition*. Chapman and Hall, New York, (2014).

[28] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–740, (1984).

[29] J.E. Gentle. *Matrix Algebra. Theory, Computations and Applications in Statistics.* Springer, New York, (2007).

[30] W.R. Gilks and P. Wild. Adaptive rejection sampling for gibbs sampling. *Journal of the Royal Statistical Society: Series C*, 41(2):337–348, (1992).

[31] P.J. Green, K. Latuszyński, M. Pereyra, and C.P. Robert. Bayesian computation: a perspective on the current state, and sampling backwards and forwards. *Statistics and Computing*, 25:835–862, (2015).

[32] G. Grimmett and D. Stirzaker. *Probability and Random Processes, 3rd Edition*. Clarendon Press, Oxford, (2009).

[33] W.K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, (1970).

[34] F.J. Hickernell. Lattice rules: How well do they measure up? *Random and Quasi-Random Pointsets*, Volume 138 of Lecture notes in Statistics:109–166, (1998).

[35] R.V. Hogg and E.A. Tanis. *Probability and Statistical Inference, 5th Edition.* Prentice-Hall, New Jersey, (1997).

[36] J.B. Illian, S.H. Sørbye, and H. Rue. A toolbox for fitting complex spatial point process models using the integrated nested Laplace approximation. *Annals of Applied Statistics*, 6(1):499 – 530, (2012).

[37] C. Joshi. *A New Method to Implement Bayesian Inference on Stochastic Differential Equation Models*. Ph.D Thesis, Trinity College Dublin, Ireland, (2011).

[38] C. Joshi, P. T. Brown, and S. Joe. Improving grid based Bayesian methods. *ArXiv e-prints*, (2016).

[39] C. Joshi, C. D'Ath, S. Curtis-Ham, D. Searle, and C. Thresher. *On predictive models of crime and their measures*. In Press, (2018).

[40] N.B. Kandala, S. Lang, S. Klasen, and L. Fahrmeir. Semiparametric analysis of the socio-demographic and spatial determinants of undernutrition in two African countries. *Research in Official Statistics*, 1:81 – 100, (2001).

[41] T. Kneib, S. Lang, and A. Brezger. Bayesian semiparametric regression based on MCMC techniques: A tutorial, (2004).

[42] N.M. Korobov. The approximate computation of multiple integrals. *Doklady Academii Nauk SSSR*, 124:1207–1210, (1959).

[43] R. Kress. *Numerical Analysis*. Springer, New York, (1998).

[44] S. Kullback and A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79 – 86, (1951).

[45] P. L'Ecuyer and C. Lemieux. Variance reduction via lattice rules. *Management Science*, 46(9):1214 – 1235, (2000).

[46] P. L'Ecuyer and D. Munger. A general software tool for constructing rank-1 lattice rules. *ACM Transactions in Mathematical Software*, 42(2):Article 15, (2016).

[47] C. Lemieux. *Monte Carlo and Quasi-Monte Carlo Sampling*. Springer, New York, (2009).

[48] F. Liese and K.J. Miescke. *Statistical Decision Theory: Estimation, Testing and Selection*. Springer, New York, (2008).

[49] F. Lindgren and H. Rue. On the second order random walk model for irregular locations. *Scandinavian Journal of Statistics*, 35(4):691 – 700, (2008).

[50] F. Lindgren, H. Rue, and J. Lindström. An explicit link between Gaussian fields and Gaussian markov random fields: the stochastic partial differential equation approach. *Journal of the Royal Statistical Society: Series B*, 73(2):423 – 498, (2011).

[51] D.J.C. Mackay. Ensemble learning and evidence maximisation. *Technical Report - Cavendish Laboratory*, (1995).

[52] J.R. Magnus and H. Neudecker. *Matrix Differential Calculus with Applications to Statistics and Economics, Revised Edition*. Wiley, West Sussex, (1999).

[53] M. Maguire and T. John. Intelligence led policing, managerialism and community engagement: Competing priorities and the role of the National Intelligence Model in the UK. *Policing and Society*, 16(1):67 – 85, (2006).

[54] S. Martino and H. Rue. Case studies in Bayesian computation using INLA. In P. Mantovan and P. Secchi, editors, *Complex Data Modelling and Computationally Intensive Statistical Methods*. Springer, Milano, (2010).

[55] Martino, S., and Rue, H. *Implementing Approximate Bayesian Inference using Integrated Nested Laplace Approximations: Manual for the INLA Program*. Department of Mathematical Sciences, NTNU, Norway, (2009).

[56] T. Martins, D. Simpson, F. Lindgren, and H. Rue. Bayesian computing with INLA: new features. *Computational Statistics and Data Analysis*, 67:68–83, (2013).

[57] T.P. Minka. Expectation-propogation for approximate Bayesian inference. *Uncertainty in Artificial Intelligence*, 17(1):362 – 369, (2001).

[58] J. Møller and R.P. Waagepetersen. *Statistical Inference and Simulation for Spatial Point Processes*. Chapman and Hall, London, (2004).

[59] Metropolis N.A., A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Chemical Physics*, 21:1087–1092, (1953).

[60] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. Volume 63 of SIAM CMBS-NSF Regional Conference Series in Applied Mathematics SIAM, Philidelphia, (1992).

[61] J.T. Ormerod. Grid based variational approximations. *Computational Statistics and Data Analysis*, 55:45–56, (2011).

[62] S.J. Press. *Subjective and Objective Bayesian Statistics.* Wiley, New Jersey, (2003).

[63] R Development Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, (2008). ISBN 3-900051-07-0.

[64] J.H. Ratcliffe. Aoristic signatures and the spatio-temporal analysis of high volume crime patterns. *Journal of Quantitative Criminology*, 18(1):23 – 43, (2002).

[65] J.H. Ratcliffe. *Intelligence-Led Policing*. Willan Publishing, Cullompton, UK, (2008).

[66] J.H. Ratcliffe and M. J. McCullagh. Hotbeds of crime and the search for spatial accuracy. *Journal of Geographical Systems*, 1(4):669 – 679, (1998).

[67] J.H. Ratcliffe and M. J. McCullagh. Identifying repeat victimization with GIS. *British Journal of Criminology*, 38(4):651 – 662, (1998).

[68] C.P. Robert. *The Bayesian Choice: From Decision-Theoretic Foundations to Computational Implementation*. Springer, New York, (2007).

[69] C.P. Robert and G. Casella. *Monte Carlo Statistical Methods*. Springer, New York, (2004).

[70] C.P. Robert and G. Casella. *Introducing Monte Carlo Methods with R*. Springer, New York, (2010).

[71] G. Roberts and R. Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(1):341–363, (1996).

[72] H. Rue and L. Held. *Gauss Markov Random Fields: Theory and Applications*. Chapman and Hall, London, (2005).

[73] H. Rue and S. Martino. Approximate Bayesian inference for hierarchical Gaussian Markov random field models. *Journal of statistical planning and inference*, 137(1):3177–3192, (2007).

[74] H. Rue, S. Martino, and N. Chopin. Approximate Bayesian inference for latent Gaussian models by using Integrated Nested Laplace Approximations. *Journal of the Royal Statistical Society: Series B*, 71(2):319–392, (2009).

[75] H. Rue, A. Riebler, S.H. Sørbye, J.B. Illian, D.P. Simpson, and S.H. Lindgren. Bayesian computing with INLA: a review. *The Annual Review of Statistics and Its Applications*, 4(1):395–421, (2016).

[76] C. Runge. Über empirische funktionen und die interpolation zwischen äquidistanten ordinaten. *Zeitschrift für Mathematik und Physik*, 46:224 – 243, (1901).

[77] D. Simpson, J.B. Illian, F. Lindgren, S.H. Sørbye, and H. Rue. Going off grid: computationally efficient inference for log-Gaussian Cox processes. *Biometrika*, 103(1):49 – 70, (2016).

[78] D. Simpson, F. Lindgren, and H. Rue. Fast approximate inference with INLA: the past, the present and the future. *ArXiv e-prints*, (2011).

[79] I.H. Sloan and S. Joe. *Lattice Methods for Multiple Integration*. Clarendon Press, Oxford, (1994).

[80] K. Takahashi, J. Fagan, and M.S. Chen. Formation of a sparse bus impedence matrix and its application to short circuit study. *Eighth PICA Conference Proceedings. IEEE Power Engineering Society*, pages 63–69, (1973).

[81] B. Taylor and P.J. Diggle. INLA or MCMC? A tutorial and comparitive evaluation for spatial prediction in log-Gaussian Cox processes. *Journal of Statistical Computation and Simulation*, 84(10):2266–2684, (2014).

[82] L. Tierney. Markov chains for exploring posterior distributions. *The Annals of Statistics*, 22:1701–1728, (1994).

[83] L. Tierney and J.B. Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81:82–86, (1986).

[84] M. Townsley, R. Homel, and J. Chaseling. Repeat burglary victimization: spatial and temporal patterns. *Australian and New Zealand Journal of Criminology*, 33(1):37 – 63, (2000).

[85] P. Whittle. On stationary processes in the plane. *Biometrika*, 41:434 – 449, (1954).

[86] C. Wikle. Hierarchical models in environmental sciences. *International Statistical Review*, 71:181–199, (2003).

[87] S. P. Wilson and J. Yoon. Bayesian ICA-based source separation of Cosmic Microwave Background by a discrete functional approximation. *ArXiv e-prints*, (2010).