A Novel Method of Marginalisation using Low Discrepancy Sequences for Integrated Nested Laplace Approximations

Paul T. Brown, Chaitanya Joshi, Stephen Joe, and Håvard Rue

December 5, 2019

Abstract

Recently, it has been shown that approximations to marginal posterior distributions obtained using a low discrepancy sequence (LDS) can outperform standard grid-based methods with respect to both accuracy and computational efficiency. This recent method, which we will refer to as LDS–StM, can also produce good approximations to multimodal posteriors. However, implementation of LDS–StM into integrated nested Laplace approximations (INLA), a methodology in which grid-based methods are used, is challenging. Motivated by this problem, we propose modifications to LDS–StM that improves the approximations and make it compatible with INLA, without sacrificing computational speed. We also present two examples to demonstrate that LDS–StM with modifications can outperform INLA's own grid approximation with respect to speed and accuracy. We also demonstrate the flexibility of the new approach for the approximation of multimodal marginals.

1 Introduction

Integrated nested Laplace approximations (INLA) [28], is a methodology developed specifically for fast approximate Bayesian inference of latent Gaussian models (LGM) [8], a large class of hierarchical Bayesian models. INLA was developed as a more computationally efficient alternative to more widely used Markov chain Monte Carlo (MCMC) methods. Initially, INLA used grid-based methods of marginalisation to compute marginal hyperparameter posterior distributions [22, 28]. Generally, a set of points is laid out in a grid structure over a relevant integration region. Along with numerical integration techniques and an interpolant, posterior marginals are constructed. Grid-based methods have been used in different applications, with several variants used, see [9, 11, 26, 33]. Grids are known to be computationally efficient only in very low dimensions, since the number of points increases

1

exponentially with dimension.

Computational advancements within INLA have given rise to new algorithms that improve computational efficiency. A set of central composite design (CCD) points [3] is laid out over the hyperparameter space, and marginals are computed using an asymmetric Gaussian interpolant. The CCD strategy requires far less points than grid-based strategies (see [30] for details of how to compute these points). The numerical integration-free method (NIFA) [23], a method bypassing numerical integration altogether, has achieved further computational gains. However, the CCD and NIFA strategies do have the drawback of only allowing for unimodal approximations. For details on both of these strategies, see [23]. Recently, a new method called Bayesian inference using sparse grid quadrature evaluation (BISQuE) [9] uses sparse grids (see [25]) to explore the hyperparameter space and estimate marginal posteriors. BISQuE is similar to INLA as a Bayesian inference framework for hierarchical models, though not necessarily just for latent Gaussian models.

An alternative strategy for hyperparameter exploration and estimation, developed by the authors of [12], show that a very general grid-based method can be improved upon by using a low discrepancy sequence (LDS) in the place of a grid. Using a least squares polynomial as an interpolant, [12] proved that this LDS-based method of marginalisation (hereafter, referred to as the standard LDS method, or LDS–StM) improved the computational efficiency by reducing the number of points needed, whilst also showing convergence to the true marginal. LDS–StM also allowed for the approximation of multimodal posterior densities.

Although LDS–StM performed better than grid-based methods in a general setting, the method itself was not easy to implement within INLA. Issues arose when choosing the degree of the polynomial. If a higher degree was needed (such as with the approximation of skewed, or multimodal distributions), the resulting matrices were sometimes ill-conditioned, and therefore some computation was inaccurate. They also encountered issues with Runge's phenomenon [7], where oscillations would form in the tails of the approximations. This paper seeks to expand on the development of the LDS–StM method by applying important modifications to the algorithm to make it more useful in a practical setting. The goal is to fully incorporate a modified LDS-based method into INLA, specifically, for the accurate and efficient estimation of hyperparameter posterior marginal distributions of an LGM. We present the modifications, and two challenging examples of estimating hyperparameter posterior marginals to demonstrate how this modified method works, and how it performs relative to INLA's current methods (INLA's grid and NIFA). The methods presented may also be used

in BISQuE as well, though this will require further investigation.

This paper is organised as follows. First, we give some necessary background on QMC integration, INLA, and grid-based methods in Section 2. Section 3 gives an overview of the LDS–StM algorithm, and the modifications for implementation in INLA. We provide two examples of the performance of the new method in Section 4, before giving our concluding remarks in Section 5.

2 Background

2.1 QMC Integration Rules

In many instances, we wish to compute an integral

$$I_f = \int_D f(\boldsymbol{x})d\boldsymbol{x}, \tag{2.1}$$

where $\boldsymbol{x} = (x_1, \ldots, x_s)$ is s-dimensional, and D is the region of integration, assumed to be the unit hypercube $[0, 1)^s$. A typical approximation to I_f in (2.1) is given by

$$I_f \approx \frac{1}{N} \times \sum_{i=1}^{N} f(\boldsymbol{x}_{(i)}). \tag{2.2}$$

where the points $\boldsymbol{x}_{(i)}, i = 1, \ldots, N$ are sampled in the region of integration. One can use a Monte Carlo rule, where the points $\boldsymbol{x}_{(i)}, i = 1, \ldots, N$ are sampled randomly, or a QMC rule, where the points are generated deterministically. LDS are a large class of sequences that are generally used in (2.2) when using a QMC rule. LDS are sequences that hold the property that, for specific values of N, its finite subsequence $x_1, \ldots, x_N$ has *low discrepancy* with respect to the Lebesgue measure on the unit hypercube. See [5, 16, 24] for different examples of low discrepancy sequences and definitions of discrepancy. It can be shown via discrepancy theory and the Koksma-Hlawka inequality [18] that for functions with bounded variation (in the sense of Hardy and Krause), the estimator in (2.2) converges to the true value faster if the points are generated using an LDS, rather than randomly sampled. This result also extends to low discrepancy point sets [19], such as the Korobov lattice which we will introduce shortly. For the purposes of this paper, we use the term LDS to refer to both low discrepancy sequences, and low discrepancy point sets.

We define here the two deterministic point sets that we use (see Figure 1). First, we define an s-dimensional n-point grid by

$$\mathcal{G}_{n,s} = \left\{ \left(\frac{g_1}{n}, \cdots, \frac{g_s}{n} \right), g_j = 0, \ldots, n-1, j = 1, \ldots, s \right\}, \tag{2.3}$$

where n is the number of unique abscissa points in each dimension, s is the dimension of the point set and $N = n^s$ is the total number of points. These point sets are deterministic but do not have low *discrepancy*. As such, the convergence rate is slow as the number of points increase exponentially as dimension increases [24].

The next deterministic point set we define is the Korobov lattice which, unlike the grid, is an LDS. The s-dimensional Korobov lattice [31] with N points is given by

$$\mathcal{K}_{N,s,\alpha} = \left\{ \frac{i-1}{N}(1, \alpha, \alpha^2, \ldots, \alpha^{s-1}) \bmod 1, i = 1, \ldots, N \right\}, \qquad (2.4)$$

where α is called the generating constant, and is typically chosen to be an integer between 1 and $N-1$, and relatively prime with N. We use the software LatticeBuilder [17], to find "good" generating constants with respect to a discrepancy measure, for fixed s, N, and constant weights. LatticeBuilder also finds generating constants for extensible Korobov lattices. Extensible lattices have the property that the number of points N can increase whilst retaining existing points and maintaining low discrepancy (see [10] for details). We denote an extensible Korobov lattice with number of points N, dimension s, and generating constant α, by $\mathcal{K}^*_{N,s,\alpha}$.

Figure 1: Point sets: An 8-point grid $\mathcal{G}_{8,2}$ (top left), a Korobov lattice $\mathcal{K}_{64,2,37}$ (top right), an extensible Korobov lattice $\mathcal{K}^*_{32,2,19}$ (bottom left), and an extensible Korobov lattice $\mathcal{K}^*_{64,2,19}$ (bottom right) based off the previous lattice. The extra 32 points are shown as hollow circles.

High dimensional integrals arise naturally in the field of Bayesian inference, where marginal posterior distributions are computed from a joint posterior distribution. These integrals are also used in other statistical ap-

plications such as the computation of marginal likelihoods. In general, we wish to estimate the following integral

$$f_k(u_k) = \int_\chi f(\boldsymbol{u}) d\boldsymbol{u}_{-k}, \quad k = 1, \ldots, s,$$

where $f_k(u_k)$ is the kth marginal function of the variable u_k, and $\boldsymbol{u}_{-k}$ is the variable $\boldsymbol{u}$ excluding the kth component. Taking χ to be some rectangular region $[\boldsymbol{a}, \boldsymbol{b})$ where $\boldsymbol{a} = (a_1, \ldots, a_s)$ and $\boldsymbol{b} = (b_1, \ldots, b_s)$, we generate a point set $\boldsymbol{x}_{(i)}, i = 1, \ldots, N$ in $[0, 1)^s$ and use a linear transformation,

$$\boldsymbol{u} = \boldsymbol{a} + (\boldsymbol{b} - \boldsymbol{a})\boldsymbol{x}, \tag{2.5}$$

so that $\boldsymbol{u}_{(i)}, i = 1, \ldots, N$ are scaled to the region $[\boldsymbol{a}, \boldsymbol{b})$. The marginal function $f_k(u_k)$ can be approximated by

$$f_k(u_k) \approx \frac{\prod_{j=1}^s (b_j - a_j)}{b_k - a_k} \times \frac{1}{N} \sum_{i=1}^N f(u_{1,(i)}, \ldots, u_k, \ldots, u_{s,(i)}), \tag{2.6}$$

where $f(u_{1,(i)}, \ldots, u_k, \ldots, u_{s,(i)})$ are the function evaluations keeping the variable u_k fixed. In the context of Bayesian inference, MCMC algorithms are used to approximate marginal posteriors. However, for some types of hierarchical models such as LGMs, these sampling methods may not necessarily be computationally efficient [28].

2.2 INLA

INLA is a computational Bayesian inference framework that was proposed as a computationally efficient alternative to MCMC methods [28]. It is particularly designed for LGMs, a widely used class of Bayesian hierarchical model, encompassing basic regression and time series models to complex spatio-temporal point process models and many others (see [29] for a comprehensive list of applications).

Given a data set with n_d observations, $\boldsymbol{y} = (y_1, \ldots, y_{n_d})$, assume that the response variable y_i follows a distribution from the exponential family, with its conditional mean μ_i associated with η_i through a link function ℓ, such that $\eta_i = \ell(\mu_i)$. The LGM has the following additive structure

$$\eta_i = \beta_0 + \sum_{j=1}^{n_\beta} \beta_j u_{ij} + \sum_{k=1}^{n_f} g_k(v_{ik}),$$

where β_0 is the intercept, β_j are the fixed effects for covariates $\boldsymbol{u}$, and g_k are the model components, associated with the covariates $\boldsymbol{v}$. These models

can be expressed in a generic three-stage hierarchical formulation. For the first stage, we assume the data $\boldsymbol{y}$ are independent of each other, given $\boldsymbol{\eta}$ and hyperparameters $\boldsymbol{\theta}_1$, so

$$\boldsymbol{y}|\boldsymbol{\eta}, \boldsymbol{\theta}_1 \sim \prod_{i=1}^{n_d} \pi(y_i|\eta_i, \boldsymbol{\theta}_1).$$

The second stage specifies $\boldsymbol{\eta}$ to be a latent Gaussian field with the density function

$$\boldsymbol{\eta}|\boldsymbol{\theta}_2 \sim \mathcal{N}(\mathbf{0}, \boldsymbol{Q}^{-1}(\boldsymbol{\theta}_2)),$$

where $\mathcal{N}$ is a multivariate Gaussian distribution. In many applications, the latent Gaussian field has conditional independence properties, translating to a multivariate Gaussian distribution with sparse precision matrix $\boldsymbol{Q}(\boldsymbol{\theta}_2)$, otherwise known as a Gaussian Markov random field (GMRF) [27]. The final stage is the hyper-prior stage

$$\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) \sim \pi(\boldsymbol{\theta}),$$

where $\boldsymbol{\theta}$ has s components.

INLA works by approximating the posterior density of hyperparameters $\boldsymbol{\theta}|\boldsymbol{y}$ using a Laplace approximation [32]. A Gaussian approximation $\tilde{\pi}_G(\boldsymbol{\eta}|\boldsymbol{\theta}, \boldsymbol{y})$ for the posterior for the latent parameters is made, and evaluated at the posterior mode $\boldsymbol{\eta}^*(\boldsymbol{\theta}) = \underset{\boldsymbol{\eta}}{\operatorname{argmax}} \pi(\boldsymbol{\eta}|\boldsymbol{\theta}, \boldsymbol{y})$. Let $\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y})$ be the approximation to the joint hyperparameter density. Then the Laplace approximation is given by

$$\tilde{\pi}(\boldsymbol{\theta}|\boldsymbol{y}) \propto \left. \frac{\pi(\boldsymbol{\theta}, \boldsymbol{\eta}|\boldsymbol{y})}{\tilde{\pi}_G(\boldsymbol{\eta}|\boldsymbol{\theta}, \boldsymbol{y})} \right|_{\boldsymbol{\eta}=\boldsymbol{\eta}^*(\boldsymbol{\theta})}. \tag{2.7}$$

The marginals of the hyperparameters $\theta_k, k = 1, \ldots, s$, are found by exploring and then taking specific points in the hyperparameter space, either in a grid structure or otherwise, and using numerical integration and an interpolant [23, 28]. Those points are reused in the approximation of the latent parameter marginals $\pi_j(\eta_j|\boldsymbol{y}), j = 1, \ldots, n_d$ via the following equation

$$\pi_j(\eta_j|\boldsymbol{y}) \approx \sum_{i=1}^{N} \tilde{\pi}(\eta_j|\boldsymbol{\theta}_{(i)}, \boldsymbol{y})\tilde{\pi}(\boldsymbol{\theta}_{(i)}|\boldsymbol{y})\Delta_i, \tag{2.8}$$

where Δ_i are the integration weights, $\tilde{\pi}$ is a Laplace approximation of π, and $\boldsymbol{\theta}_{(i)}, i = 1, \ldots, N$ are the points generated by the integration rule.

The number of latent parameters INLA can handle is very large, in the order of 10^5 [29]. However, to ensure INLA's computational efficiency, it

requires that the number of hyperparameters must be small, since those points are re-used to approximate each latent marginal [23, 28]. INLA loses much of its speed using a grid strategy for models with five or more hyperparameters, as the number of points increases exponentially with s.

During the exploration of the hyperparameter space, an optimisation algorithm is used to approximate the mode and Hessian. One way to approximate marginals would be to fit a multivariate Gaussian by matching the mode and curvature at the mode via the Hessian. However, the marginals are often skewed so this approximation is often not accurate. Instead a transformation of the hyperparameter space is performed, which is orthogonal and standardised (see [28] for details). Let $\hat{\boldsymbol{\theta}}$ denote the transformed space. In $\hat{\boldsymbol{\theta}}$, define

$$\tilde{\pi}(\hat{\boldsymbol{\theta}}|\boldsymbol{y}) \propto \prod_{k=1}^{s} \tilde{\pi}_k(\hat{\theta}_k|\boldsymbol{y}),$$

where $\tilde{\pi}_k(\hat{\theta}_k|\boldsymbol{y})$ is tangent to the marginal $\tilde{\pi}_k(\theta_k|\boldsymbol{y})$. To capture the asymmetry of $\tilde{\pi}(\hat{\theta}_k|\boldsymbol{y})$, INLA assumes the following structure

$$\tilde{\pi}_k(\hat{\theta}_k|\mathbf{y}) \propto \begin{cases} \exp\left(-\dfrac{1}{2\sigma_{k+}^2}(\hat{\theta}_k^2 - \mu_k)\right), \hat{\theta}_k > 0 \\ \exp\left(-\dfrac{1}{2\sigma_{k-}^2}(\hat{\theta}_k^2 - \mu_k)\right), \hat{\theta}_k \leq 0, \end{cases} \tag{2.9}$$

where the variances σ_{k+}^2 and σ_{k-}^2 are approximated assuming each marginal is "half-Gaussian", with a different variance either side of the mode. The NIFA method assumes the structure in (2.9) and then computes $\sigma_{k+}, \sigma_{k-}, k = 1, \ldots, s$ without numerical integration (see [23] for details). Whilst these methods greatly reduce the number of points, the grid is more accurate overall, since the approximations from the grid do not assume any kind of unimodal structure.

2.3 Grid-Based Marginalisation Methods

Grid-based methods are relatively simple in concept and straightforward to implement. A set of grid points is placed in an appropriate integration region. After evaluating the function at those points, we average out over rows and columns and fit an interpolant between the function evaluation means.

Let $\pi(\boldsymbol{\theta})$ be any s-dimensional density that we wish to marginalise. We drop the dependence on $\boldsymbol{y}$ for notational convenience. Also, let $\mathcal{G}_{n,s} = \{\boldsymbol{g}_{(1)}, \ldots, \boldsymbol{g}_{(N)}\}$ be an equally spaced, s-dimensional n-point grid, as described in (2.3). Note that the grid is generally constructed over the unit

hypercube, so a scaling transformation, given by

$$T(\boldsymbol{g}) = \boldsymbol{a} + (\boldsymbol{b} - \boldsymbol{a})\boldsymbol{g}, \tag{2.10}$$

where $\boldsymbol{g}$ is a grid point, $\boldsymbol{a}$ and $\boldsymbol{b}$ are the lower and upper vertices of the integration region, must be performed on the grid points. We denote this transformed point set by $\{\boldsymbol{\theta}_{(1)}, \ldots, \boldsymbol{\theta}_{(N)}\}$. We present a general method for grid-based marginalisation in Algorithm 1.

Algorithm 1 General grid-based method

---

1) Optimise $\pi(\boldsymbol{\theta})$ for mode π^* and Hessian H_π
2) Construct an appropriate integration region $[\boldsymbol{a}, \boldsymbol{b})$
3) Generate grid points $\mathcal{G}_{N,s}$ and transform points to $\{\boldsymbol{\theta}_{(j)}\}_{j=1}^N$ over $[\boldsymbol{a}, \boldsymbol{b})$ and evaluate $\pi(\boldsymbol{\theta}_{(j)})$ for all $j = 1, \ldots, N$

for $k = 1, \ldots, s$ **do**

    4) Orthogonally project points $(\boldsymbol{\theta}_{(j)}, \pi(\boldsymbol{\theta}_{(j)}))$ for $j = 1, \ldots, N$ onto $[a_k, b_k)$

    5) For $l = 1, \ldots, n$, where n is the number of abscissa points, compute pointwise means

$$\hat{\pi}(\theta_{k,l}) = \frac{1}{n^{s-1}} \sum_{i_1=1}^{n} \cdots \sum_{i_{s-1}=1}^{n} \pi(\theta_{1,(i_1)}, \ldots, \theta_{k,l}, \ldots, \theta_{s,(i_{s-1})}) \tag{2.11}$$

6) Fit interpolant through pointwise means and normalise for $\tilde{\pi}_k(\theta_k)$

Step 1 can be achieved through any quasi-Newton algorithm, and this is used to construct an integration region $[\boldsymbol{a}, \boldsymbol{b})$. Grid points are generated in the unit hypercube, and transformed to lie within $[\boldsymbol{a}, \boldsymbol{b})$, and function evaluations are computed. For each coordinate, an orthogonal projection is made onto each marginal axis $[a_k, b_k)$ for $k = 1, \ldots, s$. Details of the orthogonal projection used in Step 4 are provided in the Appendix. At each abscissa point, the pointwise mean (2.11) is calculated from the n^{s-1} function evaluations that correspond to the abscissa point (Step 5). An interpolant can then be fitted (such as a cubic spline) and normalised for the grid approximation to the kth marginal.

Grid-based methods are flexible in the implementation process. Changes can be made to the general method which can result in similar approximations with less computational effort. For instance, INLA chooses points by only generating important points based off some user-defined criteria which defines the denseness of the grid [28]. Thus it can save on points which improves computational efficiency. The BISQuE [9] method use sparse grids for approximation to marginal densities, reducing the number of points needed

for approximation. Methods described in [4] generate grid-like structures, such as maximal-rank lattice point sets [6], which can better capture the behaviour of the function, improving both accuracy and computational efficiency. However, the number of points still increases exponentially with the dimension, and thus a new approach is required to reduce the number of points further. We believe that the use of an LDS instead of a grid is one such approach, since the number of LDS points do not increase exponentially with dimension.

3 LDS–StM Method and Modifications

First, we give a brief overview of the LDS–StM method before presenting the modifications. There are two sets of modifications, both of which have corresponding algorithms, the LDS–QA (LDS with quadratic approximation), and the LDS–CX (LDS with correction polynomial of degree X). We present the algorithms and details later this section.

3.1 Standard LDS method (LDS–StM)

The LDS–StM was developed in [12] as a faster alternative to grid-based methods. In the place of a grid, we instead generate an LDS. Many LDS sequences would do, but we use a Korobov Lattice as described in (2.4) and use LatticeBuilder to find appropriate generating constants. Let $\mathcal{K}_{N,s,\alpha}$ be a Korobov lattice with N points in s dimensions, and generating constant α. Again, we scale the Korobov lattice points generated from the unit hypercube to the integration region $[\boldsymbol{a}, \boldsymbol{b})$ using the scaling transformation in (2.10). We present LDS–StM in Algorithm 2.

Algorithm 2 LDS–StM Algorithm

---

1) Optimise $\pi(\boldsymbol{\theta})$ for mode π^* and Hessian H_π
2) Construct an appropriate integration region $[\boldsymbol{a}, \boldsymbol{b})$
3) Generate a set of LDS points and transform to $\{\boldsymbol{\theta}_{(j)}\}_{j=1}^{N}$ over $[\boldsymbol{a}, \boldsymbol{b})$
   and evaluate $\pi(\boldsymbol{\theta}_{(j)})$ for $j = 1, \ldots, N$
for $k = 1, \ldots, s$ **do**
    4) Orthogonally project points $(\boldsymbol{\theta}_{(j)}, \pi(\boldsymbol{\theta}_{(j)}))$ for $j = 1, \ldots, N$ onto
      $[a_k, b_k)$
    5) Fit least squares polynomial through the orthogonally projected
      points $(\theta_{k,(j)}, \pi(\boldsymbol{\theta}_{(j)}))$
    6) Normalise for $\tilde{\pi}_k(\theta_k)$

Note that in the LDS–StM algorithm, after orthogonal projections of the points onto the marginal axis in Step 4, we see a scatter of ordinates. These ordinates correspond to N equally spaced and unique abscissa points along

the θ_k-axis, which is a result of a property of LDS known as fully projection regular (see [18] for more details). Figure 2 shows the orthogonal projections of the function evaluations onto a single dimension. An interpolant such as a cubic spline that could be used in the general grid method would not be appropriate here, as the resulting approximation would be a spline fitted through each point. Instead, a least squares polynomial is used in Step 5 to approximate the shape of the marginal, and normalisation of the polynomial gives the approximation of the marginal $\tilde{\pi}_k(\theta_k)$.

The LDS–StM was shown to be more computationally efficient compared with grid-based methods [12]. Most importantly, convergence theorems were proved to show that the approximations converged to the true marginal distribution as the number of points and polynomial degree increased. However, there were some shortcomings with respect to practical implementation. There was no reasonable way to choose the degree of the polynomial. For a regular unimodal, symmetric (or close to) marginal density, numerical experiments indicated that a polynomial of degree eight was found to be generally sufficient. However, a higher degree polynomial, such as one required for a skewed or multimodal density, would often encounter Runge Phenomenon, where oscillations would form in the tails of the polynomial approximation. Inverted Gramian matrices also tended to be ill-conditioned after computation. In higher dimensions, LDS–StM saved many function evaluation points. However, the number of function evaluations were still too high to be of use for a method such as INLA.

In order to improve INLA using LDS, the LDS–StM method needed to be modified such that:

1. The degree of the polynomial is not problem dependent.

2. The method will work well on any posterior without customisation.

3. It remains a computationally efficient method after modifications to achieve 1 and 2.

We now propose two modifications to the LDS–StM method in order to develop a practical method that can be used in INLA, that satisfies the three criteria above.

3.2   Initial Modifications - Quadratic Approximation

Partitioning was used in [12] to prove convergence theorems, but not in the LDS–StM algorithm. We decide to include partitioning to aid in the choice of polynomial degree, and it also is useful for the next set of modifications outlined later. For each marginal axis, we construct n partitions. We compute the pointwise mean of each partition (similar to (2.11), but calculated

Figure 2: Left: Generating a Korobov point set around a joint posterior. Right: Orthogonally projected points in one dimension.

at partition intervals rather than unique abscissa points), and fit a polynomial degree of $n - 1$ through the pointwise means.

The Laplace approximation in (2.7) assumes $\pi(\boldsymbol{\theta}|\boldsymbol{y})$ is almost Gaussian. To aid this assumption, INLA uses variance-stabilising transformations to make each hyperparameter more symmetric [29]. For instance, instead of approximating precisions, INLA approximates log precisions. Let θ_k denote a hyperparameter, and let h be the function corresponding to the reparameterisation, then

$$h(\theta_k) = \theta_{z,k},$$

where $\theta_{z,k}$ denotes the reparameterised hyperparameter. Since these transformations make the hyperparameter close to a Gaussian, this implies we can approximate the marginal using a quadratic polynomial in the log-scale. This also suggests that we should make a minimum of three partitions along the parameter axes.

In Algorithm 3, we modify Step 4 in the LDS–StM algorithm to include partitioning and transformations. We also modify part of Step 5 for changes to the quadratic polynomial fit. Since we begin with a reparameterisation of the hyperparameters, Steps 1, 2, and 3 are all performed for the purposes of approximating the marginals $\pi(\theta_{z,k})$. Approximating $\pi(\theta_k)$ requires an inverse transformation from $\theta_{z,k}$ to θ_k.

We now discuss the LDS–QA algorithm in more detail. After projection of the function evaluation coordinates, we partition the kth marginal axis with n partitions with equal interval lengths. 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, a Korobov lattice with good generating constant

11

Algorithm 3 Quadratic Approximation - LDS–QA

4a) Orthogonally project points $(\boldsymbol{\theta}_{z,(j)}, \pi(\boldsymbol{\theta}_{z,(j)}))$ for $j = 1, \ldots, N$ onto $[a_k, b_k)$

4b) Create n equally spaced partitions of $[a_k, b_k)$ (the $\theta_{z,k}$ axis). The interval $[\theta_{z,k,u}, \theta_{z,k,u+1})$ for $u = 1 \ldots, n$ is the uth partition, denoted as $\theta_{z,k,u'}$ with $u' = 1, \ldots, n$

4c) Calculate pointwise means for each partition, where the pointwise mean of the uth partition is given by $\hat{\pi}(\theta_{z,k,u'})$

5a) Transform pointwise means to $\log(\boldsymbol{\theta}_z)$-scale, and fit a least squares quadratic polynomial through $\log(\hat{\pi}(\theta_{z,k,u})), u = 1, \ldots, n$

5b) Transform polynomial to θ_z-scale and normalise marginal $\tilde{\pi}_k(\theta_{z,k})$, then transform to θ-scale for $\tilde{\pi}_k(\theta_k)$

---

produces N equally-spaced points when projected onto any marginal axis, due to the fully projection regular property. Thus, equally spaced partitions will have a very similar number of points. Let ν_u denote the number of points in the uth 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 which is the one we use in practice. Thus, the uth abscissa point would be $0.5 \times (\theta_{z,k,u+1} + \theta_{z,k,u})$. The ordinate for the uth pointwise mean is given by

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

where $\boldsymbol{\theta}_{z,u',(j)}$ are the generated points that lie in the uth partition after being orthogonally projected. Doing this for all $u = 1, \ldots, n$ partitions gives us n pointwise means.

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

$$\tilde{\pi}_{k,\log}(\theta_{z,k}) = \tilde{\beta}_{k,0} + \tilde{\beta}_{k,1}\theta_{z,k} + \tilde{\beta}_{k,2}\theta_{z,k}^2, \tag{3.1}$$

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

$$\tilde{\pi}_k(\theta_{z,k}) = \frac{\exp(\tilde{\beta}_{k,0} + \tilde{\beta}_{k,1}\theta_{z,k} + \tilde{\beta}_{k,2}\theta_{z,k}^2)}{\int_{[a_k,b_k]} \exp(\tilde{\beta}_{k,0} + \tilde{\beta}_{k,1}\theta_{z,k} + \tilde{\beta}_{k,2}\theta_{z,k}^2) d\theta_{z,k}}, \tag{3.2}$$

where $\tilde{\pi}_k(\theta_{z,k})$ is the approximation of the marginal posterior for $\theta_{z,k}$. For the approximation of the actual hyperparameter, an inverse transformation from θ_z to θ is required.

3.3   Second Modification - Polynomial Correction

From a practical standpoint, the approximation given in (3.2) is far more stable than fitting a polynomial of high degree as performed in the LDS–StM algorithm. However, it is not guaranteed that θ_z's are exactly Gaussian. Thus, the strategy of fitting a quadratic very much constricts us to a Gaussian estimate after transformation. To account for potential discrepancies in the approximation, we propose another iteration of the algorithm that fits a polynomial with a correction term that can account for skew and multiple modes. This can be done with little extra computational effort.

We propose a correction to the quadratic/Gaussian approximation in (3.1) by analysing residuals, given by the difference between the polynomial fit and the pointwise means. We name this method the LDS–CX method, where X describes the degree of the correction polynomial. In practice, we typically fit a cubic correction polynomial (LDS–C3) to the quadratic in (3.1), which corrects it for location, scale, and skewness. The process for the LDS–CX algorithm is decribed in Algorithm 4.

Algorithm 4 LDS-CX Algorithm

5a) Transform pointwise means to $\log(\boldsymbol{\theta}_z)$-scale, and fit a least squares
    quadratic polynomial through $\log(\hat{\pi}(\theta_{z,k,u})), u = 1, \ldots, n$

5a) (i) Obtain residuals found by the difference between $\tilde{\pi}_{\log}(\theta_{z,k})$
    evaluated at the uth abscissa point, and its corresponding pointwise
    mean

5a) (ii) Fit a least squares polynomial (of degree X) through residuals

5a) (iii) Correct initial quadratic approximation with the polynomial
    coefficients

The process changes the LDS–QA algorithm slightly, by adding the polynomial correction process during Step 5a. The correction is summarised in Algorithm 4. After fitting the quadratic polynomial, instead of using this to calculate the posterior marginal, we instead calculate the residuals of the fitted quadratic (the difference between each pointwise mean and the fitted quadratic). After calculating these, we fit a polynomial through these residuals, and use it to update the quadratic. In practice, we generally fit a least squares cubic polynomial, given by

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

where $\theta_{z,k} \in [a_k, b_k]$. Then, the cubic correction update for the unnormalised log-approximation to the marginal posterior $\pi(\theta_{z,k})$ is

$$\tilde{\pi}_{\log,P}(\theta_{z,k}) = (\tilde{\beta}_{k,0} - \beta'_{k,0}) + (\tilde{\beta}_{k,1} - \beta'_{k,1})\theta_{z,k} + (\tilde{\beta}_{k,2} - \beta'_{k,2})\theta^2_{z,k} + \beta'_{k,3}\theta^3_{z,k}. \quad (3.3)$$

From here, Step 5b in Algorithm 3 can be performed to find the final approximations by using the expression in (3.3), substituting into (3.2), and performing an inverse transformation.

As stated previously, we use cubic corrections, mainly to account for any skewness. These approximations almost always outperformed the quadratic approximations, and were much more stable than using the LDS–StM method. In practice, we never used polynomial corrections that were of higher degree than the cubic, with the lone exception being to approximate multimodal densities. We discuss this further in the next section.

4   Applications and Results

We have proposed modifications to the LDS–StM method, first by fitting a quadratic as our approximation to the transformed hyperparameters, then by updating the approximation via analysing residuals and correcting the quadratic approximation with a higher degree polynomial (typically a cubic) term. We apply the LDS–QA and LDS–CX method to two examples. 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 (http://www.r-inla.org) and [22]. Details and links to the data and materials for the Georgia example can be found in [2].

4.1   Child Undernutririon in Zambia

The Zambia dataset was first introduced by [13]. There, the authors used spatial factors to analyse undernutrition among children in the 57 districts (dist) 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

$$Z_i = \frac{AI_i - MAI}{\sigma}, i = 1 \dots, 57$$

where AI_i is the ith 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_1.

Several factors are considered such as age (age), body mass index (bmi) of the child's mother, and several categorical variables including gender, education, mothers employment status and locality. This dataset has been used in [14] 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 [13]

$$\boldsymbol{\eta} = \mu + \boldsymbol{z}_i^T \boldsymbol{\gamma} + bmi_i \times g_2(dist_i) + g_3(age_i) + g_4(dist_i) + g_5(dist_i). \quad (4.1)$$

Here, μ is the overall mean, $\boldsymbol{z}_i^T \boldsymbol{\gamma}$ represent the several categorical covariates $\boldsymbol{z}$ as having a linear effect. Also, $g_5(dist_i)$ is the spatially unstructured component that is i.i.d Gaussian distributed with mean 0 and unknown precision τ_5, and $g_4(dist_i)$ is the spatially structured component which varies smoothly from district to district. This is modelled as an intrinsic Gauss Markov random field (IGMRF) – a conditional autoregressive (CAR) prior [1] – with unknown precision τ_4. 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 $g_2(\cdot)$. Both of these components have unknown precision τ_2 for *bmi* and τ_3 for *age*.

We will denote the hyperparameters in (4.1) as $\boldsymbol{\tau} = \{\tau_1, \tau_2, \tau_3, \tau_4, \tau_5\}$. We assign vague Gamma priors for each element in $\boldsymbol{\tau}$. 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 3 and Algorithm 4. The reparameterisation INLA uses for the precision hyperparameters (as described in Section 3.2) is a log transformation. Let $\boldsymbol{\theta} = \{\log(\tau_1), \log(\tau_2), \log(\tau_3), \log(\tau_4), \log(\tau_5)\}$. To simplify the notation, we will refer to the components of $\boldsymbol{\theta}$ as θ_k for $k = 1, \ldots, 5$.

For the following results, to get a very precise approximation of the true posterior marginals, we compute the hyperparameters using a customised dense grid in INLA, with a total of around 60000 points. This gives a very good approximation to the true posterior, and we will consider these approximations as the true marginals for the purposes of comparison.

4.1.1 Initial Results

We apply the steps in the LDS–CX algorithm 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 $\mathcal{K}_{512,5,19}^*$. Note that 512 points is less than 116 times the number of points used by INLA's dense grid. For the purposes of this exercise, we use the approximations of the mode and Hessian found by INLA to generate the integration region (± 3

Figure 3: Process of fitting polynomials for the approximations of hyperparameters of the Zambia model. Top row: Fitting a quadratic polynomial (dashed line) and cubic correction (dashed-dotted line) through the pointwise means. Middle row: Approximation of $\boldsymbol{\theta}$ by transforming the polynomials out of the log-scale. The INLA dense grid approximation (bold line) is compared with the initial approximation (dashed line) and the cubic correction (dashed-dotted line). Bottom row: Marginal posterior approximations of $\boldsymbol{\tau}$, showing the cubic correction (dashed-dotted line) providing a more accurate approximation than the initial quadratic approximation (dashed line).

standard deviations from the mode on each $\boldsymbol{\theta}$-axis). We also use INLA to manually compute the function evaluations for each point in $\mathcal{K}^*_{512,5,19}$. After projecting function evaluations onto each axis, we make 15 equally spaced partitions and find the pointwise means. The log pointwise means and the initial quadratic polynomial fit through those points are shown in Figure 3 (top row) with the points and bold line respectively.

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 3 as the dashed-dotted line. The initial fit for θ_1 and θ_4 are very good, and as such the cubic correction does not update them. There is a slight skew to θ_2 and θ_3 which cannot be captured by the initial fit. However, the cubic correction captures this skew well. The cubic correction also updates θ_5 slightly, with a small shift in location and skew. The transformations from the approximations in the $\log(\boldsymbol{\theta})$-scale to $\boldsymbol{\theta}$ scale shows how the initial approximation has shifted towards INLA's dense grid approximation in Figure 3 (middle row). It especially highlights how the cubic correction has approximated the two more heavily skewed posterior marginals θ_2 and θ_3. For completeness, we give the results of the inverse transformation from

16

$\boldsymbol{\theta}$ to $\boldsymbol{\tau}$ in Figure 3 (bottom row)).

Figure 4: Results of INLA's low density grid. Top row: The grid structures in the dimensions θ_1 and θ_2 for INLA's low density grid and INLA's high density grid. Bottom row: Comparisons with INLA's low density grid and the cubic correction. The grid approximation (dotted line) is poor, despite having roughly five time as many points as the cubic correction (dashed-dotted line).

4.1.2 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 used INLA to generate a grid that has 2655 points, around five times more than that of our Korobov lattice, and far less dense than that of our earlier grid. Figure 4 (top row) shows both the low density grid, and high density grid in two dimensions (θ_1 and θ_2). Whilst this might not look like much difference, the number of points increases exponentially with dimension. The resulting approximations shown in Figure 4 (bottom row) show that the less dense grid failed to give appropriate approximations for all posterior marginals and is outperformed by the cubic correction.

4.1.3 Comparisons with NIFA

NIFA is the current default setting in INLA for the estimation of hyperparameters. Though very fast, there is some room for improvement with respect to accuracy due to the assumed Gaussian structure described in (2.9). We compare the cubic correction with NIFA in Figure 5, and give Kullback-Leibler divergence [15] and Hellinger distance [20] results in Table

1.

Figure 5: Zambia: Comparisons between INLA's NIFA (dotted line) and the cubic correction (dashed-dotted line). They are both accurate, though the cubic correction approximates slightly better for all marginals.

The results in Figure 5 show that there is very little difference between the NIFA and the cubic correction. However, the Kullback-Leibler divergence and Hellinger distances in Table 1 shows that for all marginals, the cubic correction did approximate better than NIFA. However, in terms of computational efficiency (points used), NIFA bypasses the need for points altogether, thus approximates much faster than the cubic correction. In our second example, we present why the assumed structure of the NIFA estimate could be a problem, and how a polynomial correction can overcome this problem which can lead to more accurate approximations.

Table 1: 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.

ZAMBIA	K-L.Div		H.Dist	
Parameter	NIFA	LDS–C3	NIFA	LDS–C3
θ_1	0.00501	0.00329	0.03961	0.03233
θ_2	0.01194	0.00495	0.05664	0.04088
θ_3	0.00329	0.00290	0.03058	0.02964
θ_4	0.01199	0.00248	0.05797	0.02655
θ_5	0.00787	0.00533	0.04953	0.03967

4.2 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, and used to perform spatio-temporal disease mapping. This particular example has been used by [2] to illustrate a spatio-temporal Poisson nonparametric approach. This includes interactions between different spatial and temporal components.

Figure 6: 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.

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

$$\eta_{i,t} = \mu + g_1(county_i) + g_2(county_i) + g_3(year_t) + g_4(year_t) + g_5(area_i \times year_t),$$

where $i = 1, \ldots, 159$ and $t = 1, \ldots, 11$. We have μ as the overall mean, $g_1(county_i)$ as the spatially structured component, modelled as a conditional autoregressive prior with unknown precision τ_1, and $g_2(county_i)$ as the spatially unstructured component modelled as i.i.d Gaussian with mean 0 and unknown precision τ_2. Our time components are $g_3(year_t)$, which is modelled as a random walk of order two [21] with unknown precision τ_3, and $g_4(year_t)$ is the unstructured time effect modelled as i.i.d Gaussian with mean 0 and precision τ_4. We also have an interaction term $g_5(area_i \times year_t)$. 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_5. Thus we have five hyperparameters $\boldsymbol{\tau} = \{\tau_1, \tau_2, \tau_3, \tau_4, \tau_5\}$. Similar to the previous example, we estimate the log of the precisions, and

denote them θ_k for $k = 1, \ldots, 5$.

Again, we use a customised, dense grid in INLA (over 100000 points) to compute approximate posteriors that are very accurate. For the purpose of comparison, we will consider these approximations from the dense grid as the true marginals. Figure 6 shows the approximations, and captures a multimodal posterior for τ_2. We present the cubic correction first, before looking at how we can use higher degree polynomials to approximate multimodal posteriors.

Figure 7: Georgia: Initial Approximation (dashed line) and cubic corrections (dashed-dotted line) for $\boldsymbol{\theta}$, and comparisons with INLA's dense grid (bold line). The cubic correction was needed as the initial fits were not very good. However, we cannot approximate θ_2 well, due to the multimodal nature of the hyperparameter.

4.2.1 Initial Results

We take the same approach as we did in the Zambia example by generating an embedded Korobov Lattice $\mathcal{K}^*_{512,5,19}$, computing the function evaluations and partitioning making 15 equally spaced partitions for each axis. We follow the processes outlined in the LDS–QA and LDS–C3, 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 approximations for the components of $\boldsymbol{\theta}$ in Figure 7. The initial approximations were not good as some of the components of $\boldsymbol{\theta}$ were quite skewed. However, we see that the cubic correction was able to improve the approximations. The approximation of θ_2 was not accurate due to it being multimodal in shape.

4.2.2 Multimodal Hyperparameters - Polynomial Correction

Up until now we have chosen 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 degree polynomial correction. Since we fit each hyperparameter independently of the other, this is easily done. We focus solely on the marginal θ_2 and fit both a quartic and quintic correction.

Figure 8: Georgia: The process of approximating τ_2 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 approximation (dashed line), and the polynomial update (dashed-dotted line), whilst the second and third columns are approximating the θ_2 and τ_2 respectively. Going to a higher degree polynomial worked well for approximating the multimodal density.

Figure 8 shows the process of fitting the marginal τ_2 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 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 an accurate approximation.

Figure 9: Georgia: Comparisons between NIFA (dotted line) and polynomial correction (dashed-dotted line). Cubic corrections were used for all hyperparameters, except for θ_2 which has a quintic correction).

4.2.3 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 unimodal approximations. We give visual comparisons in Figure 9, and Kullback-Leibler divergence and Hellinger distances between INLA's dense grid (regarded as the true density) and the approximations in Table 2.

The polynomial correction we use is cubic for all marginals, except for θ_2, for which we use a quintic correction. As expected, for θ_2, the NIFA approximated this with a unimodal density. The approximation for θ_1 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_4.

5   Discussion and Further Work

Because of its fully projection regular property, the number of LDS points needed to integrate does not increase rapidly with the number of dimensions. This makes LDS a promising alternative to grid point sets for marginalisation. The LDS–StM method was presented in [12] to perform marginalisation of joint densities using LDS point sets. Approximations based on the

22

Table 2: 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_4 according to both the Kullback-Leibler divergence and Hellinger distances.

GEORGIA	K-L.Div		H.Dist	
Parameter	NIFA	LDS–CX	NIFA	LDS–CX
θ_1	0.30047	0.00923	0.17790	0.04445
θ_2	0.25381	0.00601	0.22173	0.04080
θ_3	0.01446	0.00250	0.05836	0.02597
θ_4	0.00459	0.00671	0.03598	0.04059
θ_5	0.02541	0.01550	0.06262	0.05712

LDS–StM method converge to the true marginal, and were shown to be more computationally efficient than existing grid–based methods of marginalisation. The motivation for this work was to build upon the LDS–StM method so it can be implemented within INLA to estimate model hyperparameters. To do this, we focussed on improving the interpolation process and simplifying the polynomials used. An initial quadratic/Gaussian approximation is fit to the pointwise means of orthogonally projected function evaluations (LDS–QA method), and another polynomial is used to correct the initial approximation (LDS–CX method). In practice, we typically use a cubic correction polynomial (LDS–C3), but for more complicated marginal shapes, a higher degree polynomial can be used.

The results for the LDS–CX method are very promising. First, the approximations are far more stable than LDS–StM methods, since we do not need to use high degree polynomials in the interpolation process, thus eliminating ill-conditioned matrices and Runge's phenomenon. This makes the method far more practical than LDS–StM, and useable within INLA. Second, the results in Section 4 show that the approximations using LDS–CX outperform INLA's grid by several orders of magnitude, with respect to computational efficiency. In higher dimension, we can obtain as accurate approximations than INLA's grids with far less points. Also, for any given number of points, we can obtain better approximations by using LDS and our algorithms within INLA, than using INLA's grid. Lastly, the LDS–CX method has the potential for estimating multimodal marginals by increasing the degree of the correction polynomial. This is not possible with INLA's NIFA approximation as it assumes the marginal posterior has a unimodal, "half" Gaussian shape, with different standard deviations either side of the mode (as shown in Equation (2.9)). Further investigation is needed to de-

velop a diagnostic which allows the user to detect multimodal marginal hyperparameters, and determines which correction polynomial to use.

This paper was motivated by the efficient approximation of hyperparameters using LDS. However, there is also the potential for computing efficient approximations for the latent parameters too. To estimate latent parameters of an LGM, INLA re-uses the points generated via grid (or otherwise) in the numerical integration process. Given that the dimension of latent fields are typically large, this will require a lot of computing power if we require many points to estimate the hyperparameters. Recall from Section 2.1, that an extensible lattice can be generated to give the user the ability to add more points without discarding all the old points. We can also take points away in a similar way without affecting the low discrepancy structure of the point set (see Figure 1). For example, let $\mathcal{K}^*_{2^x,s,\alpha}$ be an extensible Korobov lattice with 2^x points in s dimensions. Expressing the pointset as a matrix form, keeping the first row and taking out every second row after will give $\mathcal{K}^*_{2^{x-1},s,\alpha}$, and keeping the first row and keeping every fourth row will give $\mathcal{K}^*_{2^{x-2},s,\alpha}$, and so on. This subset of points can be used to approximate latent posterior marginals, as described in (2.8). Recent developments by the INLA team allow any user-defined pointset for the estimation of latent parameters. Further work is needed here to test the accuracy and computational efficiency of latent parameter estimates using LDS and extensible Korobov lattices.

Appendix - Orthogonal Projections

Let $f(\boldsymbol{x})$ be an s dimensional function that is evaluated at N unique points $\boldsymbol{x}_N = \boldsymbol{x}_{(1)}, \ldots, \boldsymbol{x}_{(N)}$, where each $\boldsymbol{x}_{(i)} \in [\boldsymbol{a}, \boldsymbol{b})$, $i = 1, \ldots, N$, is an $s - tuple$ $\boldsymbol{x}_{(i)} = (x_{1,i}, \ldots, x_{s,i})$. These points along with the function evaluations, that is $(\boldsymbol{x}_{(i)}, f(\boldsymbol{x}_{(i)}))$ are $(s+1) - tuples$ that can be expressed in matrix form as

$$
\boldsymbol{\Psi} = \begin{bmatrix} x_{1,1} & \ldots & x_{s,1} & f(\boldsymbol{x}_{(1)}) \\ x_{1,2} & \ldots & x_{s,2} & f(\boldsymbol{x}_{(2)}) \\ \vdots & \ddots & \vdots & \vdots \\ x_{1,N} & \ldots & x_{s,N} & f(\boldsymbol{x}_{(N)}) \end{bmatrix}
$$

To estimate the kth marginal $f_k(x_k)$, we orthogonally project $(\boldsymbol{x}_{(i)}, f(\boldsymbol{x}_{(i)}))$ for $i = 1, \ldots, N$ on the kth marginal axis to obtain $\psi_k = P_k \boldsymbol{\Psi}$,

$$
\psi_k = \begin{bmatrix} x_{k,1} & f(\boldsymbol{x}_{(1)}) \\ x_{k,2} & f(\boldsymbol{x}_{(2)}) \\ \vdots & \vdots \\ x_{k,N} & f(\boldsymbol{x}_{(N)}) \end{bmatrix},
$$

where $P_k = A_k (A_k^\top A_k)^{-1} A_k^\top$ is a projection matrix and A_k has size $(s+1) \times 2$, and is a unit basis vector for $\mathbb{R}^2$ with the kth entry in the first column and the $(s+1)^{th}$ entry in the second column as one, all the remaining entries are zeros.

For example, if $s = 3$ and $k = 2$ then,

$$
\boldsymbol{\Psi} = \begin{bmatrix} x_{1,1} & x_{2,1} & x_{3,1} & f(\boldsymbol{x}_{(1)}) \\ x_{2,1} & x_{2,2} & x_{3,2} & f(\boldsymbol{x}_{(2)}) \\ \vdots & \ddots & \vdots & \vdots \\ x_{1,N} & x_{2,N} & x_{3,N} & f(\boldsymbol{x}_{(N)}) \end{bmatrix}, \ A_2 = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix}, \ P_2 = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}
$$

and

$$
P_2 \boldsymbol{\Psi} = \begin{bmatrix} 0 & x_{2,1} & 0 & f(\boldsymbol{x}_{(1)}) \\ 0 & x_{2,2} & 0 & f(\boldsymbol{x}_{(2)}) \\ \vdots & \vdots & \vdots & \vdots \\ 0 & x_{2,N} & 0 & f(\boldsymbol{x}_{(N)}) \end{bmatrix}, \ \text{ignoring rows with zeros} \ \begin{bmatrix} x_{2,1} & f(\boldsymbol{x}_{(1)}) \\ x_{2,2} & f(\boldsymbol{x}_{(2)}) \\ \vdots & \vdots \\ x_{2,N} & f(\boldsymbol{x}_{(N)}) \end{bmatrix} = \psi_2.
$$

References

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

[2] BLANGIARDO, M. and CAMELETTI, M. (2015). *Spatial and Spatio-Temporal Bayesian Models with R-INLA*. Wiley, Chichester.

[3] BOX, G.E.P. and WILSON, K.B. (1951). On the experimental attainment of optimal conditions (with discussion). *Journal of Royal Statistical Society- Series B*, **13**, 1, 1–45.

[4] BROWN, P.T. (2018). *Computational Bayesian Inference Using Low Discrepancy Sequences*. Ph.D Thesis, University of Waikato, NZ.

[5] DICK, J. and PILLICHSHAMMER (2010). *Digital Nets and Sequences*. University Press, Cambridge.

[6] DISNEY, S. and SLOAN, I. (1992). Lattice integration rules of maximal rank found by copying rank–1 rules. *SIAM Journal of Numerical Analysis*, **29**, 2, 566–577.

[7] EPPERSON, J.F. (1987). On the Runge phenomenon. *The American Mathematical Monthly*, **95**, 14, 329–341.

[8] FAHRMEIR, L. and TUTZ, G. (1994). *Multivariate Statistical Modelling Based on Generalized Linear Models*. Springer, Berlin.

[9] HEWITT, J. and HOETING, J.A. (2019). Approximate Bayesian inference via sparse grid quadrature evaluation. *ArXiv 1904.07270*.

[10] HICKERNELL, F.J. and NIEDERREITER, H. (2003). The existence of good rank–1 lattices. *Journal of Complexity*, **19**, 3, 286–300.

[11] JOSHI, C. (2018). *A New Method to Implement Bayesian Inference on Stochastic Differential Equation Models*. Ph.D Thesis, Trinity College, Dublin.

[12] JOSHI, C. and BROWN, P.T. and JOE, S. (2019). On approximating the shape of one dimensional functions. *ArXiv 1911.03045*.

[13] KANDALA, N.B. and LANG, S and KLASEN, S. and FAHRMEIR, L. (2001). Semi-parametric alalysis of the socio-demographic and spatial determinants of undernutrition in two African countries. *Research in Official Statistics*, **1**, 81–100.

[14] KNEIB, T. and LANG, S and BREZGER, A. (2004). *Bayesian semi-parametric regression based on MCMC techniques – a tutorial.* Technical Report – University of Munich.

[15] KULLBACK, S. and LEIBLER, A. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, **22**, 1, 79–86.

[16] L'ECUYER, P. and LEMIEUX, C. (2000). Variance reduction via lattice rules. *Management Science*, **46**, 9, 1214–1235.

[17] L'ECUYER, P. and MUNGER, D. (2016). A general software tool for constructing rank–1 lattice rules. *ACM Transactions*, **42**, 2, Article 15.

[18] LEMIEUX, C. (2009). *Monte Carlo and Quasi–Monte Carlo Sampling.* Springer, New York.

[19] LEOBACHER, G. and PILLICHSHAMMER (2014). *Introduction to Quasi–Monte Carlo Integration and Applications.* Springer, Basel.

[20] LIESE, F. and MIESCKE, K.J. (2008). *Statistical Decision Theory: Estimation, Testing, and Selection.* Springer, New York.

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

[22] MARTINO, S. and RUE, H. (2008). Case studies in Bayesian computation using INLA. In P. Mantovan and P. Secchi, editors, *Complex Data Modelling and Computationally Intensive Statistical Methods.* Springer, Milan.

[23] MARTINS, T. and SIMPSON, D. and LINDGREN, F. and RUE, H. (2013). Bayesian computing with INLA: new features. *Computational Statistics and Data Analysis*, **67**, 1, 68–83.

[24] NIEDERREITER, H. (1992). *Random Number Generation and Quasi–Monte Carlo Methods.* Volume 63 of SIAM CMBS–NSF Regional Conference Series in Applied Mathematics, Philadelphia.

[25] NOVAK, E. and RITTER, K. (1996). High dimensional integration of smooth functions over cubes. *Numerische Mathematik*, **75**, 1, 79–97.

[26] ORMEROD, J.T. (2011). Grid–based variational approximations. *Computational Statistics and Data Analysis*, **55**, 1, 45–56.

[27] RUE, H. and HELD, L. (2005). *Gaussian Markov Random Fields: Theory and Applications.* Chapman and Hall, London.

[28] Rue, H. and Martino, S. and Chopin, N. (2009). Approximate Bayesian inference for latent Gaussian models by using integrated nested Laplace approximations. *Journal of Royal Statistical Society- Series B*, **71**, 2, 319–392.

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

[30] Sanchez, S.M. and Sanchez, P.J. (2005). Very large fractional factorials and central composite designs. *ACM Transactions in Modelling and Computational Simulations*, **15**, 4, 362–377.

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

[32] Tierney, L. and Kadane, J.B. (1986). Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, **81**, 1, 82–86.

[33] Wilson, S.P. and Yoon, J. (2010). Bayesian ICA–based source separation of cosmic microwave background by a discrete functional approximation. *ArXiv 1011.4018*.