Journal of Statistical Software

January 2015, Volume 63, Issue 13. http://www.jstatsoft.org/

spBayes for Large Univariate and Multivariate Point-Referenced Spatio-Temporal Data Models

Andrew O. Finley
Michigan State University

Sudipto Banerjee
University of California, Los Angeles

Alan E. Gelfand
Duke University

Abstract

In this paper we detail the reformulation and rewrite of core functions in the **spBayes** R package. These efforts have focused on improving computational efficiency, flexibility, and usability for point-referenced data models. Attention is given to algorithm and computing developments that result in improved sampler convergence rate and efficiency by reducing parameter space; decreased sampler run-time by avoiding expensive matrix computations, and; increased scalability to large datasets by implementing a class of *predictive process* models that attempt to overcome computational hurdles by representing spatial processes in terms of lower-dimensional realizations. Beyond these general computational improvements for existing model functions, we detail new functions for modeling data indexed in both space and time. These new functions implement a class of dynamic spatio-temporal models for settings where space is viewed as continuous and time is taken as discrete.

Keywords: spatial, temporal, multivariate, Gaussian predictive process, Markov chain Monte Carlo.

1. Introduction

The scientific community is moving into an era where open-access data-rich environments provide extraordinary opportunities to understand the spatial and temporal complexity of processes at broad scales. Unprecedented access to spatial data is a result of investments to collect data for regulatory, monitoring, and resource management objectives, and technological advances in spatially-enabled sensor networks along with geospatial information storage, analysis, and distribution systems. These data sources are increasingly diverse and specialized, e.g., computer model outputs, monitoring station instruments, remotely located sensors, and georeferenced field measurements. Across scientific fields, researchers face the challenge of coupling these data with imperfect models to better understand variability in their system

of interest. The inference garnered through these analyses often supports decisions with important economic, environmental, and public health implications; therefore, it is critical to correctly estimate inferential uncertainty. However, developing modeling frameworks capable of accounting for various sources of uncertainty is not a trivial task – massive datasets from multiple sources with complex spatial dependence structures only serve to aggravate the challenges.

Proliferation of spatial data has spurred considerable development in statistical modeling; see, for example, the books by Cressie (1993), Chilès and Delfiner (2012), Møller and Waagepetersen (2003), Schabenberger and Gotway (2004), Wackernagel (2003), Diggle and Ribeiro (2007) and Cressie and Wikle (2011) for a variety of methods and applications. The statistical literature acknowledges that spatial and temporal associations are captured most effectively using models that build dependencies in different stages or hierarchies. Hierarchical models are especially advantageous with datasets having several lurking sources of uncertainty and dependence, where they can estimate much richer models with less stringent assumptions than traditional modeling paradigms. These models often follow the Bayesian framework of statistical inference (see, e.g., Carlin and Louis 2011; Gelman, Carlin, Stern, and Rubin 2004), where analysis uses sampling from the posterior distributions of model parameters.

Computational advances with regard to Markov chain Monte Carlo (MCMC) methods have contributed enormously to the popularity of hierarchical models in a wide array of disciplines (e.g., Gilks, Richardson, and Spiegelhalter 1996; Robert and Casella 2004), and spatial modeling is no exception (see, e.g., Banerjee, Carlin, and Gelfand 2004). In the realm of spatial statistics, hierarchical models have been widely applied to analyze both areally referenced as well as point-referenced or geostatistical data. For the former, a class of models known as conditionally autoregressive (CAR) models have become very popular as they are easily implemented using MCMC methods such as the Gibbs sampler. In fact, these models are somewhat naturally suited for the Gibbs sampler which draws samples from conditional distributions that are fully specified by the CAR models. Their popularity has increased in no small measure due to their automated implementation in the **OpenBUGS** software package (Thomas, O Hara, Ligges, and Sturtz 2006) which offers a flexible and user-friendly interface to construct multilevel models that are implemented using a Gibbs sampler. This is performed by identifying a multilevel model with a directed acyclic graph (DAG) whose nodes form the different components of the model and allow the language to identify the full conditional distributions that need to be updated. **OpenBUGS** is an offshoot of the BUGS (Bayesian inference Using Gibbs Sampling) project and the successor of the **WinBUGS** software (Lunn, Spiegelhalter, Thomas, and Best 2009).

From an automated implementation perspective, the challenges are somewhat greater for point-referenced models. First, expensive matrix computations are required that can become prohibitive with large datasets. Second, routines to fit unmarginalized models are less suited for direct updating using a Gibbs sampler in the BUGS paradigm and results in slower convergence of the chains. Third, investigators often encounter multivariate spatial datasets with several spatially dependent outcomes, whose analysis requires multivariate spatial models that involve matrix computations that are poorly implemented in BUGS. These issues have, however, started to wane with the delivery of relatively simpler R (R Core Team 2014) packages via the Comprehensive R Archive Network (CRAN, http://CRAN.R-project.org/) that help automate Bayesian methods for point-referenced data and diagnose convergence. The *Analysis of Spatial Data* (Bivand 2014) and *Handling and Analyzing Spatio-Temporal*

Data (Pebesma 2014) CRAN task views provide a convenient way to identify packages that offer functions for modeling such data. These packages are generally listed under the *Geostatisics* section in the task views. Here, those packages that fit Bayesian models include **geoR** (Ribeiro and Diggle 2001), **geoRglm** (Christensen and Ribeiro 2002), **spTimer** (Bakar and Sahu 2015), **spBayes** (Finley and Banerjee 2013), **spate** (Sigrist, Kuensch, and Stahel 2015), and **ramps** (Smith, Yan, and Cowles 2008). In terms of functionality, **spBayes** offers users a suite of Bayesian hierarchical models for Gaussian and non-Gaussian univariate and multivariate spatial data as well as dynamic Bayesian spatial-temporal models.

Our initial development of **spBayes** (Finley, Banerjee, and Carlin 2007) provided functions for modeling Gaussian and non-Gaussian univariate and multivariate point-referenced data. These hierarchical Bayesian spatial process models, implemented through MCMC methods, offered increased flexibility to fit models that would be infeasible with classical methods within inappropriate asymptotic paradigms. However, with this increased flexibility comes substantial computational demands. Estimating these models involves expensive matrix decompositions whose computational complexity increases in cubic order with the number of spatial locations, rendering such models infeasible for large spatial datasets. Through **spBayes** version 0.2-4, released on CRAN on 2012-04-24, very little attention was given to addressing these computational challenges. As a result, fitting models with more than a few hundred observations was very time consuming – on the order of hours to fit models with ∼1,000 locations.

spBayes version 0.3-7 (CRAN 2013-06-01) comprises a substantial reformulation and rewrite of core functions for model fitting, with a focus on improving computational efficiency, flexibility, and usability. Among other improvements, this and subsequent versions offer: *i*) improved sampler convergence rate and efficiency by reducing parameter space; *ii*) decreased sampler run-time by avoiding expensive matrix computations, and; *iii*) increased scalability to large datasets by implementing a class of *predictive process* models that attempt to overcome computational hurdles by representing spatial processes in terms of lower-dimensional realizations. Beyond these general computational improvements for existing models, new functions were added to model data indexed in both space and time. These functions implement a class of dynamic spatio-temporal models for settings where space is viewed as continuous and time is taken as discrete. The subsequent sections highlight the fundamentals of models now implemented in **spBayes**.

2. Bayesian Gaussian spatial regression models

Finley *et al.* (2007) outline the first version of **spBayes** as an R package for estimating Bayesian spatial regression models for point-referenced outcomes arising from Gaussian, binomial or Poisson distributions. For the Gaussian case, the recent version of **spBayes** offers several Bayesian spatial models emerging from the hierarchical linear mixed model framework

$$p(\boldsymbol{\theta}) \times N(\boldsymbol{\beta} \,|\, \boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) \times N(\boldsymbol{\alpha} \,|\, \mathbf{0}, \boldsymbol{K}(\boldsymbol{\theta})) \times N(\boldsymbol{y} \,|\, \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{\alpha}, \boldsymbol{D}(\boldsymbol{\theta})) \,, \tag{1}$$

where $\boldsymbol{y}$ is an $n \times 1$ vector of possibly irregularly located observations, $\boldsymbol{X}$ is a known $n \times p$ matrix of regressors ($p < n$), $\boldsymbol{K}(\boldsymbol{\theta})$ and $\boldsymbol{D}(\boldsymbol{\theta})$ are families of $r \times r$ and $n \times n$ covariance matrices, respectively, and $\boldsymbol{Z}(\boldsymbol{\theta})$ is $n \times r$ with $r \leq n$, all indexed by a set of unknown process parameters $\boldsymbol{\theta}$. The $r \times 1$ random vector $\boldsymbol{\alpha} \sim N(\mathbf{0}, \boldsymbol{K}(\boldsymbol{\theta}))$ and the $p \times 1$ slope vector $\boldsymbol{\beta} \sim N(\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta)$, where

$\boldsymbol{\mu}_\beta$ and $\boldsymbol{\Sigma}_\beta$ are known. The hierarchy is completed by assuming $\boldsymbol{\theta} \sim p(\boldsymbol{\theta})$, a proper prior distribution. The Gaussian spatial models in **spBayes** emerge as special cases of (1), which we will see later. Bayesian inference is carried out by sampling from the posterior distribution of $\{\boldsymbol{\beta}, \boldsymbol{\alpha}, \boldsymbol{\theta}\}$, which is proportional to (1).

Below, we provide some details behind Bayesian inference for (1). This involves sampling the parameters $\boldsymbol{\theta}$, $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ from their marginal posterior distributions and carrying out subsequent predictions. Direct computations usually entail inverting and multiplying dense matrices and also computing determinants. In software development, care is needed to avoid redundant operations and ensure numerical stability. Therefore, in the subsequent sections we describe how we use Cholesky factorizations, solve triangular systems, and minimize expensive matrix operations (e.g., dense matrix multiplications) to perform all the computations.

2.1. Sampling the process parameters

Sampling from (1) employs MCMC methods, in particular Gibbs sampling and random walk Metropolis steps (e.g., Robert and Casella 2004). For faster convergence, we integrate out $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ from the model and first sample from $p(\boldsymbol{\theta} \,|\, \boldsymbol{y}) \propto p(\boldsymbol{\theta}) \times N(\boldsymbol{y} \,|\, \boldsymbol{X}\boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_{y\,|\,\theta})$, where $\boldsymbol{\Sigma}_{y\,|\,\theta} = \boldsymbol{X}\boldsymbol{\Sigma}_\beta\boldsymbol{X}^\top + \boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{K}(\boldsymbol{\theta})\boldsymbol{Z}(\boldsymbol{\theta})^\top + \boldsymbol{D}(\boldsymbol{\theta})$. This matrix needs to be constructed for every update of $\boldsymbol{\theta}$. Usually $\boldsymbol{D}(\boldsymbol{\theta})$ is diagonal and $\boldsymbol{X}\boldsymbol{\Sigma}_\beta\boldsymbol{X}^\top$ is fixed, so the computation involves the matrix $\boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{K}(\boldsymbol{\theta})\boldsymbol{Z}(\boldsymbol{\theta})^\top$. Assuming that $\boldsymbol{Z}(\boldsymbol{\theta})$ and $\boldsymbol{K}(\boldsymbol{\theta})$ are computationally inexpensive to construct for each $\boldsymbol{\theta}$, $\boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{K}(\boldsymbol{\theta})\boldsymbol{Z}(\boldsymbol{\theta})^\top$ requires rn^2 flops (floating point operations).

We adopt a random-walk Metropolis step with a multivariate normal proposal (same dimension as there are parameters in $\boldsymbol{\theta}$) after transforming parameters to have support over the entire real line. This involves evaluating

$$\log p(\boldsymbol{\theta} \,|\, \boldsymbol{y}) = \text{const} + \log p(\boldsymbol{\theta}) - \frac{1}{2}\log|\boldsymbol{\Sigma}_{y\,|\,\theta}| - \frac{1}{2}Q(\boldsymbol{\theta}) \,, \tag{2}$$

where $Q(\boldsymbol{\theta}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\beta)^\top\boldsymbol{\Sigma}_{y\,|\,\theta}^{-1}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\beta)$. Generally, we compute $\boldsymbol{L} = \texttt{chol}(\boldsymbol{\Sigma}_{y\,|\,\theta})$, where $\texttt{chol}(\boldsymbol{\Sigma}_{y\,|\,\theta})$ returns the lower-triangular Cholesky factor $\boldsymbol{L}$ of $\boldsymbol{\Sigma}_{y\,|\,\boldsymbol{\theta}}$. This involves $O(n^3/3)$ flops. Next, we obtain $\boldsymbol{u} = \texttt{trsolve}(\boldsymbol{L}, \boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\beta)$, which solves the triangular system $\boldsymbol{L}\boldsymbol{u} = \boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_\beta$. This involves $O(n^2)$ flops and $Q(\boldsymbol{\theta}) = \boldsymbol{u}^\top\boldsymbol{u}$ requires another $2n$ flops. The log-determinant in (2) is evaluated as $2\sum_{i=1}^n \log l_{ii}$, where l_{ii} are the diagonal entries in $\boldsymbol{L}$. Since $\boldsymbol{L}$ has already been obtained, the log-determinant requires another n steps. Therefore, the Cholesky factorization dominates the work and computing (2) is achieved in $O(n^3)$ flops.

If $\boldsymbol{\beta}$ is flat, i.e., $\boldsymbol{\Sigma}_\beta^{-1} = \boldsymbol{O}$, the analogue of distribution (2) is

$$\log p(\boldsymbol{\theta} \,|\, \boldsymbol{y}) = \text{constant} + \log p(\boldsymbol{\theta}) - \frac{1}{2}\log|\boldsymbol{X}^\top\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}\boldsymbol{X}| - \frac{1}{2}\log|\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}| - \frac{1}{2}Q(\boldsymbol{\theta}), \tag{3}$$

where $\boldsymbol{\Sigma}_{y\,|\,\beta,\theta} = \boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{K}(\boldsymbol{\theta})\boldsymbol{Z}(\boldsymbol{\theta})^\top + \boldsymbol{D}(\boldsymbol{\theta})$ and $Q(\boldsymbol{\theta}) = \boldsymbol{y}^\top\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}^{-1}\boldsymbol{y} - \boldsymbol{b}^\top(\boldsymbol{X}^\top\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}^{-1}\boldsymbol{X})^{-1}\boldsymbol{b}$ and $\boldsymbol{b} = \boldsymbol{X}^\top\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}^{-1}\boldsymbol{y}$. Computations proceed similar to the above. We first evaluate $\boldsymbol{L} = \texttt{chol}(\boldsymbol{\Sigma}_{y\,|\,\beta,\theta})$ and then obtain $[\boldsymbol{v} : \boldsymbol{U}] = \texttt{trsolve}(\boldsymbol{L}, [\boldsymbol{y} : \boldsymbol{X}])$, so $\boldsymbol{L}\boldsymbol{v} = \boldsymbol{y}$ and $\boldsymbol{L}\boldsymbol{U} = \boldsymbol{X}$. Next, we evaluate $\boldsymbol{W} = \texttt{chol}(\boldsymbol{U}^\top\boldsymbol{U})$, $\boldsymbol{b} = \boldsymbol{U}^\top\boldsymbol{v}$ and solve $\tilde{\boldsymbol{b}} = \texttt{trsolve}(\boldsymbol{W}, \boldsymbol{b})$. Finally, (3) is evaluated as

$$\log p(\boldsymbol{\theta}) - \sum_{i=1}^p \log w_{i,i} - \sum_{i=1}^n l_{i,i} - \frac{1}{2}(\boldsymbol{v}^\top\boldsymbol{v} - \tilde{\boldsymbol{b}}^\top\tilde{\boldsymbol{b}}),$$

where $w_{i,i}$'s and l_{ii}'s are the diagonal elements in $\boldsymbol{W}$ and $\boldsymbol{L}$ respectively. The number of flops is again of cubic order in n.

Importantly, our strategy above avoids computing inverses. We use Cholesky factorizations and solve only triangular systems. If n is not large, say $\sim 10^2$, this strategy is feasible. The use of efficient numerical linear algebra routines fetch substantial reduction in computing time (see Section 3). Our implementation employs matrix-vector multiplication and avoids dense matrix-matrix multiplications wherever possible. Multiplications involving diagonal matrices are programmed using closed form expressions and inverses are obtained by solving triangular linear systems after obtaining a Cholesky decomposition. However, when $n \sim 10^3$ or higher, the computation becomes too onerous for practical use and alternative updating strategies are required. We address this in Section 2.3

2.2. Sampling the slope and the random effects

Once we have obtained marginal posterior samples $\boldsymbol{\theta}$ from $p(\boldsymbol{\theta} \,|\, \boldsymbol{y})$, we can draw posterior samples of $\boldsymbol{\beta}$ and $\boldsymbol{\alpha}$ using *composition sampling*. Suppose $\{\boldsymbol{\theta}^{(1)}, \boldsymbol{\theta}^{(2)}, \ldots, \boldsymbol{\theta}^{(M)}\}$ are M samples from $p(\boldsymbol{\theta} \,|\, \boldsymbol{y})$. Drawing $\boldsymbol{\beta}^{(k)} \sim p(\boldsymbol{\beta} \,|\, \boldsymbol{\theta}^{(k)}, \boldsymbol{y})$ and $\boldsymbol{\alpha}^{(k)} \sim p(\boldsymbol{\alpha} \,|\, \boldsymbol{\theta}^{(k)}, \boldsymbol{y})$ for $k = 1, 2, \ldots M$ results in M samples from $p(\boldsymbol{\beta} \,|\, \boldsymbol{y})$ and $p(\boldsymbol{\alpha} \,|\, \boldsymbol{y})$ respectively. Only the samples of $\boldsymbol{\theta}$ obtained after convergence (i.e., *post burn-in*) of the MCMC algorithm need to be stored.

To elucidate further, note that $\boldsymbol{\beta} \,|\, \boldsymbol{\theta}, \boldsymbol{y} \sim N_p(\boldsymbol{Bb}, \boldsymbol{B})$ with mean $\boldsymbol{Bb}$ and variance-covariance matrix $\boldsymbol{B}$, where

$$\boldsymbol{b} = \boldsymbol{\Sigma}_{\beta}^{-1}\boldsymbol{\mu}_{\beta} + \boldsymbol{X}^{\top}\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}^{-1}\boldsymbol{y} \;\text{ and }\; \boldsymbol{B} = \left(\boldsymbol{\Sigma}_{\beta}^{-1} + \boldsymbol{X}^{\top}\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}^{-1}\boldsymbol{X}\right)^{-1} . \tag{4}$$

For each $k = 1, 2, \ldots, M$, we compute $\boldsymbol{B}$ and $\boldsymbol{b}$ at the current value $\boldsymbol{\theta}^{(k)}$ and draw $\boldsymbol{\beta}^{(k)} \sim N_p(\boldsymbol{Bb}, \boldsymbol{B})$. This is achieved by computing $\boldsymbol{b} = \boldsymbol{\Sigma}_{\beta}^{-1}\boldsymbol{\mu}_{\beta} + \boldsymbol{U}^{\top}\boldsymbol{v}$, where $\boldsymbol{L} = \texttt{chol}(\boldsymbol{\Sigma}_{y\,|\,\beta,\theta^{(k)}})$ and $[\boldsymbol{v} : \boldsymbol{U}] = \texttt{trsolve}(\boldsymbol{L}, [\boldsymbol{y} : \boldsymbol{X}])$. Next, we generate p independent standard normal variables, collect them into $\boldsymbol{z}$ and set

$$\boldsymbol{\beta}^{(k)} = \texttt{trsolve}\left(\boldsymbol{L}_B^{\top}, \texttt{trsolve}(\boldsymbol{L}_B, \boldsymbol{b})\right) + \texttt{trsolve}(\boldsymbol{L}_B^{\top}, \boldsymbol{z}) \,, \tag{5}$$

where $\boldsymbol{L}_B = \texttt{chol}\left(\boldsymbol{\Sigma}_{\beta}^{-1} + \boldsymbol{U}^{\top}\boldsymbol{U}\right)$. This completes the k-th iteration. After M iterations, we obtain $\{\boldsymbol{\beta}^{(1)}, \boldsymbol{\beta}^{(2)}, \ldots, \boldsymbol{\beta}^{(M)}\}$, which are samples from $p(\boldsymbol{\beta} \,|\, \boldsymbol{y})$.

Mapping point or interval estimates of spatial random effects is often helpful in identifying missing regressors and/or building a better understanding of model adequacy. $\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta} = \boldsymbol{X}\boldsymbol{\Sigma}_{\beta}\boldsymbol{X}^{\top} + \boldsymbol{D}(\boldsymbol{\theta})$ and note that $\boldsymbol{\alpha} \,|\, \boldsymbol{\theta}, \boldsymbol{y} \sim N(\boldsymbol{Bb}, \boldsymbol{B})$, where

$$\boldsymbol{b} = \boldsymbol{Z}(\boldsymbol{\theta})^{\top}\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta}^{-1}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_{\beta}) \;\text{ and }\; \boldsymbol{B} = \left(\boldsymbol{K}(\boldsymbol{\theta})^{-1} + \boldsymbol{Z}(\boldsymbol{\theta})^{\top}\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta}^{-1}\boldsymbol{Z}(\boldsymbol{\theta})\right)^{-1} . \tag{6}$$

The vector $\boldsymbol{b}$ here is computed analogously as for $\boldsymbol{\beta}$. For each $k = 1, 2, \ldots, M$ we now evaluate $\boldsymbol{L} = \texttt{chol}(\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta^{(k)}})$, $[\boldsymbol{v} : \boldsymbol{U}] = \texttt{trsolve}(\boldsymbol{L}, [\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\mu}_{\beta} : \boldsymbol{Z}(\boldsymbol{\theta}^{(k)})])$ and set $\boldsymbol{b} = \boldsymbol{U}(\boldsymbol{\theta}^{(k)})^{\top}\boldsymbol{v}$. For computing $\boldsymbol{B}$, one could proceed as for $\boldsymbol{\beta}$ but that would involve $\texttt{chol}(\boldsymbol{K}(\boldsymbol{\theta}))$, which may become numerically unstable for certain covariance functions (e.g., the Gaussian or the Matérn with large ν). For robust software performance we define $\boldsymbol{G}(\boldsymbol{\theta})^{-1} = \boldsymbol{Z}(\boldsymbol{\theta})'\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta}^{-1}\boldsymbol{Z}(\boldsymbol{\theta})$ and utilize the identity (Henderson and Searle 1981)

$$\left(\boldsymbol{K}(\boldsymbol{\theta})^{-1} + \boldsymbol{G}(\boldsymbol{\theta})^{-1}\right)^{-1} = \boldsymbol{G}(\boldsymbol{\theta}) - \boldsymbol{G}(\boldsymbol{\theta})\left(\boldsymbol{K}(\boldsymbol{\theta}) + \boldsymbol{G}(\boldsymbol{\theta})\right)^{-1}\boldsymbol{G}(\boldsymbol{\theta})$$

to devise a numerically stable algorithm. For each $k = 1, 2, \ldots, M$, we evaluate $\boldsymbol{L} = \texttt{chol}(\boldsymbol{K}(\boldsymbol{\theta}^{(k)}) + \boldsymbol{G}(\boldsymbol{\theta}^{(k)}))$, $\boldsymbol{W} = \texttt{trsolve}(\boldsymbol{L}, \boldsymbol{G}(\boldsymbol{\theta}^{(k)}))$ and $\boldsymbol{L}_B = \texttt{chol}(\boldsymbol{G}(\boldsymbol{\theta})^{(k)} - \boldsymbol{W}^\top \boldsymbol{W})$. If $\boldsymbol{z}$ is a $r \times 1$ vector of independent standard normal variables, then we set $\boldsymbol{\alpha}^{(k)} = \boldsymbol{L}_B \boldsymbol{L}_B^\top \boldsymbol{b} + \boldsymbol{L}_B \boldsymbol{z}$. The resulting $\{\boldsymbol{\alpha}^{(1)}, \boldsymbol{\alpha}^{(2)}, \ldots, \boldsymbol{\alpha}^{(M)}\}$ are samples from $p(\boldsymbol{\alpha} \,|\, \boldsymbol{y})$.

We remark that estimating the spatial effects involves Cholesky factorizations for $n \times n$ positive definite linear system. The above steps ensure numerical stability but they can become computationally prohibitive when n becomes large. While some savings accrue from executing the above steps only for the post *burn-in* samples, for n in the order of thousands we recommend the low rank spatial models offered by **spBayes** (see Sections 2.3 and 4.2).

2.3. The special case of low-rank models

The major computational load in estimating (1) arises from unavoidable Cholesky decompositions for dense $n \times n$ positive definite matrices. The required number of flops is of cubic order and must be executed in each iteration of the MCMC. For example, when a specific form of (1) is used to analyze a dataset comprising $n = 2,000$ locations and $p = 2$ predictors, each iteration requires $\sim$0.3 seconds of CPU time (see Section 4.2.1). Marginalization, as described in Section 2.1, typically require fewer iterations to converge. But even if $10,000$ iterations are required to deliver full inferential output, the associated CPU time is $\sim$50 minutes. Clearly, large spatial datasets demand specialized models.

One strategy is to specify $\boldsymbol{Z}(\boldsymbol{\theta})$ with $r \ll n$. Such models are known as *low-rank* models. Specific choices for $\boldsymbol{Z}(\boldsymbol{\theta})$ will be discussed later – **spBayes** models $\boldsymbol{Z}(\boldsymbol{\theta})$ using the predictive process (see Section 4.2). To understand how savings accrue in low-rank models, consider the marginal Gaussian likelihood obtained by integrating out $\boldsymbol{\alpha}$ from (1)

$$p(\boldsymbol{\theta}) \times N(\boldsymbol{\beta} \,|\, \boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) \times N(\boldsymbol{y} \,|\, \boldsymbol{X}\boldsymbol{\beta}, \boldsymbol{\Sigma}_{y \,|\, \beta, \theta}) \,,$$

where $\boldsymbol{\Sigma}_{y \,|\, \beta, \theta} = \boldsymbol{Z}(\boldsymbol{\theta}) \boldsymbol{K}(\boldsymbol{\theta}) \boldsymbol{Z}(\boldsymbol{\theta})^\top + \boldsymbol{D}(\boldsymbol{\theta})$. We could have integrated out $\boldsymbol{\beta}$ too, as in Section 2.1, but this does not result in an appreciable gain in computational efficiency. For the low-rank model, each iteration of the Gibbs sampler updates $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$ from their full conditional distributions.

The $\boldsymbol{\beta}$ is drawn from $N(\boldsymbol{B}\boldsymbol{b}, \boldsymbol{B})$, where $\boldsymbol{b}$ and $\boldsymbol{B}$ are as in (4). The strategy in Section 2.2 would be expensive for large n because computing $\boldsymbol{B}$, though itself $p \times p$, involves a Cholesky factorization of the $n \times n$ matrix $\boldsymbol{\Sigma}_{y \,|\, \beta, \theta}$ for every new update of $\boldsymbol{\theta}$. Instead, we utilize the Sherman-Woodbury-Morrison formula

$$\begin{aligned} \boldsymbol{\Sigma}_{y \,|\, \beta, \theta}^{-1} &= \boldsymbol{D}(\boldsymbol{\theta})^{-1} - \boldsymbol{D}(\boldsymbol{\theta})^{-1} \boldsymbol{Z}(\boldsymbol{\theta}) \left(\boldsymbol{K}(\boldsymbol{\theta})^{-1} + \boldsymbol{Z}(\boldsymbol{\theta})^\top \boldsymbol{D}(\boldsymbol{\theta})^{-1} \boldsymbol{Z}(\boldsymbol{\theta}) \right) \boldsymbol{Z}(\boldsymbol{\theta})^\top \boldsymbol{D}(\boldsymbol{\theta})^{-1} \\ &= \boldsymbol{D}(\boldsymbol{\theta})^{-1/2} \left(\boldsymbol{I} - \boldsymbol{H}^\top \boldsymbol{H} \right) \boldsymbol{D}(\boldsymbol{\theta})^{-1/2} \,, \end{aligned} \tag{7}$$

where $\boldsymbol{H} = \texttt{trsolve}(\boldsymbol{L}, \boldsymbol{W}^\top)$, $\boldsymbol{W} = \boldsymbol{D}(\boldsymbol{\theta})^{-1/2} \boldsymbol{Z}(\boldsymbol{\theta})$ and $\boldsymbol{L} = \texttt{chol}(\boldsymbol{K}(\boldsymbol{\theta})^{-1} + \boldsymbol{W}^\top \boldsymbol{W})$. Next, we compute $[\boldsymbol{v} : \boldsymbol{V}] = \boldsymbol{D}^{-1/2} [\boldsymbol{y} : \boldsymbol{X}]$, $\tilde{\boldsymbol{V}} = \boldsymbol{H}\boldsymbol{V}$ and set

$$\boldsymbol{b} = \boldsymbol{\Sigma}_\beta^{-1} \boldsymbol{\mu}_\beta + \boldsymbol{V}^\top \boldsymbol{y} - \tilde{\boldsymbol{V}}^\top \boldsymbol{H} \boldsymbol{v} \;\text{ and }\; \boldsymbol{L}_B = \texttt{chol}(\boldsymbol{\Sigma}_\beta^{-1} + \boldsymbol{V}^\top \boldsymbol{V} - \tilde{\boldsymbol{V}}^\top \tilde{\boldsymbol{V}}) \,. \tag{8}$$

We perform the above operations for each iteration in the Gibbs sampler, using the current update of $\boldsymbol{\theta}$, and sample the $\boldsymbol{\beta}$ as in (5).

We update process parameters $\boldsymbol{\theta}$ using a random-walk Metropolis step with target log-density

$$\log p(\boldsymbol{\theta} \,|\, \boldsymbol{y}) = \text{const.} + \log p(\boldsymbol{\theta}) - \frac{1}{2}\log|\boldsymbol{\Sigma}_{y\,|\,\beta,\theta}| - \frac{1}{2}Q(\boldsymbol{\theta}) \;, \tag{9}$$

where $Q(\boldsymbol{\theta}) = (\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})^\top \boldsymbol{\Sigma}^{-1}_{y\,|\,\beta,\theta}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$. Having obtained $\boldsymbol{H}$ as above, we evaluate $\boldsymbol{v} = \boldsymbol{D}(\boldsymbol{\theta})^{-1/2}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})$, $\boldsymbol{w} = \boldsymbol{H}\boldsymbol{v}$, $\boldsymbol{T} = \texttt{chol}(\boldsymbol{I}_r - \boldsymbol{H}\boldsymbol{H}^\top)$ and compute (9) as

$$\log p(\boldsymbol{\theta}) - \frac{1}{2}\sum_{i=1}^{n}\log d_{i,i}(\boldsymbol{\theta}) + \sum_{i=1}^{n^*}\log t_{i,i} - \frac{1}{2}(\boldsymbol{v}^\top\boldsymbol{v} - \boldsymbol{w}^\top\boldsymbol{w}) \;,$$

where $d_{ii}(\boldsymbol{\theta})$ and t_{ii} are the diagonal entries of $\boldsymbol{D}(\boldsymbol{\theta})$ and $\boldsymbol{T}$ respectively.

Once the Gibbs sampler has converged and we have obtained posterior samples for $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$, obtaining posterior samples for $\boldsymbol{\alpha}$ can be achieved following closely the description in Section 2.2. In fact, since the posterior samples of $\boldsymbol{\beta}$ are already available, we can draw $\boldsymbol{\alpha}$ from its full-conditional distribution, given *both* $\boldsymbol{\beta}$ and $\boldsymbol{\theta}$. This amounts to replacing $\boldsymbol{\mu}_\beta$ with $\boldsymbol{\beta}$ and $\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta}$ with $\boldsymbol{D}(\boldsymbol{\theta})$ in (6). The algorithm now proceeds exactly as in Section 2.2 and we achieve computational savings as $\boldsymbol{D}(\boldsymbol{\theta})$ is usually cheaper to handle than $\boldsymbol{\Sigma}_{y\,|\,\alpha,\theta}$.

2.4. Spatial predictions

To predict a random $t \times 1$ vector $\boldsymbol{y}_0$ associated with a $t \times p$ matrix of predictors, $\boldsymbol{X}_0$, we assume that

$$\begin{bmatrix}\boldsymbol{y}\\ \boldsymbol{y}_0\end{bmatrix}\Bigg|\boldsymbol{\beta},\boldsymbol{\theta} \sim N_{t+n}\left(\begin{bmatrix}\boldsymbol{X}\\ \boldsymbol{X}_0\end{bmatrix}\boldsymbol{\beta},\; \begin{bmatrix}\boldsymbol{C}_{11}(\boldsymbol{\theta}) & \boldsymbol{C}_{12}(\boldsymbol{\theta})\\ \boldsymbol{C}_{12}(\boldsymbol{\theta})^\top & \boldsymbol{C}_{22}(\boldsymbol{\theta})\end{bmatrix}\right) \;, \tag{10}$$

where $\boldsymbol{C}_{11}(\boldsymbol{\theta}) = \boldsymbol{\Sigma}_{y\,|\,\beta,\theta}$, $\boldsymbol{C}_{12}(\boldsymbol{\theta})$ is the $n \times t$ cross-covariance matrix between $\boldsymbol{y}$ and $\boldsymbol{y}_0$, and $\boldsymbol{C}_{22}(\boldsymbol{\theta})$ is the variance-covariance matrix for $\boldsymbol{y}_0$. How these are constructed is crucial for ensuring a legal probability distribution or, equivalently, a positive-definite variance-covariance matrix for $(\boldsymbol{y}^\top, \boldsymbol{y}_0^\top)^\top$ in (10). A legitimate joint distribution will supply a conditional distribution $p(\boldsymbol{y}_0 \,|\, \boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{\theta})$, which is normal with mean and variance

$$\begin{aligned}\boldsymbol{\mu}_p &= \boldsymbol{X}_0\boldsymbol{\beta} + \boldsymbol{C}_{12}(\boldsymbol{\theta})^\top\boldsymbol{C}_{11}(\boldsymbol{\theta})^{-1}(\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta})\\ \boldsymbol{\Sigma}_p &= \boldsymbol{C}_{22}(\boldsymbol{\theta}) - \boldsymbol{C}_{12}(\boldsymbol{\theta})^\top\boldsymbol{C}_{11}(\boldsymbol{\theta})^{-1}\boldsymbol{C}_{12}(\boldsymbol{\theta})\end{aligned} \tag{11}$$

Bayesian prediction proceeds by sampling from the posterior predictive distribution $p(\boldsymbol{y}_0 \,|\, \boldsymbol{y}) = \int p(\boldsymbol{y}_0 \,|\, \boldsymbol{y}, \boldsymbol{\beta}, \boldsymbol{\theta})p(\boldsymbol{\beta}, \boldsymbol{\theta} \,|\, \boldsymbol{y})d\boldsymbol{\beta}d\boldsymbol{\theta}$. For each posterior sample of $\{\boldsymbol{\beta}, \boldsymbol{\theta}\}$, we draw a corresponding $\boldsymbol{y}_0 \sim N(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$. This produces samples from the posterior predictive distribution.

Observe that the posterior predictive computations involve only the retained MCMC samples after convergence. Furthermore, most of the ingredients to compute $\boldsymbol{\mu}_p$ and $\boldsymbol{\Sigma}_p$ have already been performed while updating the model parameters. For any posterior sample $\{\boldsymbol{\beta}^{(k)}, \boldsymbol{\theta}^{(k)}\}$, we solve $[\boldsymbol{u} : \boldsymbol{V}] = \texttt{trsolve}(\boldsymbol{L}, [\boldsymbol{y} - \boldsymbol{X}\boldsymbol{\beta}^{(k)} : \boldsymbol{C}_{12}(\boldsymbol{\theta}^{(k)})])$, where $\boldsymbol{L} = \texttt{chol}(\boldsymbol{C}_{11}(\boldsymbol{\theta}^{(k)}))$. Next, we set $\boldsymbol{\mu}_p^{(k)} = \boldsymbol{X}_0\boldsymbol{\beta}^{(k)} + \boldsymbol{V}^\top\boldsymbol{u}$ and $\boldsymbol{\Sigma}_p^{(k)} = \boldsymbol{C}_{22}(\boldsymbol{\theta}^{(k)}) - \boldsymbol{V}^\top\boldsymbol{V}$ and draw $\boldsymbol{y}_0^{(k)} \sim N(\boldsymbol{\mu}_p^{(k)}, \boldsymbol{\Sigma}_p^{(k)})$.

Low-rank models, where $r \ll n$, are again cheaper here. The operations are dominated by the computation of $\boldsymbol{C}_{12}(\boldsymbol{\theta})^\top\boldsymbol{C}_{11}(\boldsymbol{\theta})^{-1}\boldsymbol{C}_{12}(\boldsymbol{\theta})$, which can be evaluated as $\boldsymbol{U}^\top\boldsymbol{U} - \boldsymbol{V}^\top\boldsymbol{V}$,

where $\boldsymbol{U} = \boldsymbol{D}(\boldsymbol{\theta})^{-1/2}\boldsymbol{C}_{12}(\boldsymbol{\theta})$, $\boldsymbol{V} = \boldsymbol{H}\boldsymbol{U}$ and $\boldsymbol{H}$ is as in (7). This avoids direct evaluation of $\boldsymbol{C}_{11}(\boldsymbol{\theta})^{-1}$ and avoids redundant matrix operations.

Updating $\boldsymbol{y}_0^{(k)}$'s requires Cholesky factorization of $\boldsymbol{\Sigma}_p$, which is $t \times t$ and can be expensive if t is large. In most practical settings, it is sufficient to take $t = 1$ and perform independent individual predictions. However, if the *joint* predictive distribution is sought, say when full inference is desired for a function of $\boldsymbol{y}_0$, then the predictive step is significantly cheaper if we use the posterior samples of $\boldsymbol{\alpha}$ as well. Now posterior predictive sampling amounts to drawing $\boldsymbol{y}_0^{(k)} \sim N(\boldsymbol{X}_0\boldsymbol{\beta}^{(k)} + \boldsymbol{Z}(\boldsymbol{\theta}^{(k)})\boldsymbol{\alpha}^{(k)}, \boldsymbol{D}(\boldsymbol{\theta}^{(k)}))$, which cheap because $\boldsymbol{D}(\boldsymbol{\theta})$ is usually diagonal. Low rank models are especially useful here as posterior sampling for $\boldsymbol{\alpha}$ is much cheaper with $r \ll n$.

3. Computing environment

The MCMC algorithms described in the preceding sections are implemented in **spBayes** functions. These functions are written in C++ and leverage R's *Foreign Language Interface* to call Fortran **BLAS** (Basic Linear Algebra Subprograms, see Blackford *et al.* 2001) and **LAPACK** (Linear Algebra Package, see Anderson *et al.* 1999) libraries for efficient matrix computations. Table 1 offers a list of key **BLAS** and **LAPACK** functions used to implement the MCMC samplers. Referring to Table 1 and following from Section 2.1, `chol` corresponds to `dpotrf` and `trsolve` can be either `dtrsv` or `dtrsm` depending on the form of the equation's right-hand side. As noted previously, we try and use dense matrix-matrix multiplication, i.e., calls to `dgemm`, sparingly due to its computational overhead. Often careful formulation of the problem can result in fewer calls to `dgemm` and other *expensive* **BLAS** level 3 and **LAPACK** functions.

A heavy reliance on **BLAS** and **LAPACK** functions for matrix operations allows us to leverage multi-processor/core machines via threaded implementations of **BLAS** and **LAPACK**, e.g., Intel's Math Kernel Library (**MKL**; Intel 2013). With the exception of `dtrsv`, all functions in Table 1 are threaded in Intel's **MKL**. Use of **MKL**, or similar threaded libraries, can dramatically reduce sampler run-times. For example, the illustrative analyses offered in subsequent sections were conducted using R, and hence **spBayes**, compiled with **MKL** on an Intel Ivy Bridge i7 quad-core processor with hyperthreading. The use of these parallel matrix operations results in a near linear speadup in the MCMC sampler's run-time with the number of CPUs – at least 4 CPUs were in use in each function call. The R Installation and Administration document details how to compile R against **MKL** and similar threaded libraries.

Function	Description
`dpotrf`	**LAPACK** routine to compute the Cholesky factorization of a real symmetric positive definite matrix.
`dtrsv`	Level 2 **BLAS** routine to solve the systems of equations $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, where $\boldsymbol{x}$ and $\boldsymbol{b}$ are vectors and $\boldsymbol{A}$ is a triangular matrix.
`dtrsm`	Level 3 **BLAS** routine to solve the matrix equations $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}$, where $\boldsymbol{X}$ and $\boldsymbol{B}$ are matrices and $\boldsymbol{A}$ is a triangular matrix.
`dgemv`	Level 2 **BLAS** matrix-vector multiplication.
`dgemm`	Level 3 **BLAS** matrix-matrix multiplication.

Table 1: Common **BLAS** and **LAPACK** functions used in **spBayes** function calls.

spBayes also depends on several R packages including: **coda** (Plummer, Best, Cowles, and Vines 2006) for casting the MCMC chain results as `coda` objects for easier posterior analysis; **abind** (Plate and Heiberger 2013) and **magic** (Hankin 2013) for forming multivariate matrices, and; **Formula** (Zeileis and Croissant 2010) for interpreting symbolic model formulas.

4. Models offered by spBayes

All the models offered by **spBayes** emerge as special instances of (1). The matrix $\boldsymbol{D}(\boldsymbol{\theta})$ is always taken to be diagonal or block-diagonal (for multivariate models). The spatial random effects $\boldsymbol{\alpha}$ are assumed to arise from a partial realization of a spatial process and the spatial covariance matrix $\boldsymbol{K}(\boldsymbol{\theta})$ is constructed from the covariance function specifying that spatial process. To be precise, if $\{w(\boldsymbol{s}) : \boldsymbol{s} \in \Re^d\}$ is a Gaussian spatial process with positive definite covariance function $C(\boldsymbol{s}, \boldsymbol{t}; \boldsymbol{\theta})$ (see, e.g., Bochner 1955) and if $\{\boldsymbol{s}_1, \boldsymbol{s}_2, \ldots, \boldsymbol{s}_r\}$ is a set of any r locations in $\mathcal{D}$, then $\boldsymbol{\alpha} = (w(\boldsymbol{s}_1), w(\boldsymbol{s}_2), \ldots, w(\boldsymbol{s}_r))^\top$ and $\boldsymbol{K}(\boldsymbol{\theta})$ is its $r \times r$ covariance matrix.

4.1. Full rank univariate Gaussian spatial regression

For Gaussian outcomes, geostatistical models customarily regress a spatially referenced dependent variable, say $y(\boldsymbol{s})$, on a $p \times 1$ vector of spatially referenced predictors $\boldsymbol{x}(\boldsymbol{s})$ (with an intercept) as

$$y(\boldsymbol{s}) = \boldsymbol{x}(\boldsymbol{s})^\top \boldsymbol{\beta} + w(\boldsymbol{s}) + \varepsilon(\boldsymbol{s}) \ , \tag{12}$$

where $\boldsymbol{s} \in \mathcal{D} \subseteq \Re^2$ is a location. The residual comprises a spatial process, $w(\boldsymbol{s})$, and an independent white-noise process, $\varepsilon(\boldsymbol{s})$, that captures measurement error or micro-scale variation. With any collection of n locations, say $\mathcal{S} = \{\boldsymbol{s}_1, \ldots, \boldsymbol{s}_n\}$, we assume the independent and identically distributed $\varepsilon(\boldsymbol{s}_i)$'s follow a Normal distribution $N(0, \tau^2)$, where τ^2 is called the *nugget*. The $w(\boldsymbol{s}_i)$'s provide local adjustment (with structured dependence) to the mean and capturing the effect of unmeasured or unobserved regressors with spatial pattern.

Customarily, one assumes *stationarity*, which means that $C(\boldsymbol{s}, \boldsymbol{t}) = C(\boldsymbol{s} - \boldsymbol{t})$ is a function of the separation of sites only. *Isotropy* goes further and specifies $C(\boldsymbol{s}, \boldsymbol{t}) = C(\|\boldsymbol{s} - \boldsymbol{t}\|)$, where $\|\boldsymbol{s} - \boldsymbol{t}\|$ is the Euclidean distance between the sites $\boldsymbol{s}$ and $\boldsymbol{t}$. We further specify $C(\boldsymbol{s}, \boldsymbol{t}) = \sigma^2 \rho(\boldsymbol{s}, \boldsymbol{t}; \boldsymbol{\phi})$ in terms of spatial process parameters, where $\rho(\cdot; \boldsymbol{\phi})$ is a *correlation function* while $\boldsymbol{\phi}$ includes parameters quantifying rate of correlation decay and smoothness of the surface $w(\boldsymbol{s})$. $\mathrm{Var}(w(\boldsymbol{s})) = \sigma^2$ represents a spatial variance component. Apart from the exponential, $\rho(\boldsymbol{s}, \boldsymbol{t}; \boldsymbol{\phi}) = \exp(-\phi \|\boldsymbol{s} - \boldsymbol{t}\|)$, and the powered exponential family, $\rho(\boldsymbol{s}, \boldsymbol{t}; \boldsymbol{\phi}) = \exp(-\phi \|\boldsymbol{s} - \boldsymbol{t}\|^\alpha)$, **spBayes** also offers users the Matérn correlation function

$$\rho(\|\boldsymbol{s} - \boldsymbol{t}\|; \boldsymbol{\phi}) = \frac{1}{2^{\nu-1}\Gamma(\nu)} (\|\boldsymbol{s} - \boldsymbol{t}\|\phi)^\nu \mathcal{K}_\nu(\|\boldsymbol{s} - \boldsymbol{t}\|\phi); \quad \phi > 0, \nu > 0. \tag{13}$$

Here $\boldsymbol{\phi} = \{\phi, \nu\}$ with ϕ controlling the decay in spatial correlation and ν controlling process smoothness. Specifically, if ν lies between positive integers m and $(m+1)$, then the spatial process $w(\boldsymbol{s})$ is mean-square differentiable m times, but not $m+1$ times. Also, Γ is the usual Gamma function while $\mathcal{K}_\nu$ is a modified Bessel function of the second kind with order ν.

The hierarchical model built from (12) emerges as a special case of (1), where $\boldsymbol{y}$ is $n \times 1$ with entries $y(\boldsymbol{s}_i)$, $\boldsymbol{X}$ is $n \times p$ with $\boldsymbol{x}(\boldsymbol{s}_i)^\top$ as its rows, $\boldsymbol{\alpha}$ is $n \times 1$ with entries $w(\boldsymbol{s}_i)$, $\boldsymbol{Z}(\boldsymbol{\theta}) = \boldsymbol{I}_n$, $\boldsymbol{K}(\boldsymbol{\theta})$ is $n \times n$ with entries $C(\boldsymbol{s}_i, \boldsymbol{s}_j; \boldsymbol{\theta})$ and $\boldsymbol{D}(\boldsymbol{\theta}) = \tau^2 \boldsymbol{I}_n$. We denote by $\boldsymbol{\theta}$ the set of process

parameters in $\boldsymbol{K}(\boldsymbol{\theta})$ and $\boldsymbol{D}(\boldsymbol{\theta})$. Therefore, with the Matérn covariance function in (13), we define $\boldsymbol{\theta} = \{\sigma^2, \phi, \nu, \tau^2\}$.

Example

The marginalized specification of (12) is implemented in the `spLM` function. The primary output of this function is posterior samples of $\boldsymbol{\theta}$. As detailed in the preceding sections, sampling is conducted using a Metropolis algorithm. Hence, users must specify Metropolis proposal variances, i.e., *tuning* values, and monitor acceptance rates for these parameters. Alternately, an adaptive MCMC Metropolis-within-Gibbs algorithm, proposed by Roberts and Rosenthal (2009), is available for a more automated function call.

A key advantage of the first stage Gaussian model is that samples from the posterior distribution of $\boldsymbol{\beta}$ and $\boldsymbol{w}$ can be recovered in a posterior predictive fashion, given samples of $\boldsymbol{\theta}$. In practice we often choose to only use a subset of post burn-in $\boldsymbol{\theta}$ samples to collect corresponding samples of $\boldsymbol{\beta}$ and $\boldsymbol{w}$. This *composition* sampling, detailed in Section 2.2, is conducted by passing a `spLM` object to the `spRecover` function.

An analysis of a synthetic dataset serves to illustrate use of the `spLM` and `spRecover` functions. The data are formed by drawing 200 observations from (12) within a unit square domain. The model mean includes an intercept and covariate with associated coefficients $\beta_0 = 1$ and $\beta_1 = 5$, respectively. Model residuals are generated using an exponential spatial correlation function, with $\tau^2 = 1$, $\sigma^2 = 2$ and $\phi = 6$. This choice of ϕ corresponds to an *effective spatial range* of 0.5 distance units. For our purposes, the effective spatial range is the distance at which the correlation equals 0.05. Figure 1(a) provides a surface plot of the observed spatial random effects along with the location of the 200 observations.

All `spLM` function arguments, and those of others functions highlighted in this paper, are defined in the package manual available on CRAN. Here we illustrate only some of the possible argument specifications. In addition to a symbolic model statement, the `spLM` function requires the user to specify: *i*) the number of MCMC samples to collect; *ii*) prior distribution, with associated hyperpriors for each parameter; *iii*) starting values for each parameter, and; *iv*) tuning values for each parameter, unless the adaptive MCMC option is chosen via the `amcmc` argument.

For this analysis, we assume an inverse-Gamma (IG) distribution for the variance parameters, τ^2 and σ^2. These distributions are assigned *shape* and *scale* hyperpriors equal to 2 and 1, respectively. With a shape of 2, the mean of the IG is equal to the scale and the variance is infinite. In practice, the choice of the scale value can be guided by exploratory data analysis using a variogram or similar tools that provide estimates of the spatial and non-spatial variances. The spatial decay parameter ϕ is assigned a uniform (U) prior with support that covers the extent of the domain. Here, we assume ϕ lies in the interval between 0.1 to 1 in distance units, i.e., working from our definition of the effective spatial range this corresponds to the prior $\text{U}(-log(0.05)/1, -log(0.05)/0.1)$. In the code below, we define these priors along with the other necessary arguments that are passed to `spLM`. The resulting posterior samples of $\boldsymbol{\theta}$ are summarized using the **coda** package's `summary` function and each parameter's posterior distribution median and 95% credible interval (CI) is printed.

```
R> n.samples <- 5000
R> starting <- list("tau.sq" = 1, "sigma.sq" = 1, "phi" = 6)
R> tuning <- list("tau.sq" = 0.01, "sigma.sq" = 0.01, "phi" = 0.1)
```

```
R> priors <- list("beta.Flat", "tau.sq.IG" = c(2, 1),
+    "sigma.sq.IG" = c(2, 1), "phi.Unif" = c(3, 30))
R> m.i <- spLM(y ~ X - 1, coords = coords, starting = starting,
+    tuning = tuning, priors = priors, cov.model = "exponential",
+    n.samples = n.samples, n.report = 2500)

----------------------------------------
        General model description
----------------------------------------
Model fit with 200 observations.
Number of covariates 2 (including intercept if specified).
Using the exponential spatial correlation model.
Number of MCMC samples 5000.

Priors and hyperpriors:
        beta flat.
        sigma.sq IG hyperpriors shape=2.00000 and scale=1.00000
        tau.sq IG hyperpriors shape=2.00000 and scale=1.00000
        phi Unif hyperpriors a=3.00000 and b=30.00000
------------------------------------------------
                Sampling
------------------------------------------------
Sampled: 2500 of 5000, 50.00%
Report interval Metrop. Acceptance rate: 66.12%
Overall Metrop. Acceptance rate: 66.12%
------------------------------------------------
Sampled: 5000 of 5000, 100.00%
Report interval Metrop. Acceptance rate: 64.80%
Overall Metrop. Acceptance rate: 65.46%
------------------------------------------------

R> burn.in <- floor(0.75 * n.samples)
R> round(summary(window(m.i$p.theta.samples,
+    start = burn.in))$quantiles[, c(3, 1, 5)], 2)

          50% 2.5% 97.5%
sigma.sq 2.66 1.56  6.78
tau.sq   0.85 0.43  1.28
phi      7.17 3.01 14.94
```

Samples from the posterior distribution of $\boldsymbol{\beta}$ and $\boldsymbol{w}$ are then obtained by calling the `spRecover` function as illustrates in the code below. The samples are again returned as a `mcmc` object that can be summarized accordingly.

```
R> m.i <- spRecover(m.i, start = burn.in, thin = 5, n.report = 100)

------------------------------------------------
                Recovering beta and w
```

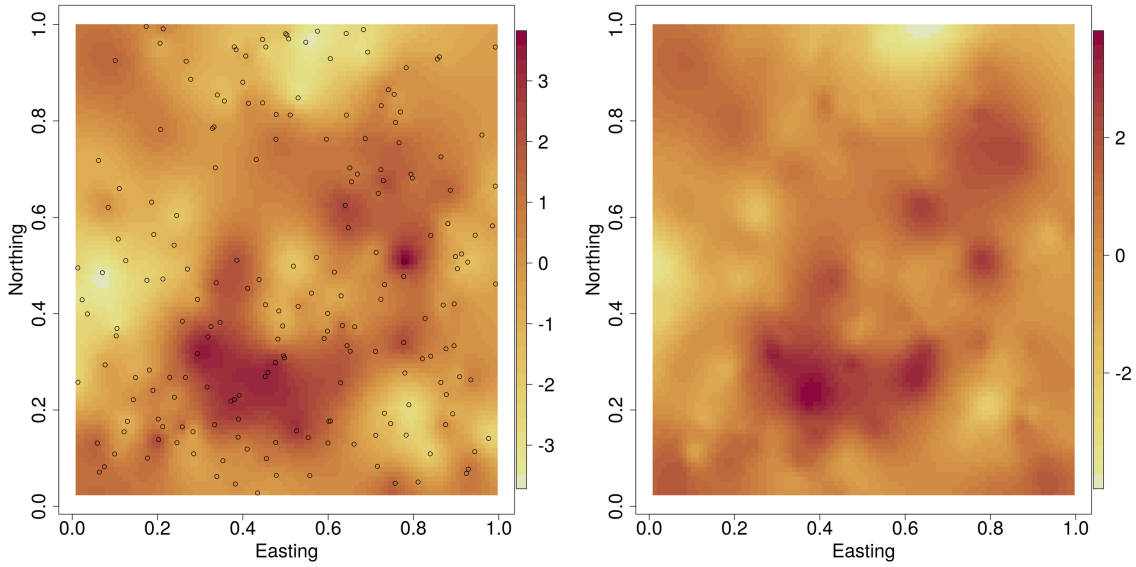

Figure 1: Interpolated surface of the observed (a) and estimated (b) spatial random effects.

```
------------------------------------------------
Sampled: 99 of 251, 39.44%
Sampled: 199 of 251, 79.28%

R> round(summary(m.i$p.beta.recover.samples)$quantiles[, c(3,1,5)], 2)

    50%  2.5% 97.5%
X1 0.71 -0.78  1.77
X2 4.96  4.79  5.17
```

In practice, it is often useful to pass the mean or median of each location's spatial random effect distribution through an interpolator to generate a surface plot. These surface estimates can be created using the `mba.surf` function available in the **MBA** (Finley and Banerjee 2010) package and plotted using the `image` or `image.plot` functions from the base **graphics** and **fields** (Nychka, Furrer, and Sain 2013) packages, respectively. Such a surface is presented in Figure 1(b) and matches closely the one depicting the synthetic data random effects in Figure 1(a).

```
R> w.hat <- apply(m.i$p.w.recover.samples, 1, median)
R> w.hat.surf <- mba.surf(cbind(coords, w.hat), no.X = res, no.Y = res,
+    extend = TRUE)$xyz.est
R> par(mar = c(5,5,0.2,0.2), cex.lab = 2, cex.axis = 2)
R> image.plot(w.hat.surf, xlab = "Easting", ylab = "Northing", xaxs = "r",
+    yaxs = "r", col = col)
```

As discussed in Section 1, reducing computing time was a key objective in reformulating and rewriting functions in **spBayes**. This same analysis conduced using the previous implementation of `spLM`, in version 0.2-4, required ∼8 minutes to generate 5,000 MCMC samples of $\boldsymbol{\theta}$. The previous implementation updated $\boldsymbol{\beta}$ from its full conditional distribution in each MCMC

iteration and sampled $\boldsymbol{\theta}$ using a Metropolis algorithm that did not take advantage of triangular solvers and other efficient computational approaches detailed in the preceding sections. For comparison, the current version of `spLM` generates the same number of samples in 0.031 minutes.

4.2. Low-rank predictive process models

spBayes offers low-rank models that allow the user to choose and fix $r \ll n$ within a hierarchical linear mixed model framework such as (1). Given the same modeling scenario as in Section 4.1, the user chooses r locations, say $\mathcal{S}^* = \{\boldsymbol{s}_1^*, \boldsymbol{s}_2^*, \ldots, \boldsymbol{s}_r^*\}$, and defines the process

$$\tilde{w}(\boldsymbol{s}) = \mathrm{E}[w(\boldsymbol{s}) \,|\, w(\boldsymbol{s}_i^*), i = 1, 2, \ldots, r] \,. \tag{14}$$

Banerjee, Gelfand, Finley, and Sang (2008) call $\tilde{w}(\boldsymbol{s})$ the *predictive process*. Replacing $w(\boldsymbol{s})$ with $\tilde{w}(\boldsymbol{s})$ in (12) yields the predictive process counterpart of the univariate Gaussian spatial regression model.

The predictive process produces a low-rank model and can be cast into (1). For example, if we take $\boldsymbol{\alpha}$ to the $r \times 1$ random vector with $w(\boldsymbol{s}_i^*)$ as its entries, then the predictive process counterpart of (12) is obtained from (1) with $\boldsymbol{D}(\boldsymbol{\theta}) = \tau^2\boldsymbol{I}$, $\boldsymbol{K}(\boldsymbol{\theta}) = \boldsymbol{C}^*(\boldsymbol{\theta})$ and $\boldsymbol{Z}(\boldsymbol{\theta}) = \mathcal{C}(\boldsymbol{\theta})^\top \boldsymbol{C}^*(\boldsymbol{\theta})^{-1}$, where $\mathcal{C}(\boldsymbol{\theta})^\top$ is $n \times r$ whose entries are the covariances between $w(\boldsymbol{s}_i)$'s and $w(\boldsymbol{s}_j^*)$'s and $\boldsymbol{C}^*(\boldsymbol{\theta})^{-1}$ is the $r \times r$ covariance matrix of the $w(\boldsymbol{s}_i^*)$'s.

When employing the computational strategy for generic low-rank models described in Section 2.3, an alternative, but equivalent, parametrization is obtained by letting $\boldsymbol{K}(\boldsymbol{\theta}) = \boldsymbol{C}^*(\boldsymbol{\theta})^{-1}$ and $\boldsymbol{Z}(\boldsymbol{\theta}) = \mathcal{C}(\boldsymbol{\theta})^\top$. This has the added benefit of avoiding the computation of $\boldsymbol{C}^*(\boldsymbol{\theta})^{-1}$, which, though not expensive for low-rank models, can become numerically unstable depending upon the choice of the covariance function. Now $\boldsymbol{\alpha} \sim N(\boldsymbol{0}, \boldsymbol{C}^*(\boldsymbol{\theta})^{-1})$ is no longer a vector of process realizations over the knots but it still is an $r \times 1$ random vector with a legitimate probability law. If the spatial effects over the knots are desired, they can be easily obtained from the posterior samples of $\boldsymbol{\alpha}$ and $\boldsymbol{\theta}$ as $\boldsymbol{C}^*(\boldsymbol{\theta})\boldsymbol{\alpha}$.

We also offer an improvement over the predictive process, which attempts to capture the residual from the low-rank approximation by adjusting for the residual variance (see, e.g., Finley, Sang, Banerjee, and Gelfand 2009). The difference between the spatial covariance matrices for the full rank model (12) and the low-rank model is $\boldsymbol{C}_w(\boldsymbol{\theta}) - \boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{K}(\boldsymbol{\theta})\boldsymbol{Z}(\boldsymbol{\theta})^\top$, where $\boldsymbol{C}_w(\boldsymbol{\theta})$ is the $n \times n$ covariance matrix of the spatial random effects for (12).

The modified predictive process model approximates this "residual" covariance matrix by absorbing its diagonal elements into $\boldsymbol{D}(\boldsymbol{\theta})$. Therefore, $\boldsymbol{D}(\boldsymbol{\theta}) = \mathrm{diag}\{\boldsymbol{C}_w(\boldsymbol{\theta}) - \boldsymbol{Z}(\boldsymbol{\theta})\boldsymbol{K}(\boldsymbol{\theta})\boldsymbol{Z}(\boldsymbol{\theta})^\top\} + \tau^2\boldsymbol{I}_n$, where $\mathrm{diag}(\boldsymbol{A})$ denotes the diagonal matrix formed with the diagonal entries of $\boldsymbol{A}$. The remaining specifications for $\boldsymbol{Z}(\boldsymbol{\theta})$, $\boldsymbol{K}(\boldsymbol{\theta})$ and $\boldsymbol{\alpha}$ in (1) remain the same as for the predictive process.

We often refer to the modified predictive process as $\tilde{w}_\varepsilon(\boldsymbol{s}) = \tilde{w}(\boldsymbol{s}) + \tilde{\epsilon}(\boldsymbol{s})$, where $\tilde{w}(\boldsymbol{s})$ is the predictive process and $\tilde{\epsilon}(\boldsymbol{s})$ is an independent process with zero mean and variance given by $\mathrm{var}\{w(\boldsymbol{s})\} - \mathrm{var}\{\tilde{w}(\boldsymbol{s})\}$. In terms of the covariance function of $w(\boldsymbol{s})$, the variance of $\tilde{\epsilon}(\boldsymbol{s})$ is $C(\boldsymbol{s}, \boldsymbol{s}; \boldsymbol{\theta}) - \boldsymbol{c}(\boldsymbol{s}, \boldsymbol{\theta})^\top \boldsymbol{C}^*(\boldsymbol{\theta})^{-1}\boldsymbol{c}(\boldsymbol{s})$, where $\boldsymbol{c}(\boldsymbol{s})$ is the $r \times 1$ vector of covariances between $w(\boldsymbol{s})$ and $w(\boldsymbol{s}^* j)$ as its entries. Also, $\boldsymbol{w}^*$, $\tilde{\boldsymbol{w}}$ and $\tilde{\boldsymbol{w}}_\epsilon$ denote the collection of $w(\boldsymbol{s}_i^*)$'s over the r knots, $\tilde{w}(\boldsymbol{s}_i)$'s over the n locations and $\tilde{w}_\epsilon(\boldsymbol{s}_i)$'s over the n locations respectively.

A key issue in low-rank models is the choice of knots. Given a computationally feasible r one could fix the knot locations using a grid over the extent of the domain, space-covering

design (e.g., Royle and Nychka 1998), or more sophisticated approach aimed at minimizing a predictive variance criterion (see, e.g., Finley *et al.* 2009; Guhaniyogi, Finley, Banerjee, and Gelfand 2011). In practice, if the observed locations are evenly distributed across the domain, we have found relatively small difference in inference based on knot locations chosen using a grid, space-covering design, or other criterion. Rather, it is the number of knots locations that has the greater impact on parameter estimates and subsequent prediction. Therefore, we often investigate sensitivity of inference to different knot intensities, within a computationally feasible range.

Example

Moving from (12) to its predictive process counterpart is as simple as passing a $r \times 2$ matrix of knot locations, via the `knots` argument, to the **spLM** function. Choice between the non-modified and modified predictive process model, i.e., $\tilde{w}(\boldsymbol{s})$ and $\tilde{w}_{\varepsilon}(\boldsymbol{s})$, is specified using the `modified.pp` logical argument. Passing a **spLM** object, specified for a predictive process model, to `spRecover` will yield posterior samples from $\tilde{\boldsymbol{w}}$ or $\tilde{\boldsymbol{w}}_{\varepsilon}$ and $\boldsymbol{w}^{*}$.

We construct a second synthetic dataset using the same model and parameter values from Section 4.1.1, but now generate 2,000 observations. Parameters are then estimated using the following candidate models: *i*) non-modified predictive process with 25 knot grid; *ii*) modified predictive process with 25 knot grid; *iii*) non-modified predictive process with 100 knot grid, and; *iv*) modified predictive process with 100 knot grid.

The `spLM` call for the 25 knot non-modified predictive process model is given below. The `starting`, `priors`, and `tuning` arguments are taken from Section 4.1.1. As noted above, the `knots` argument invokes the predictive process model. The value portion of this argument `c(5, 5, 0)` specifies a 5 by 5 knot grid with should be placed over the extent of the observed locations. The third value in this vector controls the extent of this grid, e.g., one may want the knot grid to extend beyond the convex haul of the observed locations. The placement of these knots is illustrated in Figure 2(b). Users can also pass in their own knot locations via the `knots` argument.

```
R> m.i <- spLM(y ~ X - 1, coords = coords, knots = c(5, 5, 0),
+    starting = starting, tuning = tuning, priors = priors,
+    cov.model = "exponential", modified.pp = FALSE, n.samples = n.samples,
+    n.report = 2500)

----------------------------------------
        General model description
----------------------------------------
Model fit with 2000 observations.
Number of covariates 2 (including intercept if specified).
Using the exponential spatial correlation model.
Using non-modified predictive process with 25 knots.
Number of MCMC samples 5000.

Priors and hyperpriors:
        beta flat.
        sigma.sq IG hyperpriors shape=2.00000 and scale=1.00000
```

	True	i	ii	iii	iv
β_0	1	0.64 (−0.52, 1.83)	0.63 (−0.37, 1.62)	0.77 (0.07, 1.40)	0.78 (0.03, 1.48)
β_1	5	4.99 (4.94, 5.05)	4.99 (4.94, 5.05)	4.98 (4.93, 5.03)	4.98 (4.93, 5.03)
σ^2	2	2.3 (1.45, 3.48)	1.57 (1.04, 2.13)	1.89 (1.19, 2.60)	1.65 (1.23, 3.41)
τ^2	1	1.72 (1.60, 1.84)	1.19 (0.98, 1.42)	1.41 (1.33, 1.51)	0.84 (0.56, 1.03)
ϕ	6	3.68 (3.00, 4.86)	3.39 (3.03, 4.17)	8.19 (5.62, 11.23)	7.75 (3.93, 11.3)
Time		0.13	0.14	0.70	0.77
Rel. time		0.17	0.19	0.92	1.00

Table 2: Candidate predictive process models' parameter estimates, run-time (wall time) in minutes, and run-time relative to model *iv*. Parameter posterior summary 50 (2.5, 97.5) percentiles.

```
        tau.sq IG hyperpriors shape=2.00000 and scale=1.00000
        phi Unif hyperpriors a=3.00000 and b=30.00000
------------------------------------------------
                Sampling
------------------------------------------------
Sampled: 2500 of 5000, 50.00%
Report interval Metrop. Acceptance rate: 35.20%
Overall Metrop. Acceptance rate: 35.20%
------------------------------------------------
Sampled: 5000 of 5000, 100.00%
Report interval Metrop. Acceptance rate: 32.24%
Overall Metrop. Acceptance rate: 33.72%
------------------------------------------------
```

Table 2 provides parameter estimates and run-time for all candidate models. Here, the predictive process induced upward bias, described in Section 4.2, is seen in model *i* and *iii* τ^2 estimates. This bias is removed by using the modified predictive process, as illustrated by model *ii* and *iv* variance parameter estimates. As show by the run-times, there is only a marginal difference in computation overhead between the non-modified and modified predictive process models. In most settings the modification should be used.

For comparison with Table 2, the full rank model required 5.18 minutes to generate the 5,000 posterior samples. Also parameter estimates from the full rank model were comparable to those of model *iv*. These attractive qualities of the predictive process models do not extend to all settings. For example, if the range of spatial dependence is short relative to the spacing of the knots, then covariance parameter estimation will suffer. We are obviously forgoing some information about the underlying spatial process when using an array of knots that is coarse compared to the number of observations. This is most easily seen by comparing estimated spatial random effects surfaces to the *true* surface used to generate the data, as shown in Figure 2. This smoothing of the random effects surface can translate into diminished predictive ability and, in some cases, model parameter inference, compared to a full rank model.

Following from Section 2.4, given coordinates and predictors for *new* locations, and a `spLM` object, the `spPredict` function returns posterior predictive samples from $\boldsymbol{y}_0$. The `spPredict` function provides a generic interface for prediction using most model functions in `spBayes`.

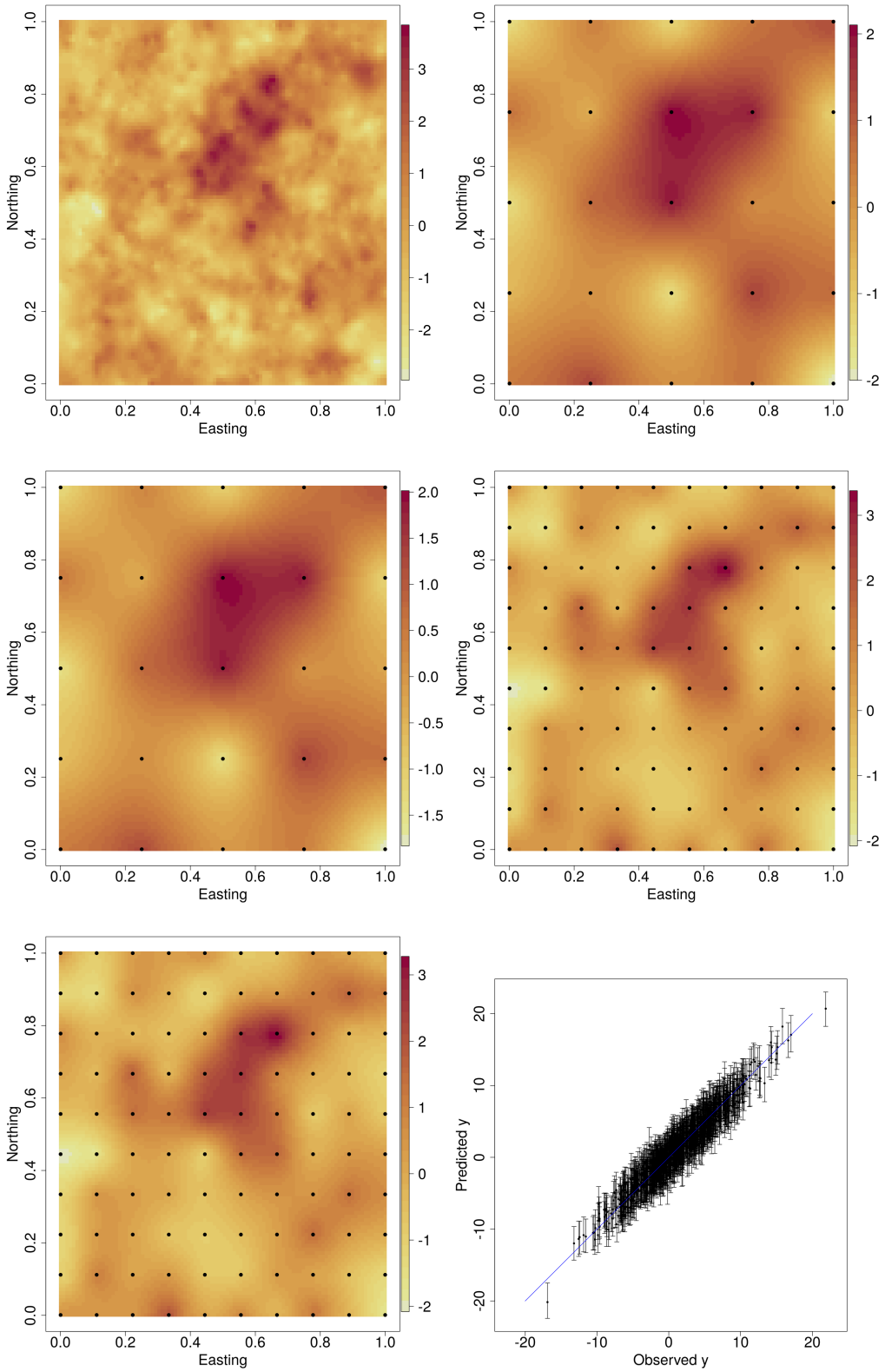

Figure 2: Interpolated surfaces of the (a) observed spatial random effects and (b), (c), (d), (e) are the estimated spatial random effects from models *i*, *ii*, *iii*, and *iv*, respectively. Filled circle symbols in (b), (c), (d), (e) show the location of predictive process knots. (f) plots holdout observed versus candidate model *iv* predicted median and 95% CI intervals with 1:1 line.

The code below illustrates prediction using model *iv* for 1,000 holdout locations. Here, `X.ho` is the $1,000 \times 2$ (i.e., $t \times p$) predictor matrix associated with the 1,000 holdout coordinates stored in `coords.ho`.

```
R> m.iv.pred <- spPredict(m.iv, start = burn.in, thin = 2,
+    pred.covars = X.ho, pred.coords = coords.ho, verbose = FALSE)
R> y.hat <- apply(m.iv.pred$p.y.predictive.samples, 1, quants)
R> par(mar = c(5, 5, 5, 5))
R> plot(y.ho, y.hat[1,], pch = 19, cex = 0.5, xlab = "Observed y",
+    ylab = "Predicted y", ylim = range(y.hat), xlim = range(y.hat),
+    cex.lab = 2, cex.axis = 2)
R> arrows(y.ho, y.hat[1,], y.ho, y.hat[2,], angle = 90, length = 0.05)
R> arrows(y.ho, y.hat[1,], y.ho, y.hat[3,], angle = 90, length = 0.05)
R> lines(-20 : 20, -20 : 20, col = "blue")
```

Figure 2(f) shows the observed versus predicted values for the holdout locations. We expect the posterior predictive 95% CIs will cover ∼950 of the *true* values in $\boldsymbol{y}_0$. For this analysis, our coverage rate was 94.4 percent.

5. Multivariate Gaussian spatial regression models

Multivariate spatial regression models consider m point-referenced outcomes that are regressed, at each location, on a known set of predictors

$$y_j(\boldsymbol{s}) = \boldsymbol{x}_j(\boldsymbol{s})^\top \boldsymbol{\beta}_j + w_j(\boldsymbol{s}) + \epsilon_j(\boldsymbol{s}) \ , \quad \text{for} \ \ j = 1, 2, \ldots, m \ , \tag{15}$$

where $\boldsymbol{x}_j(\boldsymbol{s})$ is a $p_j \times 1$ vector of predictors associated with outcome j, $\boldsymbol{\beta}_j$ is the $p_j \times 1$ slope, $w_j(\boldsymbol{s})$ and $\epsilon_j(\boldsymbol{s})$ are the spatial and random error processes associated with outcome $y_j(\boldsymbol{s})$. Customarily, we assume the unstructured residuals $\boldsymbol{\varepsilon}(\boldsymbol{s}) = (\epsilon_1(\boldsymbol{s}), \epsilon_2(\boldsymbol{s}), \ldots, \epsilon_m(\boldsymbol{s}))^\top$ follow a zero-centered multivariate normal distribution with zero mean and an $m \times m$ dispersion matrix Ψ.

Spatial variation is modeled using an $m \times 1$ Gaussian process $\boldsymbol{w}(\boldsymbol{s}) = (w_1(\boldsymbol{s}), \ldots, w_m(\boldsymbol{s}))^\top$, specified by a zero mean and a cross-covariance matrix $\boldsymbol{C}_w(\boldsymbol{s}, \boldsymbol{t})$ with entries being covariance between $w_i(\boldsymbol{s})$ and $w_j(\boldsymbol{t})$. **spBayes** uses the linear model of coregionalization (LMC) to specify the cross-covariance. This assumes that $\boldsymbol{C}_w(\boldsymbol{s}, \boldsymbol{t}) = \boldsymbol{A}\boldsymbol{M}(\boldsymbol{s}, \boldsymbol{t})\boldsymbol{A}^\top$, where $\boldsymbol{A}$ is $m \times m$ lower-triangular and $\boldsymbol{M}(\boldsymbol{s}, \boldsymbol{t})$ is $m \times m$ diagonal with each diagonal entry a spatial correlation function endowed with its own set of process parameters.

Suppose we have observed the m outcomes in each of b locations. Let $\boldsymbol{y}$ be $n \times 1$, where $n = mb$, obtained by stacking up the $\boldsymbol{y}(\boldsymbol{s}_i)$'s over the b locations. Let $\boldsymbol{X}$ be the $n \times p$ matrix of predictors associated with $\boldsymbol{y}$, where $p = \sum_{j=1}^{m} p_j$, and $\boldsymbol{\beta}$ is $p \times 1$ with the $\boldsymbol{\beta}_j$'s stacked correspondingly. Then, the hierarchical multivariate spatial regression models arise from (1) with the following specifications: $\boldsymbol{D}(\boldsymbol{\theta}) = \boldsymbol{I}_b \otimes \boldsymbol{\Psi}$, $\boldsymbol{\alpha}$ is $n \times 1$ formed by stacking the $\boldsymbol{w}_i$'s and $\boldsymbol{K}(\boldsymbol{\theta})$ is $n \times n$ partitioned into $m \times m$ blocks given by $\boldsymbol{A}\boldsymbol{M}(\boldsymbol{s}_i, \boldsymbol{s}_j)\boldsymbol{A}^\top$. The positive-definiteness of $\boldsymbol{K}(\boldsymbol{\theta})$ is ensured by the linear model of coregionalization (Gelfand, Schmidt, Banerjee, and Sirmans 2004). **spBayes** also offers low rank multivariate models involving the predictive process and the modified predictive process that can be estimated using strategies analogous to Section 2.3. Both the full rank multivariate Gaussian model and its predictive process

counterpart are implemented in the `spMvLM` function. Notation and additional background for fitting these models is given by Banerjee *et al.* (2008) and Finley *et al.* (2009) as well as example code in the `spMvLM` documentation examples.

6. Non-Gaussian models

Two typical non-Gaussian first stage settings are implemented in **spBayes**: *i*) binary response at locations modeled using logit or probit regression, and; *ii*) count data at locations modeled using Poisson regression. Diggle, Moyeed, and Tawn (1998) unify the use of generalized linear models in spatial data contexts. See also Lin, Wahba, Xiang, Gao, Klein, and Klein (2000), Kammann and Wand (2003) and Banerjee *et al.* (2004). Here we replace the Gaussian likelihood in (1) with the assumption that $\mathrm{E}[y(\boldsymbol{s})]$ is linear on a transformed scale, i.e., $\eta(\boldsymbol{s}) \equiv g(\mathrm{E}(y(\boldsymbol{s}))) = \boldsymbol{x}(\boldsymbol{s})^\top\boldsymbol{\beta} + w(\boldsymbol{s})$ where $g(\cdot)$ is a suitable link function. We refer to these as spatial generalized linear models (GLMs).

With the Gaussian first stage, we can marginalize over the spatial effects and implement our MCMC over a reduced parameter space. With a binary or Poisson first stage, such marginalization is precluded and we have to update the spatial effects in running our Gibbs sampler. We offer both the traditional random-walk Metropolis as well as the adaptive random-walk Metropolis (Roberts and Rosenthal 2009) to update the spatial effects. **spBayes** also provides low-rank predictive process versions for spatial GLMs. The analogue of (1) is

$$p(\boldsymbol{\theta}) \times N(\boldsymbol{\beta} \,|\, \boldsymbol{\mu}_\beta, \boldsymbol{\Sigma}_\beta) \times N(\boldsymbol{\alpha} \,|\, \mathbf{0}, \boldsymbol{K}(\boldsymbol{\theta})) \times \prod_{i=1}^{n} f(y(\boldsymbol{s}_i) \,|\, \eta(\boldsymbol{s}_i) \equiv \boldsymbol{x}(\boldsymbol{s}_i)^\top\boldsymbol{\beta} + \boldsymbol{z}_i(\boldsymbol{\theta})^\top\boldsymbol{\alpha}) \,, \quad (16)$$

where $f(\cdot)$ represents a Bernoulli or Poisson density with $\eta(\boldsymbol{s})$ represents the mean of $y(\boldsymbol{s})$ on a transformed scale. This model and its predictive process counterpart is implemented in the `spGLM` function. These models are extended to accommodate multivariate settings, outlined in Section 5, using the `spMvGLM` function.

7. Dynamic spatio-temporal models

There are many different flavors of spatio-temporal data and an extensive statistical literature that addresses the most common settings. The approach adopted here applies to the setting where space is viewed as continuous, but time is assumed to be discrete. Put another way, we view the data as a time series of spatial process realizations and work in the setting of dynamic models. Building upon previous work in the setting of dynamic models by West and Harrison (1997), several authors, including Stroud, Müler, and Sansó (2001) and Gelfand, Banerjee, and Gamerman (2005), proposed dynamic frameworks to model residual spatial and temporal dependence. These proposed frameworks are flexible and easily extended to accommodate nonstationary and multivariate outcomes.

Dynamic linear models, or state-space models, have gained tremendous popularity in recent years in fields as disparate as engineering, economics, genetics, and ecology. They offer a versatile framework for fitting several time-varying models (West and Harrison 1997). Gelfand *et al.* (2005) adapted the dynamic modeling framework to spatio-temporal models with spatially varying coefficients. Alternative adaptations of dynamic linear models to space-time data can be found in Stroud *et al.* (2001).

7.1. Model specification

spBayes offers a relatively simple version of the dynamic models in Gelfand *et al.* (2005). Suppose, $y_t(\boldsymbol{s})$ denotes the observation at location $\boldsymbol{s}$ and time t. We model $y_t(\boldsymbol{s})$ through a *measurement equation* that provides a regression specification with a space-time varying intercept and serially and spatially uncorrelated zero-centered Gaussian disturbances as measurement error $\epsilon_t(\boldsymbol{s})$. Next a *transition equation* introduces a $p \times 1$ coefficient vector, say $\boldsymbol{\beta}_t$, which is a purely temporal component (i.e., time-varying regression parameters), and a spatio-temporal component $u_t(\boldsymbol{s})$. Both these are generated through transition equations, capturing their Markovian dependence in time. While the transition equation of the purely temporal component is akin to usual state-space modeling, the spatio-temporal component is generated using Gaussian spatial processes. The overall model is written as

$$\begin{aligned} y_t(\boldsymbol{s}) &= \boldsymbol{x}_t(\boldsymbol{s})^\top \boldsymbol{\beta}_t + u_t(\boldsymbol{s}) + \epsilon_t(\boldsymbol{s}), \quad \epsilon_t(\boldsymbol{s}) \overset{ind.}{\sim} N(0, \tau_t^2)\ ; \\ \boldsymbol{\beta}_t &= \boldsymbol{\beta}_{t-1} + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \overset{i.i.d.}{\sim} N(0, \boldsymbol{\Sigma}_\eta)\ ; \\ u_t(\boldsymbol{s}) &= u_{t-1}(\boldsymbol{s}) + w_t(\boldsymbol{s}), \quad w_t(\boldsymbol{s}) \overset{ind.}{\sim} GP\left(\mathbf{0}, C_t(\cdot, \boldsymbol{\theta}_t)\right), \quad t = 1, 2, \ldots, N_t\ , \end{aligned} \tag{17}$$

where the abbreviations *ind.* and *i.i.d* are *independent* and *independent and identically distributed*, respectively. Here $\boldsymbol{x}_t(\boldsymbol{s})$ is a $p \times 1$ vector of predictors and $\boldsymbol{\beta}_t$ is a $p \times 1$ vector of coefficients. In addition to an intercept, $\boldsymbol{x}_t(\boldsymbol{s})$ can include location specific variables useful for explaining the variability in $y_t(\boldsymbol{s})$. The $GP(\mathbf{0}, C_t(\cdot, \boldsymbol{\theta}_t))$ denotes a spatial Gaussian process with covariance function $C_t(\cdot; \boldsymbol{\theta}_t)$. We customarily specify $C_t(\boldsymbol{s}_1, \boldsymbol{s}_2; \boldsymbol{\theta}_t) = \sigma_t^2 \rho(\boldsymbol{s}_1, \boldsymbol{s}_2; \phi_t)$, where $\boldsymbol{\theta}_t = \{\sigma_t^2, \phi_t\}$ and $\rho(\cdot; \phi)$ is a *correlation function* with ϕ controlling the correlation decay and σ_t^2 represents the spatial variance component. We further assume $\boldsymbol{\beta}_0 \sim N(\boldsymbol{m}_0, \boldsymbol{\Sigma}_0)$ and $u_0(\boldsymbol{s}) \equiv 0$, which completes the prior specifications leading to a well-identified Bayesian hierarchical model with reasonable dependence structures. In practice, estimation of model parameters are usually very robust to these hyper-prior specifications. Also note that (17) reduces to a simple spatial regression model for $t = 1$.

We consider settings where the inferential interest lies in spatial prediction or interpolation over a region for a set of discrete time points. We also assume that the same locations are monitored for each time point resulting in a space-time matrix whose rows index the locations and columns index the time points, i.e., the (i, j)-th element is $y_j(\boldsymbol{s}_i)$. Our algorithm will accommodate the situation where some cells of the space-time data matrix may have missing observations, as is common in monitoring environmental variables.

Conducting full Bayesian inference for (17) is computationally onerous and **spBayes** also offers a modified predictive process counterpart of (17). This is achieved by replacing $u_t(\boldsymbol{s})$ in (17) with $\tilde{u}_t(\boldsymbol{s}) = \sum_{k=1}^{t} [\tilde{w}_k(\boldsymbol{s}) + \tilde{\epsilon}_k(\boldsymbol{s})]$, where $\tilde{w}_k(\boldsymbol{s})$ is the predictive process as defined in (14) and the "adjustment" $\tilde{\epsilon}_t(\boldsymbol{s})$ compensates for the oversmoothing by the conditional expectation component and the consequent underestimation of spatial variability (see Finley, Banerjee, and Gelfand 2012) for details.

Example

The dynamic model (17) and its predictive process counterpart are implemented in the `spDynLM` function. Here we illustrate the full rank dynamic model using an ozone monitoring dataset that was previously analyzed by Sahu and Bakar (2011). This is a relatively

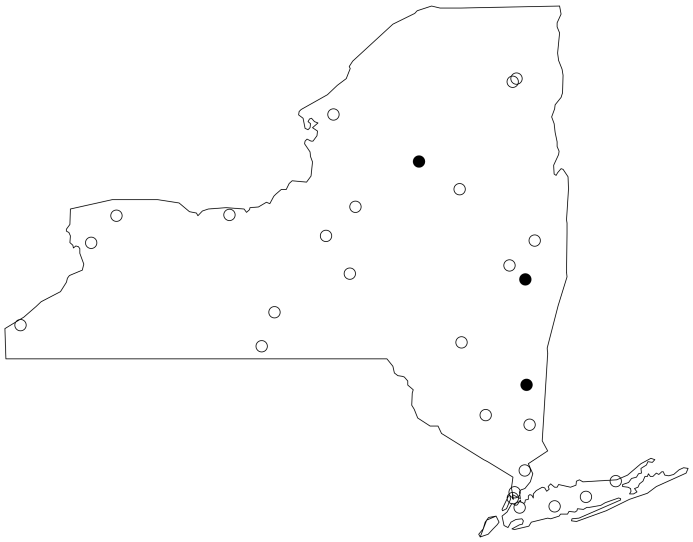

Figure 3: Open and filled circle symbols indicate the location of 28 ozone monitoring stations across New York State. Filled circle symbols identify those stations that have half of the daily ozone measurements withheld to assess model predictive performance.

small dataset and does not require dimension reduction. Note, however, similar to other **spBayes** models, moving from full to low-rank representation of $\boldsymbol{u}_t$ only requires specification of knot locations via the `knots` argument in the model call.

The dataset comprises 28 Environmental Protection Agency monitoring stations that recorded ozone from July 1 to August 31, 2006. The outcome is daily 8-hour maximum average ozone concentrations (parts per billion; O3.8HRMAX), and predictors include maximum temperature (Celsius; cMAXTMP), wind speed (knots; WDSP), and relative humidity (RM). Of the $1,736$ possible observations, i.e., n=28 locations times N_t=62 daily O3.8HRMAX measurements, 114 are missing. In this illustrative analysis we use the predictors cMAXTMP, WDSP, and RM as well as the spatially and temporally structured residuals to predict missing O3.8HRMAX values. To gain a better sense of the dynamic model's predictive performance, we withheld half of the observations from the records of three stations for subsequent validation. Figure 3 shows the monitoring station locations and identifies those stations where data were withheld.

The first `spDynLM` function argument is a list of N_t symbolic model statements representing the regression within each time step. This can be easily assembled using the `lapply` function as shown in the code below. Here too, we define the station coordinates as well as starting, tuning, and prior distributions for the model parameters. Exploratory data analysis using time step specific variograms can be helpful for defining starting values and prior support for parameters in $\boldsymbol{\theta}_t$ and τ_t^2. To avoid cluttering the code, we specify the same prior for the ϕ_t's, σ_t^2's, and τ_t^2's. As in the other `spBayes` model functions, one can choose among several popular spatial correlation functions including the exponential, spherical, Gaussian and Matérn. The exponential correlation function is specified in the `spDynLM` call below. Unlike other model functions described in the preceding sections, the `spDynLM` function will

accept NA $y_t(\boldsymbol{s})$ values. The sampler will provide posterior predictive samples for these missing values. If the `get.fitted` argument is `TRUE` then these posterior predictive samples are save along with posterior *fitted* values for locations where the outcomes are observed.

```
R> mods <- lapply(paste("O3.8HRMAX.", 1:N.t, "~cMAXTMP.", 1:N.t, "+WDSP.",
+    1:N.t, "+RH.", 1:N.t, sep = ""), as.formula)
R> p <- 4
R> coords <- NYOzone.dat[, c("X.UTM", "Y.UTM")]/1000
R> max.d <- max(iDist(coords))
R> starting <- list("beta" = rep(0, N.t * p),
+    "phi" = rep(3/(0.5*max.d), N.t), "sigma.sq" = rep(2,N.t),
+    "tau.sq" = rep(1, N.t), "sigma.eta" = diag(rep(0.01, p)))
R> tuning <- list("phi" = rep(2, N.t))
R> priors <- list("beta.0.Norm" = list(rep(0, p), diag(100000, p)),
+    "phi.Unif" = list(rep(3/(0.9 * max.d), N.t),
+      rep(3/(0.05 * max.d), N.t)),
+    "sigma.sq.IG" = list(rep(2, N.t), rep(25, N.t)),
+    "tau.sq.IG" = list(rep(2, N.t), rep(25, N.t)),
+    "sigma.eta.IW" = list(2, diag(0.001, p)))
R> n.samples <- 5000
R> m.i <- spDynLM(mods, data = NYOzone.dat, coords = as.matrix(coords),
+    starting = starting, tuning = tuning, priors = priors,
+    get.fitted = TRUE, cov.model = "exponential", n.samples = n.samples,
+    n.report = 2500)

----------------------------------------
        General model description
----------------------------------------
Model fit with 28 observations in 62 time steps.
Number of missing observations 117.
Number of covariates 4 (including intercept if specified).
Using the exponential spatial correlation model.
Number of MCMC samples 5000.

Priors and hyperpriors:
        beta normal:
        m_0:        0.000        0.000        0.000        0.000
        Sigma_0:
        100000.000        0.000        0.000        0.000
        0.000        100000.000        0.000        0.000
        0.000        0.000        100000.000        0.000
        0.000        0.000        0.000        100000.000

        sigma.sq_t=1 IG hyperpriors shape=2.00000 and scale=25.00000
        tau.sq_t=1 IG hyperpriors shape=2.00000 and scale=25.00000
        phi_t=1 Unif hyperpriors a=0.00564 and b=0.10145
        ---
```

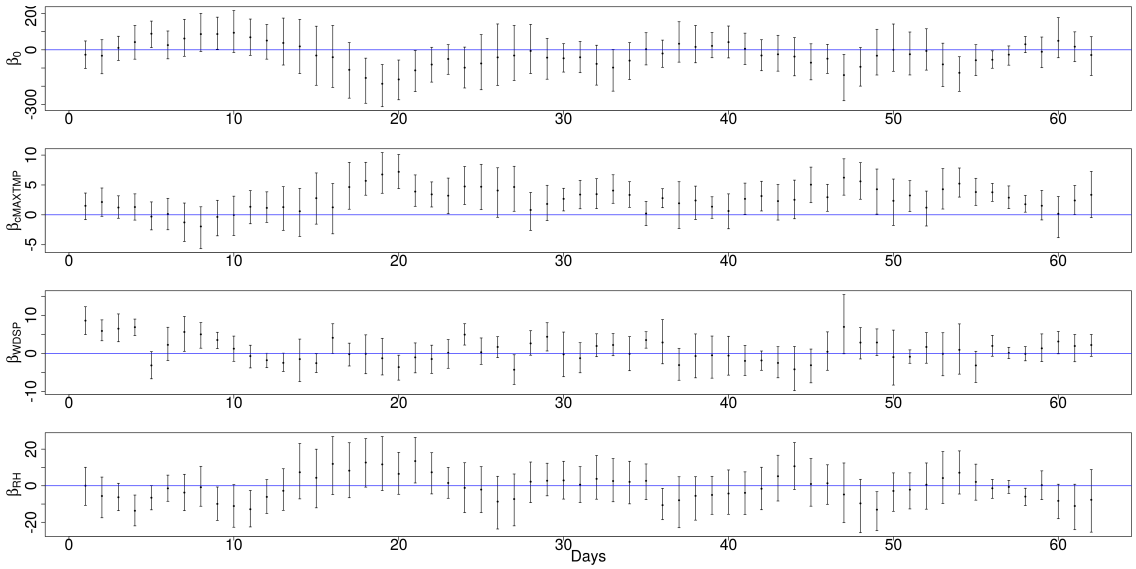

Figure 4: Posterior distribution medians and 95% credible intervals for model intercept and predictors.

```
        sigma.sq_t=2 IG hyperpriors shape=2.00000 and scale=25.00000
        tau.sq_t=2 IG hyperpriors shape=2.00000 and scale=25.00000
        phi_t=2 Unif hyperpriors a=0.00564 and b=0.10145
        ---
        ...
        ---
        sigma.sq_t=62 IG hyperpriors shape=2.00000 and scale=25.00000
        tau.sq_t=62 IG hyperpriors shape=2.00000 and scale=25.00000
        phi_t=62 Unif hyperpriors a=0.00564 and b=0.10145
        ---
-------------------------------------------------
                Sampling
-------------------------------------------------
Sampled: 2499 of 5000, 49.98%
Report interval Mean Metrop. Acceptance rate: 49.05%
Overall Metrop. Acceptance rate: 49.07%
-------------------------------------------------
Sampled: 4999 of 5000, 99.98%
Report interval Mean Metrop. Acceptance rate: 49.48%
Overall Metrop. Acceptance rate: 49.28%
-------------------------------------------------
```

Time series plots of parameters' posterior summary statistics are often useful for exploring the temporal evolution of the parameters. In the case of the regression coefficients, these plots describe the time-varying trend in the outcome and impact of covariates. For example, the sinusoidal pattern in the model intercept, β_0, seen in Figure 4, correlates strongly with both cMAXTMP, RM, and to a lesser degree with WDSP. With only a maximum of 28 observations

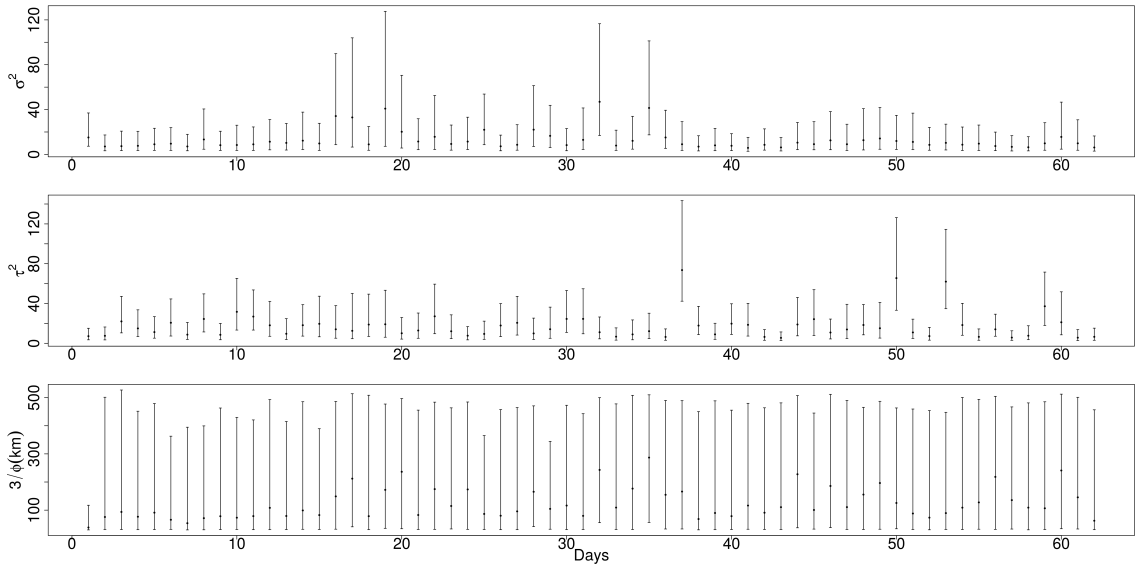

Figure 5: Posterior distribution medians and 95% credible intervals for $\boldsymbol{\theta}$ and τ^2.

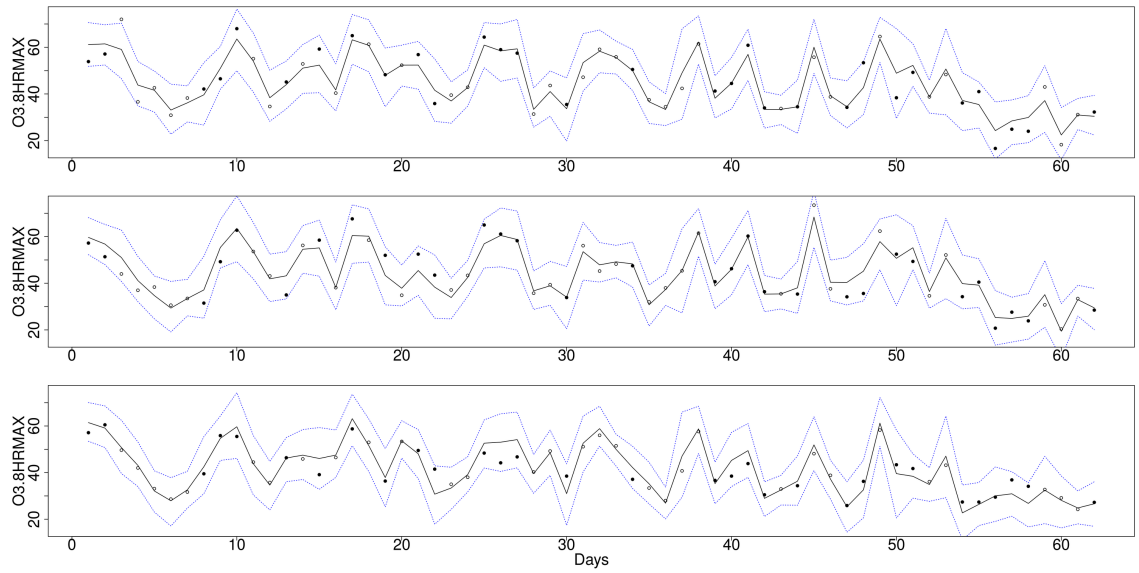

Figure 6: Posterior predicted distribution medians and 95% credible intervals, solid and dashed lines respectively, for three stations. Open circle symbols indicate those observations use for model parameter estimation and filled circle symbols indicate those observations withheld for validation.

within each time step, there is not much information to inform estimates of $\boldsymbol{\theta}$. As seen in Figure 5, this paucity of information is reflected in the imprecise CI's for the ϕ's and small deviations from the priors on σ^2 and τ^2. There are, however, noticeable trends in the variance components over time.

Figure 6 shows the observed and predicted values for the three stations used for validation. Here, open circle symbols indicate those observations used for parameter estimation and filled circles identify holdout observations. The posterior predicted median and 95% CIs are overlaid

using solid and dashed lines, respectively. Three of the 36 holdout measurements fell outside of their 95% predicted CI, a ∼92% coverage rate. As noted in Sahu and Bakar (2011), there is a noticeable reduction in ozone levels in the last two weeks in August.

8. Model choice

The `spDiag` function provides several approaches to assessing model performance and subsequent comparison for `spLM`, `spMvLM`, `spGLM`, and `spMvGLM` objects. These include the popular deviance information criterion (Spiegelhalter, Best, Carlin, and Linde 2002) as well as a measure of posterior predictive loss detailed in Gelfand and Ghosh (1998) and a scoring rule defined in Gneiting and Raftery (2007).

9. Summary and future direction

spBayes version 0.3-7 (CRAN 2013-06-01), and subsequent versions, offers a complete reformulation and rewrite of core functions for efficient estimation of univariate and multivariate models for point-referenced data using MCMC. Substantial increase in computational efficiency and flexibility in model specification, compared earlier **spBayes** package versions, is the result of careful MCMC sampler formulation that focused on reducing parameter space and avoiding expensive matrix operations. In addition, all core functions provide *predictive process* models able to accommodate large data sets that are being increasingly encountered in many fields.

We are currently developing an efficient modeling framework and sampling algorithm to accommodate multivariate spatially misaligned data, i.e., settings where not all of the outcomes are observed at all locations, that will be added to the `spMvLM` and `spMvGLM` functions. Prediction of these missing outcomes should borrow strength from the covariance among outcomes both within and across locations. In addition, we hope to add functions for non-stationary multivariate models such as those described in Gelfand *et al.* (2004) and more recent predictive process versions we developed in Guhaniyogi, Finley, Banerjee, and Kobe (2013). We will also continue developing `spDynLM` and helper functions. Ultimately, we would like to provide more flexible specifications of spatio-temporal dynamic models and allow them to accommodate non-Gaussian and multivariate outcomes.

Acknowledgments

This work was supported by National Science Foundation grants DMS-1106609, EF-1137309, EF-1241874, and EF-1253225, as well as NASA Carbon Monitoring System grants.

References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J, Croz JD, Greenbaum A, Hammarling S, McKenney A, Sorensen D (1999). ***LAPACK*** *Users' Guide*. 3rd edition. Society for Industrial and Applied Mathematics.

Bakar KS, Sahu SK (2015). "**spTimer**: Spatio-Temporal Bayesian Modelling Using R." *Journal of Statistical Software*, **63**(15), 1–32. URL http://www.jstatsoft.org/v63/i15/.

Banerjee S, Carlin CP, Gelfand AE (2004). *Hierarchical Modeling and Analysis for Spatial Data*. Chapman & Hall/CRC, Boca Raton.

Banerjee S, Gelfand AE, Finley AO, Sang H (2008). "Gaussian Predictive Process Models for Large Spatial Data Sets." *Journal of the Royal Statistical Society B*, **70**(4), 825–848.

Bivand R (2014). *CRAN Task View: Analysis of Spatial Data*. Version 2014-12-18, URL http://CRAN.R-project.org/view=Spatial.

Blackford LS, Demmel J, Dongarra J, Duff I, Hammarling S, Henry G, Heroux M, Kaufman L, Lumsdaine A, Petitet A, Pozo R, Remington K, Whaley RC (2001). "An Updated Set of Basic Linear Algebra Subprograms (**BLAS**)." *ACM Transactions on Mathematical Software*, **28**, 135–151.

Bochner S (1955). *Harmonic Analysis and the Theory of Probability*. California Monographs in mathematical sciences. University of California Press.

Carlin BP, Louis TA (2011). *Bayesian Methods for Data Analysis*. 3rd edition. Taylor & Francis.

Chilès JP, Delfiner P (2012). *Geostatistics: Modeling Spatial Uncertainty*. John Wiley & Sons.

Christensen OF, Ribeiro PJ (2002). "**geoRglm**: A Package for Generalized Linear Spatial Models." *R News*, **2**(2), 26–28. URL http://CRAN.R-project.org/doc/Rnews/.

Cressie N, Wikle CK (2011). *Statistics for Spatio-Temporal Data*. John Wiley & Sons.

Cressie NAC (1993). *Statistics for Spatial Data*. 2nd edition. John Wiley & Sons.

Diggle PJ, Moyeed RA, Tawn JA (1998). "Model-Based Geostatistics." *Applied Statistics*, **47**, 299–350.

Diggle PJ, Ribeiro PJ (2007). *Model-Based Geostatistics*. Springer-Verlag.

Finley A, Banerjee S, Gelfand A (2012). "Bayesian Dynamic Modeling for Large Space-Time Datasets Using Gaussian Predictive Processes." *Journal of Geographical Systems*, **14**(1), 29–47.

Finley AO, Banerjee S (2010). ***MBA**: Multilevel B-Spline Approximation*. R package version 0.0-7, URL http://CRAN.R-project.org/package=MBA.

Finley AO, Banerjee S (2013). ***spBayes**: Univariate and Multivariate Spatial-Temporal Modeling*. R package version 0.3-8, URL http://CRAN.R-project.org/package=spBayes.

Finley AO, Banerjee S, Carlin BP (2007). "**spBayes**: An R Package for Univariate and Multivariate Hierarchical Point-Referenced Spatial Models." *Journal of Statistical Software*, **19**(4), 1–24. URL http://www.jstatsoft.org/v19/i04/.

Finley AO, Sang H, Banerjee S, Gelfand AE (2009). "Improving the Performance of Predictive Process Modeling for Large Datasets." *Computational Statistics & Data Analysis*, **53**(8), 2873–2884.

Gelfand A, Banerjee S, Gamerman D (2005). "Univariate and Multivariate Dynamic Spatial Modelling." *Environmetrics*, **16**, 465–479.

Gelfand A, Schmidt A, Banerjee S, Sirmans C (2004). "Nonstationary Multivariate Process Modeling through Spatially Varying Coregionalization." *Test*, **13**(2), 263–312.

Gelfand AE, Ghosh SK (1998). "Model Choice: A Minimum Posterior Predictive Loss Approach." *Biometrika*, **85**, 1–11.

Gelman A, Carlin JB, Stern HS, Rubin DB (2004). *Bayesian Data Analysis*. Chapman & Hall/CRC.

Gilks WR, Richardson S, Spiegelhalter DJ (1996). *Markov Chain Monte Carlo in Practice*. Chapman and Hall.

Gneiting T, Raftery AE (2007). "Strictly Proper Scoring Rules, Prediction, and Estimation." *Journal of the American Statistical Association*, **102**, 359–378.

Guhaniyogi R, Finley AO, Banerjee S, Gelfand AE (2011). "Adaptive Gaussian Predictive Process Models for Large Spatial Datasets." *Environmetrics*, **22**(8), 997–1007.

Guhaniyogi R, Finley AO, Banerjee S, Kobe RE (2013). "Modeling Low-rank Spatially-varying Cross-covariances Using Predictive Wrocesses with Application to Soil Nutrient Data." *Journal of Agricultural, Biological, and Environmental Statistics*, **18**, 274–298.

Hankin RKS (2013). ***magic****: Create and Investigate Magic Squares*. R package version 1.5-4, URL http://CRAN.R-project.org/package=magic.

Henderson HV, Searle SR (1981). "On Deriving the Inverse of a Sum of Matrices." *SIAM Review*, **23**(1), 53–60.

Intel (2013). *Intel Math Kernel Library – Documentation*. URL http://software.intel.com/en-us/articles/intel-math-kernel-library-documentation.

Kammann EE, Wand MP (2003). "Geoadditive Models." *Applied Statistics*, **52**, 1–18.

Lin X, Wahba G, Xiang D, Gao F, Klein R, Klein B (2000). "Smoothing Spline ANOVA Models for Large Data Sets with Bernoulli Observations and the Randomized GACV." *The Annals of Statistics*, pp. 1570–1600.

Lunn D, Spiegelhalter D, Thomas A, Best N (2009). "The BUGS Project: Evolution, Critique and Future Directions." *Statistics in Medicine*, **28**, 3049–3067.

Møller J, Waagepetersen RP (2003). *Statistical Inference and Simulation for Spatial Point Processes*. Taylor & Francis.

Nychka D, Furrer R, Sain S (2013). ***fields****: Tools for Spatial Data*. R package version 6.9.1, URL http://CRAN.R-project.org/package=fields.

Pebesma E (2014). *CRAN Task View: Handling and Analyzing Spatio-Temporal Data*. Version 2014-12-18, URL http://CRAN.R-project.org/view=SpatioTemporal.

Plate T, Heiberger R (2013). ***abind**: Combine Multi-Dimensional Arrays*. R package version 1.4-0, URL http://CRAN.R-project.org/package=abind.

Plummer M, Best N, Cowles K, Vines K (2006). "**coda**: Convergence Diagnosis and Output Analysis for MCMC." *R News*, **6**(1), 7–11. URL http://CRAN.R-project.org/doc/Rnews/.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org.

Ribeiro PJ, Diggle PJ (2001). "**geoR**: A Package For Geostatistical Analysis." *R News*, **1**(2), 15–18. URL http://CRAN.R-project.org/doc/Rnews/.

Robert CP, Casella G (2004). *Monte Carlo Statistical Methods*. 2nd edition. Springer-Verlag.

Roberts GO, Rosenthal JS (2009). "Examples of Adaptive MCMC." *Journal of Computational and Graphical Statistics*, **18**(2), 349–367.

Royle J, Nychka D (1998). "An Algorithm for the Construction of Spatial Coverage Designs with Implementation in S-PLUS." *Computers & Geosciences*, **24**(5), 479–488.

Sahu SK, Bakar KS (2011). "A Comparison of Bayesian Models for Daily Ozone Concentration Levels." *Statistical Methodology*, **9**, 144–157.

Schabenberger O, Gotway CA (2004). *Statistical Methods for Spatial Data Analysis*. Taylor & Francis.

Sigrist F, Kuensch HR, Stahel WA (2015). "**spate**: An R Package for Spatio-Temporal Modeling with a Stochastic Advection-Diffusion Process." *Journal of Statistical Software*, **63**(14), 1–23. URL http://www.jstatsoft.org/v63/i14/.

Smith BJ, Yan J, Cowles MK (2008). "Unified Geostatistical Modeling for Data Fusion and Spatial Heteroskedasticity with R Package **ramps**." *Journal of Statistical Software*, **25**(10), 1–21. URL http://www.jstatsoft.org/v25/i10/.

Spiegelhalter SD, Best NG, Carlin BP, Linde AVD (2002). "Bayesian Measures of Model Complexity and Fit." *Journal of the Royal Statistical Society B*, **64**(4), 583–639.

Stroud J, Müler P, Sansó B (2001). "Dynamic Models for Spatio-Temporal Data." *Journal of the Royal Statistical Society B*, **63**, 673–689.

Thomas A, O Hara B, Ligges U, Sturtz S (2006). "Making BUGS Open." *R News*, **6**, 12–17. URL http://CRAN.R-project.org/doc/Rnews/.

Wackernagel H (2003). *Multivariate Geostatistics*. Springer-Verlag.

West M, Harrison J (1997). *Bayesian Forecasting and Dynamic Models*. 2nd edition. Springer-Verlag.

Zeileis A, Croissant Y (2010). "Extended Model Formulas in R: Multiple Parts and Multiple Responses." *Journal of Statistical Software*, **34**(1), 1–13. URL http://www.jstatsoft.org/v34/i01/.

Affiliation:

Andrew O. Finley
Departments of Forestry and Geography
Michigan State University
Natural Resources Building
480 Wilson Road, Room 126
East Lansing, MI 48824-1222, United States of America
E-mail: finleya@msu.edu

Sudipto Banerjee
Department of Biostatistics
University of California, Los Angeles
Fielding School of Public Health
Los Angeles, CA 90095-1772, United States of America
E-mail: baner009@umn.edu

Alan E. Gelfand
Department of Statistical Science
Duke University
Box 90251
Durham, NC 27708-0251, United States of America
E-mail: alan@stat.duke.edu

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 63, Issue 13 *Submitted:* 2013-06-03
January 2015 *Accepted:* 2014-09-04